

DB2 Version 9

für Linux, UNIX und Windows

BBE Version 3
für Linux, UNIX und Windows

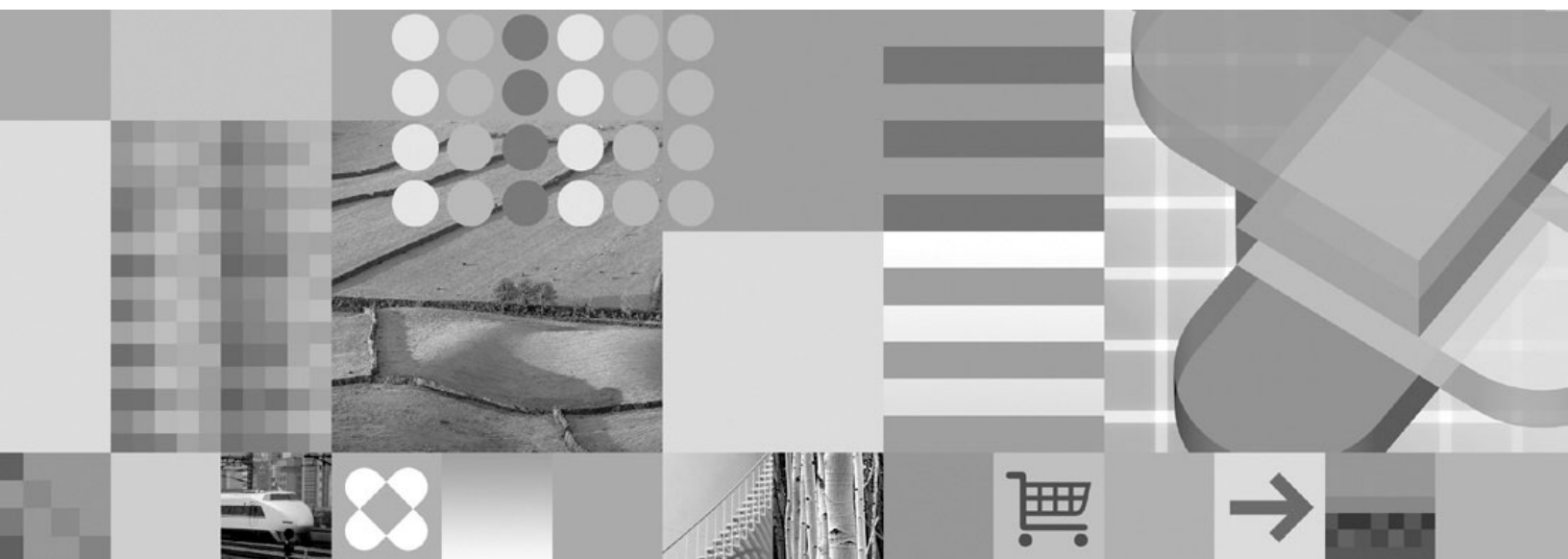


Systemverwaltung: Optimierung

DB2®

IBM

DB2 Version 9
für Linux, UNIX und Windows



Systemverwaltung: Optimierung

Hinweis

Vor Verwendung dieser Informationen und des darin beschriebenen Produkts sollten die allgemeinen Informationen unter *Bemerkungen* gelesen werden.

Erste Ausgabe (August 2006)

Diese Veröffentlichung ist eine Übersetzung des Handbuchs
IBM DB2 Version 9 for Linux, Unix, and Windows, Performance Guide,
IBM Form SC10-4222-00,
herausgegeben von International Business Machines Corporation, USA

© Copyright International Business Machines Corporation 2006
© Copyright IBM Deutschland GmbH 2006

Informationen, die nur für bestimmte Länder Gültigkeit haben und für Deutschland, Österreich und die Schweiz nicht zutreffen, wurden in dieser Veröffentlichung im Originaltext übernommen.

Möglicherweise sind nicht alle in dieser Übersetzung aufgeführten Produkte in Deutschland angekündigt und verfügbar; vor Entscheidungen empfiehlt sich der Kontakt mit der zuständigen IBM Geschäftsstelle.

Änderung des Textes bleibt vorbehalten.

Herausgegeben von:
SW TSC Germany
Kst. 2877
August 2006

Inhaltsverzeichnis

Teil 1. Einführung zur Leistung 1

Kapitel 1. Leistungsprinzipien 3

Elemente der Leistung	3
Richtlinien zur Leistungsoptimierung	3
Entwickeln eines Leistungsverbesserungsprozesses	5
Grenzen der Leistungsoptimierung	6
Leistungsinformationen, die Benutzer liefern können	6
Einstiegstipps für die Leistungsoptimierung	7
Leistungsfaktoren für Plattenspeichern.	8

Teil 2. Leistungsprobleme beim Datenbankentwurf 9

Kapitel 2. Leistungsplanung beim Datenbankentwurf 11

Funktionen zur Leistungsverbesserung	11
MQTs (MQT - Materialized Query Table, gespeicherte Abfragetabelle)	11
Tabellenpartitionierung und MDC-Tabellen	13
Optimierungsstrategien für MDC-Tabellen	19
Optimierungsstrategien für partitionierte Tabellen	21

Kapitel 3. Indizes 29

Verwenden von relationalen Indizes zur Leistungsverbesserung	29
Indexstruktur.	31
Tipps zur Planung von relationalen Indizes.	33
Tipps zur Leistung von relationalen Indizes	35
Indexieren von XML-Daten - Übersicht	39
Indizes in partitionierten Tabellen	41
Indexverhalten bei partitionierten Tabellen	41
Clusterindexverhalten bei partitionierten Tabellen	45
Asynchrone Indexbereinigung	48

Kapitel 4. Designadvisor 51

Designadvisor	51
Ausgabe des Designadvisors	52
Designadvisor-Funktionen	53
Verwenden des Designadvisors.	55
Definieren einer Auslastung für den Designadvisor	56
Verwenden des Designadvisors zur Migration von einer Datenbank mit einer einzelnen Partition auf eine Datenbank mit mehreren Partitionen	57
Begrenzungen und Einschränkungen des Designadvisors	58

Kapitel 5. Verwalten des gemeinsamen Zugriffs 61

Aspekte des gemeinsamen Zugriffs	61
Isolationsstufen	62
Isolationsstufen und Leistung	62
Angaben der Isolationsstufe	66

Sperren.	69
Sperren und Steuerung des gemeinsamen Zugriffs.	69
Sperrenattribute	71
Sperren und Arten der Anwendungsverarbeitung	73
Sperrgranularität	74
Sperrenumwandlung	75
Sperreneskalation	76
Korrigieren von Problemen der Sperreneskalation	76
Deadlocks	78
Wartestatus und Zeitlimitüberschreitungen für Sperren.	80
Angabe einer Strategie für den Modus 'Wartestatus für Sperre'	81
Verhindern von auf Sperren bezogenen Leistungsproblemen	82
Sperrtypenkompatibilität	84
Sperren und Datenzugriffsmethoden	85
Sperrmodi und Zugriffspfade für Standardtabellen.	87
Indextypen und Sperren des nächsten Schlüssels	91
Auswerten nicht festgeschriebener Daten durch Sperrenverzögerung	92
Option zum Ignorieren nicht festgeschriebener Einfügungen	95
Sperrmodi für Tabellen- und Satz-ID-Indexsuchen für MDC-Tabellen	96
Sperren für Blockindexsuchen für MDC-Tabellen	101
Sperrverhalten für partitionierte Tabellen	104

Kapitel 6. DB2 Governor. 107

Das Dienstprogramm Governor	107
Der Governor-Dämon	108
Starten und Stoppen von Governor	109
Konfigurieren von Governor	110
Die Konfigurationsdatei von Governor	111
Elemente für Governor-Regeln.	114
Beispiel für eine Governor-Konfigurationsdatei	119
Governor-Protokolldateien	120
Abfragen von Governor-Protokolldateien	124

Kapitel 7. Maximieren der E/A-Effizienz. 125

Reduzieren der E/A-Operationen durch Optimieren der SQL- und XQuery-Anweisungen	125
Angaben von Zeilenblockung zur Verringerung des Systemaufwands	125
Effiziente SELECT-Anweisungen	126
Stichprobendaten in SQL- und XQuery-Abfragen.	128
Optimieren der Sortierleistung.	129
Pufferpools und Vorablesezugriff.	131
Pufferpoolverwaltung	131
Verwaltung mehrerer Datenbankpufferpools	134
Pufferpoolverwaltung von Datenseiten	136

Illustration der Datenseitenverwaltung in	
Pufferpools	138
Aktualisierungsverarbeitung	139
Proaktive Seitenbereinigung	140
Vorablesen von Daten in den Pufferpool	141
Sequenzielles Vorablesen	142
Blockorientierte Pufferpools für besseren	
sequenziellen Vorablesezugriff.	144
Vorablesezugriff über Listen	145
Konfiguration von E/A-Servern für Vorablese-	
zugriff und Parallelität	146
Illustration des Vorablesens mit paralleler E/A	
Verwaltung der parallelen Ein-/Ausgabe	148
Verbessern der Einfügeleistung	150
Verringern des Protokollierungsaufwands zur	
Verbesserung der Abfrageleistung	151

Kapitel 8. Agentenverwaltung 153

Datenbankagenten.	153
Übersicht über die Architektur und Prozesse von	
DB2	155
Das DB2-Prozessmodell	158
Unterschiede zwischen Windows und UNIX	162
Verwaltung von Datenbankagenten	163
Agenten in einer partitionierten Datenbank	163
Konfigurationsparameter mit Auswirkung auf die	
Anzahl von Agenten	165
Client-/Serververarbeitungsmodell	166
Verbesserungen des Verbindungskonzentrators für	
Clientverbindungen	171

Kapitel 9. Entwurf für den optimalen Datenzugriff 175

Der SQL- und XQuery-Compilerprozess	175
Auswählen von Optimierungsklassen	179
Optimierungsklassen	181
Einstellen der Optimierungsklasse	184
Konfigurationsparameter mit Einfluss auf die	
Abfrageoptimierung	186
Auswirkung von Datenbankpartitionsgruppen auf	
die Abfrageoptimierung	189
Auswirkung von Tabellenbereichen auf die	
Abfrageoptimierung	189
Optimieren von Zugriffsplänen	192
Datenzugriffsmethoden	192
Datenzugriff über Indexsuchen	193
Arten des Indexzugriffs	196
Indexzugriff und Clusterverhältnisse	199
Vergleichselementtypologie und Zugriffspläne	
Auswirkungen des Sortierens und Gruppierens	
Berechnen von Gruppierungsschlüsselkardina-	
litäten mit Index- und Spaltengruppenstatistik-	
daten	204
Verbessern der Leistung durch Binden mit	
REOPT	205
Replizierte MQTs in partitionierten Datenbank-	
umgebungen	206
Optimierungsstrategien für partitionsinterne	
Parallelität	208
Statistische Sichten	211

Statistische Sichten	211
Verwenden statistischer Sichten	212
Anzeigen der für die Optimierung relevanten	
Statistikdaten	213
Szenario: Verbessern der Kardinalitätsschätzung	
mit Hilfe statistischer Sichten	214
Joins	220
Joins	220
Joinmethoden	221
Strategien zur Auswahl optimaler Joins.	224
Spaltenkorrelation für mehrere Vergleichs-	
elemente	227
Joinstrategien in partitionierten Datenbanken	
Joinmethoden in Umgebungen mit partitionier-	
ten Datenbanken	231
Umschreiben von Abfragen.	236
Methoden zum Umschreiben von Abfragen und	
Beispiele	236
Beispiel für das Umschreiben durch den Compil-	
er: Zusammenfügen von Sichten.	238
Beispiel für das Umschreiben durch den Compil-	
er: Eliminierung von DISTINCT	240
Beispiel für das Umschreiben durch den Compil-	
er: implizierte Vergleichselemente	242
Zugriffspläne in föderierten Abfragen	244
Pushdown-Analyse für föderierte Datenbanken	
Richtlinien zur Analyse, wo eine Abfrage für	
föderierte Datenbanken ausgewertet wird	249
Globale Analyse von Abfragen auf föderierte	
Datenbanken	250
Generierung von fernem SQL und globale Opti-	
mierung in föderierten Datenbanken	253
Serveroptionen mit Einfluss auf föderierte	
Datenbanken	256

Kapitel 10. Analysieren von Zugriffs- plänen mit der EXPLAIN-Funktion . . . 257

EXPLAIN-Einrichtung	257
dynexpln	258
EXPLAIN-Tools	258
Richtlinien zur Verwendung von EXPLAIN-Infor-	
mationen	260
Die EXPLAIN-Tabellen und die Organisation von	
EXPLAIN-Informationen	261
EXPLAIN-Informationen für Datenobjekte.	264
EXPLAIN-Informationen für Instanzen	264
EXPLAIN-Informationen für Datenoperatoren	268
Richtlinien zur Erfassung von EXPLAIN-Informati-	
onen	269
Richtlinien zur Analyse von EXPLAIN-Informatio-	
nen.	271
Hinweise zur Verwendung von dynexpln	273

Kapitel 11. Konfigurieren von DB2- Instanzen und -Datenbanken. 275

Konfigurationsparameter	275
Konfigurieren von DB2 mit Konfigurations-	
parametern	277
Dynamische Konfiguration von Parametern	281

Generieren von Empfehlungen für die Datenbank- konfiguration	284
Konfigurieren der DB2-Hauptspeicherzuordnung	286
Hauptspeicherzuordnung in DB2.	286
Gemeinsam genutzter Speicher des Datenbank- managers.	289
Der FCM-Pufferpool und Speicheran- forderungen.	291
Optimieren der Parameter für die Hauptspeicherzuordnung	293
Automatisches Konfigurieren mit dem Speicher mit automatischer Leistungsoptimierung	295
Speicher mit automatischer Leistungs- optimierung.	295
Aktivieren des Speichers mit automatischer Leistungsoptimierung	296
Inaktivieren des Speichers mit automatischer Leistungsoptimierung	297
Ermitteln der Speicherkonsumenten mit akti- vierter automatischer Leistungsoptimierung	299
Speicher mit automatischer Leistungs- optimierung in Umgebungen mit partitionierten Datenbanken	300
Verwenden von Speicher mit automatischer Leistungsoptimierung in Umgebungen mit par- titionierten Datenbanken	302
Betriebsmerkmale und Einschränkungen von Speicher mit automatischer Leistungs- optimierung.	304
Konfigurationsparameter - Übersicht	305

Teil 3. Erhalten der Systemleistung 317

Kapitel 12. Erfassen von Statistik- daten 319

Katalogstatistiken	319
Erfassen von Katalogstatistiken	322
Erfassen von Verteilungsstatistiken für bestimmte Spalten	324
Erfassen von Indexstatistiken	325
Richtlinien für die Erfassung und Aktualisierung von Statistiken	326
Erfassen von Statistiken an einer Stichprobe der Tabellendaten	329
Katalogstatistiktabellen	330
Verteilungsstatistiken	335
Detaillierte Indexstatistiken.	338
Unterelementstatistiken	340
Statistiken für benutzerdefinierte Funktionen.	341
Verwendung der Verteilungsstatistiken durch das Optimierungsprogramm.	343
Erweiterte Beispiele zur Verwendung von Verteilungsstatistiken.	345
Katalogstatistiken zu Modellierung und Fallstudien	349
Statistiken zur Modellierung von Produktions- datenbanken.	350
Manuelles Aktualisieren von Katalogstatistiken	353
Allgemeine Regeln zur manuellen Aktualisie- rung von Katalogstatistiken	353

Regeln zur manuellen Aktualisierung von Spaltenstatistiken	354
Regeln zur manuellen Aktualisierung von Verteilungsstatistiken.	355
Regeln zur manuellen Aktualisierung von Tabel- len- und Kurznamenstatistiken	356
Regeln zur manuellen Aktualisierung von Indexstatistiken.	357
Automatische Statistikerfassung	358
Automatische Statistikerfassung	358
Verwenden der automatischen Statistikerfassung	359
Automatische Statistikprofilerstellung	360
Erfassen von Statistiken mit einem Statistikprofil	361

Kapitel 13. Tabellen- und Index- verwaltung 365

Tabellenreorganisation	365
Indexreorganisation	366
Feststellen des Zeitpunkts zur Reorganisation von Tabellen	369
Auswählen einer Methode zur Tabellen- reorganisation	372
Aktivieren der automatischen Reorganisation von Tabellen und Indizes	374
Verwenden von Snapshot Monitor-Daten zum Überwachen der Reorganisation einer partitionier- ten Tabelle	375
Tabellen- und Indexverwaltung für Standard- tabellen	384
Tabellen- und Indexverwaltung für MDC-Tabellen	387
Indexbereinigung und Indexpflege	390
Online-Indexdefragmentierung	392

Kapitel 14. Umverteilung von Daten 395

Umverteilung von Daten	395
Feststellen der Erforderlichkeit einer Datenum- verteilung	397
Umverteilen von Daten auf Datenbankpartitionen	397
Protokollspeicheranforderungen für die Datenum- verteilung	400
Umverteilen von Daten unter Verwendung schritt- weiser Umverteilungsprozeduren.	401
Verwendungsbeispiel.	401
Fehlerbehebung nach einer Umverteilung	403

Kapitel 15. Informationen des Datenbanksystemmonitors 405

Kapitel 16. Messen der System- leistung mit Vergleichstests 409

Durchführen von Vergleichstests	409
Vorbereiten von Vergleichstests	410
Erstellung von Vergleichstests	412
Beispiele für db2batch-Tests	414
Ausführung von Vergleichstests	424
Vergleichstestanalyse - Beispiel	426

Teil 4. Anhänge und Schlussteil 429

Anhang A. DB2-Konfigurationsparameter 431

Einzelheiten zu Parametern nach Funktion	431
Kapazitätsverwaltung	432
Gemeinsam benutzter Datenbankspeicher	432
Gemeinsamer Anwendungsspeicher	448
Privater Agentenspeicher	452
Agenten-/	
Anwendungskommunikationsspeicher	461
Speicher der Datenbankmanagerinstanz	466
Sperrern	471
Ein-/Ausgabe und Speicher	475
Agenten	482
Gespeicherte Prozeduren und benutzerdefinierte	
Funktionen	493
Protokollieren und Recovery	496
Datenbankprotokolldateien	497
Datenbankprotokollierung	507
Recovery	517
DUOW-Recovery	529
Datenbankverwaltung	534
Query Enabler	534
Attribute	534
Status	538
Compilereinstellungen	540
Automatische Verwaltung	547
Kommunikation	550
Konfiguration der Kommunikationsprotokolle	550
DB2 Discovery	552
Umgebung mit partitionierter Datenbank	553
Kommunikation	554
Parallelverarbeitung	559
Instanzverwaltung.	561
Diagnose	561
Parameter für den Datenbanksystemmonitor	564
Systemverwaltung.	566
Instanzverwaltung.	574
DB2-Verwaltungsserver	589
authentication - DAS-Authentifizierungstyp	589
contact_host - Speicherposition der Liste mit	
Ansprechpartnern	590
das_codepage - DAS-Codepage	591
das_territory - DAS-Gebiet	591
dasadm_group - DASADM-Gruppenname	591
db2system - Name des DB2-Serversystems	592
discover - DAS-Discovery-Modus	593
exec_exp_task - Verfallene Tasks ausführen	593
jdk_64_path - Installationspfad für Software	
Developer's Kit (64 Bit) für Java auf DAS	594
jdk_path - Installationspfad für Software	
Developer's Kit für Java auf DAS	595
sched_enable - Schedulermodus	595
sched_userid - Benutzer-ID für Scheduler	596
smtp_server - SMTP-Server	596
toolscat_db - Toolskatalogdatenbank	597
toolscat_inst - Instanz der Toolskatalogdaten-	
bank	597
toolscat_schema - Schema der Toolskatalogda-	
tenbank	598

Anhang B. Registrierdatenbank- und Umgebungsvariablen von DB2 601

Registrierdatenbank- und Umgebungsvariablen	601
von DB2	601
Registrierdatenbank- und Umgebungsvariablen	
nach Kategorie	602
Allgemeine Registrierdatenbankvariablen	602
Systemumgebungsvariablen	607
Kommunikationsvariablen	615
Befehlszeilenvariablen	621
MPP-Konfigurationsvariablen	623
Abfragecompilervariablen	624
Leistungsvariablen	630
Data Links-Variablen	647
Verschiedene Variablen	649

Anhang C. EXPLAIN-Tabellen 663

EXPLAIN-Tabellen	663
Die Tabelle EXPLAIN_ARGUMENT	665
Tabelle EXPLAIN_DIAGNOSTIC	670
Tabelle EXPLAIN_DIAGNOSTIC_DATA	671
Die Tabelle EXPLAIN_INSTANCE	673
Die Tabelle EXPLAIN_OBJECT	676
Die Tabelle EXPLAIN_OPERATOR	679
Die Tabelle EXPLAIN_PREDICATE	681
Die Tabelle EXPLAIN_STATEMENT	684
Die Tabelle EXPLAIN_STREAM	686
Die Tabelle ADVISE_INDEX	688
Die Tabelle ADVISE_INSTANCE	692
Die Tabelle ADVISE_MQT	693
Die Tabelle ADVISE_PARTITION	695
Die Tabelle ADVISE_TABLE	697
Die Tabelle ADVISE_WORKLOAD	698

Anhang D. EXPLAIN-Operatoren 699

Operator CMPEXP	699
Operator DELETE	699
Operator EISCAN	699
Operator FETCH	700
Operator FILTER	700
Operator GENROW	701
Operator GRPBY	701
Operator HSJOIN	701
Operator INSERT	702
Operator IXAND	702
Operator IXSCAN	703
Operator MSJOIN	704
Operator NLJOIN	704
Operator PIPE	705
Operator RETURN	705
Operator RIDSCN	706
Operator RPD	706
Operator SHIP	706
Operator SORT	707
Operator TBSCAN	707
Operator TEMP	708
Operator QUEUE	709
Operator UNION	709
Operator UNIQUE	709
Operator UPDATE	710

Anhang E. SQL- und XQuery-EXPLAIN-Tools	711
SQL- und XQuery-EXPLAIN-Tools	711
db2expln	712
db2expln - SQL- und XQuery-EXPLAIN	712
Hinweise zur Verwendung von db2expln	718
Informationen der EXPLAIN-Ausgabe	719
Beschreibung der Ausgabe von db2expln und dynexpln	719
Tabellenzugriffsinformationen	720
Informationen zu temporären Tabellen	725
Joininformationen	728
Datenstrominformationen	730
Informationen zu INSERT, UPDATE und DELETE	731
Informationen zur Vorbereitung von Block- und Zeilen-IDs	731
Informationen zu Spaltenberechnungen (Aggregation)	732
Informationen zur Parallelverarbeitung	733
Informationen zu Abfragen auf föderierte Datenbanken	736
Verschiedene EXPLAIN-Informationen	737
Beispiele für die Ausgabe von db2expln und dynexpln	739
Beispiele für die Ausgabe von db2expln und dynexpln	739
Beispiel 1: keine Parallelität	739
Beispiel 2: Zugriffsplan für Einzelpartition mit partitionsinterner Parallelität	741
Beispiel 3: Zugriffsplan für mehrere Partitionen mit partitionsübergreifender Parallelität	743
Beispiel 4: Zugriffsplan für mehrere Partitionen mit partitionsübergreifender und partitionsinterner Parallelität	745
Beispiel 5: Zugriffsplan für eine föderierte Datenbank	747
Beispiel 6: Operatoren XANDOR und XISCAN	749
Beispiel 11: Operator XSCAN	751
Beispiel 8: Operator XISCAN	752

Anhang F. db2exfmt - EXPLAIN-Tabellen formatieren	755

Anhang G. Verwenden des Windows-Systemmonitors	757
Windows-Systemmonitor - Einführung	757
Registrieren von DB2 im Windows-Systemmonitor	757
Aktivieren des Fernzugriffs auf DB2-Leistungsinformationen	758
Anzeigen von Leistungswerten der DB2-Datenbank und von DB2 Connect	759
Windows-Leistungsobjekte	759
Zugreifen auf Leistungsdatenbanken	760
Zurücksetzen von DB2-Leistungswerten	761

Anhang H. Technische Informationen zu DB2-Datenbanken	763
Übersicht über technische Informationen zu DB2	763
Feedback zur Dokumentation	763
Bibliothek mit technischen Informationen zu DB2 im PDF-Format	764
Bestellen gedruckter DB2-Bücher	767
Anzeigen der Hilfe für den SQL-Status über den Befehlszeilenprozessor	768
Zugriff auf verschiedene Versionen der DB2-Informationszentrale	768
Anzeigen von Themen in der gewünschten Sprache in der DB2-Informationszentrale	768
Aktualisieren der auf Ihrem Computer oder Intranet-Server installierten DB2-Informationszentrale	769
DB2-Lernprogramme	772
Informationen zur Fehlerbehebung in DB2	772
Bedingungen	773

Anhang I. Bemerkungen	775
Marken	777

Index	779

Kontaktaufnahme mit IBM	797

Teil 1. Einführung zur Leistung

Kapitel 1. Leistungsprinzipien

Elemente der Leistung

Als *Leistung* wird die Art und Weise bezeichnet, wie ein Computersystem unter einer bestimmten Auslastung arbeitet. Die Leistung wird an der Antwortzeit, am Durchsatz und an der Verfügbarkeit gemessen. Außerdem wird die Leistung von folgenden Faktoren beeinflusst:

- Von den im System verfügbaren Ressourcen
- Von der Auslastung und vom Ausmaß der gemeinsamen Nutzung dieser Ressourcen

Im Allgemeinen optimieren Sie Ihr System, um das Kosten-Nutzen-Verhältnis zu verbessern. Die folgenden speziellen Optimierungsziele können dabei verfolgt werden:

- Verarbeiten einer größeren oder anspruchsvolleren Arbeitsbelastung ohne steigende Verarbeitungskosten
Zum Beispiel zur Steigerung der Auslastung ohne neue Hardware erwerben oder mehr Prozessorzeit in Kauf nehmen zu müssen.
- Erreichen schnellerer Systemantwortzeiten bzw. eines höheren Durchsatzes ohne Steigerung der Verarbeitungskosten
- Reduzieren von Verarbeitungskosten ohne negative Auswirkungen für Ihre Benutzer

Die Übersetzung von Leistung vom technischen Aspekt in den wirtschaftlichen Aspekt ist schwierig. Eine Leistungsoptimierung kostet natürlich Geld in Form von Zeitaufwand für Personal und Prozessorzeit, so dass vor einem Optimierungsprojekt die Kosten gegen die möglichen Vorteilsgewinne abgewogen werden müssen. Einige dieser Vorteile sind konkret messbar:

- Effizientere Ressourcennutzung
- Die Möglichkeit, weitere Benutzer dem System hinzuzufügen

Andere Vorteile, wie größere Zufriedenheit seitens der Benutzer aufgrund schnellerer Antwortzeiten, sind weniger fassbar. Jedoch sollten alle diese Vorteile in die Überlegungen mit einbezogen werden.

Zugehörige Konzepte:

- „Einstiegstipps für die Leistungsoptimierung“ auf Seite 7
- „Richtlinien zur Leistungsoptimierung“ auf Seite 3

Zugehörige Tasks:

- „Entwickeln eines Leistungsverbesserungsprozesses“ auf Seite 5

Richtlinien zur Leistungsoptimierung

Die folgenden Richtlinien sind als Hilfe für die Entwicklung eines allgemeinen Ansatzes zur Leistungsoptimierung zu verstehen.

Behalten Sie das Gesetz der abnehmenden Ertragsgewinne im Hinterkopf: Die größten Leistungsvorteile werden in der Regel durch die ersten Maßnahmen erzielt. Weitere Änderungen erbringen im Allgemeinen immer kleinere Vorteile und erfordern immer höheren Aufwand.

Optimieren Sie nicht nur des Optimierens wegen: Optimieren Sie, um erkannten Engpässen abzuweichen. Das Optimieren von Ressourcen, die nicht den Hauptgrund für Leistungsprobleme darstellen, hat wenig oder gar keine Wirkung auf Antwortzeiten, solange Sie nicht die wichtigeren Probleme behoben haben, und kann tatsächlich eine nachfolgende Optimierungsarbeit erschweren. Wenn es ein bedeutendes Verbesserungspotenzial gibt, liegt es in der Verbesserung der Leistung von Ressourcen, die wichtige Faktoren in der Antwortzeit bilden.

Betrachten Sie das gesamte System: Ein Parameter bzw. System lässt sich nicht isoliert optimieren. Bevor Sie Anpassungen vornehmen, überlegen Sie, wie sich diese Anpassungen auf das System als Ganzes auswirken werden.

Ändern Sie jeweils nur einen Parameter gleichzeitig: Ändern Sie nicht mehrere Parameter zur Leistungsoptimierung in einem Schritt. Selbst wenn Sie sich sicher sind, dass alle Änderungen vorteilhaft sind, haben Sie hinterher keine Möglichkeit, den Beitrag jeder Änderung zu bewerten. Darüber hinaus können Sie bei gleichzeitiger Änderung mehrerer Parameter den erzielten Vorteil nicht effektiv den möglicherweise in Kauf genommenen Einbußen gegenüberstellen. Von jeder Anpassung eines Parameters zur Optimierung eines Bereichs ist fast immer auch mindestens ein anderer Bereich betroffen, der vorher vielleicht nicht bedacht wurde. Wenn Sie jeweils nur einen Parameter ändern, haben Sie einen Vergleichspunkt und können feststellen, ob die Änderung die gewünschte Wirkung hat.

Führen Sie Messungen und Neukonfigurierungen nach Ebenen durch: Aus denselben Gründen, aus denen nur jeweils ein Parameter geändert werden soll, empfiehlt sich auch, die einzelnen Ebenen des Systems getrennt zu optimieren. Die folgende Liste von Ebenen innerhalb eines Systems kann Ihnen dabei als Richtlinie dienen:

- Hardware
- Betriebssystem
- Anwendungsserver und -requester
- Datenbankmanager
- SQL- und XQuery-Anweisungen
- Anwendungsprogramme

Prüfen Sie auf Hardware- und Softwareprobleme: Einige Leistungsprobleme können vielleicht durch Wartung der Hardware oder Korrektur der Software oder durch beides behoben werden. Verwenden Sie nicht zu viel Zeit auf die Überwachung und Optimierung des Systems, wenn eine Hardwarewartung oder Softwarekorrektur dies unnötig machen könnte.

Ermitteln Sie die Ursache eines Problems, bevor Sie Ihre Hardware aufrüsten: Auch wenn es so aussieht, als könnten zusätzliche Speicher- und Prozessorkapazitäten die Leistung sofort verbessern, sollten Sie sich die Zeit nehmen, die Engpässe zu lokalisieren und zu verstehen. Sie könnten ansonsten Geld für zusätzlichen Plattenspeicher ausgeben und anschließend feststellen, dass Sie nicht über die Prozessorkapazitäten oder die Kanäle verfügen, um den Speicher vorteilhaft zu nutzen.

Bereiten Sie Zurücksetzungsprozeduren vor, bevor Sie mit der Optimierung beginnen: Wie bereits früher erwähnt, können einige Optimierungsmaßnahmen zu unerwarteten Leistungsergebnissen führen. Wenn sich daraus eine schlechtere Leistung ergibt, sollten die Maßnahmen rückgängig gemacht und alternative Optimierungsmaßnahmen versucht werden. Wenn der vorige Stand in einer Weise gesichert wurde, dass er einfach wiederhergestellt werden kann, ist die Rücknahme der nicht korrekten Informationen wesentlich einfacher.

Zugehörige Konzepte:

- „Elemente der Leistung“ auf Seite 3
- „Einstiegstipps für die Leistungsoptimierung“ auf Seite 7

Zugehörige Tasks:

- „Entwickeln eines Leistungsverbesserungsprozesses“ auf Seite 5

Entwickeln eines Leistungsverbesserungsprozesses

Ein Leistungsverbesserungsprozess ist ein iteratives und langfristig angelegtes Verfahren zur Überwachung und Optimierung von Leistungsbereichen. Abhängig von den Überwachungsergebnissen passen Sie und Ihr Optimierungsteam die Konfiguration des Datenbankservers an und nehmen Änderungen an den Anwendungen vor, die den Datenbankserver verwenden.

Gehen Sie bei der Leistungsüberwachung und den Optimierungsentscheidungen von Ihren Kenntnissen über die Arten von Anwendungen, die mit den Daten arbeiten, sowie von den Datenzugriffsmustern aus. Verschiedene Arten von Anwendungen haben unterschiedliche Leistungsanforderungen.

Betrachten Sie die folgende Verfahrensstruktur für den Leistungsverbesserungsprozess als Richtlinie.

Vorgehensweise:

Gehen Sie zur Entwicklung eines Leistungsverbesserungsprozesses wie folgt vor:

1. Definieren Sie Leistungsziele.
2. Legen Sie Leistungsindikatoren für die wichtigsten Leistungsprobleme im System fest.
3. Entwickeln Sie einen Leistungsüberwachungsplan und führen Sie ihn aus.
4. Analysieren Sie die Ergebnisse der Überwachung fortlaufend, um zu ermitteln, welche Ressourcen optimiert werden müssen.
5. Nehmen Sie jeweils nur eine Anpassung vor.

Selbst wenn Sie davon überzeugt sind, dass mehr als eine Ressource eine Optimierung erfordert, oder wenn mehrere Optimierungsoptionen für die zu optimierende Ressource möglich sind, sollten Sie nur eine Änderung pro Optimierungsschritt vornehmen, so dass Sie sicherstellen können, dass Ihre Optimierungsmaßnahmen die gewünschte Wirkung erzielen. An einem bestimmten Punkt lässt sich keine weitere Verbesserung der Leistung durch Optimieren des Datenbankservers und der Anwendungen mehr realisieren. Wenn dieser Punkt erreicht ist, bleibt Ihnen nur die Möglichkeit, Ihre Hardware aufzurüsten.

Zur tatsächlichen Leistungsoptimierung müssen Kompromisslösungen für verschiedene Systemressourcen gefunden werden. Zum Beispiel könnten Sie zur Verbesse-

rung der E/A-Leistung die Pufferpools vergrößern, jedoch benötigen größere Pufferpools mehr Speicher, was wiederum andere Bereiche der Leistung beeinträchtigen könnte.

Zugehörige Konzepte:

- „Elemente der Leistung“ auf Seite 3
- „Leistungsinformationen, die Benutzer liefern können“ auf Seite 6
- „Einstiegstipps für die Leistungsoptimierung“ auf Seite 7
- „Richtlinien zur Leistungsoptimierung“ auf Seite 3
- „Grenzen der Leistungsoptimierung“ auf Seite 6

Grenzen der Leistungsoptimierung

Eine Optimierung kann die Effizienz eines Systems nur um einen bestimmten Betrag ändern. Überlegen Sie, wie viel Zeit und Geld Sie in die Optimierung der Systemleistung investieren sollten, und wie viel Aufwand an Zeit und Geld den Benutzern des Systems tatsächlich hilft.

Zum Beispiel kann eine Optimierung häufig die Leistung verbessern, wenn das System auf einen Leistungsengpass stößt. Wenn Ihr System nahe an den Leistungsgrenzen arbeitet und sich die Anzahl von Benutzern am System um zehn Prozent erhöht, verlängern sich die Antwortzeiten wahrscheinlich um wesentlich mehr als zehn Prozent. In dieser Situation müssen Sie feststellen, wie Sie dieser Beeinträchtigung der Leistung durch Optimieren des Systems entgegenwirken können.

Es gibt allerdings einen Punkt, ab dem das Optimieren keine weiteren Leistungsgewinne erzielen kann. An diesem Punkt müssen Sie Ihre Ziele und Erwartungen innerhalb der Grenzen Ihrer Umgebung überprüfen. Wenn Sie bedeutsame Leistungsverbesserungen erreichen wollen, müssen Sie vielleicht mehr Plattenpeicher, eine schnellere CPU, zusätzliche CPUs, mehr Arbeitsspeicher, schnellere Kommunikationsverbindungen oder eine Kombination aus diesen Möglichkeiten hinzufügen.

Zugehörige Konzepte:

- „Verwalten der Datenbankserverkapazität“ in *Systemverwaltung: Implementierung*

Zugehörige Tasks:

- „Entwickeln eines Leistungsverbesserungsprozesses“ auf Seite 5

Leistungsinformationen, die Benutzer liefern können

Die ersten Anzeichen dafür, dass Ihr System optimiert werden müsste, könnten Klagen von Benutzern sein. Wenn Sie nicht genügend Zeit zur Definition von Leistungszielen sowie zur Überwachung und Optimierung in umfassender Weise haben, können Sie sich mit der Leistung auseinandersetzen, indem Sie Ihren Benutzern zuhören. In der Regel können Sie bestimmen, wo mit der Untersuchung eines Problems zu beginnen ist, indem Sie einige einfache Fragen stellen. Sie könnten Ihre Benutzer zum Beispiel Folgendes fragen:

- Was meinen Sie mit „langsamer Reaktion“? Heißt dies, um zehn Prozent langsamer, als Sie erwarten, oder um das Zehnfache langsamer?
- Wann haben Sie das Problem bemerkt? Tritt es erst seit kurzem auf oder war es immer da?

- Haben andere Benutzer das gleiche Problem? Handelt es sich bei diesen Benutzern um einen oder zwei Einzelpersonen oder um eine ganze Gruppe?
- Wenn eine Gruppe von Benutzern die gleichen Probleme hat, sind sie mit demselben lokalen Netzwerk (LAN) verbunden?
- Scheinen die Probleme mit einem bestimmten Transaktions- oder Anwendungsprogramm zusammenzuhängen?
- Erkennen Sie ein Muster im Auftreten des Problems? Zum Beispiel: Tritt dieses Problem zu einer bestimmten Tageszeit, z. B. in der Mittagspause, auf oder ist es mehr oder weniger permanent spürbar?

Zugehörige Konzepte:

- „Richtlinien zur Leistungsoptimierung“ auf Seite 3

Zugehörige Tasks:

- „Entwickeln eines Leistungsverbesserungsprozesses“ auf Seite 5

Einstiegstipps für die Leistungsoptimierung

Wenn Sie eine neue Instanz von DB2 starten, ziehen Sie die folgenden Empfehlungen für eine Basiskonfiguration in Betracht:

- Verwenden Sie den Konfigurationsadvisor in der Steuerzentrale, um Empfehlungen für sinnvolle Ausgangsstandardwerte für Ihr System zu erhalten. Die Standardwerte, die für DB2 voreingestellt sind, sollten für Ihre eindeutige Hardwareumgebung optimiert werden.

Stellen Sie Informationen zur Hardware an Ihrem Standort zusammen, so dass Sie auf die Fragen des Assistenten antworten können. Sie können die vorgeschlagenen Einstellungen für Konfigurationsparameter sofort anwenden oder den Assistenten ein Script erstellen lassen, das auf Ihren Antworten beruht, und dieses Script später ausführen.

Dieses Script stellt auch eine Liste der am häufigsten optimierten Parameter zur späteren Referenz bereit.

- Verwenden Sie andere Assistenten in der Steuerzentrale und in „Clientkonfiguration - Unterstützung“ für leistungsrelevante Verwaltungsaufgaben. Solche Aufgaben sind in der Regel diejenigen, bei denen Sie beträchtliche Leistungsverbesserungen mit geringem Zeitaufwand und wenig Mühe erreichen.

Andere Assistenten können Ihnen helfen, die Leistung einzelner Tabellen und des allgemeinen Datenzugriffs zu verbessern. Solche Assistenten sind: Datenbank erstellen, Tabelle erstellen, Index erstellen und Aktualisierung auf mehreren Systemen konfigurieren. Die Diagnosezentrale stellt einen Satz von Überwachungs- und Optimierungstools bereit.

- Verwenden Sie das Designadvisor-Tool in der Steuerzentrale oder über den Befehl `db2adviz`, um zu ermitteln, welche Indizes, MQTs (MQT - Materialized Query Table, gespeicherten Abfragetabellen), MDC-Tabellen und Datenbankpartitionen die Abfrageleistung verbessern würden.
- Verwenden Sie den Befehl `ACTIVATE DATABASE`, um Datenbanken zu starten. In einer partitionierten Datenbank aktiviert dieser Befehl die Datenbank in allen Datenbankpartitionen und vermeidet die Startzeit, die zur Initialisierung der Datenbank erforderlich ist, wenn die erste Anwendung eine Verbindung herstellt.

Anmerkung: Wenn Sie den Befehl `ACTIVATE DATABASE` verwenden, müssen Sie die Datenbank mit dem Befehl `DEACTIVATE DATABASE` her-

unterfahren. Die letzte Anwendung, die ihre Verbindung zur Datenbank trennt, bewirkt nicht, dass die Datenbank heruntergefahren wird.

- Lesen Sie die Übersichtstabellen, die jeden für den Datenbankmanager und für jede Datenbank verfügbaren Konfigurationsparameter auflisten und kurz beschreiben.

Diese Übersichtstabellen enthalten eine Spalte, die angibt, ob eine Optimierung des jeweiligen Parameters einen hohen, mittleren, niedrigen oder keinen Einfluss auf die Leistung in positivem oder negativem Sinn hat. Verwenden Sie diese Tabellen, um die Parameter zu ermitteln, die Sie optimieren können, um die größten Leistungsverbesserungen zu erzielen.

Zugehörige Konzepte:

- Kapitel 15, „Informationen des Datenbanksystemmonitors“, auf Seite 405

Zugehörige Referenzen:

- „Konfigurationsparameter - Übersicht“ auf Seite 305

Leistungsfaktoren für Plattenspeichern

Die Hardware, aus der sich Ihr System zusammensetzt, kann die Leistung Ihres Systems beeinflussen. Als Beispiel für den Einfluss, den die Hardware auf die Leistung hat, werden im Folgenden einige der Auswirkungen betrachtet, die mit dem Plattenspeicher zusammenhängen.

Vier Aspekte der Plattenspeicherverwaltung können sich auf die Leistung auswirken:

- **Speichereinteilung**

Wie Sie eine begrenzte Speichergröße zwischen Indizes und Daten sowie unter Tabellenbereichen aufteilen, bestimmt in hohem Maße, welche Leistung die einzelnen Komponenten in verschiedenen Situationen erreichen.

- **Speicherverschwendung**

Verschwendeter Speicher wirkt sich vielleicht als solcher nicht auf die Leistung des Systems aus, das ihn besitzt, aber er ist eine Ressource, die zur Verbesserung der Leistung an anderer Stelle genutzt werden könnte.

- **Verteilung der Platten-E/A**

Wie ausgewogen Sie den Bedarf an Platten-E/A auf mehrere Plattenspeichereinheiten und Controller verteilen, kann sich auf die Geschwindigkeit auswirken, mit der der Datenbankmanager Informationen von Platten abrufen kann.

- **Mangel an verfügbarem Speicher**

Bei Erreichen der Grenze des verfügbaren Speichers kann die allgemeine Leistung beeinträchtigt werden.

Zugehörige Konzepte:

- „Überlegungen zu DMS-Einheiten“ in *Systemverwaltung: Konzept*
- „Vom Datenbankmanager verwaltete Tabellenbereiche“ in *Systemverwaltung: Konzept*
- „Vom Betriebssystem verwaltete Tabellenbereiche“ in *Systemverwaltung: Konzept*
- „Verzeichnisse und Dateien einer Datenbank“ in *Systemverwaltung: Konzept*
- „Tabellen- und Indexverwaltung für MDC-Tabellen“ auf Seite 387
- „Tabellen- und Indexverwaltung für Standardtabellen“ auf Seite 384

Teil 2. Leistungsprobleme beim Datenbankentwurf

Kapitel 2. Leistungsplanung beim Datenbankentwurf

Funktionen zur Leistungsverbesserung

MQTs (MQT - Materialized Query Table, gespeicherte Abfragetabelle)

MQTs bieten eine leistungsstarke Möglichkeit, die Antwortzeit für komplexe Abfragen zu verbessern, insbesondere für Abfragen, für die einige der folgenden Operationen erforderlich sind:

- Erstellen von Ergebnisdaten für eine oder mehrere Dimensionen
- Joins und Spaltenberechnungen für eine Gruppe von Tabellen
- Daten aus einer häufig genutzten Untergruppe von Daten, d. h. einer sofort für Verarbeitungsoperationen bereiten horizontalen oder vertikalen Datenbankpartition
- Erneutes Partitionieren von Daten aus einer Tabelle bzw. einem Teil einer Tabelle in einer partitionierten Datenbankumgebung

In den SQL- und XQuery-Compiler sind Kenntnisse über MQTs integriert. Im Compiler werden in der Phase des Umschreibens von Abfragen und durch das Optimierungsprogramm Abfragen mit MQTs verglichen, um zu ermitteln, ob an Stelle einer Abfrage, die auf die Basistabellen zugreift, eine MQT verwendet werden kann. Wenn eine MQT verwendet wird, kann die EXPLAIN-Einrichtung Informationen darüber bereitstellen, welche MQT ausgewählt wurde.

Da sich MQTs in vielerlei Hinsicht wie reguläre Tabellen verhalten, treffen die Richtlinien zum Optimieren des Datenzugriffs unter Verwendung von Tabellenbereichsdefinitionen, durch Erstellen von Indizes und Ausführen des Dienstprogramms RUNSTATS auch auf MQTs zu.

Zur Veranschaulichung der Leistungsfähigkeit von MQTs wird im folgenden Beispiel eine mehrdimensionale Analyseabfrage dargestellt und erläutert, wie sie von MQTs profitiert.

Nehmen Sie für dieses Beispiel ein Datenbankschema an, in dem ein Data Warehouse eine Reihe von Kunden und eine Reihe von Kreditkartenkonten enthält. Das Data Warehouse zeichnet die Gruppe der Transaktionen auf, die mit den Kreditkarten durchgeführt wurden. Alle Transaktionen enthalten eine Anzahl von Artikeln, die gemeinsam gekauft wurden. Dieses Schema wird als Mehrfachsternschema (Multi-Star) eingestuft, da zwei große Tabellen, von denen die eine Transaktions-elemente enthält und die andere die Kauftransaktionen identifiziert, zusammen das Zentrum des Sterns bilden.

Die drei hierarchischen Dimensionen, die eine Transaktion beschreiben, sind Produkt, Standort und Zeit. Die Produkthierarchie wird in zwei normalisierten Tabellen gespeichert, die die Produktgruppe und die Produktlinie darstellen. Die Standorthierarchie enthält Informationen zu Ort, Bundesland, sowie Land oder Region, die in einer einzigen denormalisierten Tabelle dargestellt werden. Die Zeithierarchie enthält Informationen zu Tag, Monat und Jahr und ist in einem einzigen Datumsfeld codiert. Die Datumsdimensionen werden aus dem Datumsfeld der

Transaktion unter Verwendung integrierter Funktionen extrahiert. Andere Tabellen in diesem Schema stellen die Kontoinformationen für Kunden und Kundeninformationen dar.

Eine MQT wird mit der Summe und der Anzahl der Verkäufe für jede Stufe der folgenden Hierarchien erstellt:

- Produkt
- Standort
- Zeit bestehend aus Jahr, Monat, Tag

Viele Abfragen können aus diesen zusammengefassten Ergebnisdaten erfüllt werden. Das folgende Beispiel zeigt, wie eine gespeicherte Abfrage erstellt wird, die die Summe und die Anzahl der Verkäufe für die Dimensionen 'Produktgruppe' (Product Group) und 'Produktlinie (Product Line), für die Dimensionen 'Ort' (City), 'Bundesland' (State) und 'Land' (Country) und für die Dimension 'Zeit' (Time) berechnet. Das Beispiel enthält auch einige weitere Spalten in der Klausel GROUP BY.

```
CREATE TABLE dba.PG_SALESSUM
AS (
  SELECT l.id AS prodline, pg.id AS pgroup,
         loc.country, loc.state, loc.city,
         l.name AS linename, pg.name AS pgname,
         YEAR(pdate) AS year, MONTH(pdate) AS month,
         t.status,
         SUM(ti.amount) AS amount,
         COUNT(*) AS count
  FROM   cube.transitem AS ti, cube.trans AS t,
         cube.loc AS loc, cube.pgroup AS pg,
         cube.prodline AS l
  WHERE  ti.transid = t.id
        AND ti.pgid = pg.id
        AND pg.lineid = l.id
        AND t.locid = loc.id
        AND YEAR(pdate) > 1990
  GROUP BY l.id, pg.id, loc.country, loc.state, loc.city,
          year(pdate), month(pdate), t.status, l.name, pg.name
)
DATA INITIALLY DEFERRED REFRESH DEFERRED;

REFRESH TABLE dba.SALESCUBE;
```

Abfragen, die solche vorberechneten Summen nutzen können, sind unter anderem:

- Verkaufsdaten nach Monat und Produktgruppe
- Gesamtvertrieb für Jahre nach 1990
- Verkauf für 1995 oder 1996
- Summe des Verkaufs für eine Produktgruppe oder Produktlinie
- Summe des Verkaufs für eine bestimmte Produktgruppe oder Produktlinie UND für 1995, 1996
- Summe des Verkaufs für ein bestimmtes Land

Obwohl die präzise Antwort für keine dieser Abfragen in der MQT enthalten ist, könnte der Aufwand zur Berechnung der Antwort mit Hilfe der MQT erheblich geringer ausfallen als bei Verwendung der umfangreichen Basistabelle, da ein Teil der für die Antwort benötigten Berechnungen bereits erfolgt ist. MQTs können die Notwendigkeit von aufwendigen Joins, Sortierungen und Spaltenberechnungen von Basisdaten verringern.

Die folgenden Beispielabfragen könnten beträchtliche Leistungsverbesserungen erfahren, da sie die bereits berechneten Ergebnisse der Beispiel-MQT nutzen könnten.

Das erste Beispiel liefert die Gesamtverkäufe für 1995 und 1996:

```
SET CURRENT REFRESH AGE=ANY

SELECT YEAR(pdate) AS year, SUM(ti.amount) AS amount
FROM   cube.transitem AS ti, cube.trans AS t,
       cube.loc AS loc, cube.pgroup AS pg,
       cube.prodline AS l
WHERE  ti.transid = t.id
       AND ti.pgid = pg.id
       AND pg.lineid = l.id
       AND t.locid = loc.id
       AND YEAR(pdate) IN (1995, 1996)
GROUP BY year(pdate);
```

Das zweite Beispiel liefert die Gesamtverkäufe nach Produktgruppe für 1995 und 1996:

```
SET CURRENT REFRESH AGE=ANY

SELECT pg.id AS "PRODUCT GROUP",
       SUM(ti.amount) AS amount
FROM   cube.transitem AS ti, cube.trans AS t,
       cube.loc AS loc, cube.pgroup AS pg,
       cube.prodline AS l
WHERE  ti.transid = t.id
       AND ti.pgid = pg.id
       AND pg.lineid = l.id
       AND t.locid = loc.id
       AND YEAR(pdate) IN (1995, 1996)
GROUP BY pg.id;
```

Je größer die Basistabellen sind, desto höher können die Leistungsverbesserungen in Antwortzeiten ausfallen, weil die MQT langsamer anwächst als die Basistabelle. MQTs können sich überlappende Arbeitsschritte von Abfragen effektiv vermeiden helfen, indem sie die Berechnungen einmal bei ihrer Erstellung und Aktualisierung (REFRESH) durchführen und ihren Inhalt anschließend vielen Abfragen zur Verfügung stellen.

Zugehörige Konzepte:

- „Replizierte MQTs in partitionierten Datenbankumgebungen“ auf Seite 206
- „Designadvisor“ auf Seite 51

Tabellenpartitionierung und MDC-Tabellen

Bei einer Tabelle kann es sich sowohl um eine MDC-Tabelle (Tabelle mit mehrdimensionalem Clustering) als auch um eine partitionierte Tabelle handeln. In einer Tabelle, die beide Merkmale aufweist, können Spalten sowohl in der Bereichspartitionsspezifikation (Range-Partition-Spec) als auch im MDC-Schlüssel für die Tabellenpartitionierung verwendet werden. Diese Möglichkeit ermöglicht eine exaktere Differenzierung des Datenpartitions- und Blockausschlusses, als dies bei Verwendung nur einer dieser Funktionalitäten der Fall wäre. Es gibt auch zahlreiche Anwendungen, in denen es sinnvoll ist, für den MDC-Schlüssel andere Spalten als die für die Tabellenpartitionierung genutzten Spalten anzugeben. Hierbei ist zu beachten, dass die Tabellenpartitionierung mit mehreren Spalten arbeitet, während beim MDC mit mehreren Dimensionen gearbeitet wird.

Merkmale eines normalen DB2 V9.1-Data Warehouses:

Die folgenden Empfehlungen beziehen sich auf typische, normale Data Warehouses, die in DB2 V9.1 neu implementiert wurden. Die folgenden Merkmale werden angenommen:

- Die Datenbank arbeitet auf mehreren Systemen oder in mehreren logischen AIX-Partitionen.
- Die Datenbankpartitionierungsfunktion (DPF) wird verwendet (Tabellen werden mit Hilfe der Klausel DISTRIBUTE BY HASH erstellt).
- Es wurden zwischen vier und 50 Datenpartitionen definiert.
- Die Tabelle, für die die Möglichkeit zur Verwendung des MDC und der Tabellenpartitionierung geprüft wird, ist eine der zentralen Fakttabellen.
- Die Tabelle umfasst zwischen 100 Millionen und 100 Milliarden Zeilen.
- Neue Daten werden in unterschiedlichen Zeitrahmen geladen: Jeweils über Nacht, wöchentlich, monatlich.
- Das tägliche Aufnahmevolumen liegt zwischen 10.000 und 10 Millionen Datensätzen.
- Die Datenvolumen schwanken: Hierbei liegt das Volumen des Spitzenmonats um das Fünffache höher als das des Monats mit dem geringsten Datenaufkommen. Entsprechend beläuft sich auch der Umfang der größten Dimensionen (Produktlinie, Bereich) auf das Fünffache des Umfangs der kleinsten Dimensionen.
- Die Datenbank enthält detaillierte Datenbestände der letzten 1 - 5 Jahre.
- Abgelaufene Daten werden in monatlichen oder vierteljährlichen Abständen mit einer Rollout-Operation ausgelagert.
- Tabellen verwenden eine Vielzahl von Abfragetypen. Der Workload besteht jedoch größtenteils aus analytischen Abfragen, die in Bezug auf OLTP-Workloads die folgenden Merkmale aufweisen:
 - Es gibt umfangreichere Ergebnismengen mit bis zu 2 Millionen Zeilen.
 - Die Mehrzahl oder alle Abfragen beziehen sich auf Sichten und nicht auf Basistabellen.
- SQL-Klauseln zur Auswahl von Daten nach Bereichen (Klausel BETWEEN), Elementen in Listen etc.

Merkmale einer normalen Fakttabelle eines DB2 V9.1-Data Warehouses:

Eine normale Data Warehouse-Fakttabelle kann z. B. das folgende Design verwenden:

- Erstellung von Datenpartitionen über die Spalte 'Month'.
- Definition einer Datenpartition für jeden Rollout-Zeitraum, z. B. 1 Monat, 3 Monate.
- Erstellung von MDC-Dimensionen für 'Day' und für 1 - 4 weitere Dimensionen. Typische Dimensionen sind: Produktlinie und Bereich.
- Alle Datenpartitionen und MDC-Cluster sind über alle Datenpartitionen verteilt.

Das MDC und die Tabellenpartitionierung bieten teilweise dieselben Vorteile. Die folgende Tabelle enthält mögliche Anforderungen innerhalb Ihres Unternehmens und Empfehlungen zu einem Organisationsschema, die auf der Basis der zuvor festgestellten Merkmale gegeben werden.

Tabelle 1. Verwendung der Tabellenpartitionierung mit MDC-Tabellen

Problemstellung	Empfohlenes Schema	Empfehlung
Datenverfügbarkeit während des Rollouts	Tabellenpartitionierung	Sie können die Klausel DETACH PARTITION verwenden, um ein Rollout großer Datenmengen unter minimaler Beeinträchtigung durchzuführen.
Abfrageleistung	Tabellenpartitionierung und MDC	MDC eignet sich am besten für die Abfrage mehrerer Dimensionen. Die Tabellenpartitionierung assistiert durch den Ausschluss von Datenpartitionen.
Minimale Reorganisation	MDC	MDC-Tabellen behalten das Clustering bei, so dass sich die Notwendigkeit von Reorganisationen verringert.
Rollout des Datenbestands eines Monats oder eines längeren Zeitraums während eines traditionellen Offlinezeitfensters	Tabellenpartitionierung	Die Datenpartitionierung kann diese Anforderung voll erfüllen. Durch das MDC würden sich keine zusätzlichen Vorteile ergeben und dieses Verfahren wäre weniger geeignet.
Rollout des Datenbestands eines Monats oder eines längeren Zeitraums während eines Mikro-Offlinezeitfensters (weniger als 1 Minute)	Tabellenpartitionierung	Die Datenpartitionierung kann diese Anforderung voll erfüllen. Durch das MDC würden sich keine zusätzlichen Vorteile ergeben und dieses Verfahren wäre weniger geeignet.
Rollout des Datenbestands eines Monats oder eines längeren Zeitraums bei gleichzeitiger Erhaltung der Tabellenverfügbarkeit für Geschäftsbenutzer, die Abfragen ohne Serviceverluste weiterhin übergeben können	MDC	MDC kann nicht alle in diesem Zusammenhang geltenden Anforderungen voll erfüllen. Die Tabellenpartitionierung eignet sich hier nicht, da die Tabelle nur für kurze Zeit in den Offlinemodus versetzt wird.
Tägliches Laden der Daten (Online oder Offline)	Tabellenpartitionierung und MDC	MDC bietet hier die meisten Vorteile. Die Tabellenpartitionierung bietet Vorteile in Teilbereichen.
Kontinuierliches Laden der Daten (Online)	Tabellenpartitionierung und MDC	MDC bietet hier die meisten Vorteile. Die Tabellenpartitionierung bietet Vorteile in Teilbereichen.
Leistung bei der Ausführung von Abfragen für traditionelle BI-Abfragen	Tabellenpartitionierung und MDC	MDC eignet sich besonders gut für Abfragen in Datenkuben und mehreren Dimensionen. Die Tabellenpartitionierung bietet Unterstützung über den Partitionsausschluss.

Tabelle 1. Verwendung der Tabellenpartitionierung mit MDC-Tabellen (Forts.)

Problemstellung	Empfohlenes Schema	Empfehlung
Minimierung des Reorganisationsaufwandes durch Vermeidung des Reorganisationsbedarfs oder Reduzierung des Aufwands für die Ausführung der Task	MDC	Beim MDC wird das Clustering beibehalten, so dass sich die Notwendigkeit von Reorganisationen verringert. Wenn das MDC verwendet wird, dann bietet die Datenpartitionierung auch in Teilbereichen keine Vorteile. Allerdings ermöglicht die Bereichspartitionierung bei Nichtverwendung des MDC die Reduzierung des Reorganisationsbedarfs, indem auf Partitionsebene ein gewisses Maß an Course-Grain-Clustering beibehalten wird.

Beispiel 1:

Sie arbeiten mit einer Tabelle mit den Schlüsselspalten 'YearAndMonth' und 'Province'. Bei der Planung dieser Tabelle wäre eine Partitionierung nach Datum mit einer Zeitspanne von zwei Monaten pro Datenpartition sinnvoll. Darüber hinaus können Sie die Daten auch nach der Spalte 'Province' organisieren, so dass alle Zeilen für ein bestimmtes Gebiet innerhalb eines Datumsbereichs von zwei Monaten in einer Gruppe zusammengefasst werden. Diese Vorgehensweise ist in Abb. 1 auf Seite 17 dargestellt.

```
CREATE TABLE orders (YearAndMonth INT, Province CHAR(2))
PARTITION BY RANGE (YearAndMonth)
(STARTING 9901 ENDING 9904 EVERY 2)
ORGANIZE BY (Province);
```

		Datenpartition (YearAndMonth)											
		AB			BC			ON			QB		
9901-9902		1	12				9	42	11				
		6					19						
							39						
							41						
		5	14	2	31	18							
		7	32	15	33								
9903-9904		8		17	43								
		3		4		16		20					
		10				22		26					
						30	36						
		13	34	50	24		45	54					
			38		25		51	56					
		44				53							

1 = Block 1

Beispiel 2:

```
CREATE TABLE orders (YearAndMonth INT, Province CHAR(2))
PARTITION BY RANGE (YearAndMonth)
(STARTING 9901 ENDING 9904 EVERY 2)
ORGANIZE BY (YearAndMonth, Province);
```

Tabelle 'Orders'

		MDC-Block (Province)			
		AB	BC	ON	QB
Datenpartition (YearAndMonth)	9901	1 12 6		9 42 19 39 41	11
	9902	5 14 7 32 8	2 31 15 33 17 43	18	
	9903	3 10	4	16 22 30 36	20 26
	9904	13	34 50 38 44	24 25	45 54 51 56 53

Legende

1	= Block 1

Abbildung 2. Eine nach 'YearAndMonth' partitionierte und nach 'Province' und 'YearAndMonth' organisierte Tabelle

Wenn die Partitionierung so festgelegt wurde, dass jeder Bereich nur einen einzigen Wert enthält, ergeben sich keine Vorteile, wenn die Tabellenpartitionierungsspalte in den MDC-Schlüssel aufgenommen wird.

Wichtige Hinweise:

- Im Vergleich mit einer Basistabelle benötigen sowohl MDC-Tabellen als auch partitionierte Tabellen mehr Speicherplatz. Diese Speicherplatzanforderungen gelten zusätzlich zu den sonstigen Anforderungen, sind jedoch unter Berücksichtigung der sich daraus ergebenden Vorteile sinnvoll.
- Wenn Sie die Tabellenpartitionierung und die MDC-Funktionalität in Ihrer partitionierten Datenbankumgebung nicht zusammen einsetzen wollen, dann sollten Sie die Tabellenpartitionierung in den Fällen einsetzen, in denen die Datenverteilung verlässlich vorausgesagt werden kann. Dies ist normalerweise bei den hier erläuterten Systemtypen der Fall. Andernfalls sollten Sie die Verwendung von MDC in Betracht ziehen.

Zugehörige Konzepte:

- „Datenpartitionen“ in *Systemverwaltung: Konzept*
- „Entwerfen von Tabellen mit mehrdimensionalem Clustering (MDC)“ in *Systemverwaltung: Konzept*
- „Erstellung, Platzierung und Verwendung von Tabellen mit mehrdimensionalem Clustering (MDC)“ in *Systemverwaltung: Konzept*
- „Optimierungsstrategien für MDC-Tabellen“ auf Seite 19
- „Optimierungsstrategien für partitionierte Tabellen“ auf Seite 21
- „Verhalten von partitionierten MQTs“ in *Systemverwaltung: Implementierung*
- „Partitionierte Tabellen“ in *Systemverwaltung: Konzept*
- „Tabellen- und Indexverwaltung für MDC-Tabellen“ auf Seite 387
- „Tabellenpartitionierung“ in *Systemverwaltung: Konzept*

Zugehörige Tasks:

- „Ändern partitionierter Tabellen“ in *Systemverwaltung: Implementierung*
- „Methoden zum Definieren von Bereichen für partitionierte Tabellen“ in *Systemverwaltung: Implementierung*
- „Methoden zum Migrieren vorhandener Tabellen und Sichten in partitionierte Tabellen“ in *Systemverwaltung: Implementierung*
- „Erstellen partitionierter Tabellen“ in *Systemverwaltung: Implementierung*

Zugehörige Referenzen:

- „Einschränkungen für den nativen XML-Datenspeicher“ in *XML-Handbuch*

Optimierungsstrategien für MDC-Tabellen

Wenn Sie Tabellen mit mehrdimensionalem Clustering (MDC - Multidimensional Clustering) erstellen, kann sich die Leistung vieler Abfragen verbessern, weil das Optimierungsprogramm zusätzliche Optimierungsstrategien anwenden kann. Diese Strategien basieren in erster Linie auf der verbesserten Effizienz von Blockindizes, jedoch ermöglicht der Vorteil des mehrdimensionalen Clustering auch einen schnelleren Datenabruf.

Anmerkung: Die Optimierungsstrategien für MDC-Tabellen können auch die Leistungsvorteile der partitionsinternen und partitionsübergreifenden Parallelität implementieren.

MDC-Tabellen bieten die folgenden spezifischen Vorteile:

- Dimensionsblockindexsuchen können die erforderlichen Teile der Tabelle ermitteln und schnell nur die angeforderten Blöcke durchsuchen.
- Da Blockindizes kleiner als Satz-ID-Indizes sind, arbeiten Blockindexsuchen schneller.
- AND- und OR-Joins von Indizes (Index ANDing und Index ORing) können auf Blockebene durchgeführt und mit Satz-IDs kombiniert werden.
- Daten werden garantiert in EXTENTSIZE-Speicherbereichen in Clustern gruppiert, was ein schnelleres Abrufen ermöglicht.
- Zeilen können schneller gelöscht werden, wenn eine Auslagerung angewendet werden kann.

Wenn die Bedingungen für eine Auslagerung gegeben sind, wird eine effizientere Lösung zum Löschen von Zeilen aus MDC-Tabellen verwendet. Die Bedingungen sind folgende:

- Die Anweisung DELETE wird gesucht, nicht positioniert (d. h. verwendet nicht die Klausel „WHERE CURRENT OF“).
- Es gibt keine WHERE-Klauseln (alle Zeilen müssen gelöscht werden) oder die einzigen Bedingungen in der Klausel WHERE gibt es für Dimensionen.
- Es gibt keine Tabellen mit DATA CAPTURE CHANGES.
- Die Tabelle ist hinsichtlich referenzieller Integrität nicht das übergeordnete Element, verfügt über keine definierten Trigger zum Löschen und wird in keinerlei MQTs verwendet, die unverzüglich aktualisiert werden.
- Eine mehrstufige Löschoperation kommt für eine Auslagerung in Frage, wenn der Fremdschlüssel eine Untermenge der Dimensionsspalten ist.
- Die Anweisung DELETE darf nicht in einer Anweisung SELECT einer Klausel OLD TABLE vorhanden sein.

Bei einer Auslagerungslöschaktion werden die gelöschten Sätze nicht protokolliert. Stattdessen werden die Seiten mit den Sätzen durch eine Neuformatierung von Teilen der Seiten optisch geleert. Die Änderungen an den neu formatierten Teilen werden protokolliert, die Sätze selbst werden allerdings nicht protokolliert. Es gibt keine Änderung hinsichtlich der Protokollierung von Indexaktualisierungen; die Leistungsverbesserung hängt also davon ab, wie viele Satz-ID-Indizes es gibt. Je weniger Satz-ID-Indizes es gibt, desto qualitativ höher ist die Verbesserung (als Prozentanteil der Gesamtzeit und des gesamten Protokollspeichers).

Mit der nachstehenden Formel können Sie eine ungefähre Angabe über die Menge des im Protokoll eingesparten Speichers machen:

$$S + 38*N - 50*P$$

Dabei ist N die Anzahl der gelöschten Sätze, S die Gesamtgröße der gelöschten Sätze einschließlich Systemaufwand (z. B. Nullanzeiger und VARCHAR-Längen) und P die Anzahl der Seiten in dem Block mit den gelöschten Sätzen:

Diese Zahl entspricht der Menge von in dem permanenten Protokoll eingesparten Speicherbereich; sie muss verdoppelt werden, damit die Einsparungen im reservierten Speicherbereich im aktiven Protokoll einbezogen werden.

Setzen Sie die Registrierdatenbankvariable DB2_MDC_ROLLOUT auf "OFF", wenn Sie das Rolloutverhalten in Löschvorgängen inaktivieren möchten.

Betrachten Sie das folgende einfache Beispiel für eine MDC-Tabelle mit dem Namen **sales**, in der Dimensionen auf den Spalten **region** und **month** definiert sind:

```
SELECT * FROM SALES
WHERE MONTH='March' AND REGION='SE'
```

Für diese Abfrage kann das Optimierungsprogramm eine Dimensionsblockindexsuche durchführen, um die Blöcke zu finden, in denen der Monat März (March) und die Region SE vorkommen. Anschließend kann es nur die resultierenden Blöcke der Tabelle durchsuchen, um die Ergebnismenge abzurufen.

Zugehörige Konzepte:

- „Tabellen- und Indexverwaltung für MDC-Tabellen“ auf Seite 387

Optimierungsstrategien für partitionierte Tabellen

Die Bezeichnung *Ausschluss von Datenpartitionen* bezieht sich auf die Fähigkeit des Datenbankservers, auf der Grundlage der Abfragevergleichselemente festzustellen, dass nur auf eine Untergruppe der Datenpartitionen einer Tabelle zugegriffen werden muss, um eine Abfrage zu erfüllen. Der Ausschluss von Datenpartitionen ist insbesondere vorteilhaft, wenn eine Entscheidungshilfeabfrage auf eine partitionierte Tabelle ausgeführt wird.

Eine partitionierte Tabelle arbeitet mit einem Datenorganisationsschema, bei dem Tabellendaten auf mehrere Speicherobjekte, die als Datenpartitionen oder Datenbereiche (*RANGE*) bezeichnet werden, entsprechend den Werten einer oder mehrerer Spalten der Tabelle, die den Tabellenpartitionierungsschlüssel bilden, verteilt werden. Daten aus einer gegebenen Tabelle werden in mehrere Speicherobjekte auf der Basis der in der Klausel *PARTITION BY* der Anweisung *CREATE TABLE* angegebenen Spezifikationen partitioniert. Diese Speicherobjekte können sich in verschiedenen Tabellenbereichen, in denselben Tabellenbereichen oder in einer Kombination beider Arten von Tabellenbereichen befinden.

Das folgende Beispiel veranschaulicht die Leistungsvorteile des Ausschlusses von Datenpartitionen. Dazu wird die folgende Anweisung zur Erstellung einer partitionierten Tabelle ausgeführt:

```
CREATE TABLE custlist(subsdate DATE, Province CHAR(2), AccountID INT)
PARTITION BY RANGE(subsdate)
(STARTING FROM '1/1/1990' IN ts1,
STARTING FROM '1/1/1991' IN ts1,
STARTING FROM '1/1/1992' IN ts1,
STARTING FROM '1/1/1993' IN ts2,
STARTING FROM '1/1/1994' IN ts2,
STARTING FROM '1/1/1995' IN ts2,
STARTING FROM '1/1/1996' IN ts3,
STARTING FROM '1/1/1997' IN ts3,
STARTING FROM '1/1/1998' IN ts3,
STARTING FROM '1/1/1999' IN ts4,
STARTING FROM '1/1/2000' IN ts4,
STARTING FROM '1/1/2001' ENDING '12/31/2001' IN ts4);
```

Nehmen Sie an, Sie interessieren sich für Kundeninformationen des Jahres 2000. Sie führen die folgende Abfrage aus:

```
SELECT * FROM custlist WHERE subsdate BETWEEN '1/1/2000' AND '12/31/2000';
```

Wie in Abb. 3 auf Seite 22 gezeigt, stellt der Datenbankserver fest, dass nur auf eine Datenpartition in Tabellenbereich 4 ('ts4') zugegriffen werden muss, um diese Abfrage zu erfüllen.

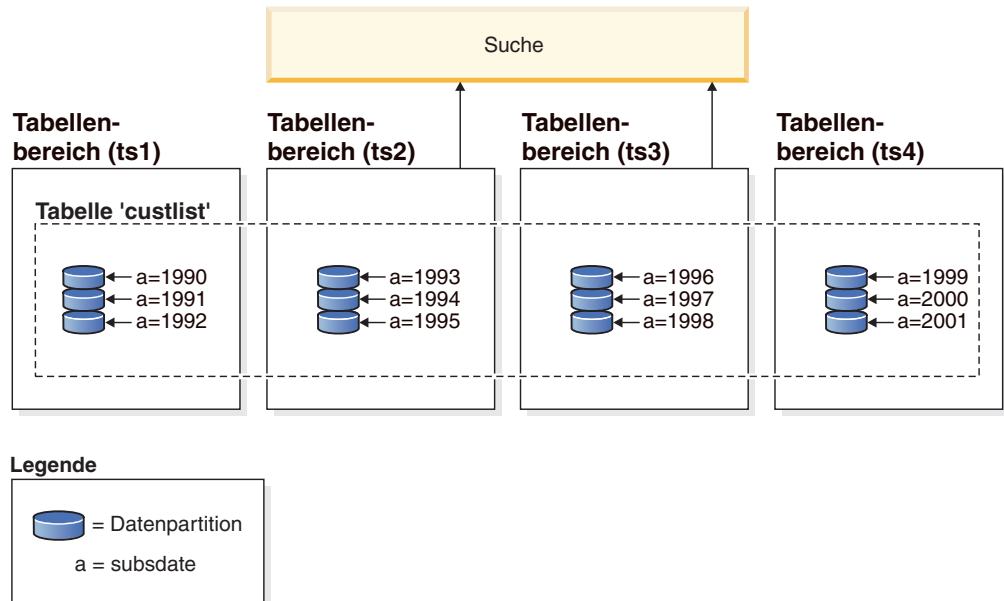


Abbildung 3. Die Leistungsvorteile des Ausschlusses von Datenpartitionen für eine partitionierte Tabelle

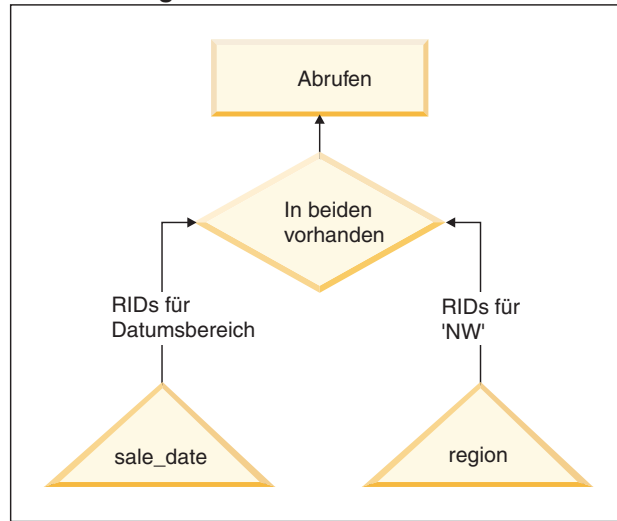
Ein weiteres Beispiel für den Ausschluss von Datenpartitionen (siehe Abb. 4 auf Seite 23) ist eine Indexsuche über zwei Indizes, die auf folgendem Schema basieren:

```
CREATE TABLE multi (sale_date date, region char(2))
PARTITION BY (sale_date)
(STARTING '01/01/2005' ENDING '12/31/2005' EVERY 1 MONTH);
CREATE INDEX sx ON multi(sale_date);
CREATE INDEX rx ON multi(region);
```

Nun wird die folgende Abfrage ausgeführt:

```
SELECT * FROM multi WHERE
sale_date BETWEEN '6/1/2005' AND '7/31/2005' AND REGION = 'NW';
```

Index ANDing



Ausschluss von Datenpartitionen

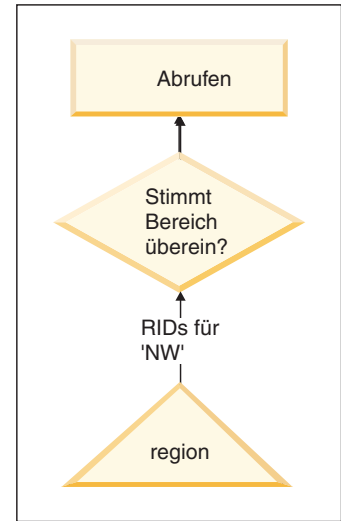


Abbildung 4. Der Entscheidungspfad des Optimierungsprogramms für die Tabellenpartitionierung und das logische Verknüpfen von Indizes über AND (Index ANDing)

Ohne Tabellenpartitionierung besteht ein wahrscheinlicher Plan in der logischen Verknüpfung von der Indizes über AND. Das Index ANDing führt die folgenden Aktionen aus:

- Lesen aller relevanten Indexeinträge aus jedem Index
- Speichern beider Gruppen von Zeilenkennungen (Satz-IDs, RIDs)
- Abgleichen der RIDs, um zu ermitteln, welche in beiden Indizes vorkommen
- Verwenden der RIDs zum Abrufen der Zeilen

Wie in Abb. 4 zu erkennen ist, wird mit der Tabellenpartitionierung der Index gelesen, um Übereinstimmungen für beide Spalten, d. h. 'region' und 'sale_date', zu finden, so dass entsprechende Zeilen schnell abgerufen werden können.

DB2-Explain:

Sie können auch mit Hilfe von DB2 Explain den Ausschluss von Datenpartitionen ermitteln, der vom DB2-Optimierungsprogramm ausgewählt wurde. Die **DP Elim Predicates**-Informationen zeigen, welche Datenpartitionen durchsucht werden, um die folgende Abfrage zu erfüllen:

```
SELECT * FROM custlist WHERE subsdate
BETWEEN '12/31/1999' AND '1/1/2001'
```

Arguments:

```
-----
DPESTFLG: (Number of data partitions accessed are Estimated)
          FALSE
DPLSTPRT: (List of data partitions accessed)
          9-11
DPNUPRT: (Number of data partitions accessed)
          3
```

DP Elim Predicates:

```
-----
Range 1)
  Stop Predicate: (Q1.A <= '01/01/2001')
  Start Predicate: ('12/31/1999' <= Q1.A)
```

Objects Used in Access Plan:

Schema: MRSRINI
Name: CUSTLIST
Type: Data Partitioned Table
Time of creation: 2005-11-30-14.21.33.857039
Last statistics update: 2005-11-30-14.21.34.339392
Number of columns: 3
Number of rows: 100000
Width of rows: 19
Number of buffer pool pages: 1200
Number of data partitions: 12
Distinct row values: No
Tablespace name: <VARIOUS>

Unterstützung mehrerer Spalten:

Der Ausschluss von Datenpartitionen funktioniert auch in Fällen, in denen mehrere Spalten als Tabellenpartitionierungsschlüssel verwendet werden.

Es wird zum Beispiel eine Tabelle mit der folgenden Anweisung erstellt:

```
CREATE TABLE sales(year INT, month INT)
PARTITION BY RANGE(year, month)
(STARTING FROM (2001, 1) ENDING AT(2001,3) IN ts1,
ENDING AT(2001,6) IN ts2,
ENDING AT(2001,9) IN ts3,
ENDING AT(2001,12) IN ts4,
ENDING AT(2002,3) IN ts5,
ENDING AT(2002,6) IN ts6,
ENDING AT(2002,9) IN ts7,
ENDING AT(2002,12) IN ts8)
```

Anschließend wird die folgende Abfrage ausgeführt:

```
SELECT * FROM sales WHERE year = 2001 AND month < 8
```

Das Abfrageoptimierungsprogramm folgert, dass zur Erfüllung dieser Abfrage nur auf die Datenpartitionen in 'ts1', 'ts2' und 'ts3' zugegriffen werden muss.

Anmerkung: Wenn der Tabellenpartitionierungsschlüssel aus mehreren Spalten gebildet wird, ist der Ausschluss von Datenpartitionen nur möglich, wenn Vergleichselemente für die führenden Spalten des zusammengesetzten Schlüssels verwendet werden, da nicht führende Spalten, die im Tabellenpartitionierungsschlüssel verwendet werden, nicht unabhängig sind.

Unterstützung mehrerer Bereiche:

Es ist möglich, einen Ausschluss von Datenpartitionen für mehrere Bereiche zu erzielen (d. h. durch logisches Verknüpfen von Bereichen über OR). An der im vorigen Beispiel erstellten Tabelle wird zum Beispiel die folgende Abfrage ausgeführt:

```
SELECT * FROM sales
WHERE (year = 2001 AND month <= 3) OR (year = 2002 and month >= 10)
```

Der Datenbankserver greift nur auf Daten für das erste Quartal von 2001 und das letzte Quartal von 2002 zu.

Generierte Spalten:

Sie können generierte Spalten als Tabellenpartitionierungsschlüssel verwenden.

Sie können zum Beispiel die folgende Anweisung ausführen:

```
CREATE TABLE sales(a INT, b INT GENERATED ALWAYS AS (a / 5))  
IN ts1,ts2,ts3,ts4,ts5,ts6,ts7,ts8,ts9,ts10  
PARTITION BY RANGE(b)  
(STARTING FROM (0) ENDING AT(1000) EVERY (50))
```

In diesem Fall werden Vergleichselemente für die generierte Spalte zum Ausschluss von Datenpartitionen verwendet. Wenn der Ausdruck zur Generierung der Spalten außerdem monoton ist, übersetzt der Datenbankserver Vergleichselemente für die Quellspalten in Vergleichselemente für die generierten Spalten, so dass der Ausschluss von Datenpartitionen über die generierten Spalten erfolgen kann.

Betrachten Sie zum Beispiel die folgende Abfrage:

```
SELECT * FROM sales WHERE a > 35
```

In diesem Fall generiert der Datenbankserver aus dem Vergleichselement für a ($a > 35$) ein zusätzliches Vergleichselement für b ($b > 7$), um den Ausschluss von Datenpartitionen zu ermöglichen.

Joinvergleichselemente:

Joinvergleichselemente können ebenfalls beim Ausschluss von Datenpartitionen verwendet werden, wenn das Joinvergleichselement auf die Ebene des Tabellenzugriffs verschoben wird (Pushdown). Das Joinvergleichselement wird nur für die innere Tabelle eines Joins mit Verschachtelungsschleife (NLJN, Nested Loop Join) auf die Tabellenzugriffsebene verschoben.

Betrachten Sie zum Beispiel die folgenden Tabellen:

```
CREATE TABLE T1(A INT, B INT)  
PARTITION BY RANGE(A, B)  
(STARTING FROM (1, 1)  
ENDING (1,10) IN ts1, ENDING (1,20) IN ts2,  
ENDING (2,10) IN ts3, ENDING (2,20) IN ts4,  
ENDING (3,10) IN ts5, ENDING (3,20) IN ts6,  
ENDING (4,10) IN ts7, ENDING (4,20) IN ts8)
```

```
CREATE TABLE T2 (A INT, B INT)
```

Folgende Vergleichselemente werden verwendet:

```
P1: T1.A = T2.A  
P2: T1.B > 15
```

In diesem Beispiel lassen sich die genauen Datenpartitionen, auf die zugegriffen wird, wegen unbekannter Werte der äußeren Tabelle des Joins beim Kompilieren nicht bestimmen. In diesem Fall und ebenso in Fällen, in denen Hostvariablen oder Parametermarken verwendet werden, erfolgt der Ausschluss von Datenpartitionen bei der Ausführung, wenn die erforderlichen Werte gebunden werden.

Wenn bei der Ausführung T1 die innere Tabelle eines NLJN-Joins ist, erfolgt der Ausschluss von Datenpartitionen entsprechend den Vergleichselementen für jeden Wert T2.A der äußeren Tabelle dynamisch. Bei der Ausführung werden die Vergleichselemente $T1.A = 3$ und $T1.B > 15$ für den Wert $T2.A = 3$ der äußeren

Tabelle angewendet. Dadurch werden die Datenpartitionen in den Tabellenbereichen 'ts6' und 'ts7' für den Zugriff ermittelt.

Nehmen Sie an, dass die Spalten A in den Tabellen T1 und T2 folgende Werte enthalten:

Äußere Tabelle T2: Spalte A	Innere Tabelle T1: Spalte A	Innere Tabelle T1: Spalte B	Innere Tabelle T1: Position der Datenpartition
2	3	20	ts6
3	2	10	ts3
3	2	18	ts4
	3	15	ts6
	1	40	ts3

Für den Join mit Verschachtelungsschleife (unter Annahme einer Tabellensuche für die innere Tabelle) führt der Datenbankmanager die folgenden Schritte aus:

1. Er liest die erste Zeile aus T2. Der Wert für A ist 2.
2. Er bindet den Wert T2.A (d. h. 2) an die Spalte T2.A im Joinvergleichselement $T1.A = T2.A$. Aus dem Vergleichselement wird $T1.A = 2$.
3. Er wendet den Ausschluss von Datenpartitionen unter Verwendung der Vergleichselemente $T1.A = 2$ und $T1.B > 15$ an. Dies qualifiziert die Datenpartitionen in den Tabellenbereichen 'ts4' und 'ts5'.
4. Er durchsucht die Datenpartitionen in den Tabellenbereichen 'ts4' und 'ts5' von Tabelle T1, bis eine Zeile nach Anwendung von $T1.A = 2$ und $T1.B > 15$ gefunden wird. Die erste qualifizierte Zeile, die gefunden wird, ist die dritte Zeile von T1.
5. Er verknüpft die übereinstimmenden Zeilen.
6. Er durchsucht die Datenpartitionen in den Tabellenbereichen 'ts4' und 'ts5', bis die nächste Übereinstimmung (mit $T1.A = 2$ und $T1.B > 15$) gefunden wird. In diesem Fall werden keine weiteren Zeilen gefunden.
7. Er wiederholt die Schritte 1 bis 6 für die nächste Zeile von T2 (d. h. er nimmt den Wert 3 aus Spalte A). Dieses Verfahren wird fortgesetzt, bis alle Zeilen von T2 durchlaufen sind.

Zugehörige Konzepte:

- „Informationen zur Vorbereitung von Block- und Zeilen-IDs“ auf Seite 731
- „Konfigurationsparameter mit Einfluss auf die Abfrageoptimierung“ auf Seite 186
- „Datenorganisationsschemata in DB2- und Informix-Datenbanken“ in *Systemverwaltung: Konzept*
- „EXPLAIN-Informationen für Datenobjekte“ auf Seite 264
- „EXPLAIN-Tools“ auf Seite 258
- „Richtlinien zur Analyse von EXPLAIN-Informationen“ auf Seite 271
- „Richtlinien zur Erfassung von EXPLAIN-Informationen“ auf Seite 269
- „Richtlinien zur Verwendung von EXPLAIN-Informationen“ auf Seite 260
- „Verschiedene EXPLAIN-Informationen“ auf Seite 737
- „Partitionierte Tabellen“ in *Systemverwaltung: Konzept*
- „EXPLAIN-Einrichtung“ auf Seite 257
- „Tabellenpartitionierung“ in *Systemverwaltung: Konzept*

Zugehörige Referenzen:

- „EXPLAIN-Tabellen“ auf Seite 663
- „CREATE TABLE statement“ in *SQL Reference, Volume 2*

Kapitel 3. Indizes

Verwenden von relationalen Indizes zur Leistungsverbesserung

Indizes können zur Leistungsverbesserung beim Zugreifen auf Tabellendaten verwendet werden. Relationale Indizes werden zum Arbeiten mit relationale Daten verwendet. Beim Zugriff auf XML-Daten werden Indizes für XML-Daten verwendet.

Obwohl das Optimierungsprogramm entscheidet, ob ein relationaler Index für den Zugriff auf Daten relationaler Tabellen verwendet werden soll, müssen Sie, abgesehen von den folgenden Ausnahmen, ermitteln, welche Indizes die Leistung verbessern könnten, und diese Indizes erstellen. Ausnahmen sind die Dimensionsblockindizes und die zusammengesetzten Blockindizes, die für jede Dimension automatisch erstellt werden, die Sie bei der Erstellung einer MDC-Tabelle (MDC - Multi-Dimensional Clustering) angeben.

Sie müssen außerdem in folgenden Situationen das Dienstprogramm RUNSTATS ausführen, um neue Statistikdaten zu den relationalen Indizes zu erfassen:

- Nachdem Sie einen relationalen Index erstellt haben.
- Nachdem Sie den Wert für PREFETCHSIZE geändert haben.

Darüber hinaus sollten Sie das Dienstprogramm RUNSTATS in regelmäßigen Abständen ausführen, um die Statistikdaten auf aktuellem Stand zu halten. Ohne aktuelle Statistiken zu Indizes ist das Optimierungsprogramm nicht in der Lage, den besten Datenzugriffsplan für Abfragen zu ermitteln.

Anmerkung: Um festzustellen, ob ein relationaler Index in einem bestimmten Paket verwendet wird, verwenden Sie die EXPLAIN-Einrichtung. Zum Planen von relationalen Indizes können Sie mit dem Designadvisor in der Steuerzentrale oder mit dem Tool db2advise Hinweise zu relationalen Indizes abrufen, die von einer oder mehreren SQL-Anweisungen verwendet werden können.

Vorteile eines relationalen Index gegenüber keinem Index

Wenn für eine Tabelle kein Index vorhanden ist, muss eine Tabellensuche für jede Tabelle durchgeführt werden, auf die in einer SQL-Abfrage verwiesen wird. Je umfangreicher die Tabelle ist, desto länger dauert die Tabellensuche, weil eine Tabellensuche erfordert, dass auf jede Tabellenzeile sequenziell zugegriffen wird. Obwohl eine Tabellensuche für eine komplexe Abfrage, die die meisten der Zeilen in einer Tabelle abrufen, effizienter sein kann, ist für eine Abfrage, die nur einige Tabellenzeilen zurückgibt, eine Indexsuche für den Zugriff auf Tabellenzeilen effizienter.

Das Optimierungsprogramm wählt eine Indexsuche aus, wenn in der SELECT-Anweisung auf die relationalen Indexspalten verwiesen wird und wenn aufgrund der Schätzungen zu erwarten ist, dass eine Indexsuche schneller als eine Tabellensuche durchgeführt werden kann. Indexdateien sind im Allgemeinen kleiner und erfordern weniger Zeit zum Lesen als eine ganze Tabelle, besonders wenn die Tabellen umfangreicher werden. Darüber hinaus braucht eventuell nicht der gesamte Index

durchsucht zu werden. Die Vergleichselemente, die auf einen Index angewandt werden, verringern die Anzahl der Zeilen, die aus den Datenseiten gelesen werden müssen.

Wenn eine angeforderte Sortierreihenfolge der Ausgabe einer Indexspalte entspricht, können die Zeilen durch eine Suche in diesem Index in der Spaltenreihenfolge gleich in der richtigen Reihenfolge ohne Sortierung abgerufen werden.

Jeder relationale Indexeintrag enthält einen Suchschlüsselwert und einen Zeiger auf die Zeile, die den entsprechenden Wert enthält. Wenn Sie den Parameter `ALLOW REVERSE SCANS` in der Anweisung `CREATE INDEX` angeben, können die Werte sowohl in aufsteigender als auch in absteigender Reihenfolge gesucht werden. Es ist daher möglich, die Suche unter Angabe geeigneter Vergleichselemente zu begrenzen. Ein relationaler Index kann auch dazu verwendet werden, Zeilen in einer geordneten Reihenfolge abzurufen und somit zu vermeiden, dass der Datenbankmanager die Zeilen nach dem Lesen aus der Tabelle in einem weiteren Arbeitsgang sortieren muss.

Neben dem Suchschlüsselwert und dem Zeilenzeiger kann ein relationaler Index auch `INCLUDE`-Spalten enthalten, bei denen es sich um nicht indexierte Spalten in der indexierten Zeile handelt. Solche Spalten machen es dem Optimierungsprogramm möglich, die angeforderten Informationen aus dem Index abzurufen, ohne auf die Tabelle selbst zugreifen zu müssen.

Anmerkung: Das Vorhandensein eines relationalen Index für die Tabelle, die abgefragt wird, garantiert jedoch keine sortierte Ergebnismenge. Nur durch die Angabe der Klausel `ORDER BY` lässt sich die Reihenfolge der Ergebnismenge sicherstellen.

Obwohl Indizes die Zugriffszeiten erheblich verringern können, können sie auch nachteilige Auswirkungen auf die Leistung haben. Vor der Erstellung von Indizes sind daher die Auswirkungen mehrerer Indizes auf den Plattenspeicherplatz und die Verarbeitungszeit zu bedenken:

- Jeder Index erfordert Speicher oder Plattenspeicherplatz. Die exakte Menge ist von der Größe der Tabelle und der Größe und Anzahl von Spalten in dem relationalen Index abhängig.
- Jede für eine Tabelle ausgeführte Operation `INSERT` oder `DELETE` erfordert eine zusätzliche Aktualisierung aller Indizes für diese Tabelle. Dies gilt auch für jede Operation `UPDATE`, mit der der Wert eines Indexschlüssels geändert wird.
- Das Dienstprogramm `LOAD` erstellt alle vorhandenen relationalen Indizes neu bzw. hängt Daten an sie an.

Der Parameter `indexfreespace MODIFIED BY` kann für den Befehl `LOAD` angegeben werden, um den Wert für `PCTREE` des Index außer Kraft zu setzen, der bei der Erstellung des relationalen Index verwendet wurde.

- Jeder relationale Index fügt potenziell einen alternativen Zugriffspfad für eine SQL-Abfrage hinzu, den das Optimierungsprogramm berücksichtigen soll. Dadurch verlängert sich die Kompilierungszeit.

Wählen Sie Indizes sorgfältig, um den Anforderungen des Anwendungsprogramms gerecht zu werden.

Zugehörige Konzepte:

- „Tipps zur Planung von relationalen Indizes“ auf Seite 33
- „Indexreorganisation“ auf Seite 366
- „Tipps zur Leistung von relationalen Indizes“ auf Seite 35

- „Tabellen- und Indexverwaltung für MDC-Tabellen“ auf Seite 387
- „Tabellen- und Indexverwaltung für Standardtabellen“ auf Seite 384
- „Tabellenreorganisation“ auf Seite 365
- „Designadvisor“ auf Seite 51
- „Indexbereinigung und Indexpflege“ auf Seite 390
- „Speicherbedarf für Indizes“ in *Systemverwaltung: Konzept*

Zugehörige Tasks:

- „Erfassen von Katalogstatistiken“ auf Seite 322
- „Erfassen von Indexstatistiken“ auf Seite 325
- „Erstellen eines Index“ in *Systemverwaltung: Implementierung*

Indexstruktur

Der Datenbankmanager verwendet eine B+-Baumstruktur zur Indexspeicherung. Eine B+-Baumstruktur hat eine oder mehrere Stufen, wie die folgende Abbildung zeigt (rid steht für Satz-/Zeilen-ID):

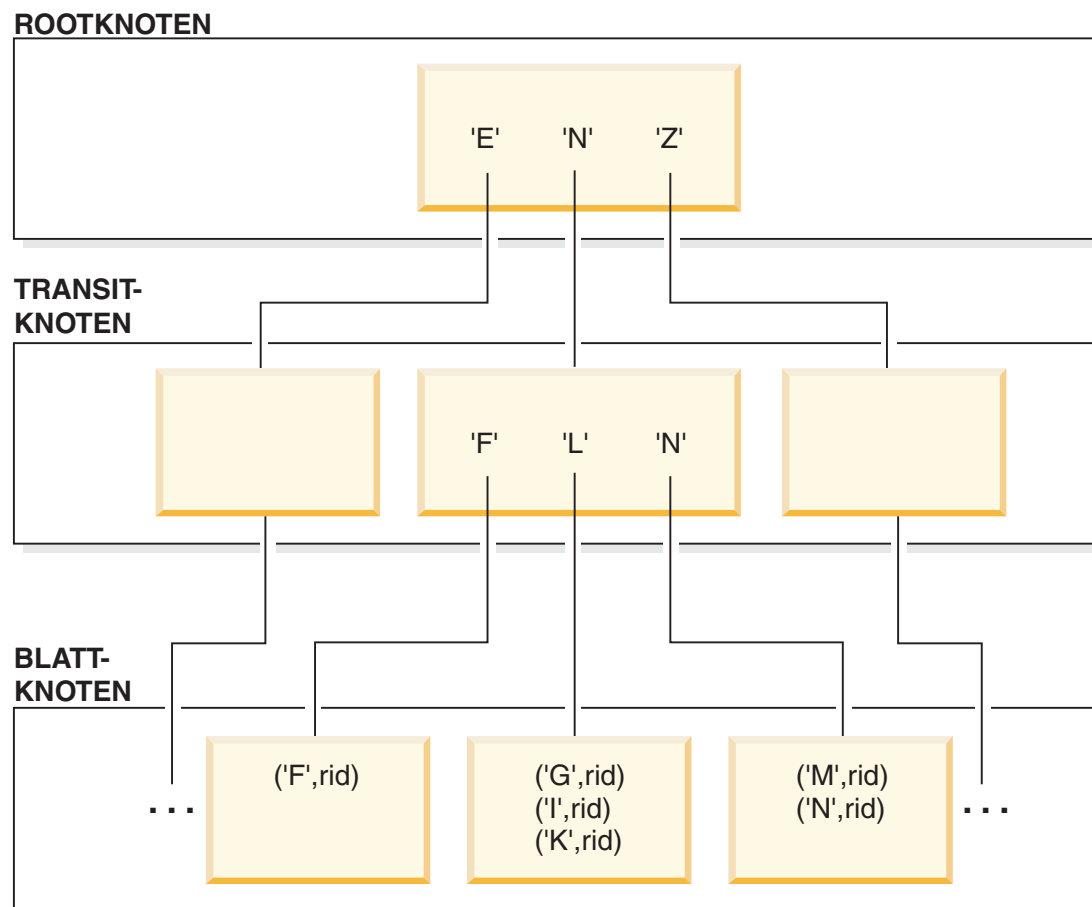


Abbildung 5. B+-Baumstruktur

Die oberste Stufe wird als *Rootknoten* bezeichnet. Die unterste Stufe besteht aus *Blattknoten*, in denen die Werte des Indexschlüssels jeweils mit Zeigern auf die Zeile in der Tabelle gespeichert sind, die den Schlüsselwert enthält. Die Stufen zwischen dem Rootknoten und dem Blattknoten werden als *Transitknoten* bezeichnet.

Bei der Suche nach einem bestimmten Indexschlüsselwert durchsucht der Indexmanager den Indexbaum ausgehend vom Rootknoten. Der Rootknoten enthält einen Schlüssel für jeden Knoten auf der folgenden Stufe. Der Wert jedes dieser Schlüssel ist jeweils der größte vorhandene Schlüsselwert für den entsprechenden Knoten auf der nächsten Stufe. Wenn zum Beispiel ein Index drei Stufen hat, wie in der Abbildung gezeigt, dann durchsucht der Indexmanager zum Auffinden eines Indexschlüsselwerts den Rootknoten nach dem ersten Schlüsselwert, der größer oder gleich dem gesuchten Schlüsselwert ist. Der Rootknotenschlüssel enthält dann einen Zeiger auf einen bestimmten Transitknoten. Der Indexmanager setzt dieses Verfahren durch die Transitknoten fort, bis er den Blattknoten findet, der den benötigten Indexschlüssel enthält.

In der Abbildung wird zum Beispiel nach dem Schlüssel „I“ gesucht. Der erste Schlüssel im Rootknoten, der größer oder gleich „I“ ist, ist „N“. Dieser Wert zeigt auf den mittleren Knoten auf der nächsten Stufe. Der erste Schlüssel in diesem Transitknoten, der größer als oder gleich „I“ ist, ist der Wert „L“. Dieser Wert zeigt auf einen bestimmten Blattknoten, in dem der Indexschlüssel für „I“ zusammen mit der entsprechenden Zeilen-ID (RID) gefunden wird. Die Zeilen-ID (oder Satz-ID) gibt die entsprechende Zeile in der Basistabelle an. Die Blattknotenstufe kann auch Zeiger auf frühere Blattknoten enthalten. Diese Zeiger geben dem Indexmanager die Möglichkeit, die Blattknoten in beide Richtungen zu durchsuchen, um einen Bereich von Werten abzurufen, nachdem er einen Wert des Bereichs gefunden hat. Die Möglichkeit, den Index in beide Richtungen zu durchsuchen, ist nur gegeben, wenn der Index mit der Klausel ALLOW REVERSE SCANS erstellt wurde.

Für MDC-Tabellen (MDC - Mehrdimensionales Clustering) wird automatisch ein Blockindex für jede Clusteringdimension erstellt, die Sie für die Tabelle angeben. Außerdem wird ein zusätzlicher zusammengesetzter Blockindex erstellt, der einen Schlüsselbestandteil für jede Spalte enthält, die an einer Dimension der Tabelle beteiligt ist. Solche Indizes enthalten Zeiger auf Block-IDs (BIDs) anstelle von Satz-IDs (RIDs) und bieten Verbesserungen beim Datenzugriff.

In DB2 Version 8.1 und späteren Versionen können Indizes den Typ 1 oder den Typ 2 aufweisen. Ein *Index des Typs 1* ist die ältere Indexdarstellung. Indizes, die in früheren Versionen von DB2 erstellt wurden, sind vom Typ 1.

Ein Index des Typs 2 ist etwas größer als ein Index des Typs 1 und verfügt über Zusatzfunktionen, die das Sperren des jeweils nächsten Schlüssels minimieren. Mit der ein Byte großen *ridFlag*-Markierung, die mit jeder Satz-ID auf der Blattseite eines Index des Typs 2 gespeichert wird, wird eine Satz-ID ggf. als logisch gelöscht markiert, so dass sie später physisch gelöscht werden kann. Für jede Spalte variabler Länge, die im Index enthalten ist, wird in einem zusätzlichen Byte die tatsächliche Länge des Spaltenwerts gespeichert. Indizes des Typs 2 können auch deshalb größer als Indizes des Typs 1 sein, weil einige Schlüssel als gelöscht markiert, jedoch noch nicht von der Indexseite physisch gelöscht wurden. Nach dem Commit der DELETE- oder UPDATE-Transaktion können die als gelöscht markierten Schlüssel bereinigt werden.

Zugehörige Konzepte:

- „Verwenden von relationalen Indizes zur Leistungsverbesserung“ auf Seite 29
- „Indexreorganisation“ auf Seite 366
- „Online-Indexdefragmentierung“ auf Seite 392

Tipps zur Planung von relationalen Indizes

Die relationalen Indizes, die Sie erstellen, sollten von den relationalen Daten und den Abfragen abhängig sein, die auf sie zugreifen.

Verwenden Sie den Designadvisor in der Steuerzentrale oder das Tool db2advise, um die besten Indizes für eine bestimmte Abfrage oder für eine Reihe von Abfragen, die eine Auslastung definieren, zu finden. Dieses Tool empfiehlt relationale Indizes mit leistungssteigernden Merkmalen wie zum Beispiel INCLUDE-Spalten, übernommenen eindeutigen Indizes und mit ALLOW REVERSE SCANS definierten Indizes.

Die folgenden Richtlinien geben Anhaltspunkte zur Erstellung nützlicher relationaler Indizes für verschiedene Zwecke:

- Definieren Sie zur Vermeidung einiger Sortieroperationen an allen möglichen Stellen Primärschlüssel und eindeutige Schlüssel, indem Sie die Anweisung CREATE UNIQUE INDEX verwenden.
- Fügen Sie eindeutigen Indizes zur Verbesserung des Datenabrufs INCLUDE-Spalten hinzu. Zu diesem Zweck bieten sich Spalten mit folgenden Eigenschaften an:
 - Auf sie wird häufig zugegriffen, so dass die Leistung durch einen reinen Indexzugriff verbessert werden kann.
 - Sie sind zur Begrenzung des Bereichs einer Indexsuche nicht erforderlich.
 - Sie haben keinen Einfluss auf die Reihenfolge oder die Eindeutigkeit des Indexschlüssels.
- Verwenden Sie für einen effizienten Zugriff auf kleine Tabellen relationale Indizes, um häufige Abfragen auf Tabellen mit einer größeren Anzahl von Datenzeilen, wie in der Spalte NPAGES in der Katalogsicht SYSCAT.TABLES eingetragen, zu optimieren. Gehen Sie wie folgt vor:
 - Erstellen Sie einen Index für jede Spalte, die beim Join von Tabellen verwendet werden soll.
 - Erstellen Sie einen Index für jede Spalte, die regelmäßig nach bestimmten Werten durchsucht werden soll.
- Treffen Sie eine Entscheidung zwischen aufsteigender und absteigender Reihenfolge von Schlüsseln, je nachdem, welche Reihenfolge am häufigsten benötigt wird, um die Suche effizienter zu machen. Obwohl die Werte in umgekehrter Richtung gesucht werden können, wenn Sie den Parameter ALLOW REVERSE SCANS in der Anweisung CREATE INDEX angeben, zeigen Suchoperationen in der angegebenen Indexreihenfolge eine etwas bessere Leistung als umgekehrte Suchen.
- Beachten Sie folgende Punkte, um den Aufwand der Indexverwaltung und den Indexspeicherbedarf möglichst gering zu halten:
 - Vermeiden Sie die Erstellung von relationalen Indizes, bei denen es sich um Teilschlüssel der anderen Indexschlüssel für die Spalten handelt. Wenn es zum Beispiel einen Index für die Spalten a, b und c gibt, ist es im Allgemeinen nicht sinnvoll, einen zweiten Index für die Spalten a und b zu erstellen.
 - Erstellen Sie nicht willkürlich für alle Spalten relationale Indizes. Unnötige Indizes belegen nicht nur Speicherplatz, sondern führen auch zu langen Vorbereitungszeiten (PREPARE). Dies spielt besonders für komplexe Abfragen eine wichtige Rolle, wenn eine Optimierungsklasse mit dynamisch programmierter Joinaufzählung (Dynamic Programming Join Enumeration) verwendet wird.

Beachten Sie die folgende allgemeine Regel für die typische Anzahl von relationalen Indizes, die Sie für eine Tabelle definieren. Diese Anzahl richtet sich nach der primären Verwendung der Datenbank:

- Erstellen Sie für Umgebungen zur Onlinetransaktionsverarbeitung (OLTP-Umgebungen) nur einen oder zwei Indizes.
 - Für Umgebungen mit reinen Leseabfragen können Sie mehr als fünf Indizes erstellen.
 - Für gemischte Abfrage- und OLTP-Umgebungen können Sie zwischen zwei und fünf Indizes erstellen.
- Erstellen Sie relationale Indizes für Fremdschlüssel, um die Leistung von DELETE- und UPDATE-Operationen an der übergeordneten Tabelle zu verbessern.
 - Erstellen Sie zur Verbesserung von DELETE- und UPDATE-Operationen mit MQTs (MQT - Materialized Query Table, gespeicherte Abfragetabelle), die mit IMMEDIATE und INCREMENTAL definiert sind, eindeutige relationale Indizes für den implizierten eindeutigen Schlüssel der MQT, das heißt, für die Spalten in der GROUP BY-Klausel der MQT-Definition.
 - Erstellen Sie relationale Indizes für Spalten, die häufig zum Sortieren der relationalen Daten verwendet werden, um schnelle Sortieroperationen zu ermöglichen.
 - Wenn Sie bei der Erstellung eines mehrspaltigen relationalen Index zur Verbesserung der Joinleistung mehrere Auswahlmöglichkeiten für die erste Spalte haben, verwenden Sie die Spalte, die am häufigsten mit dem Vergleichselement equijoin („=“) angegeben wird, oder die Spalte mit der größten Anzahl unterschiedlicher Werte als ersten Schlüssel.
 - Definieren Sie einen Clusterindex, um neu eingefügte Zeilen nach diesem Index in Cluster einzureihen und Seitentrennungen zu vermeiden. Ein Clusterindex sollte die Notwendigkeit, die Tabelle zu reorganisieren, erheblich herabsetzen.

Verwenden Sie bei der Definition einer Tabelle das Schlüsselwort PCTFREE, um anzugeben, wie viel freier Speicherplatz auf der Seite beibehalten werden soll, damit eingefügte Zeilen auf Seiten geeignet gespeichert werden können. Sie können auch die Klausel pagefreespace MODIFIED BY des Befehls LOAD angeben.

- Zur Aktivierung der Online-Indexdefragmentierung verwenden Sie bei der Erstellung von relationalen Indizes die Option MINPCTUSED. Mit der Option MINPCTUSED wird die Schwelle für die Mindestgröße des genutzten Speicherbereichs auf einer äußeren Indexseite (Blattseite) angegeben und die Online-Indexdefragmentierung aktiviert. Dadurch kann sich die Notwendigkeit für eine Reorganisation verringern. Dies geht jedoch auf Kosten einer Leistungseinbuße beim Löschen von Schlüsseln, wenn durch dieses Löschen Schlüssel physisch von der Indexseite entfernt werden.

Ziehen Sie die Erstellung eines relationalen Index unter folgenden Umständen in Betracht:

- Erstellen Sie einen relationalen Index für Spalten, die in WHERE-Klauseln der am häufigsten verarbeiteten Abfragen und Transaktionen verwendet werden.

Betrachten Sie zum Beispiel die folgende Klausel WHERE:

```
WHERE WORKDEPT='A01' OR WORKDEPT='E21'
```

Diese Klausel wird im Allgemeinen von einem Index für die Spalte WORKDEPT profitieren, sofern die Spalte WORKDEPT nicht viele doppelte Werte enthält.

- Erstellen Sie einen relationalen Index für eine Spalte bzw. Spalten, um die Zeilen in der für die Abfrage erforderlichen Reihenfolge zu sortieren. Das Ordnen ist nicht nur für die Klausel ORDER BY erforderlich, sondern auch für andere Klauseln wie DISTINCT und GROUP BY.

Im folgenden Beispiel wird die Klausel DISTINCT verwendet:

```
SELECT DISTINCT WORKDEPT  
FROM EMPLOYEE
```


Der Datenbankmanager kann einen Index verwenden, der in aufsteigender oder absteigender Reihenfolge für die Spalte WORKDEPT definiert ist, um doppelte Werte zu eliminieren. Derselbe Index könnte auch verwendet werden, um Werte wie in folgendem Beispiel mit einer Klausel GROUP BY zu gruppieren:

```
SELECT WORKDEPT, AVERAGE(SALARY)
FROM EMPLOYEE
GROUP BY WORKDEPT
```

- Erstellen Sie einen relationalen Index mit einem zusammengesetzten Schlüssel, der jede Spalte nennt, auf die in einer Anweisung verwiesen wird. Wenn ein Index in dieser Weise angegeben wird, können relationale Daten nur aus dem Index abgerufen werden, was effizienter als der Zugriff auf eine Tabelle ist. Betrachten Sie zum Beispiel die folgende SQL-Anweisung:

```
SELECT LASTNAME
FROM EMPLOYEE
WHERE WORKDEPT IN ('A00','D11','D21')
```

Wenn ein relationaler Index für die Spalten WORKDEPT und LASTNAME der Tabelle EMPLOYEE definiert ist, kann die Anweisung möglicherweise effizienter verarbeitet werden, indem nur der Index und nicht die gesamte Tabelle durchsucht wird. Beachten Sie, dass hier die Spalte WORKDEPT die erste Spalte des relationalen Index sein sollte, da sich das Vergleichselement auf diese Spalte bezieht.

- Erstellen Sie einen relationalen Index mit INCLUDE-Spalten, um die Verwendung von Indizes für Tabellen zu verbessern. Für das vorige Beispiel ließe sich ein eindeutiger relationaler Index folgendermaßen definieren:

```
CREATE UNIQUE INDEX x ON employee (workdept) INCLUDE (lastname)
```

Die Angabe von lastname als INCLUDE-Spalte und nicht als Teil des Indexschlüssels bedeutet, dass lastname nur auf den äußeren Seiten (Blattseiten) des Index gespeichert wird.

Zugehörige Konzepte:

- „Erstellung, Platzierung und Verwendung von Tabellen mit mehrdimensionalem Clustering (MDC)“ in *Systemverwaltung: Konzept*
- „Verwenden von relationalen Indizes zur Leistungsverbesserung“ auf Seite 29
- „Indexreorganisation“ auf Seite 366
- „Online-Indexdefragmentierung“ auf Seite 392
- „Tipps zur Leistung von relationalen Indizes“ auf Seite 35
- „Designadvisor“ auf Seite 51

Tipps zur Leistung von relationalen Indizes

Beachten Sie die folgenden Vorschläge zur Verwendung und Verwaltung relationaler Indizes:

- **Angeben der Parallelität bei Indexerstellung und -reorganisation**

Bei der Erstellung bzw. Reorganisation von relationalen Indizes für umfangreiche Tabellen, die auf einer SMP-Maschine bereitgestellt werden, sollten Sie in Betracht ziehen, den Parameter *intra_parallel* auf 1 (YES) bzw. -1 (SYSTEM) zu setzen, um die Leistungsvorteile der Parallelverarbeitung zu nutzen.

Zum Durchsuchen und Sortieren von relationalen Daten können mehrere Prozessoren verwendet werden.

- **Angeben eines großen Dienstprogrammzwischenspeichers**

Sowohl für die Indexerstellung (CREATE INDEX) als auch die Reorganisation von Indizes (REORG INDEXES) wird Schreibzugriff anderer Benutzer bzw.

Anwendungen auf die zugrunde liegende Tabelle unterstützt. Wenn Sie ein hohes Aktualisierungsaufkommen an der zugrunde liegenden Tabelle für den relationalen Index erwarten, der erstellt oder reorganisiert wird, sollten Sie die Konfiguration eines großen Dienstprogrammzwischenspeichers in Betracht ziehen. Ein großer Dienstprogrammzwischenspeicher beschleunigt die Erstellung bzw. Reorganisation der Indizes während der Übernahmephase. Alle Schreibvorgänge für die erstellten bzw. reorganisierten Indizes werden in den DB2-Protokollen und im internen Speicherpufferbereich aufgezeichnet. Der interne Speicherpufferbereich ist ein ausgewiesener Speicherbereich, der bei Bedarf vom Dienstprogrammzwischenspeicher zugeordnet wird, um die Änderungen an den erstellten bzw. reorganisierten Indizes zu speichern. Die Verwendung dieses Speichers beschleunigt die Übernahmephase. Der zugeordnete Speicher wird nach Beendigung der Indexerstellung bzw. -reorganisation wieder freigegeben. Die Leistung während der Übernahmephase lässt sich stark verbessern, wenn sichergestellt wird, dass ein ausreichender Dienstprogrammzwischenspeicher zur Verfügung steht, um alle oder zumindest die meisten Änderungen an den erstellten bzw. reorganisierten Indizes aufzunehmen.

- **Angeben separater Tabellenbereiche für relationale Indizes**

Indizes können in einem anderen Tabellenbereich als die Tabellendaten gespeichert werden. Dadurch kann Plattenspeicher eventuell effizienter genutzt werden, indem die Bewegungen der Schreib-/Leseköpfe beim Indexzugriff reduziert werden. Sie können auch Indextabellenbereiche auf schnelleren physischen Einheiten erstellen. Darüber hinaus können Sie den Tabellenbereich für Indizes einem anderen Pufferpool zuordnen, wodurch die Indexseiten vielleicht länger im Puffer bleiben, weil sie nicht mit den Tabellendaten konkurrieren.

Wenn Sie Indizes nicht in getrennten Tabellenbereichen speichern, verwenden sowohl die Datenseiten als auch die Indexseiten dieselben Werte für EXTENT-SIZE und PREFETCHSIZE. Wenn Sie für Indizes einen anderen Tabellenbereich verwenden, können Sie unterschiedliche Werte für alle Merkmale eines Tabellenbereichs auswählen. Da Indizes in der Regel kleiner als Tabellen sind und sich über weniger Container erstrecken, haben Indizes in der Regel auch kleinere Werte für EXTENTSIZE (z. B. 8 und 16 Seiten). Das Abfrageoptimierungsprogramm beachtet die Geschwindigkeit der Einheit für einen Tabellenbereich bei der Auswahl eines Zugriffsplans.

- **Sicherstellen des Grads der Clusterbildung**

Wenn für Ihre SQL-Anweisung eine Reihenfolge erforderlich ist (wenn sie zum Beispiel die SQL-Klauseln ORDER BY, GROUP BY und DISTINCT enthält), wählt das Optimierungsprogramm den Index möglicherweise nicht aus, selbst wenn die Reihenfolge in folgenden Fällen erfüllt wird:

- Wenn der Grad der Index-Clusterbildung gering ist. Untersuchen Sie die Spalten CLUSTERRATIO und CLUSTERFACTOR des Katalogs SYSCAT.INDEXES, um Informationen zu erhalten.
- Wenn die Tabelle so klein ist, dass es weniger aufwendig ist, die Tabelle zu durchsuchen und die Ergebnismenge im Hauptspeicher zu sortieren.
- Wenn für den Zugriff auf die Tabelle konkurrierende Indizes gibt

Führen Sie nach der Erstellung eines Clusterindex einen Befehl REORG TABLE im klassischen Modus aus, wodurch ein perfekt organisierter Index erstellt wird. Zur erneuten Herstellung einer Clusterbildung der Tabelle müssen Sie eventuell eine Sortieroperation oder eine LOAD-Operation durchführen. Im Allgemeinen ist die Clusterbildung in einer Tabelle nur anhand eines Index möglich. Erstellen Sie weitere Indizes, nachdem Sie den Clusterindex erstellt haben. Ein Clusterindex versucht, eine bestimmte Reihenfolge der Daten zu erhalten, wodurch die vom Dienstprogramm RUNSTATS gesammelten statistischen Werte für CLUSTERRATIO bzw. CLUSTERFACTOR verbessert werden.

Zur Erhaltung des Clusterverhältnisses können Sie einen geeigneten Wert für PCTFREE beim Ändern einer Tabelle angeben, bevor Sie diese Tabelle mit Daten füllen (LOAD) oder reorganisieren. Der durch PCTFREE angegebene freie Speicherplatz auf jeder Seite reserviert Platz für Einfügungen, so dass diese Einfügungen der Clusterbildung entsprechend vorgenommen werden können. Wenn Sie PCTFREE für die Tabelle nicht angeben, wird der gesamte freie Speicherplatz durch die Reorganisation beseitigt.

Anmerkung: Die Clusterbildung wird zurzeit bei Aktualisierungen (UPDATE) nicht beibehalten, sofern es sich nicht um Bereichsclustertabellen handelt. Das heißt, wenn Sie einen Datensatz aktualisieren, so dass sich der Schlüsselwert im Clusterindex ändert, wird dieser Datensatz nicht unbedingt auf eine entsprechend andere Seite versetzt, um die Clusterreihenfolge beizubehalten. Verwenden Sie zur Beibehaltung der Clusterbildung anstelle der Anweisung UPDATE die Anweisung DELETE und anschließend INSERT.

- **Aktualisieren der Tabellen- und Indexstatistiken**

Führen Sie nach der Erstellung eines neuen relationalen Index das Dienstprogramm RUNSTATS aus, um Indexstatistikdaten zu sammeln. Anhand dieser Statistikdaten kann das Optimierungsprogramm bestimmen, ob durch die Verwendung des Index die Zugriffsleistung verbessert werden kann.

- **Aktivieren der Online-Indexdefragmentierung**

Die Online-Indexdefragmentierung wird aktiviert, wenn die Klausel MINPCTUSED auf einen Wert größer null für den relationalen Index gesetzt wird. Die Online-Indexdefragmentierung ermöglicht eine Komprimierung von Indizes durch Zusammenfügen von Blattseiten, wenn der freie Speicherplatz auf einer Seite den angegebenen Wert erreicht oder unterschreitet, wobei der Index jedoch verfügbar bleibt.

- **Reorganisieren von relationalen Indizes nach Bedarf**

Um optimale Leistungsvorteile aus den Indizes zu ziehen, sollten Sie eine regelmäßige Reorganisation der Indizes in Betracht ziehen, da Aktualisierungen an Tabellen dazu führen, dass der Vorablesezugriff auf Indexseiten an Effizienz einbüßt.

Zur Reorganisation eines Index können Sie ihn entweder löschen und neu erstellen oder das Dienstprogramm REORG verwenden.

Geben Sie zur Verringerung der Notwendigkeit für häufige Reorganisationen bei der Erstellung eines relationalen Index einen geeigneten Wert für PCTFREE an, um einen Prozentsatz an freiem Speicherplatz auf jeder Indexblattseite, wenn sie erstellt wird, freizuhalten. So ist bei zukünftigen Aktivitäten, durch die Datensätze in den Index eingefügt werden, die Wahrscheinlichkeit geringer, dass Indexseitentrennungen verursacht werden. Seitentrennungen bewirken, dass Indexseiten nicht mehr direkt aufeinander folgen oder sequenziell sind, was wiederum zu einer verminderten Effizienz des Vorablesezugriffs auf Indexseiten führt.

Anmerkung: Der bei der Erstellung des relationalen Index angegebene PCTFREE wird beibehalten, wenn der Index reorganisiert wird.

Durch Löschen und erneutes Erstellen bzw. Reorganisieren des relationalen Index wird auch eine neue Gruppe von Seiten erstellt, die weitgehend zusammenhängend und sequenziell sind; außerdem wird der Vorablesezugriff auf Indexseiten verbessert. Obwohl das Dienstprogramm REORG TABLE mehr Aufwand an Zeit und Ressourcen erfordert, stellt es sicher, dass die Datenseiten in

Clustern angeordnet werden. Die Clusterbildung bietet größere Vorteile bei Indexsuchen, durch die auf eine bedeutende Anzahl von Datenseiten zugegriffen wird.

In einer symmetrischen Multiprozessorumgebung (SMP) kann der „klassische“ Modus des Dienstprogramms REORG TABLE, bei dem eine Spiegeltabelle zur schnellen Tabellenreorganisation verwendet wird, mehrere Prozessoren zur erneuten Erstellung von Indizes nutzen, sofern der Konfigurationsparameter *intra_parallel* des Datenbankmanagers den Wert YES oder ANY hat.

- **Analysieren von EXPLAIN-Informationen zur Nutzung des relationalen Index**

Führen Sie in regelmäßigen Abständen EXPLAIN für Ihre am häufigsten verwendeten Abfragen aus, und überprüfen Sie, ob jeder Ihrer relationalen Indizes wenigstens einmal verwendet wird. Wenn ein Index in keiner Abfrage verwendet wird, empfiehlt es sich, diesen Index zu löschen.

EXPLAIN-Informationen geben Ihnen auch Einblick, ob Tabellensuchen in großen Tabellen als innere Tabellen bei Joins mit Verschachtelungsschleife verarbeitet werden. Dies würde darauf hinweisen, dass ein Index für die Spalte des Joinvergleichselements entweder fehlt oder als ineffektiv für die Anwendung des Joinvergleichselements betrachtet wird.

- **Verwenden flüchtiger Tabellen für Tabellen mit stark variierender Größe**

Eine *flüchtige* Tabelle ist eine Tabelle, deren zur Ausführungszeit zwischen leer und sehr groß schwanken kann. Für diese Art von Tabelle, bei der die Kardinalität stark variiert, kann das Optimierungsprogramm einen Zugriffsplan generieren, der eine Tabellensuche einer Index vorzieht.

Durch die Deklaration einer Tabelle als „flüchtig“ in der Anweisung ALTER TABLE...VOLATILE ist es dem Optimierungsprogramm möglich, eine Indexsuche auf der flüchtigen Tabelle auszuführen. In den folgenden Fällen verwendet das Optimierungsprogramm unabhängig von den statistischen Daten eine Indexsuche anstelle einer Tabellensuche:

- Wenn alle Spalten, auf die verwiesen wird, im Index vorhanden sind.
- Wenn der Index bei der Indexsuche ein Vergleichselement anwenden kann.

Wenn es sich bei der Tabelle um eine typisierte Tabelle handelt, wird die Verwendung der Anweisung ALTER TABLE...VOLATILE nur für die Stammtabelle der Hierarchie der typisierten Tabelle unterstützt.

Zugehörige Konzepte:

- „Indexzugriff und Clusterverhältnisse“ auf Seite 199
- „Verwenden von relationalen Indizes zur Leistungsverbesserung“ auf Seite 29
- „Tipps zur Planung von relationalen Indizes“ auf Seite 33
- „Indexreorganisation“ auf Seite 366
- „Indexstruktur“ auf Seite 31
- „Online-Indexdefragmentierung“ auf Seite 392
- „Tabellen- und Indexverwaltung für MDC-Tabellen“ auf Seite 387
- „Tabellen- und Indexverwaltung für Standardtabellen“ auf Seite 384
- „Tabellenreorganisation“ auf Seite 365
- „Indexbereinigung und Indexpflege“ auf Seite 390

Zugehörige Referenzen:

- „intra_parallel - Partitionsinterne Parallelität aktivieren“ auf Seite 559

Indexieren von XML-Daten - Übersicht

Ein Index für XML-Daten kann die Effizienz von Abfragen auf XML-Dokumente verbessern, die in einer XML-Spalte gespeichert sind.

Im Unterschied zu traditionellen relationalen Indizes, bei denen Indexschlüssel aus einer oder mehreren von Ihnen angegebenen Tabellenspalten zusammengesetzt sind, verwendet ein Index für XML-Daten einen bestimmten XML-Musterausdruck zum Indexieren von Pfaden und Werten in XML-Dokumenten, die in nur einer XML-Spalte gespeichert sind. Der Datentyp dieser Spalte muss XML sein.

An Stelle des Zugriffs auf den Anfang eines Dokuments ermöglichen die Indexeinträge in einem Index für XML-Daten den Zugriff auf Knoten innerhalb des Dokuments, indem Indexschlüssel auf der Basis von XML-Musterausdrücken erstellt werden. Da ein XML-Muster von verschiedenen Teilen eines XML-Dokuments erfüllt werden kann, können für ein einzelnes Dokument mehrere Indexschlüssel in den Index eingefügt werden.

Ein Index für XML-Daten wird mit Hilfe der Anweisung `CREATE INDEX` erstellt und mit Hilfe der Anweisung `DROP INDEX` gelöscht. Die Klausel `GENERATE KEY USING XMLPATTERN`, die Sie in die Anweisung `CREATE INDEX` einfügen, gibt die zu indexierenden Daten an.

Einige der Schlüsselwörter, die in der Anweisung `CREATE INDEX` für Indizes von Nicht-XML-Spalten verwendet werden, sind auf Indizes für XML-Daten nicht anwendbar. Auch das Schlüsselwort `UNIQUE` hat bei Indizes für XML-Daten eine andere Bedeutung.

Beispiel: Erstellen eines Index für XML-Daten: Nehmen Sie an, dass die Tabelle 'companyinfo' eine XML-Spalte mit dem Namen 'companydocs' enthält, in der XML-Dokumentfragmente wie die folgenden gespeichert werden:

Dokument für 'Company1':

```
<company name="Company1">
  <emp id="31201" salary="60000" gender="Female">
    <name>
      <first>Laura</first>
      <last>Brown</last>
    </name>
    <dept id="M25">
      Finance
    </dept>
  </emp>
</company>
```

Dokument für 'Company2':

```
<company name="Company2">
  <emp id="31664" salary="60000" gender="Male">
    <name>
      <first>Chris</first>
      <last>Murphy</last>
    </name>
    <dept id="M55">
      Marketing
    </dept>
  </emp>
  <emp id="42366" salary="50000" gender="Female">
    <name>
      <first>Nicole</first>
```

```

        <last>Murphy</last>
    </name>
    <dept id="K55">
        Sales
    </dept>
</emp>
</company>

```

Benutzer der Tabelle 'companyinfo' rufen häufig Mitarbeiterinformationen mit Hilfe der Mitarbeiter-ID ('emp id') ab. In diesem Fall können Sie einen Index wie den folgenden verwenden, um die Effizienz solcher Abrufe zu erhöhen:

```

CREATE INDEX empindex on companyinfo(companydocs)
GENERATE KEY USING XMLPATTERN '/company/emp/@id'
AS SQL DOUBLE

```

1
2
3

Abbildung 6. Beispiel eines Index für XML-Daten

Anmerkungen zu Abb. 6:

- 1** Der Index für XML-Daten wird für die Spalte 'companydocs' der Tabelle 'companyinfo' definiert. Die Spalte 'companydocs' muss den Datentyp XML besitzen.
- 2** Die Klausel GENERATE KEY USING XMLPATTERN gibt Informationen zu den zu indexierenden Daten an. Diese Klausel wird als XML-Indexspezifikation bezeichnet. Die XML-Indexspezifikation enthält eine XML-Musterklausel. Die XML-Musterklausel in diesem Beispiel gibt an, dass die Werte des Attributs 'id' für jedes Mitarbeiterelement ('emp') zu indexieren sind.
- 3** AS SQL DOUBLE gibt an, dass die indexierten Werte als DOUBLE-Werte gespeichert werden.

Zugehörige Referenzen:

- „CREATE INDEX statement“ in *SQL Reference, Volume 2*

Indizes in partitionierten Tabellen

Indexverhalten bei partitionierten Tabellen

Indizes für partitionierte Tabellen sind Indizes für normale Tabellen insofern ähnlich, als dass jeder Index Zeiger auf Zeilen in allen Datenpartitionen der Tabelle enthält. Ein wichtiger Unterschied besteht jedoch darin, dass jeder Index für eine partitionierte Tabelle ein unabhängiges Objekt ist. In Umgebungen mit partitionierten Datenbanken wird der Index über die Datenbankpartitionen in der gleichen Weise verteilt wie die Tabelle. Da ein Index für eine partitionierte Tabelle unabhängig von anderen Indizes einsetzbar ist, sind einige besondere Aspekte im Hinblick darauf zu beachten, welcher Tabellenbereich beim Erstellen eines Indexes für eine partitionierte Tabelle verwendet wird.

Ein Index für eine partitionierte Tabelle wird in einem einzigen Tabellenbereich erstellt, auch wenn sich die Datenpartitionen der Tabelle über mehrere Tabellenbereiche erstrecken. Sowohl DMS- als auch SMS-Tabellenbereiche unterstützen die Verwendung von Indizes, die sich an einer anderen Position befinden als die Tabelle. Alle angegebenen Tabellenbereiche müssen sich in derselben Datenbankpartitionsgruppe befinden. Jeder Index kann in einem eigenen Tabellenbereich, auch in großen Tabellenbereichen, abgelegt werden. Jeder Tabellenbereich für einen Index muss denselben Speichermechanismus wie die Datenpartitionen verwenden (entweder DMS oder SMS). Indizes in großen Tabellenbereichen können bis zu 2^{29} Seiten enthalten.

Ein Index für eine partitionierte Tabelle bietet folgende zusätzliche Vorteile:

- Beim Löschen und bei der Onlineerstellung von Indizes wird eine bessere Leistung erzielt.
- Es besteht die Möglichkeit, verschiedene Werte für die Tabellenbereichsmerkmale für verschiedene Indizes der Tabelle zu verwenden (z. B. können verschiedene Seitengrößen für die einzelnen Indizes zu einer besseren Speicherplatznutzung beitragen).
- Weniger E/A-Konkurrenzsituationen lassen einen effizienteren gemeinsamen Zugriff auf die Indexdaten für die Tabelle zu.
- Wenn einzelne Indizes gelöscht werden, wird der Speicherplatz sofort für das System verfügbar, ohne dass eine Indexreorganisation erforderlich ist.
- Wenn eine Indexreorganisation ausgeführt werden soll, kann ein einzelner Index reorganisiert werden.

Abb. 7 auf Seite 42 zeigt einen nicht partitionierten Index für eine partitionierte Tabelle, die sich in einem einzigen Tabellenbereich befindet.

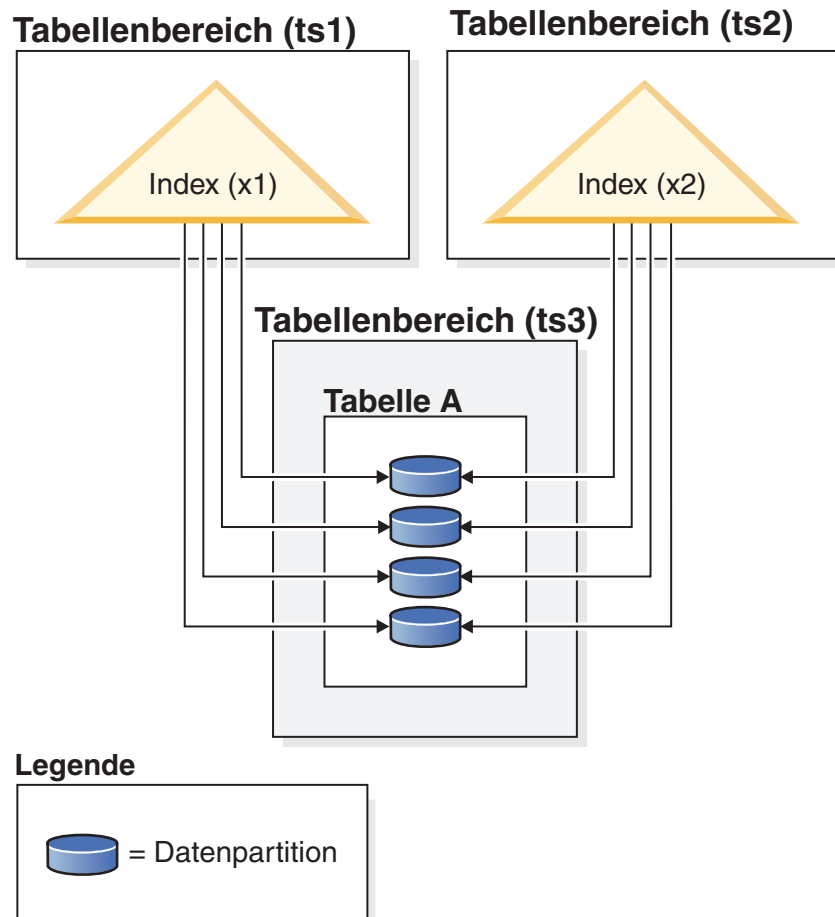
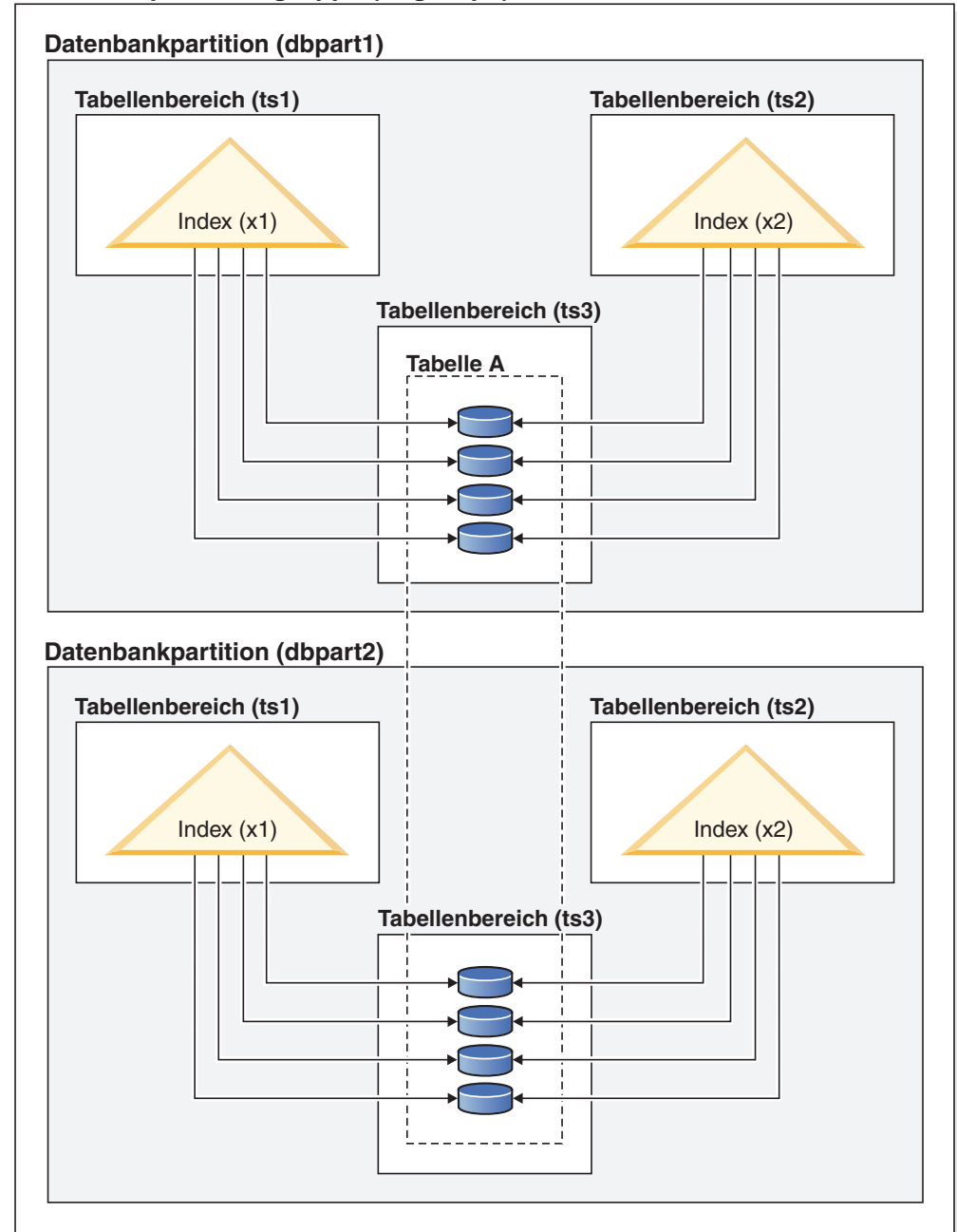


Abbildung 7. Indexverhalten bei einer partitionierten Tabelle

Abb. 8 auf Seite 43 zeigt das Indexverhalten bei einer partitionierten Tabelle, die gleichzeitig über mehrere Datenbankpartitionen verteilt ist.

Datenbankpartitionsgruppe (dbgroup1)



Legende



Abbildung 8. Indexverhalten bei einer Tabelle, die sowohl verteilt als auch partitioniert ist.

Sie können einen Tabellenbereich für Indizes für eine partitionierte Tabelle in der Anweisung `CREATE INDEX ...IN <tb_bereich1>` angeben, der ein anderer ist, als der Tabellenbereich für Indizes, der in der Anweisung `CREATE TABLE .. INDEX IN <tb_bereich2>` angegeben wurde.

Nur für partitionierte Tabellen könne Sie die Indexspeicherposition durch die IN-Klausel der Anweisung CREATE INDEX überschreiben, so dass Sie eine Tabellenbereichsposition für den Index angeben können. Durch diese Methode können Sie verschiedene Indizes für eine partitionierte Tabelle je nach Bedarf in einem anderen Tabellenbereich speichern. Wenn Sie eine partitionierte Tabelle erstellen, ohne die Speicherposition für die zugehörigen nicht partitionierten Indizes anzugeben, und einen Index mit der Anweisung CREATE INDEX erstellen, ohne einen bestimmten Tabellenbereich anzugeben, wird der Index im Tabellenbereich der ersten zugeordneten bzw. sichtbaren Datenpartition erstellt. Zur Bestimmung, wo der Index zu erstellen ist, werden die drei folgenden möglichen Fälle in der angegebenen Reihenfolge (beginnend mit Fall 1) ausgewertet. Die Auswertung wird beendet, wenn einer der Fälle zutrifft:

Fall 1:

Wenn ein Tabellenbereich für den Index in der Anweisung CREATE INDEX ... IN <tb_bereich1> angegeben wird, wird der in <tb_bereich1> angegebene Tabellenbereich für diesen Index verwendet.

Fall 2:

Wenn ein Tabellenbereich für den Index in der Anweisung CREATE TABLE .. INDEX IN <tb_bereich2> angegeben wird, wird der in <tb_bereich2> angegebene Tabellenbereich für diesen Index verwendet.

Fall 3:

Wenn kein Tabellenbereich definiert ist, wird der Tabellenbereich verwendet, der von der ersten zugeordneten oder sichtbaren Datenpartition verwendet wird.

Wo der Index erstellt wird, hängt davon ab, was bei der Ausführung der Anweisung CREATE TABLE geschieht. Wenn Sie für nicht partitionierte Tabellen keine Klausel INDEX IN angeben, fügt die Datenbank diese Information für Sie ein und gibt denselben Tabellenbereich wie für die Datentabelle an. Wenn Sie für partitionierte Tabelle diese Information weglassen, wird sie nicht nachgetragen und es gilt Fall 3.

Beispiel 1: Dieses Beispiel geht von der Annahme aus, dass eine partitionierte Tabelle 'sales' mit den Spalten (a int, b int, c int) vorhanden ist. Es wird ein eindeutiger Index 'a_idx' im Tabellenbereich 'ts1' erstellt.

```
CREATE UNIQUE INDEX a_idx ON sales ( a ) IN ts1
```

Beispiel 2: Dieses Beispiel geht von der Annahme aus, dass eine partitionierte Tabelle 'sales' mit den Spalten (a int, b int, c int) vorhanden ist. Es wird ein Index 'b_idx' im Tabellenbereich 'ts2' erstellt.

```
CREATE INDEX b_idx ON sales ( b ) IN ts2
```

Zugehörige Konzepte:

- „Verwenden von relationalen Indizes zur Leistungsverbesserung“ auf Seite 29
- „Clusterindexverhalten bei partitionierten Tabellen“ auf Seite 45
- „Indexes“ in *SQL Reference, Volume 1*
- „Large object behavior in partitioned tables“ in *SQL Reference, Volume 1*
- „Optionen der Anweisung CREATE INDEX“ in *Systemverwaltung: Implementierung*
- „Partitionierte Tabellen“ in *Systemverwaltung: Konzept*
- „Speicherbedarf für Indizes“ in *Systemverwaltung: Konzept*

Zugehörige Tasks:

- „Ändern partitionierter Tabellen“ in *Systemverwaltung: Implementierung*

- „Erstellen partitionierter Tabellen“ in *Systemverwaltung: Implementierung*
- „Aktivieren der Parallelität beim Erstellen von Indizes“ in *Systemverwaltung: Implementierung*

Zugehörige Referenzen:

- „CREATE INDEX statement“ in *SQL Reference, Volume 2*
- „REORG INDEXES/TABLE command“ in *Command Reference*
- „Command Line Processor (CLP) samples“ in den *Themen mit Beispielen*

Clusterindexverhalten bei partitionierten Tabellen

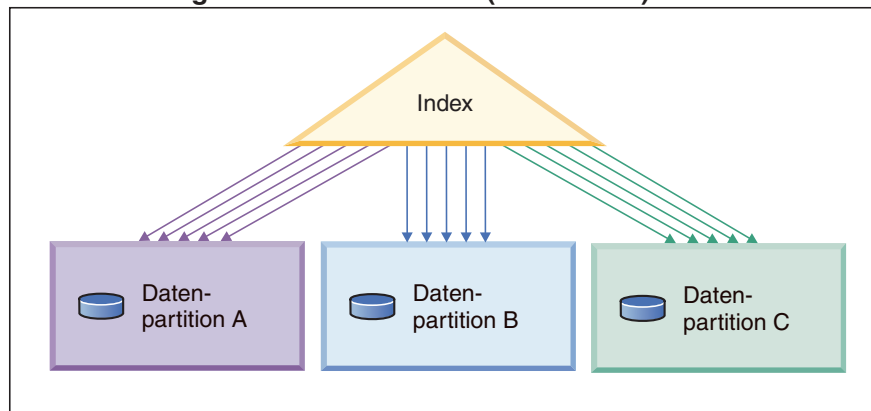
Clusterindizes bieten für partitionierte Tabellen die gleichen Vorteile wie für reguläre Tabellen. Allerdings ist bei der Auswahl eines Clusterindex im Hinblick auf die Definitionen von Tabellenpartitionierungsschlüsseln Sorgfalt geboten.

Sie können Clusterindizes für eine partitionierte Tabelle mit einem beliebigen Clusterschlüssel erstellen. Der Datenbankserver versucht den Clusterindex zu verwenden, um die Daten lokal in jeder Datenpartition in Clustern zusammenzufassen. Bei einer Einfügung über einen Clusterindex wird der Index durchsucht, um eine passende Zeilenkennung (Satz-ID, RID) zu ermitteln. Diese RID wird als Ausgangspunkt für die Suche nach einem Platz in der Tabelle zum Einfügen des Datensatzes verwendet. Zur Erzielung einer gut gepflegten Clusterbildung mit guter Leistung sollte es eine Korrelation zwischen den Indexspalten und den Spalten des Tabellenpartitionierungsschlüssels geben. Eine Methode, eine solche Korrelation sicherzustellen, besteht darin, den Indexspalten die Spalten des Tabellenpartitionierungsschlüssels als Präfix voranzustellen, wie im folgenden Beispiel gezeigt:

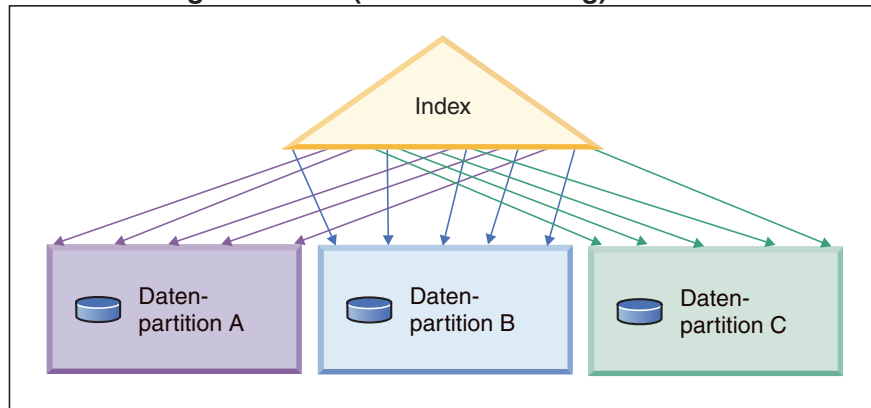
```
PARTITION BY RANGE (Month, Region)
CREATE INDEX ... (Month, Region, Department) CLUSTER
```

Obwohl der Datenbankserver diese Korrelation nicht zwingend umsetzt, ist zu erwarten, dass alle Schlüssel im Index nach Partitions-IDs zusammen gruppiert werden, um eine gute Clusterbildung zu erreichen. Wenn zum Beispiel eine Tabelle nach Quartal partitioniert ist und ein Clusterindex für Datum definiert wird, weil zwischen Quartal und Datum eine Beziehung besteht, kann eine optimale Clusterbildung der Daten mit guter Leistung erzielt werden, weil alle Schlüssel einer Datenpartition im Index zusammen gruppiert werden.

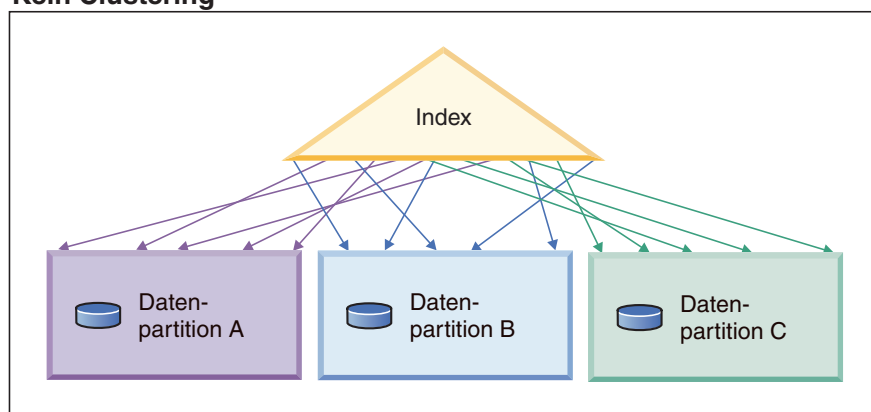
Clustering mit dem Partitionierungsschlüssel als Präfix (Korrelation)



Clustering entspricht nicht dem Partitionierungsschlüssel (lokales Clustering)



Kein Clustering



Legende

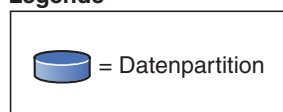


Abbildung 9. Die möglichen Effekte eines Clusterindex für eine partitionierte Tabelle. In der ersten Abbildung werden die Daten sowohl global als auch lokal zu Clustern zusammengefasst.

Wie in Abb. 9 auf Seite 46 gezeigt, wird bei dem gegebenen Layout des Indexes und der Daten in den einzelnen Beispielen eine optimale Suchleistung erzielt, wenn die Clusterbildung mit dem Tabellenpartitionierungsschlüssel korreliert ist. Wenn die Clusterbildung nicht mit dem Tabellenpartitionierungsschlüssel korreliert ist, ist es unwahrscheinlich, dass die Werte im Index lokal zu Clustern zusammengefasst werden. Da eine Korrelation zwischen den Tabellenpartitionierungsspalten und den Indexspalten erwartet wird, ist ein Szenario mit perfekter lokaler Clusterbildung sehr unwahrscheinlich.

Das Clustering bietet folgende Vorteile:

- Innerhalb jeder Datenpartition sind die Zeilen in der Reihenfolge des Schlüssels angeordnet.
- Clusterindizes verbessern die Leistung von Suchoperationen, welche die Tabelle in der Reihenfolge der Schlüssel durchqueren. Dies liegt daran, dass die Suche die erste Zeile der ersten Seite abrufen, dann jede Zeile auf eben dieser Seite, bis alle Zeilen für diese Seite abgerufen sind. Anschließend fährt die Suche mit der nächsten Seite fort. Dies bedeutet, dass sich nur eine Seite der Tabelle zu einem gegebenen Zeitpunkt im Pufferpool befinden muss. Wenn die Tabelle hingegen nicht zu Clustern zusammengefasst ist, wird jede Zeile mit hoher Wahrscheinlichkeit von einer anderen Seite abgerufen. Sofern im Pufferpool nicht genügend Platz zur Aufnahme der gesamten Tabelle ist, führt dies dazu, dass jede Seite mehr als einmal abgerufen wird und sich die Suche erheblich verlangsamt.

Für partitionierte Tabellen kann der Idealfall, dass jede Seite bei der Suche jeweils nur einmal abgerufen wird, nur dann garantiert werden, wenn der Tabellenpartitionierungsschlüssel ein Präfix des Clusterschlüssels (siehe erstes Beispiel in Abb. 9 auf Seite 46 ist. Wenn der Clusterschlüssel jedoch nicht wie zuvor beschrieben mit dem Tabellenpartitionierungsschlüssel korreliert ist und die Daten lokal zu Clustern zusammengefasst sind, können Sie immer noch den vollen Vorteil des Clusterindex ausschöpfen, wenn im Pufferpool genügend Platz ist, um eine Seite jeder Datenpartition aufzunehmen. Der Grund hierfür ist, dass jede Zeile, die für eine gegebene Datenpartition abgerufen wird, in der Nähe einer vorigen Zeile liegt, die für dieselbe Datenpartition abgerufen wurde (siehe zweites Beispiel in Abb. 9 auf Seite 46). Wie zuvor erwähnt, wird die Clusterbildung möglicherweise nicht gut aufrechterhalten, wenn der Clusterschlüssel nicht mit dem Tabellenpartitionierungsschlüssel korreliert ist. Wenn Sie jedoch kein hohes Volumen an Einfüge-, Aktualisierungs- und Löschoperationen für Ihre Tabelle erwarten, sollte diese Lösung von Vorteil sein.

Selbst wenn nicht genügend Platz für eine Seite jeder Datenpartition im Pufferpool vorhanden ist, lassen sich dennoch einige Vorteile aus der Definition eines Clusterindex gewinnen.

Zugehörige Konzepte:

- „Verwenden von relationalen Indizes zur Leistungsverbesserung“ auf Seite 29
- „Optimierungsstrategien für partitionierte Tabellen“ auf Seite 21
- „Partitionierte Tabellen“ in *Systemverwaltung: Konzept*
- „Clustered indexes“ in *Developing SQL and External Routines*
- „Non-clustered indexes“ in *Developing SQL and External Routines*

Zugehörige Tasks:

- „Ändern partitionierter Tabellen“ in *Systemverwaltung: Implementierung*
- „Erstellen partitionierter Tabellen“ in *Systemverwaltung: Implementierung*
- „Erstellen eines Index“ in *Systemverwaltung: Implementierung*
- „Creating indexes“ in *Developing SQL and External Routines*

- „Designing clustered indexes“ in *Developing SQL and External Routines*
- „Designing non-clustered indexes“ in *Developing SQL and External Routines*

Zugehörige Referenzen:

- „CREATE INDEX statement“ in *SQL Reference, Volume 2*

Asynchrone Indexbereinigung

Als asynchrone Indexbereinigung (AIC, Asynchronous Index Cleanup) wird die verzögerte Bereinigung von Indizes nach Operationen bezeichnet, die Indexeinträge ungültig machen. Abhängig vom Typ des Indexes können die Einträge Satz-IDs (RIDs) oder Block-IDs (BIDs) sein. In beiden Fällen werden diese Einträge durch die Indexbereinigungsfunktionen entfernt, die asynchron im Hintergrund ausgeführt werden.

Die AIC beschleunigt das Aufheben der Zuordnung einer Datenpartition zu einer partitionierten Tabelle. Wenn die partitionierte Tabelle einen oder mehrere nicht partitionierte Indizes enthält, wird die AIC eingeleitet. In diesem Fall entfernt die AIC alle nicht partitionierten Indexeinträge, die sich auf die Datenpartition mit aufgehobener Zuordnung beziehen, und außerdem alle pseudo-gelöschten Einträge. Wenn alle Indizes bereinigt sind, wird die Kennung (ID), die zu der Datenpartition mit aufgehobener Zuordnung gehört, aus dem Systemkatalog entfernt.

Anmerkung: Wenn für die partitionierte Tabelle abhängige MQTs (Materialized Query Tables, gespeicherte Abfragetabellen) definiert sind, wird die AIC erst nach der Ausführung einer SET INTEGRITY-Operation eingeleitet.

Während der Ausführung der AIC bleibt der normale Tabellenzugriff erhalten. Abfragen, die auf die Indizes zugreifen, ignorieren einfach alle ungültigen Einträge, die noch nicht bereinigt wurden.

In den meisten Fällen wird eine Bereinigungsfunktion für jeden nicht partitionierten Index gestartet, der der partitionierten Tabelle zugeordnet ist. Ein interner Taskverteilungsdämonprozess ist für die Verteilung der AIC-Tasks an die jeweiligen Datenpartitionen sowie für die Zuweisung von Datenbankagenten zuständig.

Der Verteilungsdämon und die Bereinigungsagenten sind interne Systemanwendungen. Sie werden in der Ausgabe des Befehls LIST APPLICATION mit dem Anwendungsnamen **db2taskd** bzw. **db2aic** aufgeführt. Zur Vermeidung versehentlicher Unterbrechungen lässt sich der Abbruch von Systemanwendungen nicht erzwingen. Der Verteilungsdämon bleibt online, solange die Datenbank aktiv ist. Die Bereinigungsfunktionen bleiben aktiv, bis die Bereinigung abgeschlossen ist. Falls die Datenbank während der Bereinigung inaktiviert wird, nimmt die AIC die Arbeit wieder auf, wenn die Datenbank reaktiviert wird.

Leistung:

Die AIC wirkt sich nur minimal auf die Leistung aus.

Ein sofortiger Zeilensperrentest ist erforderlich, um festzustellen, ob ein pseudo-gelöschter Eintrag festgeschrieben wurde. Da die Sperre jedoch nie aktiviert wird, wird der gemeinsame Zugriff nicht beeinträchtigt.

Jede Bereinigungsfunktion aktiviert eine minimale Tabellenbereichssperre (IX) und eine Tabellensperre (IS). Die Sperren werden freigegeben, wenn die Bereinigungsfunktion feststellt, dass andere Anwendungen auf die Sperren warten. Wenn dieser Fall eintritt, setzt die Bereinigungsfunktion die Verarbeitung vorübergehend für fünf Minuten aus.

Bereinigungsfunktionen sind darüber hinaus in die Drosselungseinrichtung für Dienstprogramme integriert. Standardmäßig ist für jede Bereinigungsfunktion der Wert 50 für die Leistungseinflusspriorität von Dienstprogrammen eingestellt. Diese Priorität kann mit Hilfe des Befehls SET UTIL_IMPACT_PRIORITY oder über die API *db2UtilityControl* geändert werden.

Überwachung:

Während der Ausführung kann die AIC mit Hilfe des Befehls LIST UTILITIES überwacht werden. Jede Indexbereinigungsfunktion wird im Monitor als separates Dienstprogramm aufgeführt.

Beispiel 1

Das folgende Beispiel veranschaulicht, wie die AIC-Aktivität in der Datenbank WSDB in der aktuellen Datenbankpartition über die Schnittstelle des Befehlszeilenprozessors (CLP) angezeigt wird.

```
$ db2 list utilities show detail
```

```
ID                      = 2
Typ                     = ASYNCHRONOUS INDEX CLEANUP
Datenbankname           = WSDB
Partitionsnummer        = 0
Beschreibung            = Tabelle: USER1.SALES, Index: USER1.I2
Startzeit               = 2005-12-15 11:15:01.967939
Status                  = Wird ausgeführt
Aufruftyp               = Automatisch
Drosselung:
  Priorität              = 50
  Fortschrittsüberwachung:
    Gesamte Arbeit      = 5 Seiten
    Abgeschlossene Arbeit = 0 Seiten
    Startzeit           = 2005-12-15 11:15:01.979033

ID                      = 1
Typ                     = ASYNCHRONOUS INDEX CLEANUP
Datenbankname           = WSDB
Partitionsnummer        = 0
Beschreibung            = Tabelle: USER1.SALES, Index: USER1.I1
Startzeit               = 2005-12-15 11:15:01.978554
Status                  = Wird ausgeführt
Aufruftyp               = Automatisch
Drosselung:
  Priorität              = 50
  Fortschrittsüberwachung:
    Gesamte Arbeit      = 5 Seiten
    Abgeschlossene Arbeit = 0 Seiten
    Startzeit           = 2005-12-15 11:15:01.980524
```

In gezeigten Fall sind zwei Bereinigungsfunktionen an der Tabelle USERS1.SALES aktiv. Eine Bereinigungsfunktion bearbeitet den Index I1, die andere den Index I2.

Dem Abschnitt unter 'Fortschrittsüberwachung' ist die geschätzte Gesamtzahl von zu bereinigenden Indexseiten sowie die aktuelle Anzahl der bereits bereinigten Indexseiten zu entnehmen.

Das Feld 'Status' gibt den aktuellen Status der Bereinigungsfunktion an. Der normale Status ist 'Wird ausgeführt'. Die Bereinigungsfunktion kann 'Im Wartestatus' sein, wenn sie darauf wartet, einem verfügbaren Datenbankagenten zugeordnet zu werden oder wenn sie wegen eines Sperrenkonflikts vorübergehend ausgesetzt ist.

Anmerkung: Verschiedene Tasks in verschiedenen Datenbankpartitionen können die gleiche Dienstprogramm-ID haben, da den Tasks in jeder Datenbankpartition IDs nur für diese Datenbankpartition zuordnet werden.

Zugehörige Konzepte:

- „Attribute von Datenpartitionen mit aufgehobener Zuordnung“ in *Systemverwaltung: Implementierung*
- „Datenpartitionen“ in *Systemverwaltung: Konzept*

Zugehörige Tasks:

- „Aufheben der Zuordnung einer Datenpartition“ in *Systemverwaltung: Implementierung*

Zugehörige Referenzen:

- „Richtlinien und Einschränkungen beim Ändern partitionierter Tabellen mit Datenpartitionen, die zugeordnet sind bzw. deren Zuordnung aufgehoben wurde“ in *Systemverwaltung: Implementierung*
- „ALTER TABLE statement“ in *SQL Reference, Volume 2*
- „SET UTIL_IMPACT_PRIORITY command“ in *Command Reference*

Kapitel 4. Designadvisor

Designadvisor

Der Designadvisor von DB2 ist ein Tool, das Ihnen helfen kann, die Auslastungsleistung Ihres Systems erheblich zu verbessern. Die Aufgabe, zu entscheiden, welche Indizes, MQTs (MQT - Materialized Query Table, gespeicherte Abfragetabelle), Clusteringdimensionen oder Datenbankpartitionen für eine komplexe Auslastung zu erstellen sind, kann sich als äußerst schwierig gestalten. Der Designadvisor ermittelt alle Objekte, die zur Verbesserung der Leistung Ihrer Auslastung erforderlich sind. Für einen gegebenen Satz von SQL-Anweisungen in einer Auslastung generiert der Designadvisor Empfehlungen für folgende Objekte und Maßnahmen:

- Neue Indizes
- Neue gespeicherte MQTs
- Umwandlung in MDC-Tabellen (mit mehrdimensionalem Clustering)
- Umverteilen von Tabellen
- Löschen von Indizes und MQTs, die durch die angegebene Auslastung nicht genutzt werden (über das GUI-Tool)

Über den Designadvisor können Sie einige oder alle dieser Empfehlungen unverzüglich implementieren oder ihre Implementierung für einen späteren Zeitpunkt terminieren.

Über die grafische Benutzerschnittstelle (GUI) des Designadvisors bzw. das Befehlszeilentool können Sie den Designadvisor zur Vereinfachung der folgenden Aufgaben einsetzen:

Planen oder Einrichten einer neuen Datenbank

Beim Entwurf Ihrer Datenbank können Sie den Designadvisor zu folgenden Zwecken einsetzen:

- Generieren von Entwurfsalternativen in einer Testumgebung für eine partitionierte Datenbankumgebung sowie für Indizes, MQTs und MDC-Tabellen.
- Für partitionierte Datenbankumgebungen kann der Designadvisor mit folgenden Zielen verwendet werden:
 - Ermitteln der Strategie für eine Datenbankpartitionierung vor dem Laden von Daten in eine Datenbank.
 - Unterstützen der Migration von einer DB2-Einzelpartitionsdatenbank auf eine DB2-Datenbank mit mehreren Partitionen.
 - Unterstützen der Migration von einem anderen Datenprodukt auf eine DB2-Datenbank mit mehreren Partitionen.
- Bewerten von Indizes, MQTs, MDC-Tabellen oder Strategien für eine Datenbankpartitionierung, die manuell generiert wurden.

Optimieren der Auslastungsleistung

Nach der Einrichtung Ihrer Datenbank können Sie den Designadvisor zu folgenden Zwecken einsetzen:

- Verbessern der Leistung einer bestimmten Anweisung oder Auslastung
- Verbessern der allgemeinen Datenbankanleistung unter Verwendung der Leistung einer Musterauslastung als Messgröße
- Verbessern der Leistung der am häufigsten ausgeführten Abfragen, wie sie zum Beispiel durch den Aktivitätsmonitor ermittelt werden

- Bestimmen, wie sich die Leistung einer neuen kritischen Abfrage optimieren lässt
- Reagieren auf Empfehlungen der Diagnosezentrale im Hinblick auf Probleme mit dem gemeinsamen Dienstprogrammspeicher oder dem Sortierspeicher bei einer sortierintensiven Auslastung
- Ermitteln von Objekten, die in keiner Auslastung verwendet werden

Ausgabe des Designadvisors

Über die grafische Benutzerschnittstelle (GUI) des Designadvisors können Sie Empfehlungen aus dem Designadvisor anzeigen, speichern oder implementieren. Wenn Sie den Designadvisor über die Befehlszeile ausführen, wird die Ausgabe standardmäßig an die Standardausgabeeinheit (stdout) gesendet und in den Tabellen ADVISE_TABLE und ADVISE_INDEX gespeichert.

- Die Tabelle ADVISE_TABLE enthält USE_TABLE='Y' für MQT- und MDC-Tabellen sowie Empfehlungen zu Strategien für die Datenbankpartitionierung. Die MQT-Empfehlungen sind außerdem in der Tabelle ADVISE_MQT, die MDC-Empfehlungen in der Tabelle ADVISE_TABLE und die Empfehlungen zu Strategien für die Datenbankpartitionierung in der Tabelle ADVISE_PARTITION zu finden. Der Wert RUN_ID in diesen Tabellen entspricht dem Wert START_TIME in der Tabelle ADVISE_INSTANCE für jede Ausführung des Designadvisors.
- Die Tabelle ADVISE_INDEX enthält den Wert USE_INDEX='Y' oder 'R' für Indexempfehlungen.

Die Tabelle ADVISE_INSTANCE wird ebenfalls mit einer Zeile pro Ausführung des Designadvisors aktualisiert:

- Das Feld START_TIME und das Feld END_TIME zeigen die Start- bzw. Stoppzeiten des Dienstprogramms.
- In das Feld STATUS wird der Wert 'COMPLETED' eingetragen, wenn das Dienstprogramm erfolgreich beendet wird.
- Das Feld MODE zeigt an, ob die Option -m verwendet wurde.
- Das Feld COMPRESSION gibt den Typ der verwendeten Komprimierung an.

Sie können die Empfehlungen des Designadvisors über die Option -o in einer Datei speichern. Die gespeicherte Ausgabe des Designadvisors besteht aus den folgenden Elementen:

- Die CREATE-Anweisungen, die für neue Indizes, MQTs, Strategien für die Datenbankpartitionierung und MDC-Tabellen angegeben werden
- REFRESH-Anweisungen für MQTs
- RUNSTATS-Befehle für neue Objekte
- Empfohlene, bereits vorhandene MQTs, sofern sie zur Ausführung der Auslastung verwendet wurden und werden

Anmerkung: Die Spalte COLSTATS der Tabelle ADVISE_MQT enthält die Spaltenstatistiken für eine MQT. Die Statistiken besitzen eine XML-Struktur wie die folgende:

```
<?xml version="1.0" encoding="USASCII"?>
<colstats>
  <column>
    <name>SPALTENNAME1</name>
    <colcard>1000</colcard>
    <high2key>999</high2key>
    <low2key>2</low2key>
  </column>
  ....

```

```

<column>
  <name>SPALTENNAME100</name>
  <colcard>55000</colcard>
  <high2key>49999</high2key>
  <low2key>100</low2key>
</column>
</colstats>

```

Beachten Sie, dass die XML-Struktur mehr als eine Spalte beinhalten kann. Für jede Spalte wird die Spaltenkardinalität (d. h. die Anzahl der Werte in der Spalte) sowie optional die Werte für HIGH2KEY und LOW2KEY angegeben.

Wenn MQT-, Datenbankpartitionierungs- oder MDC-Empfehlungen angegeben werden, wird der relevante Befehl zum Aufrufen der gespeicherten Prozedur ALTER TABLE in der Spalte ALTER_COMMAND von ADVISE_TABLE positioniert.

Anmerkung: Der Befehl zum Aufrufen der gespeicherten Prozedur ALTER TABLE schlägt möglicherweise auf Grund von Tabelleneinschränkungen für die gespeicherte Prozedur ALTOBJ fehl.

Nach einigen kleineren Modifikationen können Sie diese Ausgabedatei als Script für den Befehlszeilenprozessor (CLP) zur Erstellung der empfohlenen Objekte verwenden. Folgende Modifikationen bieten sich eventuell an:

- Kombinieren aller RUNSTATS-Befehlsanweisungen zu einem einzigen RUNSTATS-Aufruf für die neuen oder modifizierten Objekte
- Angeben geeigneterer Objektnamen als die systemgenerierten IDs
- Entfernen aller DDL-Anweisungen für Objekte, die nicht sofort implementiert werden sollen, aus dem Code durch Löschen oder Verwenden von Kommentarzeichen

Zugehörige Konzepte:

- „Verwenden von relationalen Indizes zur Leistungsverbesserung“ auf Seite 29
- „MQTs (MQT - Materialized Query Table, gespeicherte Abfragetabelle)“ auf Seite 11
- „Umgebungen mit partitionierten Datenbanken“ in *Systemverwaltung: Konzept*
- „Tipps zur Planung von relationalen Indizes“ auf Seite 33
- „Tabellen mit mehrdimensionalem Clustering“ in *Systemverwaltung: Konzept*

Zugehörige Referenzen:

- „db2advis - DB2 design advisor command“ in *Command Reference*

Designadvisor-Funktionen

Der Designadvisor berücksichtigt bei der Formulierung von Designempfehlungen die Kosten und Vorteile der einzelnen Funktionen und Komponenten. Er wertet die Merkmale Ihrer Auslastung, die Kenndaten der Datenbank und die Hardwareressourcen aus. Dann wägt er die Leistungsverbesserungen gegen den Aufwand ab, der mit der Implementierung der Empfehlungen verbunden ist.

Bevor Sie eine der Empfehlungen des Designadvisors implementieren, sollten Sie unbedingt überlegen, ob Sie den Designadvisor mit repräsentativen Informationen

zu Ihrer Abfrageauslastung versorgt haben. Wenn die Auslastung, die Sie an den Designadvisor übergeben, nicht für die Gesamtdatenbankaktivität repräsentativ ist, bieten die Empfehlungen möglicherweise keine Leistungsverbesserungen für Abfragen, die nicht zur angegebenen Auslastung gehören. Darüber hinaus ist es von entscheidender Bedeutung, dass die für jedes an der Auslastung beteiligte Objekt erfassten Statistikdaten aktuell sind. Unpräzise Statistikdaten können sich negativ auf die Qualität der Empfehlungen des Designadvisors auswirken.

Da der Designadvisor eine Kosten-Nutzen-Analyse für Sie durchführt, werden standardmäßig alle Funktionen und Komponenten zur Berücksichtigung in der Analyse ausgewählt. Es wird empfohlen, diese Standardeinstellung zu belassen und den Designadvisor feststellen zu lassen, welche Funktionen bzw. Komponenten die größten Gesamtleistungsvorteile für die von Ihnen angegebene Auslastung bieten.

Wenn Sie bestimmte Leistungsverbesserungsfunktionen nicht implementieren möchten, können Sie festlegen, dass diese zur Auswertung nicht ausgewählt werden. Durch die Auslassung von Funktionen benötigt der Designadvisor weniger Zeit zur Auswertung der verschiedenen Möglichkeiten zur Leistungsverbesserung für Ihre Datenbank.

Die folgenden Informationen enthalten Richtlinien für die Bestimmung, welche Funktionen oder Komponenten für Ihr System vielleicht nicht geeignet sind.

Nachteile von MQTs

MQTs (Materialized Query Tables, gespeicherte Abfragetabellen) verursachen Aufwand im Hinblick auf den Speicherbedarf sowie durch die mit ihnen verbundenen Aktualisierungen. Die Daten in einer MQT sind redundant: Sie werden einmal in der MQT und einmal in der zugrunde liegenden Basistabelle bzw. den Basistabellen gespeichert. Dies bedeutet, dass jede Aktualisierung an den Daten in den Basistabellen zwei Aktualisierungen erfordert, wenn MQTs vorhanden sind, die mit REFRESH IMMEDIATE definiert sind. Bei einer Auslastung mit zahlreichen Aktualisierungen kann dieser Mehraufwand im Vergleich zu den Leistungsvorteilen von MQT-Kandidaten überwiegen.

Darüber hinaus sind MQTs auch mit Verwaltungsaufwand verbunden, wenn Sie Dienstprogramme wie LOAD, BACKUP, RESTORE oder RUNSTATS ausführen.

Zur Bestimmung, ob MQTs signifikante Leistungsvorteile bieten würden, betrachtet der Designadvisor den mit MQTs verbundenen Aufwand in Verbindung mit den Merkmalen der Auslastung, die Sie angeben.

Nachteile von MDC-Tabellen

Die Implementierung von Empfehlungen zu MDC-Tabellen (Multidimensional Clustering-Tabellen) erfordert das Löschen und erneute Erstellen von Tabellen sowie das Erstellen zusätzlichen Speicherbereichs zum Speichern der Tabellen. Wenn Sie sicher sind, dass Sie dies nicht wünschen, wählen Sie diese Funktionalität nicht aus.

Nachteile der erneuten Partitionierung von Tabellen

Die erneute Partitionierung von Tabellen ist nur in einer partitionierten Umgebung ohne gemeinsame Ressourcen relevant. Die Implementierung von Empfehlungen

zur erneuten Partitionierung erfordert das Löschen und erneute Erstellen von Tabellen. Wenn Sie sicher sind, dass Sie dies nicht wünschen, wählen Sie diese Funktionalität nicht aus.

Zugehörige Konzepte:

- „Informationen zu MQTs“ in *Systemverwaltung: Konzept*
- „Designadvisor-Frequenz“ in *Systemverwaltung: Konzept*
- „Prozess der Designadvisor-Empfehlungen“ in *Systemverwaltung: Konzept*
- „Designadvisor-Zeitlimit“ in *Systemverwaltung: Konzept*
- „Designadvisor-Auslastung“ in *Systemverwaltung: Konzept*
- „Tabellen mit mehrdimensionalem Clustering“ in *Systemverwaltung: Konzept*
- „Partitionierte Datenbanken“ in *Systemverwaltung: Konzept*

Verwenden des Designadvisors

Sie können den Designadvisor über die Befehlszeile ausführen oder die grafische Benutzerschnittstelle (GUI) des Designadvisors in der Steuerzentrale verwenden.

Vorgehensweise:

- **In der Steuerzentrale:**

1. Definieren Sie Ihre Auslastung.
2. Öffnen Sie die Steuerzentrale.
3. Führen Sie den Designadvisor über die Steuerzentrale aus, indem Sie die gewünschte Datenbank oder einen bestimmten Index in der Datenbank mit der rechten Maustaste anklicken und **Designadvisor** im Kontextmenü auswählen.
4. Wenn der Designadvisor die Generierung von Empfehlungen abgeschlossen hat, können Sie auswählen, ob diese Empfehlungen in einem Bericht gespeichert werden sollen oder ob der Designadvisor einige oder alle der Empfehlungen für Sie implementieren soll.

- **Über die Befehlszeile:**

1. Definieren Sie Ihre Auslastung.
2. Führen Sie den Befehl **db2adv** für die Auslastung aus.

Anmerkung: Wenn die Statistikdaten für Ihre Datenbank keinen aktuellen Stand haben, sind die generierten Empfehlungen weniger zuverlässig.

3. Interpretieren Sie die Ausgabe des Befehls **db2adv**, und nehmen Sie alle erforderlichen Modifikationen vor.
4. Implementieren Sie die ausgewählten Empfehlungen des Designadvisors.

Zugehörige Konzepte:

- „Begrenzungen und Einschränkungen des Designadvisors“ auf Seite 58
- „Designadvisor“ auf Seite 51

Zugehörige Tasks:

- „Definieren einer Auslastung für den Designadvisor“ auf Seite 56
- „Verwenden des Designadvisors zur Migration von einer Datenbank mit einer einzelnen Partition auf eine Datenbank mit mehreren Partitionen“ auf Seite 57

Definieren einer Auslastung für den Designadvisor

Eine *Auslastung* ist eine Gruppe von SQL-Anweisungen, die der Datenbankmanager in einem bestimmten Zeitraum verarbeiten muss. Zum Beispiel könnte der Datenbankmanager in einem Monat 1000 INSERT-, 10000 UPDATE-, 10000 SELECT- und 1000 DELETE-Anweisungen verarbeiten müssen. Der Designadvisor analysiert eine angegebene Auslastung und zieht Faktoren, wie den Typ der Auslastungsanweisungen, die Häufigkeit, mit der eine bestimmte Anweisung auftritt, sowie Merkmale Ihrer Datenbank in Betracht, um Empfehlungen zu generieren, die den Gesamtaufwand zur Ausführung der Auslastung minimieren.

Vorgehensweise:

Über die Designadvisor-GUI:

Über die Seite „Auslastung“ der grafischen Benutzerschnittstelle des Designadvisors können Sie eine neue Auslastungsdatei erstellen oder eine bereits vorhandene Auslastungsdatei modifizieren. Sie können Anweisungen in die Datei aus verschiedenen Quellen importieren:

- Eine Textdatei mit Begrenzern
- Eine Ereignismonitortabelle
- Query Patroller-Protokolldatentabellen durch die Option -qp über die Befehlszeile
- Mit EXPLAIN bearbeitete Anweisungen in der Tabelle EXPLAINED_STATEMENT
- Neue SQL-Anweisungen, die mit einer DB2-Momentaufnahme erfasst wurden

Nach dem Importieren Ihrer SQL-Anweisungen können Sie Anweisungen hinzufügen, ändern, modifizieren oder entfernen sowie die Häufigkeit der Anweisungen modifizieren.

Über die Befehlszeile:

Über die Befehlszeile führen Sie den Designadvisor wie folgt aus:

- Mit einer einzigen SQL-Anweisung, die Sie in der Befehlszeile zusammen mit dem Befehl angeben
- Mit einer Gruppe dynamischer SQL-Anweisungen, die in einer DB2-Momentaufnahme erfasst wurden
- Mit einer Gruppe von SQL-Anweisungen, die in einer Auslastungsdatei enthalten sind

Gehen Sie wie folgt vor, um den Designadvisor für dynamische SQL-Anweisungen auszuführen:

1. Setzen Sie den Datenbankmonitor mit dem folgenden Befehl zurück:
`db2 reset monitor for database datenbankname`
2. Warten Sie einen geeigneten Zeitraum für die Ausführung dynamischer SQL-Anweisungen für die Datenbank ab.
3. Führen Sie den Befehl **db2adv** mit der Option -g aus. Wenn Sie die dynamischen SQL-Anweisungen in der Tabelle ADVISE_WORKLOAD zur späteren Referenz speichern wollen, verwenden Sie außerdem die Option -p.

Gehen Sie wie folgt vor, um den Designadvisor für eine Gruppe von SQL-Anweisungen auszuführen, die in einer Auslastungsdatei enthalten sind:

1. Erstellen Sie manuell eine Auslastungsdatei, indem Sie jede SQL-Anweisung mit einem Semikolon trennen, oder importieren Sie SQL-Anweisungen aus einer oder mehreren der oben aufgeführten Quellen.
2. Legen Sie die Häufigkeit der Anweisungen in der Auslastung fest. Jeder Anweisung in der Auslastungsdatei wird standardmäßig die Häufigkeit 1 zugeordnet. Die Häufigkeit einer SQL-Anweisung stellt die Anzahl der Vorkommen der Anweisung innerhalb der Auslastung im Verhältnis zur Anzahl der Vorkommen anderer Anweisungen dar. Eine bestimmte SELECT-Anweisung könnte zum Beispiel 100-mal in einer Auslastung vorkommen, während eine andere SELECT-Anweisung 10-mal vorkommt. Zur Darstellung der relativen Häufigkeit dieser beiden Anweisungen würden Sie der ersten SELECT-Anweisung eine Häufigkeit von 10 und der zweiten SELECT-Anweisung eine Häufigkeit von 1 zuordnen. Sie können die Häufigkeit bzw. die Wertigkeit einer bestimmten Anweisung in der Auslastung manuell ändern, indem Sie die folgende Zeile nach der Anweisung einfügen: `- # SET FREQUENCY n`. Dabei ist *n* der Häufigkeitswert, den Sie der Anweisung zuordnen wollen.
3. Geben Sie den Befehl **db2adv** mit der Option `-i` gefolgt vom Namen der Auslastungsdatei ein.

Zur Ausführung des Designadvisors für eine Auslastung, die in der Tabelle ADVISE_WORKLOAD enthalten ist, geben Sie den Befehl **db2adv** mit der Option `-w` gefolgt vom Namen der Auslastung ein.

Zugehörige Konzepte:

- „Designadvisor“ auf Seite 51

Verwenden des Designadvisors zur Migration von einer Datenbank mit einer einzelnen Partition auf eine Datenbank mit mehreren Partitionen

Sie können den Designadvisor zur Unterstützung einer Migration von einer Datenbank mit einer Einzelpartition auf eine Datenbank mit mehreren Partitionen verwenden. Der Designadvisor kann Ihnen Empfehlungen für die Verteilung Ihrer Daten und gleichzeitig Empfehlungen für neue Indizes, MQTs (MQT - Materialized Query Table, gespeicherte Abfragetabelle), und MDC-Tabellen (mit mehrdimensionalem Clustering) geben.

Vorgehensweise:

1. Aktualisieren Sie den Produktlizenzschlüssel für DB2 ESE.
2. Erstellen Sie mindestens einen Tabellenbereich in einer Partitionsgruppe einer Datenbank mit mehreren Partitionen.

Anmerkung: Da der Designadvisor nur eine Umverteilung für vorhandene Tabellenbereiche empfehlen kann, müssen die Tabellenbereiche, die der Designadvisor in Betracht ziehen soll, in der partitionierten Datenbank vorhanden sein, bevor der Designadvisor ausgeführt wird.

3. Führen Sie den Designadvisor mit ausgewählter Datenbankpartitionierungsfunktion über die Designadvisor-GUI oder unter Angabe der Partitionierungsoption mit dem Befehl **db2adv** aus.
4. Wenn Sie mit dem Designadvisor in der Steuerzentrale arbeiten, können Sie die Datenbankpartitionierungsempfehlungen automatisch implementieren. Wenn

Sie den Befehl **db2adv** verwenden, müssen Sie die **db2adv**-Ausgabedatei geringfügig modifizieren, bevor Sie die vom Designadvisor generierten DDL-Anweisungen ausführen.

Zugehörige Konzepte:

- „Designadvisor“ auf Seite 51

Zugehörige Tasks:

- „Registrieren des Lizenzschlüssels für ein DB2-Produkt oder -Feature mit dem Befehl **db2licm**“ in *Installation und Konfiguration Ergänzung*

Begrenzungen und Einschränkungen des Designadvisors

1. Einschränkungen für Indexempfehlungen

- Indizes, die für MQTs (MQT - Materialized Query Table, gespeicherte Abfragetabelle) empfohlen werden, dienen zur Verbesserung der Auslastungsleistung, und nicht der REFRESH TABLE-Leistung. Wenn außerdem Aktualisierungs-, Einfüge- und Löschoperationen kein Bestandteil der Auslastung sind, würde die Leistung einer Änderung (z. B. Aktualisierung) der MQT für mit IMMEDIATE definierte MQTs nicht berücksichtigt.
- Der RID-Clusterindex wird nur empfohlen, wenn mehrdimensionales Clustering auszuwählen ist. Der Designadvisor schließt RID-Clusterindizes als Option anstelle der Erstellung einer MDC-Struktur für eine Tabelle mit ein.

2. Einschränkungen für MQT-Empfehlungen

- Der Designadvisor empfiehlt keine inkrementellen gespeicherten MQTs. Wenn Sie inkrementelle MQTs erstellen wollen, können Sie mit REFRESH IMMEDIATE definierte MQTs nehmen und diese in inkrementelle Abfragetabellen mit einer eigenen Auswahl an Zwischenspeichertabellen konvertieren.
- Indizes, die für MQTs empfohlen werden, dienen zur Verbesserung der Auslastungsleistung, und nicht der REFRESH-Leistung für MQTs.
- Wenn Aktualisierungs-, Einfüge- und Löschoperationen kein Bestandteil der angegebenen Auslastung sind, wird der Leistungsaufwand zur Aktualisierung einer empfohlenen, mit REFRESH IMMEDIATE definierten MQT nicht berücksichtigt. Es wird empfohlen, für MQTs mit REFRESH IMMEDIATE eindeutige Indizes für den implizierten eindeutigen Schlüssel zu erstellen, der sich aus den Spalten in der GROUP BY-Klausel der Definition der MQT zusammensetzt.

3. Einschränkungen für MDC-Empfehlungen

- Vorhandene Tabellen müssen Daten enthalten. Ansonsten wird MDC für die Tabelle nicht in Betracht gezogen.
- MDC-Empfehlungen für neue MQTs werden nicht in Betracht gezogen, sofern nicht die Stichprobenoption -r mit dem Befehl verwendet wird oder die MQT-Stichprobenerstellung im GUI-Tool ausgewählt wird.
- Der Designadvisor gibt keine MDC-Empfehlungen für typisierte oder temporäre Tabellen.
- Der Designadvisor gibt keine MDC-Empfehlungen für föderierte Tabellen.
- Der Speicherplatz für die Stichprobendaten, der während der Ausführung des Designadvisors verwendet wird, muss vorhanden sein. Ansonsten wird die Stichprobentabelle nur für Basisspalten unter der Annahme unkorrelierter Daten untersucht. In diesem Fall wird eine Warnung generiert.
- Tabellen ohne erfasste Statistiken werden bei der Analyse übersprungen.

- Der Designadvisor gibt keine Empfehlungen für mehrspaltige Dimensionen.
- Vorhandene Tabellen müssen Daten enthalten, damit die Stichprobenerstellung bei der MDC-Auswahl funktioniert.

4. Einschränkungen für Datenbankpartitionierungsempfehlungen

Der Designadvisor kann eine Datenbankpartitionierung nur unter DB2 Enterprise Server Edition empfehlen. Wenn die Datenbankpartitionierungsoptionen mit dem Befehl **db2adv** angegeben werden, wird ein Fehler zurückgegeben. In der grafischen Benutzerschnittstelle (GUI) des Designadvisors kann die Datenbankpartitionierungsfunktion in einer Einzelpartitionsdatenbankumgebung nicht ausgewählt werden.

5. Weitere Einschränkungen

Während der Ausführung des Designadvisors werden Simulationskatalogtabellen erstellt. Diese Tabellen werden gelöscht, wenn die Ausführung des Designadvisors beendet ist. Eine unvollständige Ausführung des Designadvisors kann dazu führen, dass einige dieser Tabellen nicht gelöscht werden. In diesem Fall können Sie den Designadvisor dazu verwenden, diese Tabellen zu löschen, indem Sie das Dienstprogramm über die Befehlszeile erneut starten. Zum Entfernen der Simulationskatalogtabellen geben Sie die Option **-f** und die Option **-n** an (geben Sie mit **-n** den gleichen Benutzernamen ein, der bei der unvollständigen Ausführung verwendet wurde). Wenn Sie die Option **-f** nicht angeben, generiert der Designadvisor die DROP-Anweisungen, die zum Entfernen der Tabellen benötigt werden.

Sie sollten einen getrennten Tabellenbereich zum Speichern dieser simulierten Katalogtabellen erstellen und **DROP TABLE RECOVERY** auf den Wert **OFF** setzen. Dies ermöglicht eine einfachere Bereinigung und eine schnellere Ausführung des Designadvisors.

Zugehörige Konzepte:

- „Designadvisor“ auf Seite 51

Kapitel 5. Verwalten des gemeinsamen Zugriffs

Aspekte des gemeinsamen Zugriffs

Da viele Benutzer auf Daten in einer relationalen Datenbank zugreifen und sie ändern, muss der Datenbankmanager in der Lage sein, einerseits den Benutzern diese Änderungen zu ermöglichen, und andererseits sicherzustellen, dass die Datenintegrität gewahrt bleibt. Der Begriff *gemeinsamer Zugriff* bezeichnet die Möglichkeit, dass mehrere interaktive Benutzer oder Anwendungsprogramme die Ressourcen zur gleichen Zeit verwenden können. Der Datenbankmanager steuert diesen Zugriff, um unerwünschte Folgen zu vermeiden. Solche Folgen sind zum Beispiel:

- **Verlorene Aktualisierungen.** Zwei Anwendungen A und B lesen beispielsweise dieselbe Zeile aus einer Datenbank und berechnen aufgrund der Daten, die von den Anwendungen gelesen werden, neue Werte für eine ihrer Spalten. Die Anwendung A aktualisiert die Zeile mit ihrem neuen Wert und anschließend aktualisiert B die Zeile ebenfalls, so dass die von A durchgeführte Aktualisierung verloren geht.
- **Zugriff auf nicht festgeschriebene Daten.** Anwendung A könnte zum Beispiel einen Wert in der Datenbank aktualisieren, den Anwendung B liest, bevor er festgeschrieben wurde. Wenn dann der Wert von der Anwendung A später nicht mit COMMIT festgeschrieben, sondern zurückgesetzt wird, basieren die Berechnungen der Anwendung B auf nicht festgeschriebenen (und daher wahrscheinlich ungültigen) Daten.
- **Nichtwiederholbare Lesevorgänge.** Einige Anwendungen bewirken die folgende Reihenfolge von Ereignissen: Anwendung A liest eine Zeile aus der Datenbank und verarbeitet anschließend weitere Anforderungen. In der Zwischenzeit ändert oder löscht Anwendung B die Zeile und schreibt diese Aktion fest. Später, wenn Anwendung A versucht, die ursprüngliche Zeile erneut zu lesen, empfängt sie die geänderte Zeile oder stellt fest, dass die ursprüngliche Zeile gelöscht wurde.
- **Das Phänomen der Phantomzeilen.** Phantomzeilen treten unter folgenden Umständen auf:
 1. Eine Anwendung führt eine Abfrage aus, die eine Menge von Zeilen nach einer Suchbedingung liest.
 2. Eine andere Anwendung fügt neue Daten ein oder aktualisiert vorhandene Daten, die die Abfragebedingungen der ersten Anwendung erfüllen würden.
 3. Die erste Anwendung wiederholt die Abfrage aus Schritt 1 (innerhalb derselben UOW (Unit of Work, Arbeitseinheit)).Einige zusätzliche Zeilen („Phantomzeilen“) werden als Teil der Ergebnismenge zurückgegeben, die bei der ersten Ausführung der Abfrage (Schritt 1) nicht zurückgegeben wurden.

Anmerkung: Bei deklarierten temporären Tabellen gibt es keine Probleme bezüglich des gemeinsamen Zugriffs, da sie nur für die Anwendung verfügbar sind, die sie deklariert hat. Diese Art von Tabelle besteht nur von dem Zeitpunkt ihrer Deklaration durch die Anwendung bis zu dem Zeitpunkt, zu dem die Anwendung beendet wird oder die Verbindung trennt.

Steuerung des gemeinsamen Zugriffs in Systemen föderierter Datenbanken

Ein *System föderierter Datenbanken* unterstützt von Anwendungen und Benutzern übergebene SQL-Anweisungen, die auf zwei oder mehr Datenbankverwaltungssysteme (DBMS) bzw. Datenbanken innerhalb einer Anweisung verweisen. Als Verweise auf die Datenquellen, die aus einem DBMS und Daten bestehen, verwendet DB2 so genannte *Kurznamen*. Kurznamen sind Aliasnamen für Objekte in anderen Datenbankmanagern. In einem föderierten System verlässt sich DB2 auf die Protokolle zur Steuerung des gemeinsamen Zugriffs des Datenbankmanagers, der die angeforderten Daten bereitstellt.

In einem föderierten DB2-System besteht für Datenbankobjekte *Positionstransparenz*. Durch die Positionstransparenz ist es möglich, wenn Informationen von Tabellen und Sichten an andere Positionen versetzt werden, die Verweise auf diese Informationen über Kurznamen zu aktualisieren, ohne die Anwendungen ändern zu müssen, die diese Informationen anfordern. Wenn eine Anwendung auf Daten über Kurznamen zugreift, verlässt sich DB2 darauf, dass die Protokolle zur Steuerung des gemeinsamen Zugriffs der Datenbankmanager der Datenquellen die Isolationsstufen sicherstellen. Obwohl DB2 versucht, die angeforderte Isolationsstufe an der Datenquelle mit einem logischen Äquivalent abzugleichen, können die Ergebnisse je nach Funktionsspektrum der Datenquelle unterschiedlich sein.

Zugehörige Konzepte:

- „Isolationsstufen und Leistung“ auf Seite 62

Zugehörige Tasks:

- „Angabe der Isolationsstufe“ auf Seite 66

Zugehörige Referenzen:

- „maxlocks - Maximale Anzahl Sperren pro Anwendung “ auf Seite 474
- „locklist - Maximaler Speicher für Sperrenliste “ auf Seite 438

Isolationsstufen

Isolationsstufen und Leistung

Eine *Isolationsstufe* legt fest, wie Daten gesperrt oder von anderen Prozessen während des Zugriffs auf die Daten isoliert werden. Die Isolationsstufe ist während der Dauer der UOW (Unit of Work, Arbeitseinheit) in Kraft. Anwendungen, die einen Cursor verwenden, der mit der Anweisung DECLARE CURSOR und mit der Klausel WITH HOLD deklariert wurde, behalten die gewählte Isolationsstufe für die Dauer der UOW bei, in der die Anweisung OPEN CURSOR durchgeführt wurde. DB2 unterstützt die folgenden Isolationsstufen:

- Wiederholtes Lesen (Repeatable Read)
- Lesestabilität (Read Stability)
- Cursorstabilität (Cursor Stability)
- Nicht festgeschriebener Lesevorgang (Uncommitted Read, UR)

Anmerkung: Einige Hostdatenbankserver unterstützen die Isolationsstufe *kein Commit* (No Commit). In anderen Datenbanken verhält sich diese Isolationsstufe wie die Isolationsstufe *Nicht festgeschriebener Lesevorgang* (Uncommitted Read).

Im Folgenden sind detaillierte Erläuterungen zu den Isolationsstufen aufgeführt. Diese sind in absteigender Reihenfolge bezüglich der Auswirkungen auf die Leistung, aber in aufsteigender Reihenfolge bezüglich der Vorsicht geordnet, die beim Zugriff auf und Aktualisieren von Daten erforderlich ist.

Wiederholtes Lesen (Repeatable Read)

Die Isolationsstufe *Wiederholtes Lesen* (RR - Repeatable Read) sperrt alle Zeilen, auf die eine Anwendung innerhalb einer UOW verweist. Bei Verwendung der Isolationsstufe RR liefert eine Anweisung SELECT, die von einer Anwendung zweimal innerhalb derselben UOW, in der der Cursor geöffnet wurde, ausgegeben wird, jedes Mal dasselbe Ergebnis. Bei der Isolationsstufe RR können verlorene Aktualisierungen, Zugriffe auf nicht festgeschriebene Daten und Phantomzeilen nicht auftreten.

Eine Anwendung mit der Isolationsstufe RR kann, bis die UOW beendet wird, so oft wie erforderlich Zeilen abrufen und Aktionen an ihnen vornehmen. Andere Anwendungen können jedoch keine Zeile aktualisieren, löschen oder einfügen, die sich auf die Ergebnistabelle auswirken würde, bis die UOW beendet ist. Für Anwendungen mit der Isolationsstufe RR sind nicht festgeschriebene Änderungen anderer Anwendungen nicht sichtbar.

Durch die Isolationsstufe RR werden alle Zeilen, auf die verwiesen wird, und nicht nur die Zeilen, die abgerufen werden, gesperrt. Die Sperrung wird so ausgeführt, dass eine andere Anwendung keine Zeile einfügen oder aktualisieren kann, die der Liste von Zeilen, auf die in einer Abfrage zugegriffen wird, hinzugefügt würde, wenn diese Abfrage erneut ausgeführt würde. Dadurch wird das Auftreten von Phantomzeilen verhindert. Wenn Sie zum Beispiel 10.000 Zeilen unter Verwendung von Vergleichselementen durchsuchen, werden alle 10.000 Zeilen gesperrt, auch wenn letzten Endes nur 10 Zeilen den Vergleichselementen entsprechen.

Anmerkung: Durch die Isolationsstufe RR wird sichergestellt, dass alle zurückgelieferten Daten bis zu dem Zeitpunkt, zu dem die Daten für die Anwendung sichtbar werden, unverändert bleiben, auch wenn temporäre Tabellen oder Zeilenblockung verwendet werden.

Da die Isolationsstufe RR eine beträchtliche Anzahl an Sperren erfordern kann, können diese Sperren die Anzahl von Sperren, die durch die Konfigurationsparameter *locklist* und *maxlocks* festgelegt wird, überschreiten. Um eine Sperreneskulation zu vermeiden, kann das Optimierungsprogramm eventuell von vornherein eine einzige Sperre auf Tabellenebene für eine Indexsuche anfordern, wenn das Auftreten einer Sperreneskulation wahrscheinlich ist. Dies funktioniert so, als würde der Datenbankmanager für Sie eine Anweisung LOCK TABLE absetzen. Wenn Sie keine Sperre auf Tabellenebene aktivieren lassen wollen, stellen Sie sicher, dass für die Transaktion ausreichend Sperren verfügbar sind, oder verwenden Sie die Isolationsstufe der Lesestabilität (RS).

Bei der Auswertung referenzieller Integritätsbedingungen kann DB2 unabhängig von der durch den Benutzer festgelegten Isolationsstufe in einigen Fällen die für die Suche in der untergeordneten Tabelle verwendete Isolationsstufe intern auf RR heraufsetzen. Dadurch werden zusätzliche Sperren bis zur COMMIT-Operation aktiviert, so dass die Wahrscheinlichkeit eines Deadlocks oder einer Sperrzeitlimitüberschreitung zunimmt. Um dies zu vermeiden, wird empfohlen, einen Index zu erstellen, der nur die Spalte bzw. Spalten des Fremdschlüssels enthält, so dass die RI-Suche diesen Index als Alternative nutzen kann.

Lesestabilität (RS)

Die Isolationsstufe *Lesestabilität* (RS - Read Stability) sperrt nur die Zeilen, die von einer Anwendung innerhalb einer UOW abgerufen werden. Sie stellt sicher, dass jede den Vergleichselementen entsprechende gelesene Zeile während einer UOW nicht von Prozessen anderer Anwendungen geändert wird und dass keine von einem Prozess einer anderen Anwendung geänderte Zeile gelesen wird, bis die Änderung von dem Prozess festgeschrieben wurde. Das heißt, „nichtwiederholbare Lesevorgänge“ sind **nicht** möglich.

Anders als bei der Isolationsstufe RR (Wiederholtes Lesen) können bei der Isolationsstufe RS (Lesestabilität), wenn die Anwendung dieselbe Abfrage mehr als einmal absetzt, zusätzliche *Phantomzeilen* auftreten (*Phänomen der Phantomzeilen*). In dem bereits angeführten Beispiel der Suche über 10.000 Zeilen werden durch die Isolationsstufe RS nur die den Vergleichselementen entsprechenden Zeilen gesperrt. Da durch die Abfrage nur 10 Zeilen abgerufen werden, werden unter der Isolationsstufe RS nur für diese zehn Zeilen Sperren aktiviert. Unter der Isolationsstufe RR (Wiederholtes Lesen) werden in diesem Beispiel hingegen sämtliche 10.000 Zeilen gesperrt. Bei den aktivierten Sperren kann es sich um Sperren der Modi Share, Next Share, Update oder Exclusive handeln.

Anmerkung: Durch die Isolationsstufe RS wird sichergestellt, dass alle zurückgelieferten Daten bis zu dem Zeitpunkt, zu dem die Daten für die Anwendung *sichtbar* werden, unverändert bleiben, auch wenn temporäre Tabellen oder Zeilenblockung verwendet werden.

Ein Zweck der Isolationsstufe RS besteht darin, sowohl einen hohen Grad des gemeinsamen Zugriffs als auch eine stabile Sicht der Daten zur Verfügung zu stellen. Um diesen Zweck zu erfüllen, stellt das Optimierungsprogramm sicher, dass Sperren auf Tabellenebene erst aktiviert werden, wenn eine Sperreneskulation auftritt.

Die Isolationsstufe RS eignet sich am besten für Anwendungen, die alle folgenden Merkmale aufweisen:

- Sie arbeiten in einer Umgebung mit gemeinsamem Zugriff.
- Sie fordern Zeilen an, die für die Dauer der UOW stabil bleiben müssen.
- Sie verwenden dieselbe Abfrage in einer UOW nur einmal, oder es ist nicht notwendig, dass bei mehrmaliger Verwendung der Abfrage innerhalb derselben UOW stets dieselben Abfrageergebnisse erzielt werden.

Cursorstabilität (CS)

Die Isolationsstufe *Cursorstabilität* (CS - Cursor Stability) sperrt jede Zeile, auf die von einer Transaktion einer Anwendung zugegriffen wird, während sich der Cursor auf der Zeile befindet. Diese Sperre bleibt aktiv, bis die nächste Zeile abgerufen oder die Transaktion beendet wird. Wenn jedoch Daten in einer Zeile geändert werden, muss die Sperre aktiv bleiben, bis die Änderung in der Datenbank festgeschrieben wurde.

Keine anderen Anwendungen können eine Zeile aktualisieren oder löschen, die von einer Anwendung mit der Isolationsstufe CS abgerufen wurde, während sich ein Aktualisierungscursor auf der Zeile befindet. Für Anwendungen mit der Isolationsstufe CS sind nicht festgeschriebene Änderungen anderer Anwendungen nicht sichtbar.

In dem bereits angeführten Beispiel einer Suche über 10.000 Zeilen heißt dies, dass unter der Isolationsstufe CS lediglich eine Sperre für die Zeile, die unter der aktuellen Cursorposition liegt, aktiv ist. Die Sperre wird entfernt, wenn der Cursor von der Zeile fortbewegt wird (sofern die Zeile nicht aktualisiert wurde).

Bei der Isolationsstufe CS können sowohl unterschiedliche Abfrageergebnisse innerhalb derselben UOW (d. h. nicht wiederholbare Lesevorgänge) als auch Phantomzeilen auftreten. Die Isolationsstufe CS ist die Standardisolationsstufe, die verwendet werden sollte, wenn es auf den höchstmöglichen Grad des gemeinsamen Zugriffs ankommt und nur festgeschriebene Zeilen anderer Anwendungen sichtbar sein sollen.

Nicht festgeschriebener Lesevorgang (Uncommitted Read, UR)

Die Isolationsstufe *Nicht festgeschriebener Lesevorgang* (UR - Uncommitted Read) erlaubt einer Anwendung den Zugriff auf nicht festgeschriebene Änderungen anderer Transaktionen. Die Anwendung sperrt die Zeile, die sie liest, auch für keine andere Anwendung, sofern die andere Anwendung nicht versucht, die Tabelle zu löschen oder zu ändern. Die Isolationsstufe UR wirkt sich auf Lescursor und Aktualisierungscursor unterschiedlich aus.

Cursor mit Lesezugriff können auf die meisten nicht festgeschriebenen Änderungen anderer Transaktionen zugreifen. Tabellen, Sichten und Indizes, die momentan von anderen Transaktionen erstellt oder gelöscht werden, sind während der Verarbeitung der Transaktion jedoch nicht verfügbar. Alle anderen Änderungen durch andere Transaktionen können gelesen werden, bevor sie mit COMMIT festgeschrieben oder mit ROLLBACK rückgängig gemacht wurden.

Anmerkung: Aktualisierungscursor, die unter der Isolationsstufe UR arbeiten, verhalten sich genauso wie unter der Isolationsstufe CS (Cursorstabilität).

Wenn eine Anwendung ein Programm unter der Isolationsstufe UR ausführt, kann sie die Isolationsstufe CS verwenden. Dies liegt an der Mehrdeutigkeit der Cursor, die in den Anwendungsprogrammen verwendet werden. Mehrdeutige Cursor können durch eine BLOCKING-Option auf die Isolationsstufe CS eskaliert werden. Der Standardwert der BLOCKING-Option ist UNAMBIG. Das bedeutet, dass mehrdeutige Cursor als aktualisierbar behandelt werden und die Eskalation der Isolationsstufe auf CS durchgeführt wird. Zur Vermeidung dieser Eskalation haben Sie die folgenden beiden Möglichkeiten:

- Ändern Sie die Cursor im Anwendungsprogramm, so dass die Mehrdeutigkeit beseitigt wird. Ändern Sie die SELECT-Anweisungen, um die Klausel FOR READ ONLY hinzuzufügen.
- Belassen Sie die mehrdeutigen Cursor im Anwendungsprogramm, aber kompilieren Sie das Programm mit der Option BLOCKING ALL vor oder binden Sie es mit dieser Option, um zuzulassen, dass alle mehrdeutigen Cursor als reine Lescursor behandelt werden, wenn das Programm ausgeführt wird.

Wie in dem für die Isolationsstufe RR genannten Beispiel einer Suche über 10.000 Zeilen werden bei der Isolationsstufe UR keine Zeilensperren aktiviert.

Bei der Isolationsstufe UR können sowohl unterschiedliche Leseergebnisse innerhalb derselben UOW (d. h. nicht wiederholbare Lesevorgänge) als auch Phantomzeilen auftreten. Die Isolationsstufe UR wird in der Regel für Abfragen auf Tabellen verwendet, die sich im Lesezugriff befinden, oder wenn Anweisungen SELECT

ausgeführt werden und es dabei keine Rolle spielt, ob nicht festgeschriebene Daten anderer Anwendungen angezeigt werden.

Zusammenfassung der Isolationsstufen

Die folgende Tabelle fasst die verschiedenen Isolationsstufen im Hinblick auf unerwünschte Nebeneffekte zusammen.

Tabelle 2. Zusammenfassung der Isolationsstufen

Isolationsstufe	Zugriff auf nicht festgeschriebene Daten	Nichtwiederholbare Lesevorgänge	Phänomen der Phantomzeilen
Wiederholtes Lesen (RR)	Nicht möglich	Nicht möglich	Nicht möglich
Lesestabilität (RS)	Nicht möglich	Nicht möglich	Möglich
Cursorstabilität (CS)	Nicht möglich	Möglich	Möglich
Nicht festgeschriebener Lesevorgang (UR)	Möglich	Möglich	Möglich

Die folgende Tabelle gibt einfache Anhaltspunkte, die Ihnen bei der Wahl der ersten Isolationsstufe für Ihre Anwendungen helfen können. Betrachten Sie die Angaben dieser Tabelle als Ausgangspunkt, und lesen Sie die Beschreibung der verschiedenen Isolationsstufen in den vorigen Abschnitten, um Hinweise zu erhalten, die vielleicht eine geeignetere Isolationsstufe empfehlen.

Tabelle 3. Richtlinien zur Wahl einer Isolationsstufe

Anwendungsart	Hohe Datenstabilität erforderlich	Hohe Datenstabilität nicht erforderlich
Schreib-/Lese-Transaktionen	RS	CS
Transaktionen mit Lesezugriff	RR oder RS	UR

Die Wahl der geeigneten Isolationsstufe für eine Anwendung ist zur Vermeidung von Erscheinungen, die für die Anwendung nicht tolerierbar sind, äußerst wichtig. Von der Isolationsstufe sind nicht nur die verschiedenen Grade der Isolation zwischen Anwendungen, sondern auch die Leistungsmerkmale einer einzelnen Anwendung betroffen, da die CPU- und Speicherressourcen, die zur Aktivierung und Freigabe von Sperren erforderlich sind, mit der Isolationsstufe variieren. Die Möglichkeit, dass Deadlocks auftreten, ist ebenfalls je nach Isolationsstufe unterschiedlich.

Zugehörige Konzepte:

- „Aspekte des gemeinsamen Zugriffs“ auf Seite 61

Zugehörige Tasks:

- „Angaben der Isolationsstufe“ auf Seite 66

Angeben der Isolationsstufe

Da die Isolationsstufe festlegt, wie Daten gesperrt und von anderen Prozessen während des Zugriffs auf die Daten isoliert werden, sollten Sie eine Isolationsstufe auswählen, die die Anforderungen des gemeinsamen Zugriffs und der Datenintegrität angemessen berücksichtigt. Die Isolationsstufe, die Sie angeben, gilt für die Dauer der UOW (Unit of Work, Arbeitseinheit).

Anmerkung: Isolationsstufen können für XQuery-Anweisungen nicht auf der Anweisungsebene angegeben werden.

Die Isolationsstufe kann auf verschiedene Arten angegeben werden. Die folgenden heuristischen Methoden können verwendet werden, um die Isolationsstufe festzulegen, die zum Kompilieren einer SQL- oder XQuery-Anweisung verwendet wird:

Statisches SQL:

- Wenn eine Isolationsklausel in der Anweisung angegeben wird, wird der Wert dieser Klausel verwendet.
- Wird in der Anweisung keine Isolationsklausel angegeben, ist die verwendete Isolationsstufe die für das Paket zum Zeitpunkt der Bindung des Pakets an die Datenbank angegebene Klausel.

Dynamisches SQL:

- Wenn eine Isolationsklausel in der Anweisung angegeben wird, wird der Wert dieser Klausel verwendet.
- Wurde in der Anweisung keine Isolationsklausel angegeben, und wurde eine Anweisung SET CURRENT ISOLATION in der aktuellen Sitzung abgesetzt, wird der Wert des Sonderregisters CURRENT ISOLATION verwendet.
- Wird in der Anweisung keine Isolationsklausel angegeben, und wurde keine Anweisung SET CURRENT ISOLATION in der aktuellen Sitzung abgesetzt, ist die verwendete Isolationsstufe die für das Paket zum Zeitpunkt der Bindung des Pakets an die Datenbank angegebene Klausel.

Statische oder dynamische XQuery-Anweisungen:

- Die Isolationsstufe der Umgebung legt die Isolationsstufe fest, wenn der XQuery-Ausdruck ausgewertet wird.

Anmerkung: Viele kommerziell geschriebene Anwendungen ermöglichen eine Auswahl der Isolationsstufe. Informationen dazu finden Sie in der Dokumentation zur jeweiligen Anwendung.

Vorgehensweise:

Gehen Sie wie folgt vor, um die Isolationsstufe anzugeben:

1. Beim Vorkompilieren oder Binden:

Für eine Anwendung, die in einer unterstützten kompilierten Sprache geschrieben ist, wird die Option ISOLATION der Befehle PREP oder BIND des Befehlszeilenprozessors verwendet. Sie können die Isolationsstufe auch über die APIs PREP und BIND angeben.

- Wenn Sie beim Vorkompilieren eine Bindedatei erstellen, wird die Isolationsstufe in der Bindedatei gespeichert. Wenn Sie beim Binden keine Isolationsstufe angeben, wird standardmäßig die beim Vorkompilieren verwendete Isolationsstufe verwendet.
- Wenn Sie keine Isolationsstufe angeben, wird die Standardisolationsstufe Cursorstabilität (CS) verwendet.

Anmerkung: Führen Sie zur Feststellung der Isolationsstufe eines Pakets die folgende Abfrage aus:

```
SELECT ISOLATION FROM SYSCAT.PACKAGES
WHERE PKGNAME = 'XXXXXXXX'
AND PKGSCHEMA = 'YYYYYYYY'
```

Dabei steht `XXXXXXXX` für den Namen des Pakets und `YYYYYYYY` für den Namen des Schemas des Pakets. Beide Namen müssen vollständig in Großbuchstaben angegeben werden.

2. Auf Datenbankservern, die REXX unterstützen:

Wenn eine Datenbank erstellt wird, werden mehrere Bindedateien, die die verschiedenen Isolationsstufen für SQL in REXX unterstützen, an die Datenbank gebunden. Andere Befehlszeilenprozessorpakete werden ebenfalls an die Datenbank gebunden, wenn die Datenbank erstellt wird.

REXX und der Befehlszeilenprozessor stellen die Verbindung zu einer Datenbank mit der Standardisolationsstufe CS (Cursorstabilität) her. Durch die Änderung in eine andere Isolationsstufe wird der Status der Verbindung nicht geändert. Dies muss im Status `CONNECTABLE AND UNCONNECTED` oder `IMPLICITLY CONNECTABLE` ausgeführt werden.

Die Isolationsstufe, die von einer REXX-Anwendung verwendet wird, lässt sich durch Prüfen des Werts für die REXX-Variable `SQLISL` feststellen. Dieser Wert wird jedes Mal aktualisiert, wenn der Befehl `CHANGE SQLISL` ausgeführt wird.

3. Auf Anweisungsebene:

Verwenden Sie die Klausel `WITH`. Die Isolationsstufe auf Anweisungsebene setzt die Isolationsstufe des Pakets außer Kraft, in dem die Anweisung auftritt.

Für die folgenden SQL-Anweisungen können Sie eine Isolationsstufe angeben:

- `SELECT`
- `SELECT INTO`
- `DELETE` mit Suche
- `INSERT`
- `UPDATE` mit Suche
- `DECLARE CURSOR`

Die folgenden Bedingungen gelten für Isolationsstufen, die für Anweisungen angegeben werden:

- Die `WITH`-Klausel kann nicht in Unterabfragen verwendet werden.
- Die Option `WITH UR` gilt nur für Operationen mit Lesezugriff. Anderenfalls wird die Anweisung automatisch von `UR` in `CS` geändert.

4. Über CLI oder ODBC zur Laufzeit:

Verwenden Sie den Befehl `CHANGE ISOLATION LEVEL`. Für DB2 Call Level Interface (CLI) können Sie die Isolationsstufe bei der Konfiguration von CLI ändern. Verwenden Sie zur Laufzeit die Funktion `SQLSetConnectAttr` mit dem Attribut `SQL_ATTR_TXN_ISOLATION`, um die Transaktionsisolationsstufe für die aktuelle Verbindung zu definieren, auf die durch `ConnectionHandle` verwiesen wird. Darüber hinaus können Sie das Schlüsselwort `TXNISOLATION` in der Datei `db2cli.ini` verwenden.

5. Über JDBC oder SQLJ zur Laufzeit:

Anmerkung: JDBC und SQLJ sind in DB2 mit CLI implementiert. Dies bedeutet, dass die Einstellungen der Datei `"db2cli.ini"` Einfluss auf die Projekte haben können, die unter Verwendung von JDBC und SQLJ geschrieben und ausgeführt werden.

In SQLJ verwenden Sie die SQLJ-Profilanpassungsfunktion (Befehl `db2sqljcustomize`), um ein Paket zu erstellen. Zu den Optionen, die Sie für dieses Paket angeben können, gehört auch die Isolationsstufe.

6. Für dynamisches SQL in der aktuellen Sitzung:

Verwenden Sie die Anweisung `SET CURRENT ISOLATION`, um die Isolationsstufe für dynamisches SQL festzulegen, die innerhalb einer Sitzung abgesetzt

wird. Wird diese Anweisung abgesetzt, wird das Sonderregister CURRENT ISOLATION auf einen Wert festgelegt, der die Isolationsstufe für alle dynamischen SQL-Anweisungen angibt, die innerhalb der aktuellen Sitzung abgesetzt wurden. Sobald dieser festgelegt wurde, stellt das Sonderregister CURRENT ISOLATION die Isolationsstufe für alle nachfolgenden dynamischen SQL-Anweisungen zur Verfügung, die innerhalb der Sitzung kompiliert wurden, unabhängig davon, welches Paket die Anweisung abgesetzt hat. Diese Isolationsstufe wird angewendet bis die Sitzung beendet wird oder bis eine Anweisung SET CURRENT ISOLATION mit der Option RESET abgesetzt wird.

Zugehörige Konzepte:

- „Aspekte des gemeinsamen Zugriffs“ auf Seite 61
- „Isolation levels“ in *SQL Reference, Volume 1*

Zugehörige Referenzen:

- „CONNECT (Type 1) statement“ in *SQL Reference, Volume 2*
- „SQLSetConnectAttr function (CLI) - Set connection attributes“ in *Call Level Interface Guide and Reference, Volume 2*
- „Statement attributes (CLI) list“ in *Call Level Interface Guide and Reference, Volume 2*

Sperren

Sperren und Steuerung des gemeinsamen Zugriffs

Zur Steuerung des gemeinsamen Zugriffs sowie zur Verhinderung eines unkontrollierten Datenzugriffs richtet der Datenbankmanager Sperren für Pufferpools, Tabellen, Datenpartitionen, Tabellenblöcke oder Tabellenzeilen ein. Eine *Sperre* ordnet eine Ressource des Datenbankmanagers einer Anwendung zu, die als *Sperreneigner* bezeichnet wird, um die Zugriffsmöglichkeiten anderer Anwendungen auf dieselbe Ressource zu steuern.

Der Datenbankmanager verwendet Sperren entweder auf Zeilen- oder auf Tabellenebene, wobei folgende Faktoren die Grundlage für die Auswahl der jeweils geeigneten Sperre bilden:

- Die Isolationsstufe, die bei der Vorkompilierung oder beim Binden einer Anwendung an die Datenbank angegeben wird. Folgende Isolationsstufen sind möglich:
 - Nicht festgeschriebener Lesevorgang (UR)
 - Cursorstabilität (CS)
 - Lesestabilität (RS)
 - Wiederholtes Lesen (RR)

Die unterschiedlichen Isolationsstufen werden zur Steuerung des Zugriffs auf nicht festgeschriebene Daten, zur Verhinderung verlorener Aktualisierungen, zur Ermöglichung nicht wiederholter Lesevorgänge für Daten sowie zur Verhinderung von Phantomzeilen verwendet. Verwenden Sie zur Minimierung des Leistungseinflusses die minimale Isolationsstufe, die den Erfordernissen Ihrer jeweiligen Anwendung entspricht.

- Der vom Optimierungsprogramm ausgewählte Zugriffsplan. Tabellensuchen, Indexsuchen und andere Datenzugriffsmethoden erfordern jeweils verschiedene Arten des Zugriffs auf die Daten.

- Das Attribut LOCKSIZE für die Tabelle. Die Klausel LOCKSIZE in der Anweisung ALTER TABLE gibt die Unterteilung (Granularität) der Sperren an, die beim Zugriff auf die Tabellen verwendet werden. Ausgewählt werden können ROW für Zeilsperren, TABLE für Tabellensperren oder BLOCKINSERT für lediglich Blocksperren in MDC-Tabellen. Wenn die Klausel BLOCKINSERT für eine MDC-Tabelle verwendet wird, wird das Sperren auf Zeilenebene durchgeführt, mit Ausnahme einer INSERT-Operation, bei der stattdessen das Sperren auf Blockebene durchgeführt wird. Verwenden Sie für MDC-Tabellen die Anweisung ALTER TABLE ... LOCKSIZE BLOCKINSERT, wenn Transaktionen umfangreiche Einfügeoperationen in zerlegten Zellen durchführen. Verwenden Sie die Anweisung ALTER TABLE ... LOCKSIZE TABLE für Tabellen, auf die lediglich Lesezugriff besteht. Dadurch wird die Zahl der Sperren verringert, die durch die Datenbankaktivität erforderlich werden. Bei partitionierten Tabellen werden erst Tabellensperren und anschließend Datenpartitionssperren angefordert, was die Daten, auf die zugegriffen wird, vorgeben.
- Der Speicherplatz, der für das Sperren aufgewendet wird. Der Speicherplatz, der für das Sperren aufgewendet wird, wird durch den Konfigurationsparameter locklist (Sperrenliste) der Datenbank gesteuert. Wenn sich die Sperrenliste füllt, kann sich die Leistung aufgrund von Sperreneskalationen und verringertem gemeinsamen Zugriff auf gemeinsam verwendete Objekte in der Datenbank verschlechtern. Wenn Sperreneskalationen häufig stattfinden, erhöhen Sie den Wert des Parameters locklist, des Parameters maxlocks oder die Werte beider Parameter. Stellen Sie des Weiteren sicher, dass zur Reduzierung von gleichzeitigen Sperren Transaktionen häufig festgeschrieben werden, damit die Sperren gelöst werden.

Die meisten Sperren werden für Tabellen erstellt. Beim Erstellen, Ändern oder Löschen von Pufferpools wird jedoch eine Pufferpoolsperre gesetzt. Mit dieser Sperre wird der Modus EXCLUSIVE (X) verwendet. Beim Erfassen von Systemüberwachungsdaten ist es möglich, dass Sie auf diesen Sperrentyp treffen. Beim Anzeigen der Momentaufnahme wird ersichtlich, dass der Name der Sperre der Kennung (ID) des Pufferpools selbst entspricht.

Im Allgemeinen werden Sperren auf Zeilenebene verwendet, sofern nicht einer der folgenden Fälle vorliegt:

- Die ausgewählte Isolationsstufe ist nicht festgeschriebenes Lesen (Uncommitted Read, UR).
- Die ausgewählte Isolationsstufe ist wiederholtes Lesen (Repeatable Read, RR) und der Zugriffsplan erfordert eine Suche ohne Vergleichselemente.
- Das Tabellenattribut ist „TABLE“.
- Die Sperrenliste füllt sich und verursacht Sperreneskalation.
- Über die Anweisung LOCK TABLE erfolgt eine explizite Sperrung einer Tabelle. Die Anweisung LOCK TABLE verhindert, dass gleichzeitig ablaufende Anwendungsprozesse eine Tabelle ändern oder verwenden.

Wenn es sich hierbei um eine MDC-Tabelle handelt, gibt es verschiedene andere Fälle, in denen das Sperren auf Blockebene an Stelle des Sperrens auf Zeilenebene verwendet wird, einschließlich der folgenden Fälle:

- Das Tabellenattribut LOCKSIZE lautet „BLOCKINSERT“.
- Die ausgewählte Isolationsstufe ist wiederholtes Lesen (Repeatable Read, RR) und der Zugriffsplan beinhaltet Vergleichselemente.
- Eine durchsuchte Aktualisierungs- oder Löschoperation, die lediglich Vergleichselemente in Dimensionsspalten umfasst.

Sperreneskalationen verringern den gemeinsamen Zugriff. Bedingungen, die Sperreneskalationen verursachen könnten, sollten vermieden werden.

Die Dauer einer Zeilensperre variiert in Abhängigkeit von der verwendeten Isolationsstufe:

- UR-Suchen: Zeilensperren werden nicht aktiviert, sofern keine Zeilendaten geändert werden.
- CS-Suchen: Zeilensperren bleiben nur für die Zeit aktiviert, die sich der Cursor auf der Zeile befindet.
- RS-Suchen: Sperren bleiben nur für die Dauer der Transaktion für Zeilen aktiviert, die den Suchbedingungen entsprechen.
- RR-Suchen: Alle Zeilensperren bleiben für die Dauer der Transaktion aktiviert.

Zugehörige Konzepte:

- „Verhindern von auf Sperren bezogenen Leistungsproblemen“ auf Seite 82
- „Sperrenattribute“ auf Seite 71
- „Sperrgranularität“ auf Seite 74

Zugehörige Referenzen:

- „Sperrmodi für Tabellen- und Satz-ID-Indexsuchen für MDC-Tabellen“ auf Seite 96
- „Sperrtypenkompatibilität“ auf Seite 84
- „Sperrmodi und Zugriffspfade für Standardtabellen“ auf Seite 87
- „Sperren für Blockindexsuchen für MDC-Tabellen“ auf Seite 101
- „diaglevel - Aufzeichnungsebene bei Fehlerdiagnose “ auf Seite 561
- „locktimeout - Zeitlimit für Sperren “ auf Seite 472
- „locklist - Maximaler Speicher für Sperrenliste “ auf Seite 438
- „dlchktime - Zeitintervall für Prüfung von Deadlocks “ auf Seite 471

Sperrenattribute

Sperren des Datenbankmanagers verfügen über folgende Grundattribute:

Modus

Die Art des Zugriffs, die dem Sperreneigner gewährt wird, sowie die Art des Zugriffs, die Benutzern gewährt wird, die das gesperrte Objekt gleichzeitig verwenden. Dies wird manchmal auch als *Status* der Sperre bezeichnet.

Objekt

Die Ressource, die gesperrt ist. Der einzige Typ von Objekt, den Sie explizit sperren können, ist eine Tabelle. Der Datenbankmanager implementiert Sperren auch für andere Arten von Ressourcen, wie Zeilen, Tabellen und Tabellenbereiche. Für MDC-Tabellen (Tabellen mit mehrdimensionalem Clustering) können außerdem Blocksperren aktiviert werden; für partitionierte Tabellen können Datenpartitionssperren aktiviert werden. Das Objekt, das gesperrt wird, bestimmt die *Granularität* der Sperre.

Dauer Der Zeitraum, für den eine Sperre aktiv ist. Die Isolationsstufe, in der die Abfrage ausgeführt wird, beeinflusst die Sperrdauer.

Die folgende Tabelle zeigt die Modi und ihre Auswirkungen in der Reihenfolge zunehmender Ressourcenbeschränkung. Detaillierte Informationen über Sperren auf verschiedenen Stufen finden Sie in den Referenztabelle zu den Sperrmodi.

Tabelle 4. Zusammenfassung der Sperrmodi

Sperrmodus	Gültiger Objekttyp	Beschreibung
IN (Intent None)	Tabellenbereiche, Blöcke, Tabellen, Datenpartitionen	Der Sperreignier kann alle Daten im Objekt lesen, einschließlich der nicht festgeschriebenen Daten, aber er kann keine Daten aktualisieren. Andere gleichzeitig ausgeführte Anwendungen können die Tabelle lesen oder aktualisieren.
IS (Intent Share)	Tabellenbereiche, Blöcke, Tabellen, Datenpartitionen	Der Sperreignier kann Daten in der gesperrten Tabelle lesen, aber nicht aktualisieren. Andere Anwendungen können die Tabelle lesen oder aktualisieren.
NS (Next Key Share)	Zeilen	Der Sperreignier und jede gleichzeitig ausgeführte Anwendung kann die gesperrten Zeilen lesen, aber nicht aktualisieren. Diese Sperre wird anstatt einer Sperre im Modus S für Zeilen einer Tabelle aktiviert, wenn die Isolationsstufe der Anwendung entweder RS oder CS ist. Der Sperrmodus NS wird nicht zum Sperren der nächsten Zeile (Next-Key locking) verwendet. Er wird anstelle des Modus S bei CS- und RS-Suchen verwendet, um die Auswirkungen des Sperrens der nächsten Zeilen bei diesen Suchen zu minimieren.
S (Share)	Zeilen, Blöcke, Tabellen, Datenpartitionen	Der Sperreignier und jede gleichzeitig ausgeführte Anwendung kann die gesperrten Daten lesen, aber nicht aktualisieren.
IX (Intent Exclusive)	Tabellenbereiche, Blöcke, Tabellen, Datenpartitionen	Der Sperreignier und gleichzeitig ausgeführte Anwendungen können Daten lesen und aktualisieren. Andere gleichzeitig ausgeführte Anwendungen können die Tabelle sowohl lesen als auch aktualisieren.
SIX (Share with Intent Exclusive)	Tabellen, Blöcke, Datenpartitionen	Der Sperreignier kann Daten lesen und aktualisieren. Andere gleichzeitig ausgeführte Anwendungen können die Tabelle lesen.
U (Update)	Zeilen, Blöcke, Tabellen, Datenpartitionen	Der Sperreignier kann Daten aktualisieren. Andere UOWs (UOW = Unit of Work, Arbeitseinheit) können die Daten im gesperrten Objekt lesen, aber nicht versuchen, sie zu aktualisieren.
NW (Next Key Weak Exclusive)	Zeilen	Wenn eine Zeile in einen Index eingefügt wird, wird eine NW-Sperre für die nächste Zeile aktiviert. Für Indizes des Typs 2 findet dies nur statt, wenn die nächste Zeile momentan von einer RR-Suche gesperrt wird. Der Sperreignier kann die gesperrte Zeile lesen, aber nicht aktualisieren. Dieser Sperrmodus ist dem Sperrmodus X ähnlich, ist jedoch auch mit den Sperrmodi W und NS kompatibel.
X (Exclusive)	Zeilen, Blöcke, Tabellen, Pufferpools, Datenpartitionen	Der Sperreignier kann Daten im gesperrten Objekt lesen und auch aktualisieren. Nur Anwendungen mit der Isolationsstufe UR (nicht festgeschriebener Lesevorgang) können auf das gesperrte Objekt zugreifen.
W (Weak Exclusive)	Zeilen	Eine Sperre in diesem Modus wird für die Zeile aktiviert, wenn eine Zeile in eine Tabelle eingefügt wird, für die keine Indizes des Typs 2 definiert sind. Der Sperreignier kann die gesperrte Zeile ändern. Dieser Sperrmodus wird auch während des Einfügens in einen eindeutigen Index verwendet, um festzustellen, ob ein doppelter Wert festgeschrieben wurde, wenn ein doppelter Wert angetroffen wird. Dieser Sperrmodus ist dem Sperrmodus X ähnlich, jedoch ist er mit dem Sperrmodus NW kompatibel. Nur Anwendungen mit der Isolationsstufe UR (nicht festgeschriebener Lesevorgang) können auf die gesperrte Zeile zugreifen.
Z (Super Exclusive)	Tabellenbereiche, Tabellen, Datenpartitionen	Diese Sperre wird für eine Tabelle unter bestimmten Umständen aktiviert, zum Beispiel, wenn die Tabelle geändert (ALTER) oder gelöscht (DROP) wird, ein Index für die Tabelle erstellt oder gelöscht wird oder bestimmte Arten von Tabellenreorganisation durchgeführt werden. Keine anderen gleichzeitig ausgeführten Anwendungen können die Tabelle lesen oder aktualisieren.

Zugehörige Konzepte:

- „Sperren und Steuerung des gemeinsamen Zugriffs“ auf Seite 69
- „Sperrgranularität“ auf Seite 74

Zugehörige Referenzen:

- „maxlocks - Maximale Anzahl Sperren pro Anwendung “ auf Seite 474
- „Sperrtypenkompatibilität“ auf Seite 84
- „Sperrmodi und Zugriffspfade für Standardtabellen“ auf Seite 87

Sperren und Arten der Anwendungsverarbeitung

Zum Zweck der Bestimmung von Sperrenattributen kann die Verarbeitung durch Anwendungen einem der folgenden Typen zugeordnet werden:

- Lesezugriff

Dieser Typ umfasst alle SELECT-Anweisungen, die von sich aus nur Lesezugriff haben, eine explizite Klausel FOR READ ONLY enthalten oder mehrdeutig sind, jedoch vom Abfragecompiler aufgrund der im Befehl PREP oder BIND angegebenen Option BLOCKING als Anweisungen mit Lesezugriff eingestuft werden. Für diesen Verarbeitungstyp sind nur S-Sperren (S, NS oder IS) erforderlich.

- Änderungsabsicht

Dieser Typ umfasst alle SELECT-Anweisungen mit der Klausel FOR UPDATE, der Klausel USE AND KEEP UPDATE LOCKS, der Klausel USE AND KEEP EXCLUSIVE LOCKS bzw. Anweisungen, die der Abfragecompiler als mehrdeutige Anweisung interpretiert, um anzunehmen, dass eine Änderung beabsichtigt wird. Für diesen Typ werden Share- und Update-Sperren (S, U und X für Zeilen; IX, U, X und S für Blöcke; IX, U und X für Tabellen) verwendet.

- Änderung

Dieser Typ umfasst Anweisungen mit UPDATE, INSERT und DELETE, aber keine Anweisungen mit UPDATE WHERE CURRENT OF oder DELETE WHERE CURRENT OF. Für diesen Typ sind exklusive Sperren (X oder IX) erforderlich.

- Cursorgesteuert

Dieser Typ umfasst Anweisungen mit UPDATE WHERE CURRENT OF und DELETE WHERE CURRENT OF. Für diesen Typ sind ebenfalls Sperren des exklusiven Modus (X oder IX) erforderlich.

Eine Anweisung, die an einer Zieltabelle Einfüge-, Aktualisierungs- oder Löschope-rationen (INSERT, UPDATE oder DELETE) auf der Grundlage des Ergebnisses eines Subselects vornimmt, führt zwei Typen der Verarbeitung aus. Die Regeln für Lesezugriffsverarbeitung bestimmen die Sperren für die Tabellen, die in der Anweisung des Subselects zurückgegeben werden. Die Regeln für die Änderungsverarbeitung bestimmen die Sperren für die Zieltabelle.

Zugehörige Konzepte:

- „Deadlocks“ auf Seite 78
- „Verhindern von auf Sperren bezogenen Leistungsproblemen“ auf Seite 82
- „Indextypen und Sperren des nächsten Schlüssels“ auf Seite 91
- „Sperrenattribute“ auf Seite 71
- „Sperren und Steuerung des gemeinsamen Zugriffs“ auf Seite 69
- „Sperren und Datenzugriffsmethoden“ auf Seite 85
- „Sperrgranularität“ auf Seite 74

Zugehörige Tasks:

- „Korrigieren von Problemen der Sperreneskalation“ auf Seite 76

Zugehörige Referenzen:

- „Sperrentypenkompatibilität“ auf Seite 84

Sperrgranularität

Wenn eine Anwendung eine Sperre für ein Datenbankobjekt aktiviert hat, kann eine andere Anwendung auf dieses Objekt vielleicht nicht zugreifen. Aus diesem Grund sind Sperren auf Zeilenebene in Bezug auf einen maximalen gemeinsamen Zugriff besser als Sperren auf Blockebene, Datenpartitionsebene oder Tabellenebene; dadurch wird das gesperrte Datenvolumen, auf das nicht zugegriffen werden kann, minimiert. Jedoch benötigen Sperren Speicherplatz und Verarbeitungszeit, so dass eine einzelne Tabellensperre den Sperrenaufwand minimiert.

Die Klausel LOCKSIZE der Anweisung ALTER TABLE legt den Geltungsbereich (Granularität) von Sperren auf die Zeilen-, Datenpartitions-, Block- oder Tabellenebene fest. Standardmäßig werden Zeilensperren verwendet. Nur Sperren der Modi S (Shared) und X (Exclusive) werden durch diese definierten Tabellensperren angefordert. Die Klausel LOCKSIZE ROW in der Anweisung ALTER TABLE verhindert normale Sperreneskalationen nicht.

Eine permanente Tabellensperre, die durch eine Anweisung ALTER TABLE definiert wird, kann einer Tabellensperre für eine einzelne Transaktion durch die Anweisung LOCK TABLE in folgenden Fällen vorzuziehen sein:

- Die Tabelle ist schreibgeschützt und benötigt immer nur S-Sperren. Andere Benutzer können ebenfalls S-Sperren für die Tabelle aktivieren.
- Auf die Tabelle greifen in der Regel reine Leseanwendungen zu, jedoch greift manchmal ein einzelner Benutzer zu kurzen Pflegemaßnahmen auf sie zu, und dieser Benutzer benötigt eine X-Sperre. Während das Pflegeprogramm ausgeführt wird, werden die reinen Leseanwendungen ausgesperrt. Unter anderen Umständen können reine Leseanwendung jedoch auf die Tabelle bei minimalem Sperrenaufwand gleichzeitig zugreifen.

Bei einer MDC-Tabelle können Sie BLOCKINSERT für die Klausel LOCKSIZE angeben, um das Sperren auf Blockebene nur bei INSERT-Operationen zu verwenden. Wenn die Angabe stattgefunden hat, wird das Sperren auf Zeilenebene für alle anderen Operationen durchgeführt, jedoch in nur geringfügigem Maße für INSERT-Operationen. D. h., das Sperren auf Blockebene wird beim Einfügen von Zeilen verwendet, das Sperren auf Zeilenebene wird jedoch für das Sperren des nächsten Schlüssels verwendet, wenn bei der Aktualisierung in den Indizes RR-Suchen aufgefunden werden. Das Sperren mit BLOCKINSERT kann in den folgenden Fällen nützlich sein:

- Es gibt Mehrfachtransaktionen, bei denen umfangreiche Einfügungen in getrennte Zellen stattfinden.
- Normalerweise werden vom System keine Einfügeoperationen ausgeführt, bei denen mehrere Transaktionen gleichzeitig Daten in eine einzelne Zelle einfügen. Sind derartige Einfügeoperationen doch erforderlich, so werden pro Zelle von jeder der beteiligten Transaktionen so viele Daten eingefügt, dass der Benutzer nicht überprüfen muss, ob die Daten durch die einzelnen Transaktionen in separate Blöcke eingefügt werden.

Die Anweisung ALTER TABLE gibt Sperren global an, wodurch alle Anwendungen und Benutzer, die auf die Tabelle zugreifen, betroffen werden. Einzelne Anwendungen können die Anweisung LOCK TABLE verwenden, um stattdessen Tabellensperren auf Anwendungsebene zu definieren.

Zugehörige Konzepte:

- „Deadlocks“ auf Seite 78
- „Sperren und Steuerung des gemeinsamen Zugriffs“ auf Seite 69

Zugehörige Tasks:

- „Korrigieren von Problemen der Sperreneskulation“ auf Seite 76

Zugehörige Referenzen:

- „diaglevel - Aufzeichnungsebene bei Fehlerdiagnose “ auf Seite 561
- „locktimeout - Zeitlimit für Sperren “ auf Seite 472

Sperrenumwandlung

Die Änderung des Modus einer bereits aktivierten Sperre wird als *Umwandlung* bezeichnet. Die Sperrenumwandlung erfolgt, wenn ein Prozess auf ein Datenobjekt zugreift, für das er bereits eine Sperre aktiviert hat, und der Zugriffsmodus eine noch stärker einschränkende als die aktuelle Sperre erfordert. Für einen Prozess kann immer nur eine Sperre für ein Datenobjekt aktiv sein, obwohl er mehrfach eine Sperre für dasselbe Datenobjekt indirekt durch eine Abfrage anfordern kann.

Einige Sperrmodi gelten nur für Tabellen, andere nur für Zeilen oder Blöcke. Für Zeilen oder Blöcke findet in der Regel eine Umwandlung statt, wenn eine X-Sperre benötigt wird und eine S- oder U-Sperre (Update) zurzeit aktiviert ist.

Sperren der Modi IX (Intent Exclusive) und S (Shared) stellen jedoch in Bezug auf die Sperrenumwandlung einen Sonderfall dar. Weder der Modus S noch der Modus IX wird als stärker einschränkend angesehen als der jeweils andere Modus. Wenn einer dieser Modi aktiv ist und der andere angefordert wird, erfolgt eine Sperrenumwandlung in den Modus SIX (Share with Intent Exclusive). Alle anderen Umwandlungen werden so ausgeführt, dass der angeforderte Sperrmodus zum Modus der aktiven Sperre wird, wenn der angeforderte Modus einen höheren Grad der Einschränkung bewirkt.

Es kann auch eine doppelte Umwandlung stattfinden, wenn eine Abfrage eine Zeile aktualisiert. Wenn die Zeile über einen Indexzugriff gelesen wird und im Modus S gesperrt ist, hat die Tabelle, die diese Zeile enthält, eine abdeckende Intent-Sperre. Wenn der Sperrentyp jedoch IS und nicht IX ist und die Zeile nachfolgend geändert wird, wird die Tabellensperre in eine IX-Sperre und die Zeilensperre in eine X-Sperre umgewandelt.

Eine Sperrenumwandlung findet normalerweise implizit bei der Ausführung einer Abfrage statt. Das Verständnis der verschiedenen Arten von Sperren für verschiedene Abfragen sowie Tabellen- und Indexkombinationen kann beim Entwurf und bei der Optimierung von Anwendungen hilfreich sein.

Die Systemmonitorelemente *lock_current_mode* und *lock_mode* können Informationen zu Sperrenumwandlungen bereitstellen, die in Ihrer Datenbank stattfinden.

Zugehörige Konzepte:

- „Sperrattribute“ auf Seite 71
- „Sperrern und Steuerung des gemeinsamen Zugriffs“ auf Seite 69

Zugehörige Referenzen:

- „lock_current_mode - Original Lock Mode Before Conversion monitor element“ in *Systemmonitor Handbuch und Referenz*
- „lock_mode - Lock Mode monitor element“ in *Systemmonitor Handbuch und Referenz*
- „Sperrtypenkompatibilität“ auf Seite 84

Sperreneskalation

Eine Sperreneskalation findet statt, wenn die Anzahl von Sperren, die für Zeilen und Tabellen in der Datenbank aktiviert sind, dem Prozentsatz der Sperrenliste entspricht, der durch den Datenbankkonfigurationsparameter `maxlocks` definiert ist. Die Sperreneskalation kann für die Tabelle, in der die Sperre aktiviert wird, durch die die Eskalation ausgelöst wird, ohne Auswirkung bleiben. Zur Verringerung der Anzahl von Sperren auf ungefähr die Hälfte der Sperren, die zu Anfang der Sperreneskalation aktiviert waren, beginnt der Datenbankmanager damit, viele kleine Zeilensperren in Tabellensperren für alle aktiven Tabellen umzuwandeln, wobei er mit eventuell vorhandenen Sperren für LOB- oder LONG VARCHAR-Elemente beginnt. Eine *exklusive Sperreneskalation* ist eine Sperreneskalation, bei der die aktivierte Tabellensperre eine exklusive Sperre ist.

Für partitionierte Tabellen erfolgt die Sperreneskalation auf Datenpartitionsebene. Dies bedeutet, dass andere Transaktionen auf die Tabelle zugreifen können. Dies gilt auch dann, wenn für eine Datenpartition eine Eskalation auf gemeinsame (Share) oder exklusive (Exclusive oder Super Exclusive) Sperren auftritt, da andere Datenpartitionen, die nicht eskaliert werden, nicht betroffen sind. Die Transaktion setzt das Aktivieren von Zeilensperren in anderen Datenpartitionen nach der Eskalation für eine bestimmte Datenpartition möglicherweise fort. Bei partitionierten Tabellen enthalten die Nachrichten für das Benachrichtigungsprotokoll für Eskalationen die eskalierte Datenpartition und den Tabellennamen.

Korrigieren von Problemen der Sperreneskalation

Der Datenbankmanager kann Sperren automatisch von Zeilen- oder Blockebene auf Tabellenebene eskalieren (hochstufen). Bei partitionierten Tabellen kann Sperren automatisch von Zeilen- oder Blockebene auf Datenpartitionsebene eskalieren. Der Datenbankkonfigurationsparameter `maxlocks` gibt an, wann die Sperreneskalation ausgelöst wird. Die Tabelle, für die die Sperre aktiviert wird, die die Sperreneskalation auslöst, kann selbst unberührt bleiben. Sperren werden zuerst für die Tabelle mit den meisten Sperren eskaliert, angefangen bei Tabellen, für die LOB- und LONG VARCHAR-Deskriptoren gesperrt sind, dann für die Tabelle mit der nächsthöchsten Anzahl von Sperren usw., bis die Anzahl der aktiven Sperren auf ungefähr die Hälfte des durch `maxlocks` definierten Werts gesenkt wurde.

In einer adäquat entwickelten Datenbank kommt eine Sperreneskalation nur selten vor. Wenn die Sperreneskalation jedoch den gemeinsamen Zugriff auf ein nicht akzeptables Maß reduziert (worauf das Monitorelement `lock_escalation` bzw. der Diagnoseanzeiger `db.lock_escal_rate` hinweist), müssen Sie das Problem analysieren und entscheiden, wie es zu lösen ist.

Voraussetzungen:

Stellen Sie sicher, dass Informationen zur Sperreneskalation aufgezeichnet werden. Setzen Sie den Konfigurationsparameter *notifylevel* (Benachrichtigungsstufe) des Datenbankmanagers auf den Wert 3 (Standardwert) oder 4. Wenn *notifylevel* den Wert 2 hat, wird nur der SQLCODE-Wert des Fehlers protokolliert. Wenn bei der Benachrichtigungsstufe 3 oder 4 eine Sperreneskalation fehlschlägt, werden Informationen zum SQLCODE-Wert des Fehlers sowie zur Tabelle aufgezeichnet, für die die Eskalation fehlgeschlagen ist. Die aktuelle Abfrageanweisung wird nur protokolliert, wenn es sich um eine momentan ausgeführte dynamische Abfrageanweisung handelt und der Parameter *notifylevel* auf den Wert 4 gesetzt ist.

Vorgehensweise:

Befolgen Sie zur Diagnose der Ursache für nicht akzeptable Sperreneskalationen sowie zur Anwendung von Abhilfemaßnahmen die folgenden Schritte:

1. Analysieren Sie das Benachrichtigungsprotokoll für die Systemverwaltung für alle Tabellen, für die Sperren eskaliert werden. Diese Protokolldatei enthält die folgenden Informationen:
 - Die Anzahl der momentan aktiven Sperren.
 - Die Anzahl der bis zum Ende der Sperreneskalation erforderlichen Sperren.
 - Die Tabellenkennung und der Tabellename jeder eskalierten Tabelle.
 - Die Anzahl der momentan aktiven Sperren für andere Objekte als Tabellen.
 - Die neue Sperre auf Tabellenebene, die als Teil der Eskalation aktiviert werden soll. Dabei handelt es sich in der Regel um eine Sperre im Modus „S“ (Share) oder „X“ (eXclusive).
 - Der interne Rückkehrcode für das Ergebnis des Aktivierens der neuen Tabellensperrenebene.
2. Verwenden Sie die Informationen im Benachrichtigungsprotokoll für die Systemverwaltung, um zu entscheiden, wie das Eskalationsproblem zu lösen ist. Ziehen Sie die folgenden Möglichkeiten in Betracht:
 - Erhöhen der Anzahl der global zulässigen Sperren durch Heraufsetzen des Werts des Parameters *maxlocks*, des Parameters *locklist* oder beider Parameter in der Konfigurationsdatei der Datenbank. In einer partitionierten Datenbank führen Sie diese Änderungen in allen Datenbankpartitionen aus.
Sie könnten diese Methode wählen, wenn der gleichzeitige Zugriff auf die Tabelle durch andere Prozesse von höchster Wichtigkeit ist. Jedoch kann der Systemaufwand, der mit der Aktivierung von Sperren auf Datensatzebene verbunden ist, zu größeren Verzögerungen für andere Prozesse führen, als durch den gleichzeitigen Zugriff auf eine Tabelle an Zeit eingespart werden kann.
 - Anpassen des Prozesses bzw. der Prozesse, die die Eskalation verursacht haben. Für diese Prozesse könnten Sie die Anweisung LOCK TABLE explizit absetzen.
 - Ändern der Isolationsstufe. Beachten Sie, dass diese Maßnahme jedoch zu einer Verringerung des gemeinsamen Zugriffs führen kann.
 - Erhöhen der Häufigkeit von Commits zur Verringerung der Anzahl von Sperren, die zu einem gegebenen Zeitpunkt aktiv sind.
 - Verwenden häufiger COMMIT-Anweisungen für Transaktionen, die LONG VARCHAR-Daten und verschiedene Arten von LOB-Daten anfordern. Diese Art von Daten wird zwar erst vom Datenträger gelesen, wenn die Ergebnismenge im Speicher erstellt wird, jedoch wird der Deskriptor bereits beim ers-

ten Verweis auf die Daten gesperrt. Infolgedessen werden möglicherweise wesentlich mehr Sperren aktiviert als für Zeilen, die gängigere Arten von Daten enthalten.

Zugehörige Referenzen:

- „diaglevel - Aufzeichnungsebene bei Fehlerdiagnose “ auf Seite 561
- „maxlocks - Maximale Anzahl Sperren pro Anwendung “ auf Seite 474

Deadlocks

Ein Deadlock wird erstellt, wenn zwei Anwendungen Daten sperren, die die jeweils andere Anwendung benötigt; daraufhin kommt es zu einer Situation, in der weder die eine noch die andere Anwendung weiter ausgeführt werden kann. Beispiel: Im folgenden Diagramm werden zwei Anwendungen gleichzeitig ausgeführt: Anwendung A und Anwendung B. Der erste Schritt von Anwendung A ist die Aktualisierung der ersten Zeile von Tabelle 1; der zweite Schritt ist die Aktualisierung der zweiten Zeile von Tabelle 2. Anwendung B aktualisiert zuerst die zweite Zeile von Tabelle 2 und anschließend die erste Zeile von Tabelle 1. Zu einem bestimmten Zeitpunkt wird in Tabelle 1, Anwendung A der erste Schritt ausgeführt; dabei wird die erste Zeile von Tabelle 1 zur Aktualisierung gesperrt. Gleichzeitig sperrt Anwendung B die zweite Zeile in Tabelle 2, um eine Aktualisierung durchzuführen. In Tabelle 2 versucht Anwendung A den nächsten Schritt auszuführen und fordert eine Sperre für die zweite Zeile in Tabelle 2 für eine Aktualisierung an. Allerdings versucht Anwendung B gleichzeitig, die erste Zeile in Tabelle 1 zu sperren und zu aktualisieren. Da Anwendung A ihre Sperre für die erste Zeile von Tabelle 1 erst entriegeln wird, wenn sie eine Aktualisierung der zweiten Zeile in Tabelle 2 ausführen kann, und Anwendung B ihre Sperre für die zweite Zeile in Tabelle 2 erst entriegeln wird, wenn sie die erste Zeile von Tabelle 1 sperren und aktualisieren kann, tritt ein Deadlock auf. Die Anwendungen können unbegrenzt darauf warten, dass eine der Anwendungen die aktive Sperre für die Daten freigibt.

Deadlock-Konzept

Anwendung A

- T₁: Zeile 1 in Tabelle 1 aktualisieren
- T₂: Zeile 2 in Tabelle 2 aktualisieren
- T₃: Deadlock

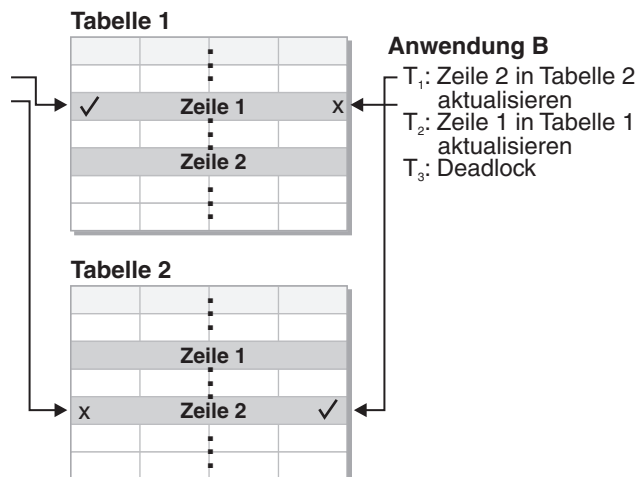


Abbildung 10. Deadlock zwischen Anwendungen

Da Anwendungen Sperren für Daten, die sie benötigen, nicht von sich aus freigeben, ist ein Detektorprozess erforderlich, der Deadlocks auflöst und eine Fortsetzung der Anwendungsverarbeitung ermöglicht. Wie der Name schon sagt, über-

wacht der Deadlock-Detektor die Informationen zu Agenten, die an Sperren warten, die in Intervallen aktiviert werden, die durch den Konfigurationsparameter *dlchktime* angegeben werden.

Wenn der Deadlock-Detektor einen Deadlock ermittelt, wählt er per Zufallsprinzip einen Deadlock-Prozess als *betroffenen Prozess* aus, für den ein Rollback durchgeführt werden muss. Der betroffene Prozess wird aktiviert und gibt den SQLCODE -911 (SQLSTATE 40001) mit Ursachencode 2 an die aufrufende Anwendung zurück. Der Datenbankmanager macht den ausgewählten Prozess automatisch rückgängig (ROLLBACK). Wenn der Rollback beendet ist, werden die Sperren freigegeben, die zu dem betroffenen Prozess gehörten, und die anderen Prozesse, die ebenfalls am Deadlock beteiligt waren, können fortfahren.

Zur Gewährleistung einer guten Leistung müssen Sie ein geeignetes Intervall für den Deadlock-Detektor auswählen. Ein zu kurzes Intervall verursacht einen unnötigen Systemaufwand, ein zu langes Intervall verlängert die Verzögerung eines Prozesses durch einen Deadlock auf ein nicht tolerierbares Maß. Zum Beispiel lässt ein Aktivierungsintervall von fünf Minuten möglicherweise zu, dass ein Deadlock annähernd fünf Minuten bestehen kann, was für eine kurze Transaktionsverarbeitung bereits ein langer Zeitraum sein kann. Es ist wichtig, die möglichen Verzögerungen durch die Auflösung von Deadlocks gegen den Aufwand ihrer Erkennung abzuwägen.

Anmerkungen:

1. In einer Umgebung mit partitionierten Datenbanken wird das durch den Konfigurationsparameter *dlchktime* definierte Intervall nur auf den Katalogknoten angewendet. Werden in einer Umgebung mit partitionierten Datenbanken zahlreiche Deadlocks festgestellt, erhöhen Sie den Wert des Parameters *dlchktime*, um den Wartestatus für Sperren und die Übertragung Rechnung zu tragen.
2. In einer partitionierten Datenbank sendet jede Datenbankpartition *Sperren-diagramme* an die Datenbankpartition, die die Systemkatalogsichten enthält. In dieser Datenbankpartition findet eine umfassende Deadlock-Erkennung fest.

Ein anderes Problem tritt auf, wenn eine Anwendung mit mehr als einem unabhängigen Prozess, der auf die Datenbank zugreift, so strukturiert ist, dass die Entstehung von Deadlocks wahrscheinlich ist. Ein Beispiel wäre eine Anwendung, in der mehrere Prozesse auf dieselbe Tabelle zuerst zu Leseoperationen und anschließend zu Schreiboperationen zugreifen. Wenn die Prozesse schreibgeschützte SQL- oder XQuery-Abfragen durchführen und anschließend SQL-Aktualisierungen für dieselbe Tabelle verarbeiten, steigt die Wahrscheinlichkeit von Deadlocks, weil es zwischen den Prozessen zu Konkurrenzsituationen beim Zugriff auf dieselben Daten kommen kann. Wenn zum Beispiel zwei Prozesse die Tabelle lesen und anschließend aktualisieren, könnte Prozess A versuchen, eine Sperre des Modus X für eine Zeile zu erhalten, für die Prozess B eine Sperre des Modus S hat. Zur Vermeidung solcher Deadlocks sollten Anwendungen, die auf Daten zugreifen, um diese zu ändern, auf eine der folgenden Arten verfahren:

- Verwenden der Klausel FOR UPDATE OF bei der Ausführung einer SELECT-Anweisung. Durch diese Klausel wird sichergestellt, dass eine Sperre des Modus U aktiviert wird, wenn der Prozess A versucht, die Daten zu lesen. Die Zeilenblockung ist inaktiviert.
- Verwenden der Klausel WITH RR USE AND KEEP UPDATE LOCKS oder der Klausel WITH RS USE AND KEEP UPDATE LOCKS bei der Ausführung der

Abfrage. Durch beide Klauseln wird sichergestellt, dass eine Sperre des Modus U aktiviert wird, wenn der Prozess A versucht, die Daten zu lesen, wobei Zeilenblockung zulässig ist.

Bei der Erstellung einer Datenbank wird gleichzeitig ein detaillierter Ereignismonitor für Deadlocks erstellt. Wie bei jedem Monitor fällt für diesen Ereignismonitor etwas Systemaufwand an.

Zur Begrenzung der Größe des Plattenspeicherplatzes, den dieser Ereignismonitor beansprucht, wird der Ereignismonitor inaktiviert und eine Nachricht wird in das Protokoll mit Benachrichtigungen für die Systemverwaltung geschrieben, wenn die maximale Anzahl von Ausgabedateien erreicht ist. Durch das Entfernen der Ausgabedateien, die nicht mehr benötigt werden, kann der Ereignismonitor bei der nächsten Datenbankaktivierung erneut aktiviert werden.

Wenn Sie den detaillierten Ereignismonitor für Deadlocks entfernen möchten, kann er mit dem folgenden Befehl gelöscht werden:

```
DROP EVENT MONITOR db2detaildeadlock
```

In einer föderierten Systemumgebung, in der eine Anwendung auf Kurznamen zugreift, ist es möglich, dass die von der Anwendung angeforderten Daten aufgrund eines Deadlocks an einer Datenquelle nicht verfügbar sind. In diesem Fall ist DB2 von den Einrichtungen zur Behandlung von Deadlocks an der Datenquelle abhängig. Wenn Deadlocks in mehreren Datenquellen auftreten, ist DB2 für die Auflösung der Deadlocks auf die Zeitlimitmechanismen der Datenquellen angewiesen.

Wenn Sie mehr Informationen über Deadlocks protokollieren wollen, setzen Sie den Konfigurationsparameter *diaglevel* des Datenbankmanagers auf den Wert 4. Die protokollierten Informationen geben das gesperrte Objekt, den Sperrmodus und die Anwendung an, die die Sperre aktiviert hat. Die aktuellen dynamischen SQL- und XQuery-Anweisung bzw. die Namen der statischen Pakete können ebenfalls protokolliert werden. Dynamische SQL- und XQuery-Anweisungen werden nur protokolliert, wenn *diaglevel* gleich vier ist.

Zugehörige Konzepte:

- „Übersicht über die Architektur und Prozesse von DB2“ auf Seite 155
- „Sperrgranularität“ auf Seite 74

Wartestatus und Zeitlimitüberschreitungen für Sperren

Die Erkennung von Zeitlimitüberschreitungen für Sperren ist eine Datenbankmanagerfunktion, die verhindert, dass Anwendungen unendlich lange auf die Freigabe einer Sperre in einer abnormalen Situation warten. Zum Beispiel könnte eine Transaktion auf eine Sperre warten, die von der Anwendung eines anderen Benutzers aktiviert wurde. Der andere Benutzer hat jedoch seine Workstation verlassen, ohne seine Anwendung die Transaktion festschreiben zu lassen, wodurch die Sperre freigegeben würde. Um die Blockierung einer Anwendung in einem solchen Fall zu vermeiden, setzen Sie den Konfigurationsparameter *locktimeout* auf die maximale Zeitdauer, die eine beliebige Anwendung auf eine Sperre warten sollte.

Durch die Einstellung dieses Parameters können globale Deadlocks besser vermieden werden, insbesondere in Anwendungen mit verteilten Arbeitseinheiten (DUOW). Wenn die Zeit, die eine Sperrenanforderung ansteht, länger ist als die im Wert des Parameters *locktimeout* definierte Zeit, empfängt die anfordernde Anwen-

dung einen Fehler und ihre Transaktion wird mit ROLLBACK rückgängig gemacht. Wenn zum Beispiel *Programm 1* versucht, eine Sperre zu erhalten, die bereits für *Programm 2* aktiv ist, liefert *Programm 1* den SQLCODE -911 (SQLSTATE 40001) mit dem Ursachencode 68 zurück, wenn das Zeitlimit abläuft. Der Standardwert für den Parameter *locktimeout* ist -1, d. h. die Erkennung der Überschreitung des Sperrzeitlimits ist ausgeschaltet.

Anmerkung: Für Tabellen-, Zeilen-, Datenpartitions- und MDC-Blocksperrungen kann eine Anwendung die Einstellung für *locktimeout* auf der Datenbankebene mit der Anweisung SET CURRENT LOCK TIMEOUT überschreiben.

Wenn Sie mehr Informationen zu Zeitlimitüberschreitungen bei Sperrenanforderungen im Protokoll 'db2diag.log' protokollieren wollen, setzen Sie den Konfigurationsparameter *diaglevel* des Datenbankmanagers auf den Wert 4. Die protokollierten Informationen geben das gesperrte Objekt, den Sperrmodus und die Anwendung an, die die Sperre aktiviert hat. Der Name der aktuellen dynamischen SQL- oder XQuery-Anweisung oder des statischen Pakets kann ebenfalls protokolliert werden. Eine dynamische SQL- oder XQuery-Anweisung wird nur protokolliert, wenn *diaglevel* gleich vier ist.

Informationen über Wartestatus für Sperren und Zeitlimitüberschreitungen durch Sperren können Sie mit Hilfe der Systemmonitorelemente zum Wartestatus von Sperren ('lock_wait') oder über den Diagnoseanzeiger 'db.apps_waiting_locks' abrufen.

Zugehörige Konzepte:

- „Angabe einer Strategie für den Modus 'Wartestatus für Sperre'“ auf Seite 81

Zugehörige Referenzen:

- „locktimeout - Zeitlimit für Sperren “ auf Seite 472

Angabe einer Strategie für den Modus 'Wartestatus für Sperre'

Eine einzelne Sitzung kann nun eine Strategie für den Modus 'Wartestatus für Sperre' angeben, die verwendet wird, wenn für die Sitzung eine Sperre erforderlich ist, die nicht unmittelbar abgerufen werden kann. Die Strategie gibt an, ob die Sitzung folgende Bedingungen erfüllt:

- Zurückgeben von SQLCODE und SQLSTATE, wenn keine Sperre abgerufen werden kann
- Andauerndes Warten auf eine Sperre
- Warten auf eine Sperre über einen angegebenen Zeitraum
- Verwenden des Werts für den Datenbankkonfigurationsparameter *locktimeout* beim Warten auf eine Sperre

Die Strategie für den Modus 'Wartestatus für Sperre' wird über die neue Anweisung SET CURRENT LOCK TIMEOUT angegeben, durch die der Wert des Sonderregisters CURRENT LOCK TIMEOUT geändert wird. Das Sonderregister CURRENT LOCK TIMEOUT gibt die Anzahl an Sekunden an, die auf eine Sperre gewartet werden soll, bevor ein Fehler zurückgegeben wird, der darauf hinweist, dass keine Sperre abgerufen werden kann.

Traditionelle Sperrmethoden können Anwendungen zur Folge haben, die sich gegenseitig blockieren. Dies geschieht dann, wenn eine Anwendung auf eine andere Anwendung warten muss, um die Sperre freizugeben. Strategien zum

Umgang mit Auswirkungen auf solche Blockungen stellen in der Regel einen Mechanismus bereit, um die annehmbare Höchstdauer des Blocks anzugeben. Dies ist die Menge an Zeit, die eine Anwendung vor der Rückgabe ohne Sperre wartet. Vorher war dies nur auf der Datenbankebene durch Ändern des Werts des Datenbankkonfigurationsparameters *locktimeout* möglich.

Der Wert des Parameters *locktimeout* gilt für alle Sperren; die Sperrtypen, die diese neue Funktion enthält, umfassen Blocksperren für Zeilen, Tabellen, Indexschlüssel und MDC.

Zugehörige Referenzen:

- „locktimeout - Zeitlimit für Sperren “ auf Seite 472

Verhindern von auf Sperren bezogenen Leistungsproblemen

Berücksichtigen Sie die folgenden Richtlinien, wenn Sie das Sperren im Hinblick auf den gemeinsamen Zugriff und die Datenintegrität optimieren:

- Erstellen Sie kleine UOWs (UOW = Unit of Work, Arbeitseinheit) mit häufigen Anweisungen COMMIT, um den gemeinsamen Zugriff auf Daten durch zahlreiche Benutzer zu fördern.

Nehmen Sie Anweisungen COMMIT in die Anwendung an den Stellen auf, an denen sich die Anwendung logisch in einem Konsistenzzustand befindet, das heißt, wenn die geänderten Daten konsistent sind. Wenn eine Anweisung COMMIT abgesetzt wird, werden Sperren freigegeben, abgesehen von Tabellsperren, die mit der Klausel WITH HOLD deklarierten Cursors zugeordnet sind.

- Geben Sie eine geeignete Isolationsstufe an.

Sperren werden aktiviert, auch wenn die Anwendung Zeilen lediglich liest, so dass auch UOWs im Lesezugriff mit COMMIT festgeschrieben werden müssen. Der Grund dafür ist der, dass in den Isolationsstufen RR, RS und CS in Anwendungen mit Lesezugriff gemeinsame Sperren (Shared) aktiviert werden. Bei den Isolationsstufen RR und RS werden alle Sperren beibehalten, bis eine Anweisung COMMIT ausgeführt wird, so dass keine anderen Prozesse die gesperrten Daten aktualisieren können, es sei denn, Sie schließen den Cursor mit der Klausel WITH RELEASE. Außerdem werden auch Katalogsperren in Anwendungen mit dynamischen SQL- oder XQuery-Anweisungen unter der Isolationsstufe UR aktiviert.

Der Datenbankmanager stellt sicher, dass die Anwendung keine noch nicht festgeschriebenen Daten (Zeilen, die von anderen Anwendungen aktualisiert, aber noch nicht mit der Anweisung COMMIT festgeschrieben wurden) abrufen, sofern Sie nicht mit der Isolationsstufe UR (Uncommitted Read) arbeiten.

- Verwenden Sie die Anweisung LOCK TABLE in geeigneter Weise.

Diese Anweisung sperrt eine ganze Tabelle. Es wird nur die in der Anweisung LOCK TABLE angegebene Tabelle gesperrt. Übergeordnete und abhängige Tabellen der angegebenen Tabelle werden nicht gesperrt. Sie müssen selbst entscheiden, ob das Sperren anderer Tabellen, auf die möglicherweise zugegriffen wird, erforderlich ist, um das gewünschte Ergebnis hinsichtlich des gemeinsamen Zugriffs und der Leistung zu erzielen. Die Sperre wird erst freigegeben, wenn für die UOW ein Commit oder ein Rollback durchgeführt wurde.

LOCK TABLE IN SHARE MODE

Sie wollen auf Daten zugreifen, die *zeitlich konsistent* sind, d. h. Daten, die für eine Tabelle zu einem bestimmten Zeitpunkt aktuell sind. Wenn auf die Tabelle häufig zugegriffen wird, liegt die einzige Möglichkeit, die Tabelle in einem stabilen Zustand zu erhalten, darin, die Tabelle zu sperren. Die Anwendung soll zum Beispiel eine Momentaufnahme der Tabelle machen. Während der Zeit jedoch, die die Anwendung zur Verarbeitung einiger Zeilen der Tabelle benötigt, aktualisieren andere Anwendungen Zeilen, die die Anwendung noch nicht verarbeitet hat. Dies ist unter der Isolationsstufe RR (Wiederholtes Lesen) zulässig, jedoch nicht beabsichtigt.

Als alternative Methode kann Ihre Anwendung die Anweisung LOCK TABLE IN SHARE MODE absetzen: Es können keine Zeilen geändert werden, egal ob sie von Ihnen abgerufen wurden oder nicht. Anschließend kann die Anwendung die benötigte Anzahl von Zeilen abrufen und gleichzeitig davon ausgehen, dass die abgerufenen Zeilen nicht kurz vor dem Abrufen geändert wurden.

Nach Ausführung der Anweisung LOCK TABLE IN SHARE MODE können andere Benutzer Daten aus der Tabelle abrufen, jedoch keine Zeilen in der Tabelle aktualisieren, löschen oder einfügen.

LOCK TABLE IN EXCLUSIVE MODE

Sie beabsichtigen, einen großen Teil der Tabelle zu aktualisieren. Es ist weniger aufwendig und daher effektiver, alle anderen Benutzer vom Zugriff auf die Tabelle auszuschließen, als jede Zeile zur Aktualisierung zu sperren und anschließend die Sperren wieder freizugeben, wenn alle Änderungen festgeschrieben sind.

Durch LOCK TABLE IN EXCLUSIVE MODE werden alle anderen Benutzer ausgeschlossen, d. h. keine anderen Anwendungen können auf die Tabelle zugreifen, außer wenn es sich um Anwendungen mit der Isolationsstufe UR (nicht festgeschriebener Lesevorgang) handelt.

- Verwenden Sie Anweisungen ALTER TABLE in Anwendungen.

Die Anweisung ALTER TABLE mit dem Parameter LOCKSIZE stellt eine Alternative zur Anweisung LOCK TABLE dar. Über den Parameter LOCKSIZE können Sie eine Sperrgranularität für den nächsten Tabellenzugriff angeben. Gültige Werte sind ROW für Zeilensperre oder TABLE für Tabellensperre. Für MDC-Tabellen können Sie eine Sperrgranularität der Klausel BLOCKINSERT angeben.

Die Auswahl von Zeilensperren (ROW) entspricht der Auswahl der Standardsperrgranularität bei der Erstellung einer Tabelle. Die Auswahl von Tabellensperren (TABLE) verbessert eventuell die Leistung von Abfragen durch die dadurch verringerte Anzahl benötigter Sperren. Allerdings könnte der gemeinsame Zugriff eingeschränkt werden, weil alle Sperren für die gesamte Tabelle aktiviert werden. Bei MDC-Tabellen kann die Auswahl der Klausel BLOCKINSERT die Leistung von INSERT-Operationen durch Sperren auf Blockebene und Vermeiden von Zeilensperren für Einfügungen verbessern. Sperren auf Zeilenebene werden immer noch für alle anderen Operationen durchgeführt; die Durchführung findet für Einfügungen von Schlüsseln statt, um RR-Scanner zu schützen. Die Option BLOCKINSERT ist für große Einfügungen in Zellen durch einzelne Transaktionen nützlich. Keine der LOCKSIZE-Auswahlmöglichkeiten verhindert eine normale Sperreneskulation.

- Schließen Sie Cursor, um die von ihnen aktivierten Sperren freizugeben.
Wenn Sie einen Cursor mit der Anweisung `CLOSE CURSOR` schließen, die die Klausel `WITH RELEASE` enthält, versucht der Datenbankmanager alle vorhandenen Lesesperren freizugeben, die für diesen Cursor aktiviert sind. Tabellenlesesperren sind IS-, S- und U-Tabellensperren. Zeilenlesesperren sind S-, NS- und U-Zeilensperren. Blocklesesperren sind IS-, S- und U-Blocksperrern.
Die Klausel `WITH RELEASE` hat keine Auswirkung auf Cursor, die unter den Isolationsstufen CS oder UR verwendet werden. Wenn die Klausel `WITH RELEASE` für Cursor angegeben wird, die unter der Isolationsstufe RS oder RR verwendet werden, hebt sie einige Merkmale dieser Isolationsstufen auf. Dies heißt im Einzelnen, dass beim einem RS-Cursor *nichtwiederholbare Lesevorgänge* und bei einem RR-Cursor *nichtwiederholbare Lesevorgänge* oder *Phantomzeilen* auftreten können.
Wird ein Cursor, der ursprünglich ein RR- oder RS-Cursor ist, nachdem er mit der Klausel `WITH RELEASE` geschlossen wurde, erneut geöffnet, werden neue Lesesperren aktiviert.
In CLI-Anwendungen kann das DB2-CLI-Verbindungsattribut `SQL_ATTR_CLOSE_BEHAVIOR` verwendet werden, um die gleichen Ergebnisse wie mit `CLOSE CURSOR WITH RELEASE` zu erzielen.
- Wenn Sie in einer Umgebung mit partitionierten Datenbanken die Konfigurationsparameter ändern, die sich auf Sperren auswirken, stellen Sie sicher, dass die Änderungen in allen Datenbankpartitionen durchgeführt werden.

Zugehörige Konzepte:

- „Sperren und Steuerung des gemeinsamen Zugriffs“ auf Seite 69
- „Sperrenattribute“ auf Seite 71
- „Sperrgranularität“ auf Seite 74

Sperrentypenkompatibilität

Die Sperrenkompatibilität wird zu einem Problem, wenn eine Anwendung derzeit eine Sperre über einem Objekt verhängt hat und eine weitere Anwendung eine Sperre für dasselbe Objekt anfordert. Wenn die beiden Sperrmodi kompatibel sind, ist die Anforderung für eine zweite Sperre für das Objekt zulässig.

Wenn der Sperrmodus der angeforderten Sperre nicht mit der bereits aktiven Sperre kompatibel ist, ist die Sperrenanforderung nicht zulässig. Stattdessen muss die Anforderung warten, bis die erste Anwendung die Sperre freigegeben hat und alle weiteren bestehenden inkompatiblen Sperren freigegeben wurden.

Die folgende Tabelle enthält Informationen zu den Umständen, in denen eine Sperrenanforderungen erfüllt werden kann, wenn ein anderer Prozess eine Sperre für dieselbe Ressource in einem bestimmten Status aktiviert oder angefordert hat. Ein **Nein** bedeutet, dass der Anforderer warten muss, bis alle inkompatiblen Sperren von anderen Prozessen inaktiviert werden. Beachten Sie, dass es zu einer Zeitlimitüberschreitung kommen kann, wenn ein Anforderer auf eine Sperre wartet. Ein **Ja** bedeutet, dass die Sperre aktiviert wird, sofern kein früherer Anforderer auf die Ressource wartet.

Tabelle 5. Kompatibilität der Sperrmodi

Angeforderter Status	Status der aktiven Sperre											
	NONE	IN	IS	NS	S	IX	SIX	U	X	Z	NW	W
NONE	Ja	Ja	Ja	Ja	Ja	Ja	Ja	Ja	Ja	Ja	Ja	Ja
IN	Ja	Ja	Ja	Ja	Ja	Ja	Ja	Ja	Ja	Nein	Ja	Ja
IS	Ja	Ja	Ja	Ja	Ja	Ja	Ja	Ja	Nein	Nein	Nein	Nein
NS	Ja	Ja	Ja	Ja	Ja	Nein	Nein	Ja	Nein	Nein	Ja	Nein
S	Ja	Ja	Ja	Ja	Ja	Nein	Nein	Ja	Nein	Nein	Nein	Nein
IX	Ja	Ja	Ja	Nein	Nein	Ja	Nein	Nein	Nein	Nein	Nein	Nein
SIX	Ja	Ja	Ja	Nein	Nein	Nein	Nein	Nein	Nein	Nein	Nein	Nein
U	Ja	Ja	Ja	Ja	Ja	Nein	Nein	Nein	Nein	Nein	Nein	Nein
X	Ja	Ja	Nein	Nein	Nein	Nein	Nein	Nein	Nein	Nein	Nein	Nein
Z	Ja	Nein	Nein	Nein	Nein	Nein	Nein	Nein	Nein	Nein	Nein	Nein
NW	Ja	Ja	Nein	Ja	Nein	Nein	Nein	Nein	Nein	Nein	Nein	Ja
W	Ja	Ja	Nein	Nein	Nein	Nein	Nein	Nein	Nein	Nein	Ja	Nein

Anmerkung:

I	Intent
N	NONE
NS	Next Key Share
S	Share
X	Exclusive
U	Update
Z	Super Exclusive
NW	Next Key Weak Exclusive
W	Weak Exclusive

Anmerkung:

- Ja - angeforderte Sperre sofort **erteilen**
- Nein - **warten**, bis aktive Sperre freigegeben oder Zeitlimit überschritten wird

Zugehörige Konzepte:

- „Sperrattribute“ auf Seite 71
- „Sperrungen und Steuerung des gemeinsamen Zugriffs“ auf Seite 69
- „Sperrgranularität“ auf Seite 74

Zugehörige Referenzen:

- „Sperrmodi und Zugriffspfade für Standardtabellen“ auf Seite 87
- „Sperrungen für Blockindexsuchen für MDC-Tabellen“ auf Seite 101

Sperrungen und Datenzugriffsmethoden

Ein *Zugriffsplan* ist die Methode, die das Optimierungsprogramm zum Abrufen von Daten aus einer bestimmten Tabelle auswählt. Der Zugriffsplan kann erhebliche Auswirkungen auf Sperrmodi haben. Wenn zum Beispiel eine Indexsuche zum Auffinden einer bestimmten Zeile verwendet wird, wählt das Optimierungsprogramm wahrscheinlich Sperren auf Zeilenebene (IS) für die Tabelle aus. Wenn die Tabelle EMPLOYEE zum Beispiel einen Index für die Spalte EMPNO hat, könnte ein Zugriff über einen Index ausgeführt werden, um Informationen zu einem einzelnen Mitarbeiter mit Hilfe einer Anweisung auszuwählen, die die

folgende SELECT-Klausel enthält:

```
SELECT *  
FROM EMPLOYEE  
WHERE EMPNO = '000310';
```

Wenn kein Index verwendet wird, muss die gesamte Tabelle der Reihe nach durchsucht werden, um die ausgewählten Zeilen zu finden. Dafür ist möglicherweise nur eine Sperre auf Tabellenebene (S) erforderlich. Wenn zum Beispiel kein Index für die Spalte SEX vorhanden ist, könnte eine Tabellensuche verwendet werden, um alle männlichen Mitarbeiter mit einer Anweisung auszuwählen, die folgende SELECT-Klausel enthält:

```
SELECT *  
FROM EMPLOYEE  
WHERE SEX = 'M';
```

Anmerkung: Bei der cursorgesteuerten Verarbeitung wird der Sperrmodus des zugrunde liegenden Cursors verwendet, bis die Anwendung eine Zeile findet, die zu aktualisieren oder zu löschen ist. Für diesen Verarbeitungstyp wird unabhängig vom Sperrmodus eines Cursors immer eine Sperre des Modus Exclusive aktiviert, um die Aktualisierung oder Löschung durchzuführen.

Das Sperren in Bereichsclustertabellen funktioniert etwas anders als beim Sperren von Standardschlüsseln und nächsten Schlüsseln. Beim Zugriff auf einen Bereich von Zeilen in einer Bereichsclustertabelle werden alle Zeilen in diesem Bereich gesperrt, selbst wenn einige dieser Zeilen leer sind. Beim Sperren von Standardschlüsseln oder nächsten Schlüsseln werden nur Zeilen mit vorhandenen Datensätzen gesperrt.

Referenztabellen enthalten detaillierte Informationen darüber, welche Sperren für welche Art von Zugriffsplan aktiviert werden.

Beim verzögerten Zugriff auf die Datenseiten erfolgt der Zugriff auf die Zeile in zwei Schritten, wodurch komplexere Sperrsituationen auftreten. Die Zeitpunkte der Aktivierung von Sperren und die Dauer der Sperren hängen von der Isolationsstufe ab. Da bei der Isolationsstufe RR (*Wiederholtes Lesen*) alle Sperren bis zum Ende der Transaktion beibehalten werden, bleiben die im ersten Schritt aktivierten Sperren bestehen, und im zweiten Schritt müssen keine weiteren Sperren aktiviert werden. Für die Isolationsstufen Lesestabilität und Cursorstabilität müssen Sperren im zweiten Schritt aktiviert werden. Um den gemeinsamen Zugriff zu maximieren, werden im ersten Schritt keine Sperren aktiviert und alle Vergleichselemente erneut angewandt, so dass nur den Auswahlbedingungen entsprechende Zeilen zurückgegeben werden.

Zugehörige Konzepte:

- „Verhindern von auf Sperren bezogenen Leistungsproblemen“ auf Seite 82
- „Indextypen und Sperren des nächsten Schlüssels“ auf Seite 91
- „Sperrenattribute“ auf Seite 71
- „Sperren und Steuerung des gemeinsamen Zugriffs“ auf Seite 69
- „Sperrgranularität“ auf Seite 74
- „Sperren und Arten der Anwendungsverarbeitung“ auf Seite 73

Zugehörige Tasks:

- „Korrigieren von Problemen der Sperreneskalation“ auf Seite 76

Zugehörige Referenzen:

- „Sperrmodi und Zugriffspfade für Standardtabellen“ auf Seite 87
- „Sperrmodi für Tabellen- und Satz-ID-Indexsuchen für MDC-Tabellen“ auf Seite 96
- „Sperrtypenkompatibilität“ auf Seite 84
- „Sperrmodi für Blockindexsuchen für MDC-Tabellen“ auf Seite 101

Sperrmodi und Zugriffspfade für Standardtabellen

Dieser Abschnitt enthält Referenzinformationen zu Sperrmethoden für Standardtabellen für verschiedene Datenzugriffspläne.

In den folgenden Tabellen werden die Typen von Sperren aufgelistet, die für Standardtabellen in der jeweiligen Isolationsstufe für verschiedene Zugriffspläne aktiviert werden. Jeder Eintrag besteht aus zwei Teilen: Tabellensperre und Zeilensperre. Ein Strich zeigt an, dass für eine bestimmte Stufe keine Sperrung erfolgt.

Anmerkungen:

1. In einer MDC-Umgebung (Multidimensional Clustering) wird eine zusätzliche Sperrstufe verwendet: der Block.
2. Sperrmodi können explizit mit einer Sperranforderungsklausel (LOCK REQUEST) in einer SELECT-Anweisung geändert werden.

Tabelle 6. Sperrmodi für Tabellensuchen ohne Vergleichselemente

Isolationsstufe	Suchen: mehrdeutig und Lesezugriff	Cursorgesteuerte Operationen		UPDATE oder DELETE mit Suche	
		Suche	WHERE CURRENT OF	Suche	UPDATE oder DELETE
RR	S/-	U/-	SIX/X	X/-	X/-
RS	IS/NS	IX/U	IX/X	IX/X	IX/X
CS	IS/NS	IX/U	IX/X	IX/X	IX/X
UR	IN/-	IX/U	IX/X	IX/X	IX/X

Tabelle 7. Sperrmodi für Tabellensuchen mit Vergleichselementen

Isolationsstufe	Suchen: mehrdeutig und Lesezugriff	Cursorgesteuerte Operationen		UPDATE oder DELETE mit Suche	
		Suche	WHERE CURRENT OF	Suche	UPDATE oder DELETE
RR	S/-	U/-	SIX/X	U/-	SIX/X
RS	IS/NS	IX/U	IX/X	IX/U	IX/X
CS	IS/NS	IX/U	IX/X	IX/U	IX/X
UR	IN/-	IX/U	IX/X	IX/U	IX/X

Anmerkung: Wenn in der Isolationsstufe UR mit IN-Sperren für Indizes des Typs 1 Vergleichselemente für INCLUDE-Spalten im Index angewendet werden, werden die Isolationsstufe auf CS erhöht und die Sperren in eine IS-Tabellensperre und in NS-Zeilensperren umgewandelt.

Tabelle 8. Sperrmodi für Satz-ID-Indexsuchen (RID) ohne Vergleichselemente

Isolations- stufe	Suchen: mehrdeu- tig und Lesezu- griff	Cursorgesteuerte Operationen		UPDATE oder DELETE mit Suche	
		Suche	WHERE CURRENT OF	Suche	UPDATE oder DELETE
RR	S/-	IX/S	IX/X	X/-	X/-
RS	IS/NS	IX/U	IX/X	IX/X	IX/X
CS	IS/NS	IX/U	IX/X	IX/X	IX/X
UR	IN/-	IX/U	IX/X	IX/X	IX/X

Tabelle 9. Sperrmodi für Satz-ID-Indexsuchen (RID) mit einer einzigen Qualifikationszeile

Isolations- stufe	Suchen: mehrdeu- tig und Lesezu- griff	Cursorgesteuerte Operationen		UPDATE oder DELETE mit Suche	
		Suche	WHERE CURRENT OF	Suche	UPDATE oder DELETE
RR	IS/S	IX/U	IX/X	IX/X	IX/X
RS	IS/NS	IX/U	IX/X	IX/X	IX/X
CS	IS/NS	IX/U	IX/X	IX/X	IX/X
UR	IN/-	IX/U	IX/X	IX/X	IX/X

Tabelle 10. Sperrmodi für Satz-ID-Indexsuchen (RID) nur mit Start- und Stopppvergleichs-
elementen

Isolations- stufe	Suchen: mehrdeu- tig und Lesezu- griff	Cursorgesteuerte Operationen		UPDATE oder DELETE mit Suche	
		Suche	WHERE CURRENT OF	Suche	UPDATE oder DELETE
RR	IS/S	IX/S	IX/X	IX/X	IX/X
RS	IS/NS	IX/U	IX/X	IX/X	IX/X
CS	IS/NS	IX/U	IX/X	IX/X	IX/X
UR	IN/-	IX/U	IX/X	IX/X	IX/X

Tabelle 11. Sperrmodi für Satz-ID-Indexsuchen (RID) nur mit Index und anderen Vergleichs-
elementen (sargs, resids)

Isolations- stufe	Suchen: mehrdeu- tig und Lesezu- griff	Cursorgesteuerte Operationen		UPDATE oder DELETE mit Suche	
		Suche	WHERE CURRENT OF	Suche	UPDATE oder DELETE
RR	IS/S	IX/S	IX/X	IX/S	IX/X
RS	IS/NS	IX/U	IX/X	IX/U	IX/X
CS	IS/NS	IX/U	IX/X	IX/U	IX/X
UR	IN/-	IX/U	IX/X	IX/U	IX/X

Die folgenden Tabellen zeigen die Sperrmodi für Fälle, in denen der Lesezugriff auf die Datenseiten verzögert erfolgt, um folgende Operationen an der Zeilenliste durchführen zu können:

- Eine weitere Qualifizierung mit Hilfe mehrerer Indizes
- Eine Sortierung für einen effizienten Vorablesezugriff

Tabelle 12. Sperrmodi für Indexsuchen mit verzögertem Zugriff auf Datenseiten: Satz-ID-Indexsuche ohne Vergleichselemente

Isolations- stufe	Suchen: mehrdeu- tig und Lesezu- griff	Cursorgesteuerte Operationen		UPDATE oder DELETE mit Suche	
		Suche	WHERE CURRENT OF	Suche	UPDATE oder DELETE
RR	IS/S	IX/S		X/-	
RS	IN/-	IN/-		IN/-	
CS	IN/-	IN/-		IN/-	
UR	IN/-	IN/-		IN/-	

Tabelle 13. Sperrmodi für Indexsuchen mit verzögertem Zugriff auf Datenseiten: nach einer Satz-ID-Indexsuche ohne Vergleichselemente

Isolations- stufe	Suchen: mehrdeu- tig und Lesezu- griff	Cursorgesteuerte Operationen		UPDATE oder DELETE mit Suche	
		Suche	WHERE CURRENT OF	Suche	UPDATE oder DELETE
RR	IN/-	IX/S	IX/X	X/-	X/-
RS	IS/NS	IX/U	IX/X	IX/X	IX/X
CS	IS/NS	IX/U	IX/X	IX/X	IX/X
UR	IN/-	IX/U	IX/X	IX/X	IX/X

Tabelle 14. Sperrmodi für Indexsuchen mit verzögertem Zugriff auf Datenseiten: Satz-ID-Indexsuche mit Vergleichselementen (sargs, resids)

Isolations- stufe	Suchen: mehrdeu- tig und Lesezu- griff	Cursorgesteuerte Operationen		UPDATE oder DELETE mit Suche	
		Suche	WHERE CURRENT OF	Suche	UPDATE oder DELETE
RR	IS/S	IX/S		IX/S	
RS	IN/-	IN/-		IN/-	
CS	IN/-	IN/-		IN/-	
UR	IN/-	IN/-		IN/-	

Tabelle 15. Sperrmodi für Indexsuchen mit verzögertem Zugriff auf Datenseiten: Satz-ID-Indexsuche nur mit Start- und Stoppvergleichselementen

Isolationsstufe	Suchen: mehrdeutig und Lesezugriff	Cursorgesteuerte Operationen		UPDATE oder DELETE mit Suche	
		Suche	WHERE CURRENT OF	Suche	UPDATE oder DELETE
RR	IS/S	IX/S		IX/X	
RS	IN/-	IN/-		IN/-	
CS	IN/-	IN/-		IN/-	
UR	IN/-	IN/-		IN/-	

Tabelle 16. Sperrmodi für Indexsuchen mit verzögertem Zugriff auf Datenseiten: Satz-ID-Indexsuche nur mit Start- und Stoppvergleichselementen

Isolationsstufe	Suchen: mehrdeutig und Lesezugriff	Cursorgesteuerte Operationen		UPDATE oder DELETE mit Suche	
		Suche	WHERE CURRENT OF	Suche	UPDATE oder DELETE
RR	IN/-	IX/S	IX/X	IX/X	IX/X
RS	IS/NS	IX/U	IX/X	IX/U	IX/X
CS	IS/NS	IX/U	IX/X	IX/U	IX/X
UR	IS/-	IX/U	IX/X	IX/U	IX/X

Tabelle 17. Sperrmodi für Indexsuchen mit verzögertem Zugriff auf Datenseiten: nach einer Satz-ID-Indexsuche mit Vergleichselementen

Isolationsstufe	Suchen: mehrdeutig und Lesezugriff	Cursorgesteuerte Operationen		UPDATE oder DELETE mit Suche	
		Suche	WHERE CURRENT OF	Suche	UPDATE oder DELETE
RR	IN/-	IX/S	IX/X	IX/S	IX/X
RS	IS/NS	IX/U	IX/X	IX/U	IX/X
CS	IS/NS	IX/U	IX/X	IX/U	IX/X
UR	IN/-	IX/U	IX/X	IX/U	IX/X

Zugehörige Konzepte:

- „Sperrattribute“ auf Seite 71
- „Sperrgranularität“ auf Seite 74

Zugehörige Referenzen:

- „Sperrmodi für Tabellen- und Satz-ID-Indexsuchen für MDC-Tabellen“ auf Seite 96
- „Sperrtypenkompatibilität“ auf Seite 84
- „Sperrn für Blockindexsuchen für MDC-Tabellen“ auf Seite 101

Indextypen und Sperren des nächsten Schlüssels

Wenn Transaktionen Änderungen an Indizes des Typs 1 verursachen, treten Sperren der jeweils nächsten Schlüssel (Next-Key Locking) auf. Für Indizes des Typs 2 findet das Sperren nächster Schlüssel nur in minimalem Ausmaß statt.

- Sperren der nächsten Schlüssel für Indizes des Typs 2:

Das Sperren eines nächsten Schlüssels findet statt, wenn ein Schlüssel in einen Index eingefügt wird.

Während der Einfügung eines Schlüssels in einen Index, wird die Zeile, die dem Schlüssel entspricht, der auf den neuen Schlüssel im Index als Nächstes folgt, nur gesperrt, wenn diese Zeile momentan durch eine RR-Indextsuche gesperrt ist. Der Spermodus, der für die Sperre des nächsten Schlüssels verwendet wird, ist NW. Diese Sperre des nächsten Schlüssels wird wieder freigegeben, bevor die eigentliche Einfügung des Schlüssels erfolgt. Die Einfügung des Schlüssels erfolgt, wenn eine Zeile in eine Tabelle eingefügt wird.

Wenn Aktualisierungen an einer Zeile zu einer Änderung am Wert des Indexschlüssels für diese Zeile führen, erfolgt die Einfügung des Schlüssels außerdem deshalb, weil der ursprüngliche Schlüsselwert als gelöscht markiert wird und der neue Schlüsselwert in den Index eingefügt wird. Bei Aktualisierungen, die nur die INCLUDE-Spalten eines Index betreffen, kann der Schlüssel an seinem Platz aktualisiert werden, und es findet keine Sperrung des nächsten Schlüssels statt.

Bei RR-Suchen wird die Zeile, die dem Schlüssel entspricht, der auf das Ende des Suchbereichs folgt, im S-Modus gesperrt. Wenn auf das Ende des Suchbereichs keine Schlüssel folgen, wird eine Tabellenende-Sperre aktiviert, um das Ende des Index zu sperren. Wenn der Schlüssel, der auf das Ende des Suchbereichs folgt, als gelöscht markiert ist, fährt die Suche mit dem Sperren der entsprechenden Zeilen solange fort, bis ein Schlüssel angetroffen wird, der nicht als gelöscht markiert ist, wenn die Suche die entsprechende Zeile für diesen Schlüssel sperrt, oder solange bis das Ende des Index gesperrt ist.

- Sperren der nächsten Schlüssel für Indizes des Typs 1:

Sperren der nächsten Schlüssel werden bei Lösch- und Einfügeoperationen an Indizes und bei Indextsuchen aktiviert. Wenn eine Zeile in einer Tabelle aktualisiert, aus ihr gelöscht oder in sie eingefügt wird, wird für diese Zeile eine X-Sperre aktiviert. Bei Einfügungen kann der Spermodus auf eine W-Sperre herabgesetzt werden.

Wenn der Schlüssel aus dem Tabellenindex gelöscht oder in ihn eingefügt wird, wird die Tabellenzeile gesperrt, die dem Schlüssel entspricht, der auf den gelöschten oder eingefügten Schlüssel im Index folgt. Bei Aktualisierungen, die den Wert des Schlüssels betreffen, wird der ursprüngliche Schlüsselwert zuerst gelöscht und der neue Wert eingefügt, so dass zwei Sperren für nächste Schlüssel aktiviert werden. Die Dauer dieser Sperren ist wie folgt festgelegt:

- Beim Löschen eines Indexschlüssels ist der Spermodus für den nächsten Schlüssel X und die Sperre bleibt bis zum Commit aktiv.
- Beim Einfügen eines Indexschlüssels ist der Spermodus für den nächsten Schlüssel NW. Diese Sperre wird nur aktiviert, wenn es eine Konkurrenzsituation für die Sperre gibt. In diesem Fall wird die Sperre freigegeben, bevor der Schlüssel tatsächlich in den Index eingefügt wird.
- Bei RR-Suchen wird die Tabellenzeile, die dem Schlüssel direkt nach dem Ende des Indextsuchbereichs entspricht, im Modus S gesperrt und die Sperre bis zum Commit beibehalten.
- Bei CS/RS-Suchen wird die Zeile, die dem Schlüssel direkt nach dem Ende des Indextsuchbereichs entspricht, im Modus NS gesperrt, wenn eine

Konkurrenzsituation für die Sperre vorliegt. Diese Sperre wird freigegeben, wenn das Ende des Suchbereichs einmal geprüft ist.

Das Sperren nächster Schlüssel für Indizes des Typs 1 bei Einfügungen und Löschungen von Schlüsseln kann zu Deadlocks führen. Das folgende Beispiel zeigt, wie sich zwei Transaktionen gegenseitig sperren könnten. Bei Indizes des Typs 2 treten solche Deadlocks nicht auf.

Betrachten Sie das folgende Beispiel eines Index, der sechs Zeilen mit den folgenden Werten enthält: 1 5 6 7 8 12.

1. Transaktion 1 löscht die Zeile mit dem Schlüsselwert 8. Die Zeile mit dem Wert 8 wird im Modus X gesperrt. Wenn der entsprechende Schlüssel aus dem Index gelöscht wird, wird die Zeile mit dem Wert 12 im Modus X gesperrt.
2. Transaktion 2 löscht die Zeile mit dem Schlüsselwert 5. Die Zeile mit dem Wert 5 wird im Modus X gesperrt. Wenn der entsprechende Schlüssel aus dem Index gelöscht wird, wird die Zeile mit dem Wert 6 im Modus X gesperrt.
3. Transaktion 1 fügt eine Zeile mit dem Schlüsselwert 4 ein. Diese Zeile wird im Modus W gesperrt. Wenn versucht wird, den neuen Schlüssel in den Index einzufügen, wird die Zeile mit dem Wert 6 im Modus NW gesperrt. Dieser Sperrversuch muss auf die X-Sperre warten, die von Transaktion 2 für diese Zeile aktiviert wurde.
4. Transaktion 2 fügt eine Zeile mit dem Schlüsselwert 9 ein. Diese Zeile wird im Modus W gesperrt. Wenn versucht wird, den neuen Schlüssel in den Index einzufügen, wird die Zeile mit dem Schlüsselwert 12 im Modus NW gesperrt. Dieser Sperrversuch muss auf die X-Sperre warten, die von Transaktion 1 für diese Zeile aktiviert wurde.

Bei der Verwendung von Indizes des Typs 1 führt ein solches Szenario zu einem Deadlock, und für eine der Transaktionen wird ein Rollback durchgeführt.

Zugehörige Konzepte:

- „Indexbereinigung und Indexpflege“ auf Seite 390
- „Verwenden von relationalen Indizes zur Leistungsverbesserung“ auf Seite 29
- „Indexreorganisation“ auf Seite 366
- „Indexstruktur“ auf Seite 31
- „Online-Indexdefragmentierung“ auf Seite 392
- „Tipps zur Leistung von relationalen Indizes“ auf Seite 35

Auswerten nicht festgeschriebener Daten durch Sperrenverzögerung

Zur Verbesserung des gemeinsamen Zugriffs lässt DB2 jetzt die Verzögerung von Zeilensperren für Suchen mit der Isolationsstufe CS oder RS in einigen Fällen zu, bis ein Datensatz festgestellt wird, der die Vergleichselemente einer Abfrage erfüllt. Standardmäßig sperrt DB2 beim Sperren von Zeilen während einer Tabellen- oder Indexsuche jede Zeile, die durchsucht wird, bevor festgestellt wird, ob die Zeile der Abfrage entspricht. Zur Verbesserung des gemeinsamen Zugriffs durch Suchoperationen ist es möglich, die Zeilensperre zu verzögern, bis festgestellt ist, dass eine Zeile einer Abfrage entspricht.

Zur Nutzung dieser Funktion aktivieren Sie die Registrierdatenbankvariable DB2_EVALUNCOMMITTED.

Wenn diese Variable aktiviert ist, kann eine Auswertung von Vergleichselementen für nicht festgeschriebene Daten stattfinden. Das bedeutet, dass eine Zeile, die eine nicht festgeschriebene Aktualisierung enthält, die Abfrage vielleicht nicht erfüllt, während sie der Abfrage möglicherweise entsprechen würde, wenn mit der Auswertung der Vergleichselemente gewartet würde, bis die aktualisierte Transaktion beendet ist. Darüber hinaus werden nicht festgeschriebene gelöschte Zeilen bei Tabellensuchen übersprungen. DB2 überspringt gelöschte Schlüssel beim Durchsuchen von Indizes des Typs 2, wenn die Registrierdatenbankvariable DB2_SKIPDELETED aktiviert ist.

Diese Registrierdatenbankvariableneinstellungen gelten bei der Kompilierung für dynamische SQL- oder XQuer-Anweisungen sowie beim Binden für statische SQL- oder XQuery-Anweisungen. Dies bedeutet, dass selbst wenn die Registrierdatenbankvariable bei der Ausführung aktiviert ist, die Sperrenvermeidungsstrategie nicht angewendet wird, wenn die Registrierdatenbankvariable DB2_EVALUNCOMMITTED beim Binden nicht aktiviert war. Wenn die Registrierdatenbankvariable beim Binden aktiviert, bei der Ausführung jedoch nicht aktiviert ist, wird die Sperrenvermeidungsstrategie trotzdem angewendet. Für statische SQL- oder XQuery-Anweisungen gilt beim Rebind eines Pakets die Einstellung der Registrierdatenbankvariablen zum Zeitpunkt des Bindens. Ein impliziter Rebind von statischen SQL- oder XQuery-Anweisungen verwendet die aktuelle Einstellung der Registrierdatenbankvariablen DB2_EVALUNCOMMITTED.

Anwendbarkeit der Auswertung nicht festgeschriebener Daten für verschiedene Zugriffspläne

Tabelle 18. Zugriff nur über Satz-ID-Index

Vergleichselemente	Nicht festgeschriebene Daten auswerten
Keine	Nein
Als Suchargument verwendbare Vergleichselemente	Ja

Tabelle 19. Reiner Datenzugriff (relational oder mit verzögerter Satz-ID-Liste)

Vergleichselemente	Nicht festgeschriebene Daten auswerten
Keine	Nein
Als Suchargument verwendbare Vergleichselemente	Ja

Tabelle 20. Satz-ID-Index und Datenzugriff

Vergleichselemente		Nicht festgeschriebene Daten auswerten	
Index	Daten	Indexzugriff	Datenzugriff
Keine	Keine	Nein	Nein
Keine	Als Suchargument verwendbare Vergleichselemente	Nein	Nein
Als Suchargument verwendbare Vergleichselemente	Keine	Ja	Nein
Als Suchargument verwendbare Vergleichselemente	Als Suchargument verwendbare Vergleichselemente	Ja	Nein

Tabelle 21. Blockindex und Datenzugriff

Vergleichselemente		Nicht festgeschriebene Daten auswerten	
Index	Daten	Indexzugriff	Datenzugriff
Keine	Keine	Nein	Nein
Keine	Als Suchargument verwendbare Vergleichselemente	Nein	Ja
Als Suchargument verwendbare Vergleichselemente	Keine	Ja	Nein
Als Suchargument verwendbare Vergleichselemente	Als Suchargument verwendbare Vergleichselemente	Ja	Ja

Beispiel

Das folgende Beispiel zeigt einen Vergleich der Standardfunktionsweise von Sperren und der neuen Funktionsweise mit Auswertung nicht festgeschriebener Daten.

Die folgende Tabelle ist die Tabelle ORG aus der Beispieldatenbank SAMPLE.

DEPTNUMB	DEPTNAME	MANAGER	DIVISION	LOCATION
10	Head Office	160	Corporate	New York
15	New England	50	Eastern	Boston
20	Mid Atlantic	10	Eastern	Washington
38	South Atlantic	30	Eastern	Atlanta
42	Great Lakes	100	Midwest	Chicago
51	Plains	140	Midwest	Dallas
66	Pacific	270	Western	San Francisco
84	Mountain	290	Western	Denver

Die folgenden Transaktionen werden an dieser Tabelle mit der Standardisolationstufe der Cursorstabilität (CS) ausgeführt.

Tabelle 22. Transaktionen an der Tabelle ORG mit der Isolationsstufe CS

SITZUNG 1	SITZUNG 2
connect to SAMPLE	connect to SAMPLE
+c update org set deptnumb=5 where manager=160	
	select * from org where deptnumb >= 10

Die nicht festgeschriebene UPDATE-Operation in Sitzung 1 aktiviert eine exklusive Satzsperrung für die erste Zeile in der Tabelle, welche die SELECT-Abfrage in Sitzung 2 daran hindert, zurückzukehren, obwohl die in Sitzung 1 zu aktualisierende Zeile momentan die Abfrage in Sitzung 2 nicht erfüllt. Das liegt daran, dass die Isolationsstufe CS festlegt, dass jede Zeile, auf die durch eine Abfrage zugegriffen wird, gesperrt werden muss, während sich der Cursor auf dieser Zeile befindet. Sitzung 2 kann solange keine Sperre für die erste Zeile aktivieren, bis Sitzung 1 ihre Sperre freigibt.

Beim Durchsuchen der Tabelle kann das Warten auf die Sperre in Sitzung 2 durch die Verwendung der Funktion zur Auswertung nicht festgeschriebener Daten vermieden werden. Dabei wird zunächst das Vergleichselement ausgewertet und anschließend eine Sperre für die Zeile aktiviert, um das Vergleichselement endgültig

tig auszuwerten. In diesem Fall würde die Abfrage in Sitzung 2 nicht versuchen, die erste Zeile in der Tabelle zu sperren, so dass der gemeinsame Zugriff durch Anwendungen verbessert wird. Beachten Sie, dass dies auch bedeuten würde, dass die Auswertung des Vergleichselements in Sitzung 2 im Hinblick auf den nicht festgeschriebenen Wert mit `deptnumb=5` in Sitzung 1 erfolgen würde. Die Abfrage in Sitzung 2 würde die erste Zeile aus ihrer Ergebnismenge herauslassen, obwohl ein Rollback der UPDATE-Operation in Sitzung 1 die Abfrage in Sitzung 2 erfüllen würde.

Wenn die Reihenfolge der Operationen umgekehrt wäre, ergäbe sich mit der Funktion zur Auswertung nicht festgeschriebener Daten trotzdem eine Verbesserung für den gemeinsamen Zugriff. Bei der Standardfunktionsweise von Sperren würde Sitzung 2 zunächst eine Zeilensperre anfordern, die die Ausführung der Aktualisierung mit Suche in Sitzung 1 verhindern würde, obwohl die Aktualisierung in Sitzung 1 die von der Abfrage in Sitzung 2 gesperrte Zeile nicht ändern würde. Wenn die Aktualisierung mit Suche in Sitzung 1 versuchen würde, die Zeilen zunächst zu untersuchen und sie nur zu sperren, wenn sie die Vergleichselemente erfüllen, würde die Abfrage in Sitzung 1 nicht blockiert.

Einschränkungen

Die folgenden externen Einschränkungen gelten für diese neue Funktionalität:

- Die Registrierdatenbankvariable `DB2_EVALUNCOMMITTED` muss aktiviert sein.
- Die Isolationsstufe muss CS oder RS sein.
- Zeilensperren müssen auftreten.
- Als Suchargument verwendbare Vergleichselemente sind vorhanden.
- Die Auswertung nicht festgeschriebener Daten gilt nicht für Suchen in den Katalogtabellen.
- Für MDC-Tabellen können Blocksperrern für eine Indexsuche verzögert werden. Bei Tabellensuchen werden Blocksperrern jedoch nicht verzögert.
- Das verzögerte Sperren tritt nicht für eine Tabelle auf, die eine Inplace-Tabellenreorganisation ausführt.
- Das verzögerte Sperren findet nicht bei einer Indexsuche statt, wenn es sich um einen Index des Typs 1 handelt.
- Für Iscan-Fetch-Pläne werden Zeilensperren nicht bis zum Datenzugriff verzögert, sondern die Zeile wird beim Indexzugriff gesperrt, bevor auf die Zeile in der Tabelle zugegriffen wird.
- Gelöschte Zeilen werden bei Tabellensuchen bedingungslos übersprungen, während gelöschte Schlüssel von Indizes des Typs 2 nur übersprungen werden, wenn die Registrierdatenbankvariable `DB2_SKIPDELETED` aktiviert ist.

Option zum Ignorieren nicht festgeschriebener Einfügungen

Die Registrierdatenbankvariable `DB2_SKIPINSERTED` steuert, ob nicht festgeschriebene Einfügungen für Cursor ignoriert werden können, die die Isolationsstufe Cursorstabilität oder Lesestabilität verwenden. Das DB2-Datenbanksystem kann nicht festgeschriebene Einfügungen wie folgt handhaben:

- Das DB2-Datenbanksystem kann so lange warten, bis die INSERT-Transaktion durchgeführt (d. h. mit COMMIT festgeschrieben oder mit ROLLBACK rückgängig gemacht) wurde, und die Daten dementsprechend verarbeiten. Die Standardoption ist OFF.

In den folgenden Beispielen sehen Sie Instanzen, bei denen die Standardoption OFF bevorzugt wird:

- Angenommen, zwei Anwendungen verwenden eine Tabelle, um Daten untereinander auszutauschen, wobei die erste Anwendung Daten in die Tabelle einfügt und die zweite Anwendung die Daten liest. Die Daten müssen von der zweiten Anwendung in der Reihenfolge in der Tabelle so verarbeitet werden, damit, falls die nächste zu lesende Zeile von der ersten Anwendung eingefügt wurde, die zweite Anwendung warten muss, bis die eingefügten Daten festgeschrieben werden. In einem solchen Fall muss der Standardwert für DB2_SKIPINSERTED verwendet werden.
- Angenommen, eine Anwendung nimmt Änderungen an Daten vor, indem die Daten gelöscht und das neue Datenimage eingefügt wird. In den Fällen, in denen UPDATE-Anweisungen vermieden werden, muss der Standardwert für DB2_SKIPINSERTED verwendet werden.
- Das DB2-Datenbanksystem kann nicht festgeschriebene Einfügungen ignorieren, was in vielen Fällen den gemeinsamen Zugriff verbessern kann. Wenn Sie dieses Verhalten wünschen, muss die Registrierdatenbankvariable auf ON gesetzt sein. Im Allgemeinen wird durch ON der gemeinsame Zugriff verbessert für die meisten Anwendungen bevorzugt verwendet. Wenn die Registrierdatenbankvariable aktiviert ist, werden nicht festgeschriebene eingefügte so Zeilen betrachtet, als wären sie noch nicht eingefügt worden.

Zugehörige Konzepte:

- „Auswerten nicht festgeschriebener Daten durch Sperrenverzögerung“ auf Seite 92

Sperrmodi für Tabellen- und Satz-ID-Indexsuchen für MDC-Tabellen

In einer MDC-Umgebung (Multidimensional Clustering) wird eine zusätzliche Sperrstufe verwendet: der Block. In den folgenden Tabellen werden die Typen von Sperren aufgelistet, die in der jeweiligen Isolationsstufe für verschiedene Zugriffspläne aktiviert werden. Jeder Eintrag besteht aus drei Teilen: Tabellensperre, Blocksperrung und Zeilensperre. Ein Strich zeigt an, dass für eine bestimmte Stufe keine Sperrung erfolgt.

Anmerkung: Sperrmodi können explizit mit einer Sperranforderungsklausel (LOCK REQUEST) in einer SELECT-Anweisung geändert werden.

Tabelle 23. Sperrmodi für Tabellensuchen ohne Vergleichselemente

Isolations- stufe	Suchen: mehrdeu- tig und Lesezu- griff	Cursorgesteuerte Operationen		UPDATE oder DELETE mit Suche	
		Suche	WHERE CURRENT OF	Suche od. DELETE	UPDATE
RR	S/-/-	U/-/-	SIX/IX/X	X/-/-	X/-/-
RS	IS/IS/NS	IX/IX/U	IX/IX/U	IX/X/-	IX/I/-
CS	IS/IS/NS	IX/IX/U	IX/IX/X	IX/X/-	IX/X/-
UR	IN/IN/-	IX/IX/U	IX/IX/X	IX/X/-	IX/X/-

Tabelle 24. Sperrmodi für Tabellensuchen mit Vergleichselementen nur mit Dimensionsspalten

Isolationsstufe	Suchen: mehrdeutig und Lesezugriff	Cursorgesteuerte Operationen		UPDATE oder DELETE mit Suche	
		Suche	WHERE CURRENT OF	Suche od. DELETE	UPDATE
RR	S/-/-	U/-/-	SIX/IX/X	U/-/-	SIX/X/-
RS	IS/IS/NS	IX/IX/U	IX/IX/X	IX/U/-	X/X/-
CS	IS/IS/NS	IX/IX/U	IX/IX/X	IX/U/-	X/X/-
UR	IN/IN/-	IX/IX/U	IX/IX/X	IX/U/-	X/X/-

Tabelle 25. Sperrmodi für Tabellensuchen mit anderen Vergleichselementen (sargs, residu)

Isolationsstufe	Suchen: mehrdeutig und Lesezugriff	Cursorgesteuerte Operationen		UPDATE oder DELETE mit Suche	
		Suche	WHERE CURRENT OF	Suche od. DELETE	UPDATE
RR	S/-/-	U/-/-	SIX/IX/X	U/-/-	SIX/IX/X
RS	IS/IS/NS	IX/IX/U	IX/IX/X	IX/IX/U	IX/IX/X
CS	IS/IS/NS	IX/IX/U	IX/IX/X	IX/IX/U	IX/IX/X
UR	IN/IN/-	IX/IX/U	IX/IX/X	IX/IX/U	IX/IX/X

Die beiden folgenden Tabellen zeigen Sperrmodi für Satz-ID-Indizes für MDC-Tabellen.

Tabelle 26. Sperrmodi für Satz-ID-Indexsuchen (RID) ohne Vergleichselemente

Isolationsstufe	Suchen: mehrdeutig und Lesezugriff	Cursorgesteuerte Operationen		UPDATE oder DELETE mit Suche	
		Suche	WHERE CURRENT OF	Suche od. DELETE	UPDATE
RR	S/-/-	IX/IX/S	IX/IX/X	X/-/-	X/-/-
RS	IS/IS/NS	IX/IX/U	IX/IX/X	X/X/X	X/X/X
CS	IS/IS/NS	IX/IX/U	IX/IX/X	X/X/X	X/X/X
UR	IN/IN/-	IX/IX/U	IX/IX/X	X/X/X	X/X/X

Tabelle 27. Sperrmodi für Satz-ID-Indexsuchen (RID) mit einer einzigen Qualifikationszeile

Isolationsstufe	Suchen: mehrdeutig und Lesezugriff	Cursorgesteuerte Operationen		UPDATE oder DELETE mit Suche	
		Suche	WHERE CURRENT OF	Suche od. DELETE	UPDATE
RR	IS/IS/S	IX/IX/U	IX/IX/X	X/X/X	X/X/X
RS	IS/IS/NS	IX/IX/U	IX/IX/X	X/X/X	X/X/X
CS	IS/IS/NS	IX/IX/U	IX/IX/X	X/X/X	X/X/X
UR	IN/IN/-	IX/IX/U	IX/IX/X	X/X/X	X/X/X

Tabelle 28. Sperrmodi für Satz-ID-Indextsuchen (RID) nur mit Start- und Stoppvergleichselementen

Isolationsstufe	Suchen: mehrdeutig und Lesezugriff	Cursorgesteuerte Operationen		UPDATE oder DELETE mit Suche	
		Suche	WHERE CURRENT OF	Suche od. DELETE	UPDATE
RR	IS/IS/S	IX/IX/S	IX/IX/X	IX/IX/X	IX/IX/X
RS	IS/IS/NS	IX/IX/U	IX/IX/X	IX/IX/X	IX/IX/X
CS	IS/IS/NS	IX/IX/U	IX/IX/X	IX/IX/X	IX/IX/X
UR	IN/IN/-	IX/IX/U	IX/IX/X	IX/IX/X	IX/IX/X

Tabelle 29. Sperrmodi für Satz-ID-Indextsuchen (RID) nur mit Indexvergleichselementen

Isolationsstufe	Suchen: mehrdeutig und Lesezugriff	Cursorgesteuerte Operationen		UPDATE oder DELETE mit Suche	
		Suche	WHERE CURRENT OF	Suche od. DELETE	UPDATE
RR	IS/S/S	IX/IX/S	IX/IX/X	IX/IX/S	IX/IX/X
RS	IS/IS/NS	IX/IX/U	IX/IX/X	IX/IX/U	IX/IX/X
CS	IS/IS/NS	IX/IX/U	IX/IX/X	IX/IX/U	IX/IX/X
UR	IN/IN/-	IX/IX/U	IX/IX/X	IX/IX/U	IX/IX/X

Tabelle 30. Sperrmodi für Satz-ID-Indextsuchen (RID) mit anderen Vergleichselementen (sargs, resids)

Isolationsstufe	Suchen: mehrdeutig und Lesezugriff	Cursorgesteuerte Operationen		UPDATE oder DELETE mit Suche	
		Suche	WHERE CURRENT OF	Suche od. DELETE	UPDATE
RR	IS/S/S	IX/IX/S	IX/IX/X	IX/IX/S	IX/IX/X
RS	IS/IS/NS	IX/IX/U	IX/IX/X	IX/IX/U	IX/IX/X
CS	IS/IS/NS	IX/IX/U	IX/IX/X	IX/IX/U	IX/IX/X
UR	IN/IN/-	IX/IX/U	IX/IX/X	IX/IX/U	IX/IX/X

Anmerkung: In den folgenden Tabellen, die Sperrmodi für Satz-ID-Indextsuchen für verzögerte Datenseitenzugriffe anzeigen, wird die Isolationsstufe auf CS erhöht und die Sperren in eine IS-Tabellensperre, eine IS-Blocksperrre und NS-Zeilensperren hochgestuft, wenn die Isolationsstufe UR mit einer IN-Sperre für Indizes des Typs 1 vorliegt oder wenn es Vergleichselemente für INCLUDE-Spalten im Index gibt.

Tabelle 31. Sperrmodi für Satz-ID-Indexsuchen mit verzögertem Zugriff auf Datenseiten: Satz-ID-Indexsuche ohne Vergleichselemente

Isolations- stufe	Suchen: mehrdeu- tig und Lesezu- griff	Cursorgesteuerte Operationen		UPDATE oder DELETE mit Suche	
		Suche	WHERE CURRENT OF	Suche od. DELETE	UPDATE
RR	IS/S/S	IX/IX/S		X/-/-	
RS	IN/IN/-	IN/IN/-		IN/IN/-	
CS	IN/IN/-	IN/IN/-		IN/IN/-	
UR	IN/IN/-	IN/IN/-		IN/IN/-	

*Tabelle 32. Sperrmodi für Satz-ID-Indexsuchen mit verzögertem Zugriff auf Datenseiten: Ver-
zögerter Zugriff auf Datenseiten nach einer Satz-ID-Indexsuche ohne Vergleichselemente*

Isolations- stufe	Suchen: mehrdeu- tig und Lesezu- griff	Cursorgesteuerte Operationen		UPDATE oder DELETE mit Suche	
		Suche	WHERE CURRENT OF	Suche od. DELETE	UPDATE
RR	IN/IN/-	IX/IX/S	IX/IX/X	X/-/-	X/-/-
RS	IS/IS/NS	IX/IX/U	IX/IX/X	IX/IX/X	IX/IX/X
CS	IS/IS/NS	IX/IX/U	IX/IX/X	IX/IX/X	IX/IX/X
UR	IN/IN/-	IX/IX/U	IX/IX/X	IX/IX/X	IX/IX/X

Tabelle 33. Sperrmodi für Satz-ID-Indexsuchen mit verzögertem Zugriff auf Datenseiten: Satz-ID-Indexsuche mit Vergleichselementen (sargs, resids)

Isolations- stufe	Suchen: mehrdeu- tig und Lesezu- griff	Cursorgesteuerte Operationen		UPDATE oder DELETE mit Suche	
		Suche	WHERE CURRENT OF	Suche od. DELETE	UPDATE
RR	IS/S/-	IX/IX/S		IX/IX/S	
RS	IN/IN/-	IN/IN/-		IN/IN/-	
CS	IN/IN/-	IN/IN/-		IN/IN/-	
UR	IN/IN/-	IN/IN/-		IN/IN/-	

*Tabelle 34. Sperrmodi für Satz-ID-Indexsuchen mit verzögertem Zugriff auf Datenseiten: Ver-
zögerter Zugriff auf Datenseiten nach einer Satz-ID-Indexsuche mit Vergleichselementen
(sargs, resids)*

Isolations- stufe	Suchen: mehrdeu- tig und Lesezu- griff	Cursorgesteuerte Operationen		UPDATE oder DELETE mit Suche	
		Suche	WHERE CURRENT OF	Suche od. DELETE	UPDATE
RR	IN/IN/-	IX/IX/S	IX/IX/X	IX/IX/S	IX/IX/X
RS	IS/IS/NS	IX/IX/U	IX/IX/X	IX/IX/U	IX/IX/X
CS	IS/IS/NS	IX/IX/U	IX/IX/X	IX/IX/U	IX/IX/X

Tabelle 34. Sperrmodi für Satz-ID-Indexsuchen mit verzögertem Zugriff auf Datenseiten: Verzögerter Zugriff auf Datenseiten nach einer Satz-ID-Indexsuche mit Vergleichselementen (sargs, resids) (Forts.)

Isolationsstufe	Suchen: mehrdeutig und Lesezugriff	Cursorgesteuerte Operationen		UPDATE oder DELETE mit Suche	
		Suche	WHERE CURRENT OF	Suche od. DELETE	UPDATE
UR	IN/IN/-	IX/IX/U	IX/IX/X	IX/IX/U	IX/IX/X

Tabelle 35. Sperrmodi für Satz-ID-Indexsuchen mit verzögertem Zugriff auf Datenseiten: Satz-ID-Indexsuche nur mit Start- und Stoppvergleichselementen

Isolationsstufe	Suchen: mehrdeutig und Lesezugriff	Cursorgesteuerte Operationen		UPDATE oder DELETE mit Suche	
		Suche	WHERE CURRENT OF	Suche od. DELETE	UPDATE
RR	IS/IS/S	IX/IX/S		IX/IX/X	
RS	IN/IN/-	IN/IN/-		IN/IN/-	
CS	IN/IN/-	IN/IN/-		IN/IN/-	
UR	IN/IN/-	IN/IN/-		IN/IN/-	

Tabelle 36. Sperrmodi für Satz-ID-Indexsuchen mit verzögertem Zugriff auf Datenseiten: Verzögerter Zugriff auf Datenseiten nach einer Satz-ID-Indexsuche nur mit Start- und Stoppvergleichselementen

Isolationsstufe	Suchen: mehrdeutig und Lesezugriff	Cursorgesteuerte Operationen		UPDATE oder DELETE mit Suche	
		Suche	WHERE CURRENT OF	Suche od. DELETE	UPDATE
RR	IN/IN/-	IX/IX/S	IX/IX/X	IX/IX/X	IX/IX/X
RS	IS/IS/NS	IX/IX/U	IX/IX/X	IX/IX/U	IX/IX/X
CS	IS/IS/NS	IX/IX/U	IX/IX/X	IX/IX/U	IX/IX/X
UR	IS/-/-	IX/IX/U	IX/IX/X	IX/IX/U	IX/IX/X

Zugehörige Konzepte:

- „Sperrattribute“ auf Seite 71
- „Sperrren und Steuerung des gemeinsamen Zugriffs“ auf Seite 69
- „Sperrgranularität“ auf Seite 74

Zugehörige Referenzen:

- „Sperrmodi und Zugriffspfade für Standardtabellen“ auf Seite 87
- „Sperrren für Blockindexsuchen für MDC-Tabellen“ auf Seite 101
- „Sperrrentypenkompatibilität“ auf Seite 84

Sperrungen für Blockindexsuchen für MDC-Tabellen

In den folgenden Tabellen werden die Typen von Sperrungen aufgelistet, die in der jeweiligen Isolationsstufe für verschiedene Zugriffspläne aktiviert werden. Jeder Eintrag besteht aus drei Teilen: Tabellensperre, Blocksperre und Zeilensperre. Ein Strich zeigt an, dass für eine bestimmte Stufe keine Sperrung erfolgt.

Anmerkung: Sperrmodi können explizit mit einer Sperranforderungsklausel (LOCK REQUEST) in einer SELECT-Anweisung geändert werden.

Tabelle 37. Sperrmodi für Indexsuchen ohne Vergleichselemente

Isolationsstufe	Suchen: mehrdeutig und Lesezugriff	Cursorgesteuerte Operationen		UPDATE oder DELETE mit Suche	
		Suche	WHERE CURRENT OF	Suche od. DELETE	UPDATE
RR	S/--/--	IX/IX/S	IX/IX/X	X/--/--	X/--/--
RS	IS/IS/NS	IX/IX/U	IX/IX/X	X/X/--	X/X/--
CS	IS/IS/NS	IX/IX/U	IX/IX/X	X/X/--	X/X/--
UR	IN/IN/-	IX/IX/U	IX/IX/X	X/X/--	X/X/--

Tabelle 38. Sperrmodi für Indexsuchen nur mit Dimensionsvergleichselementen

Isolationsstufe	Suchen: mehrdeutig und Lesezugriff	Cursorgesteuerte Operationen		UPDATE oder DELETE mit Suche	
		Suche	WHERE CURRENT OF	Suche od. DELETE	UPDATE
RR	IS/-/-	IX/IX/S	IX/IX/X	X/-/-	X/-/-
RS	IS/IS/NS	IX/IX/U	IX/IX/X	IX/X/-	IX/X/-
CS	IS/IS/NS	IX/IX/U	IX/IX/X	IX/X/-	IX/X/-
UR	IN/IN/-	IX/IX/U	IX/IX/X	IX/X/-	IX/X/-

Tabelle 39. Sperrmodi für Indexsuchen nur mit Start- und Stoppvergleichselementen

Isolationsstufe	Suchen: mehrdeutig und Lesezugriff	Cursorgesteuerte Operationen		UPDATE oder DELETE mit Suche	
		Suche	WHERE CURRENT OF	Suche od. DELETE	UPDATE
RR	IS/S/-	IX/IX/S	IX/IX/S	IX/IX/S	IX/IX/S
RS	IX/IX/S	IX/IX/U	IX/IX/X	IX/IX/-	IX/IX/-
CS	IX/IX/S	IX/IX/U	IX/IX/X	IX/IX/-	IX/IX/-
UR	IN/IN/-	IX/IX/U	IX/IX/X	IX/IX/-	IX/IX/-

Tabelle 40. Sperrmodi für Indexsuchen mit Vergleichselementen

Isolations- stufe	Suchen: mehrdeu- tig und Lesezu- griff	Cursorgesteuerte Operationen		UPDATE oder DELETE mit Suche	
		Suche	WHERE CURRENT OF	Suche od. DELETE	UPDATE
RR	IS/S/-	IX/IX/S	IX/IX/X	IX/IX/S	IX/IX/X
RS	IS/IS/NS	IX/IX/U	IX/IX/X	IX/IX/U	IX/IX/X
CS	IS/IS/NS	IX/IX/U	IX/IX/X	IX/IX/U	IX/IX/X
UR	IN/IN/-	IX/IX/U	IX/IX/X	IX/IX/U	IX/IX/X

Die folgende Tabelle listet Sperrmodi für Blockindexsuchen mit verzögertem Zugriff auf Datenseiten auf:

Tabelle 41. Sperrmodi für Blockindexsuchen mit verzögertem Zugriff auf Datenseiten: Blockindexsuche ohne Vergleichselemente

Isolations- stufe	Suchen: mehrdeu- tig und Lesezu- griff	Cursorgesteuerte Operationen		UPDATE oder DELETE mit Suche	
		Suche	WHERE CURRENT OF	Suche od. DELETE	UPDATE
RR	IS/S/--	IX/IX/S		X/--/--	
RS	IN/IN/--	IN/IN/--		IN/IN/--	
CS	IN/IN/--	IN/IN/--		IN/IN/--	
UR	IN/IN/--	IN/IN/--		IN/IN/--	

Tabelle 42. Sperrmodi für Blockindexsuchen mit verzögertem Zugriff auf Datenseiten: Verzögerter Zugriff auf Datenseiten nach einer Blockindexsuche ohne Vergleichselemente

Isolations- stufe	Suchen: mehrdeu- tig und Lesezu- griff	Cursorgesteuerte Operationen		UPDATE oder DELETE mit Suche	
		Suche	WHERE CURRENT OF	Suche od. DELETE	UPDATE
RR	IN/IN/--	IX/IX/S	IX/IX/X	X/--/--	X/--/--
RS	IS/IS/NS	IX/IX/U	IX/IX/X	X/X/--	X/X/--
CS	IS/IS/NS	IX/IX/U	IX/IX/X	X/X/--	X/X/--
UR	IN/IN/--	IX/IX/U	IX/IX/X	X/X/--	X/X/--

Tabelle 43. Sperrmodi für Blockindexsuchen mit verzögertem Zugriff auf Datenseiten: Blockindexsuche nur mit Dimensionsvergleichselementen

Isolations- stufe	Suchen: mehrdeu- tig und Lesezu- griff	Cursorgesteuerte Operationen		UPDATE oder DELETE mit Suche	
		Suche	WHERE CURRENT OF	Suche od. DELETE	UPDATE
RR	IS/S/--	IX/IX/--		IX/S/--	
RS	IS/IS/NS	IX/--/--		IX/--/--	
CS	IS/IS/NS	IX/--/--		IX/--/--	

Tabelle 43. Sperrmodi für Blockindexsuchen mit verzögertem Zugriff auf Datenseiten: Blockindexsuche nur mit Dimensionsvergleichselementen (Forts.)

Isolationsstufe	Suchen: mehrdeutig und Lesezugriff	Cursorgesteuerte Operationen		UPDATE oder DELETE mit Suche	
		Suche	WHERE CURRENT OF	Suche od. DELETE	UPDATE
UR	IN/IN/--	IX/--/--		IX/--/--	

Tabelle 44. Sperrmodi für Blockindexsuchen mit verzögertem Zugriff auf Datenseiten: Verzögerter Zugriff auf Datenseiten nach einer Blockindexsuche nur mit Dimensionsvergleichselementen

Isolationsstufe	Suchen: mehrdeutig und Lesezugriff	Cursorgesteuerte Operationen		UPDATE oder DELETE mit Suche	
		Suche	WHERE CURRENT OF	Suche od. DELETE	UPDATE
RR	IN/IN/--	IX/IX/S	IX/IX/X	IX/S/--	IX/X/--
RS	IS/IS/NS	IX/IX/U	IX/IX/X	IX/U/--	IX/X/--
CS	IS/IS/NS	IX/IX/U	IX/IX/X	IX/U/--	IX/X/--
UR	IN/IN/--	IX/IX/U	IX/IX/X	IX/U/--	IX/X/--

Tabelle 45. Sperrmodi für Blockindexsuchen mit verzögertem Zugriff auf Datenseiten: Blockindexsuche nur mit Start- und Stoppvergleichselementen

Isolationsstufe	Suchen: mehrdeutig und Lesezugriff	Cursorgesteuerte Operationen		UPDATE oder DELETE mit Suche	
		Suche	WHERE CURRENT OF	Suche od. DELETE	UPDATE
RR	IS/S/--	IX/IX/--		IX/X/--	
RS	IN/IN/--	IN/IN/--		IN/IN/--	
CS	IN/IN/--	IN/IN/--		IN/IN/--	
UR	IN/IN/--	IN/IN/--		IN/IN/--	

Tabelle 46. Sperrmodi für Blockindexsuchen mit verzögertem Zugriff auf Datenseiten: Verzögerter Zugriff auf Datenseiten nach einer Blockindexsuche nur mit Start- und Stoppvergleichselementen

Isolationsstufe	Suchen: mehrdeutig und Lesezugriff	Cursorgesteuerte Operationen		UPDATE oder DELETE mit Suche	
		Suche	WHERE CURRENT OF	Suche od. DELETE	UPDATE
RR	IN/IN/--	IX/IX/X		IX/X/--	
RS	IS/IS/NS	IN/IN/--		IN/IN/--	
CS	IS/IS/NS	IN/IN/--		IN/IN/--	
UR	IS/--/--	IN/IN/--		IN/IN/--	

Tabelle 47. Sperrmodi für Blockindexsuchen mit verzögertem Zugriff auf Datenseiten: Blockindexsuche mit anderen Vergleichselementen (sargs, resids)

Isolationsstufe	Suchen: mehrdeutig und Lesezugriff	Cursorgesteuerte Operationen		UPDATE oder DELETE mit Suche	
		Suche	WHERE CURRENT OF	Suche od. DELETE	UPDATE
RR	IS/S/--	IX/IX/--		IX/IX/--	
RS	IN/IN/--	IN/IN/--		IN/IN/--	
CS	IN/IN/--	IN/IN/--		IN/IN/--	
UR	IN/IN/--	IN/IN/--		IN/IN/--	

Tabelle 48. Sperrmodi für Blockindexsuchen mit verzögertem Zugriff auf Datenseiten: Verzögerter Zugriff auf Datenseiten nach einer Blockindexsuche mit anderen Vergleichselementen (sargs, resids)

Isolationsstufe	Suchen: mehrdeutig und Lesezugriff	Cursorgesteuerte Operationen		UPDATE oder DELETE mit Suche	
		Suche	WHERE CURRENT OF	Suche od. DELETE	UPDATE
RR	IN/IN/--	IX/IX/S	IX/IX/X	IX/IX/S	IX/IX/X
RS	IS/IS/NS	IX/IX/U	IX/IX/X	IX/IX/U	IX/IX/X
CS	IS/IS/NS	IX/IX/U	IX/IX/X	IX/IX/U	IX/IX/X
UR	IN/IN/--	IX/IX/U	IX/IX/X	IX/IX/U	IX/IX/X

Zugehörige Konzepte:

- „Sperrgranularität“ auf Seite 74

Zugehörige Referenzen:

- „Sperrmodi für Tabellen- und Satz-ID-Indexsuchen für MDC-Tabellen“ auf Seite 96
- „Sperrtypenkompatibilität“ auf Seite 84
- „Sperrmodi und Zugriffspfade für Standardtabellen“ auf Seite 87

Sperrverhalten für partitionierte Tabellen

Zusätzlich zu einer Sperre für die gesamte Tabelle können Sie eine Sperre für jede Datenpartition einer partitionierten Tabelle setzen. Auf diese Weise können Sie die zu sperrenden Bereiche besser differenzieren und so den gemeinsamen Zugriff im Vergleich zu einer nicht partitionierten Tabelle verbessern. Die neue Datenpartitionssperre wird in der Ausgabe des Befehls `db2pd`, des Ereignismonitors, der Verwaltungssichten und der Tabellenfunktionen ausgewiesen.

Beim Zugriff auf eine Tabelle wird von der Sperrfunktion zuerst die Tabellensperre gesetzt. Anschließend werden dann die erforderlichen Datenpartitionssperren gesetzt, die sich nach den Daten richten, auf die zugegriffen werden soll. Durch die verwendeten Zugriffsmethoden und Isolationsstufen ist es möglicherweise erforderlich, bestimmte Datenpartitionen zu sperren, die sich nicht in der Ergebnismenge befinden. Nach dem Setzen dieser Datenpartitionssperren können diese über denselben Zeitraum gehalten werden wie die Tabellensperre. Für einen CS-

Suchvorgang (CS = Cursorstabilität) in einem Index kann es z. B. sinnvoll sein, die Sperren der zuvor verwendeten Datenpartition beizubehalten, um den Aufwand für das erneute Setzen der Datenpartitionssperre zu vermeiden, wenn diese Datenpartition von nachfolgenden Schlüsseln referenziert wird. Die Datenpartitionssperre umfasst auch den Aufwand für die Sicherstellung des Zugriffs auf die Tabellenbereiche. Bei nicht partitionierten Tabellen wird der Tabellenbereichszugriff über die Tabellensperre gesteuert. Aus diesem Grund werden die Datenpartitionen auch dann gesperrt, wenn eine exklusive oder gemeinsam verwendete Sperre auf Tabellenebene für eine partitionierte Tabelle definiert wurde.

Durch die genauere Differenzierung kann einer Transaktion der exklusive Zugriff auf eine bestimmte Datenpartition gewährt und die Zeilensperren vermieden werden, während andere Transaktionen auf die anderen verfügbaren Datenpartitionen zugreifen können. Dieser Fall kann auf Grund eines Zugriffsplans für eine Massenaktualisierung oder auf Grund einer Sperreneskalation auf Datenpartitionsebene eintreten. Als Tabellensperre für zahlreiche Zugriffsmethoden wird normalerweise eine Intent-Sperre verwendet. Dies gilt auch dann, wenn für die Datenpartitionen eine gemeinsame oder eine exklusive Sperre gesetzt wird. Auf diese Weise kann der gemeinsame Zugriff verbessert werden. Wenn jedoch auf Datenpartitionsebene Nicht-Intent-Sperren erforderlich sind und im Zugriffsplan angegeben ist, dass auf alle Datenpartitionen zugegriffen werden kann, dann wird möglicherweise auf Tabellenebene eine Nicht-Intent-Sperre ausgewählt, um Deadlocks zwischen Datenpartitionssperren bei gleichzeitig zugreifenden Transaktionen zu vermeiden.

Sperren für die SQL-Anweisung LOCK TABLE:

Bei partitionierten Tabellen werden für die Anweisung LOCK TABLE ausschließlich Sperren auf Tabellenebene gesetzt. Datenpartitionssperren werden hingegen nicht definiert. Auf diese Weise kann sichergestellt werden, dass bei nachfolgenden DML-Anweisungen keine Zeilensperren für die Tabelle gesetzt werden. Außerdem können so Deadlocks auf Zeilen-, Block- oder Datenpartitionsebene vermieden werden. Mit LOCK TABLE IN EXCLUSIVE MODE kann beim Aktualisieren von Indizes ebenfalls der exklusive Zugriff sichergestellt werden. Dies ist sinnvoll, um das Anwachsen des Umfangs von Indizes des Typs 2 während umfangreicher Aktualisierungsoperationen zu begrenzen.

Auswirkung des Parameters LOCKSIZE TABLE der Anweisung ALTER TABLE:

Die Anweisung ALTER TABLE verfügt über die Option zum Setzen von LOCKSIZE TABLE, durch die sichergestellt werden kann, dass für die Tabelle eine gemeinsame oder exklusive Sperre ohne Intent-Sperren definiert wird. Bei einer partitionierten Tabelle wird diese Sperrenstrategie sowohl bei der Tabellensperre als auch bei den Datenpartitionssperren für alle benutzten Datenpartitionen angewendet.

Zeilen- und Blocksperreneskalation:

Bei partitionierten Tabellen bezieht sich der Bereich für die Zeilen- und Blocksperreneskalation auf die Datenpartitionsebene. Dies bedeutet wiederum, dass andere Transaktionen besser auf die Tabelle zugreifen können. Dies gilt auch dann, wenn für eine Datenpartition eine Eskalation für gemeinsame oder exklusive (Exclusive oder Super Exclusive) Sperren auftritt, andere Datenpartitionen, die nicht eskaliert werden, jedoch nicht betroffen sind. Die Transaktion setzt das Definieren von Zeilensperren für andere Datenpartitionen nach der Eskalation für eine angegebene Datenpartition möglicherweise fort. Die Benachrichtigungsprotokollnachricht für

Eskalationen umfasst Informationen zu der Datenpartition, in der die Eskalation aufgetreten ist, sowie zum Namen der betroffenen partitionierten Tabelle. Aus diesem Grund kann der exklusive Zugriff auf einen Index durch die Sperreneskalation nicht sichergestellt werden. Entweder muss in der Anweisung eine exklusive Sperre auf Tabellenebene gesetzt, eine explizite Anweisung LOCK TABLE IN EXCLUSIVE MODE ausgegeben oder von der Tabelle das Attribut LOCKSIZE TABLE verwendet werden. Die Sperre für die gesamte Tabelle, die für die Zugriffsmethode definiert ist, wird vom Optimierungsprogramm ausgewählt und ist abhängig vom Datenpartitionsausschluss. Eine umfangreiche Aktualisierung an einer Tabelle kann zur Auswahl einer exklusiven Tabellensperre führen, wenn der Datenpartitionsausschluss nicht durchgeführt wird.

Zugehörige Konzepte:

- „Sperrenattribute“ auf Seite 71
- „Sperrgranularität“ auf Seite 74
- „Sperren und Steuerung des gemeinsamen Zugriffs“ auf Seite 69

Zugehörige Referenzen:

- „Sperrmodi für Tabellen- und Satz-ID-Indexsuchen für MDC-Tabellen“ auf Seite 96
- „Sperrtypenkompatibilität“ auf Seite 84
- „Sperren für Blockindexsuchen für MDC-Tabellen“ auf Seite 101
- „LOCK TABLE statement“ in *SQL Reference, Volume 2*
- „LOCKS_HELD administrative view – Retrieve information on locks held“ in *Administrative SQL Routines and Views*
- „LOCKWAITS administrative view – Retrieve current lockwaits information“ in *Administrative SQL Routines and Views*
- „SNAPLOCK administrative view and SNAP_GET_LOCK table function – Retrieve lock logical data group snapshot information“ in *Administrative SQL Routines and Views*
- „SNAPLOCKWAIT administrative view and SNAP_GET_LOCKWAIT table function – Retrieve lockwait logical data group snapshot information“ in *Administrative SQL Routines and Views*
- „SNAPSHOT_LOCK table function“ in *Administrative SQL Routines and Views*
- „SNAPSHOT_LOCKWAIT table function“ in *Administrative SQL Routines and Views*

Kapitel 6. DB2 Governor

Das Dienstprogramm Governor

Das Dienstprogramm Governor kann das Verhalten von Anwendungen überwachen, die an der Datenbank ausgeführt werden, und abhängig von den Regeln, die Sie in der Konfigurationsdatei von Governor angeben, bestimmte Funktionsweisen ändern.

Eine Governor-Instanz besteht aus einem Front-End-Dienstprogramm und mindestens einem Dämon. Jede Governor-Instanz, die Sie starten, ist für eine Instanz des Datenbankmanagers spezifisch. Wenn Sie das Dienstprogramm Governor starten, wird standardmäßig ein Governor-Dämon in jeder Datenbankpartition einer partitionierten Datenbank gestartet. Sie können jedoch angeben, dass ein Dämon in einer einzelnen Datenbankpartition gestartet werden soll, die Sie überwachen wollen.

Anmerkung: Wenn Governor aktiv ist, können die Momentaufnahmenanforderungen des Dienstprogramms die Leistung des Datenbankmanagers beeinträchtigen. Zur Verbesserung der Leistung können Sie das Aktivierungsintervall (wake-up) von Governor vergrößern, um den CPU-Bedarf des Dienstprogramms zu verringern.

Jeder Governor-Dämon sammelt Informationen über die Anwendungen, die an einer Datenbank ausgeführt werden. Anschließend prüft er diese Informationen anhand der Regeln, die Sie in der Konfigurationsdatei von Governor für diese Datenbank angeben.

Governor verwaltet Anwendungstransaktionen, wie es durch die Regeln in der Konfigurationsdatei angegeben ist. Zum Beispiel könnte die Anwendung einer Regel darauf hinweisen, dass eine Anwendung eine bestimmte Ressource übermäßig beansprucht. Eine Regel gibt die durchzuführende Aktion an, wie zum Beispiel das Ändern der Priorität der Anwendung oder das zwangsweise Trennen der Anwendung von der Datenbank.

Wenn die einer Regel zugeordnete Aktion die Priorität der Anwendung ändert, ändert Governor die Priorität von Agenten in der Datenbankpartition, in der die Ressourcenverletzung aufgetreten ist. Wenn in einer partitionierten Datenbank die Anwendung zwangsweise von der Datenbank getrennt wird, findet die Aktion auch dann statt, wenn der Dämon, der die Verletzung festgestellt hat, auf dem Koordinatorknoten der Anwendung ausgeführt wird.

Governor protokolliert alle von ihm durchgeführten Aktionen. Zur Prüfung der Aktionen können Sie die Protokolldateien abfragen.

Zugehörige Konzepte:

- „Governor-Protokolldateien“ auf Seite 120
- „Die Konfigurationsdatei von Governor“ auf Seite 111
- „Der Governor-Dämon“ auf Seite 108

Zugehörige Tasks:

- „Konfigurieren von Governor“ auf Seite 110

- „Starten und Stoppen von Governor“ auf Seite 109

Zugehörige Referenzen:

- „db2gov - DB2 governor command“ in *Command Reference*

Der Governor-Dämon

Wenn der Governor-Dämon gestartet wird, d. h. wenn entweder Sie das Dienstprogramm db2gov ausführen oder Governor aktiv wird (wake-up), führt er die folgende wiederkehrende Abfolge von Operationen aus.

1. Er prüft, ob seine Governor-Konfigurationsdatei geändert bzw. noch nicht gelesen wurde. Trifft eine der beiden Bedingungen zu, liest der Dämon die Regeln in der Datei. Dadurch können Sie das Verhalten des Governor-Dämons ändern, während er aktiv ist.
2. Er fordert Momentaufnahmeninformationen zur Ressourcennutzungsstatistik für jede Anwendung und jeden Agenten an, die bzw. der in der Datenbank arbeitet.

Anmerkung: Auf einigen Plattformen sind die CPU-Statistikdaten auf dem DB2-Monitor nicht verfügbar. In diesem Fall stehen die Abrechnungsregel (account) und die CPU-Begrenzung (cpu) nicht zur Verfügung.

3. Er überprüft die Statistik für jede einzelne Anwendung anhand der Regeln in der Governor-Konfigurationsdatei. Wenn eine Regel auf eine Anwendung zutrifft, führt Governor die angegebene Aktion aus.

Anmerkung: Der Governor vergleicht aufgelaufene Informationen mit den in der Konfigurationsdatei definierten Werten. Dies bedeutet: Falls die Konfigurationsdatei mit neuen Werten aktualisiert wird, die eine Anwendung möglicherweise bereits überschritten hat, werden die Governor-Regeln, die für die betreffende Überschreitung gelten, im nächsten Governor-Intervall unverzüglich auf die Anwendung angewandt.

4. Er schreibt für jede ausgeführte Aktion einen Datensatz in die Governor-Protokolldatei.

Anmerkung: Governor kann nicht zur Anpassung der Agentenprioritäten verwendet werden, wenn der Konfigurationsparameter *agentpri* des Datenbankmanagers einen anderen Wert als den Systemstandardwert enthält.

Wenn Governor seine Operationen abgeschlossen hat, wird er für die in der Konfigurationsdatei angegebene Zeitdauer inaktiviert. Nach Ablauf dieses Intervalls wird Governor wieder aktiviert (wake-up) und führt dieselbe Abfolge von Operationen erneut aus.

Trifft Governor auf einen Fehler oder ein Stoppsignal, führt er vor der Beendigung eine Bereinigung durch. Bei der Bereinigung werden mit Hilfe einer Liste von Anwendungen, deren Prioritäten geändert wurden, die Prioritäten aller Anwendungsagenten zurückgesetzt. Anschließend werden die Prioritäten derjenigen Agenten zurückgesetzt, die nicht mehr für eine Anwendung arbeiten. Dadurch wird sichergestellt, dass keine Agenten nach Beendigung von Governor mit anderen als den Standardprioritäten aktiv bleiben. Falls ein Fehler auftritt, schreibt

Governor eine Nachricht in das Benachrichtigungsprotokoll für die Verwaltung, um seine abnormale Beendigung anzuzeigen.

Anmerkung: Obwohl der Governor-Dämon keine Datenbankanwendung ist und daher auch keine Verbindung (Connect) zur Datenbank unterhält, hat er dennoch eine Instanzverbindung (Attach). Da er Anforderungen für Momentaufnahmen absetzen kann, kann der Governor-Dämon erkennen, wenn der Datenbankmanager beendet wird.

Zugehörige Konzepte:

- „Das Dienstprogramm Governor“ auf Seite 107

Zugehörige Tasks:

- „Starten und Stoppen von Governor“ auf Seite 109

Starten und Stoppen von Governor

Das Dienstprogramm Governor überwacht Anwendungen, die eine Verbindung zu einer Datenbank herstellen, und ändert ihre Funktionsweise gemäß den Regeln, die Sie in einer Konfigurationsdatei von Governor für diese Datenbank angeben.

Voraussetzungen:

Bevor Sie Governor starten, müssen Sie die Konfigurationsdatei erstellen.

Einschränkungen:

Zum Starten oder Stoppen von Governor benötigen Sie die Berechtigung *sysadm* oder *sysctrl*.

Vorgehensweise:

Gehen Sie wie folgt vor, um Governor zu starten oder zu stoppen:

1. Führen Sie zum Starten von Governor den Befehl *db2gov* in der DB2-Befehlszeile aus. Geben Sie die folgenden erforderlichen Parameter an:

- *START datenbankname*

Der von Ihnen angegebene Datenbankname muss mit dem Namen der Datenbank in der Konfigurationsdatei übereinstimmen, die Sie angeben. Stimmen die Namen nicht überein, wird ein Fehler zurückgegeben. Beachten Sie, dass bei der Ausführung von Governor für mehrere Datenbanken für jede Datenbank Dämonen gestartet werden.

- *konfig_dateiname*

Der Name der Konfigurationsdatei von Governor für diese Datenbank. Wenn sich die Datei nicht an der Standardposition befindet, d. h. im Verzeichnis *sqllib*, müssen Sie außer dem Dateinamen auch den Pfad mit angeben.

- *protokolldateiname*

Der Basisname der Protokolldatei für diesen Governor. In einer partitionierten Datenbank wird die Nummer der Datenbankpartition für jede Datenbankpartition hinzugefügt, in der ein Dämon für diese Instanz von Governor ausgeführt wird.

Zum Starten von Governor in einer einzelnen Datenbankpartition für eine partitionierte Datenbank fügen Sie die Option *nodenum* hinzu. Um zum Beispiel Governor für eine Datenbank mit dem Namen *sales* nur auf Knoten 3 einer par-

tionierten Datenbank mit einer Konfigurationsdatei mit dem Namen *salescfg* und einer Protokolldatei mit dem Namen *saleslog* zu starten, geben Sie den folgenden Befehl ein:

```
db2gov START sales nodenum 3 salescfg saleslog
```

Wenn Governor in allen Datenbankpartitionen der Datenbank *sales* gestartet werden soll, geben Sie den folgenden Befehl ein:

```
db2gov START sales salescfg saleslog
```

2. Zum Stoppen von Governor geben Sie den Befehl *db2gov* mit der Option STOP ein.

Wenn Governor zum Beispiel in allen Datenbankpartitionen der Datenbank *sales* gestoppt werden soll, geben Sie den folgenden Befehl ein:

```
db2gov STOP sales
```

Soll Governor nur in Datenbankpartition 3 gestoppt werden, geben Sie den folgenden Befehl ein:

```
db2gov START sales nodenum 3
```

Zugehörige Konzepte:

- „Der Governor-Dämon“ auf Seite 108
- „Das Dienstprogramm Governor“ auf Seite 107

Zugehörige Referenzen:

- „db2gov - DB2 governor command“ in *Command Reference*

Konfigurieren von Governor

Zum Konfigurieren von Governor erstellen Sie eine Konfigurationsdatei, in der die Datenbank, die von einer Governor-Instanz überwacht wird, und die Verwaltung von Abfragen festgelegt werden.

Die Konfigurationsdatei besteht aus einer Menge von Regeln. Die ersten drei Regeln geben die zu überwachende Datenbank, das Intervall, in dem Protokollsätze zu schreiben sind, und das Intervall an, in dem der Governor-Dämon zur Überwachung aktiv wird. Die übrigen Regeln geben an, wie der Datenbankserver zu überwachen ist und welche Aktionen unter bestimmten Umständen auszuführen sind.

Vorgehensweise:

Gehen Sie wie folgt vor, um eine Governor-Konfigurationsdatei zu erstellen:

1. Erstellen Sie in einem Verzeichnis, das von allen Datenbankpartitionen aus zugänglich ist bzw. für sie angehängt ist, eine ASCII-Datei mit einem beschreibenden Namen. Zum Beispiel könnte eine Konfigurationsdatei für eine Governor-Instanz, die eine Datenbank **sales** überwacht, den Namen *govcfgsales* haben.
2. Öffnen Sie die Datei in einem beliebigen Texteditor und geben Sie Konfigurationsinformationen und Aktionsbedingungen ein.

Schließen Sie jede Regel mit einem Semikolon (;) ab. Die folgenden Konfigurationsdaten werden empfohlen:

- **dbname:** Der Name oder Aliasname der zu überwachenden Datenbank.
- **account:** Die Zahl Minuten, nach denen die Governor-Instanz Statistikdaten zur CPU-Nutzung in die zugehörige Protokolldatei schreibt.

- interval: Die Zahl Sekunden, nach denen der Governor-Dämon aktiv wird, um die Überwachungsfunktion auszuführen. Wenn Sie kein Intervall angeben, wird der Standardwert von 120 Sekunden verwendet.

Zum Beispiel könnten die ersten drei Regeln in der Konfigurationsdatei wie folgt aussehen:

```
{ Einmal pro Sekunde aktiv werden, Datenbankname lautet sales,
  Abrechnungsdatensätze alle 30 Minuten }
interval 1; dbname sales; account 30;
```

Fügen Sie Regeln hinzu, welche die zu überwachenden Bedingungen und die Aktion angeben, die auszuführen ist, wenn die jeweilige Regel als wahr ausgewertet wird. Sie könnten zum Beispiel wie folgt eine Regel hinzufügen, welche die Zeit, die eine UOW (Unit of Work, Arbeitseinheit) ausgeführt werden kann, bevor die Anwendung zwangsweise von der Datenbank getrennt wird, auf eine Stunde begrenzt:

```
setlimit uowtime 3600 action force;
```

3. Sichern Sie die Datei.

Zugehörige Konzepte:

- „Beispiel für eine Governor-Konfigurationsdatei“ auf Seite 119
- „Governor-Protokolldateien“ auf Seite 120
- „Elemente für Governor-Regeln“ auf Seite 114
- „Die Konfigurationsdatei von Governor“ auf Seite 111
- „System monitor switches“ in *Systemmonitor Handbuch und Referenz*

Zugehörige Referenzen:

- „appl_priority - Application Agent Priority monitor element“ in *Systemmonitor Handbuch und Referenz*
- „db2gov - DB2 governor command“ in *Command Reference*

Die Konfigurationsdatei von Governor

Beim Starten von Governor geben Sie die Konfigurationsdatei an, die die Regeln zum Überprüfen der in einer Datenbank aktiven Anwendungen enthält. Governor wertet jede Regel aus und führt die angegebene Aktion aus, wenn die entsprechende Regel zutrifft.

Wenn sich Ihre Anforderungen für die Regeln ändern, editieren Sie die Konfigurationsdatei, ohne Governor zu stoppen. Jeder Governor-Dämon erkennt die geänderte Datei und liest sie erneut.

Die Konfigurationsdatei muss in einem Verzeichnis erstellt werden, das für alle Datenbankpartitionen angehängt ist, sodass der Governor-Dämon für die einzelnen Datenbankpartitionen dieselbe Konfigurationsdatei lesen kann.

Die Konfigurationsdatei besteht aus drei erforderlichen Regeln, die die zu überwachende Datenbank, das Intervall, in dem Protokolldatensätze geschrieben werden und das Inaktivierungsintervall der Governor-Dämonen festlegen. Im Anschluss an diese Parameter enthält die Konfigurationsdatei eine Reihe optionaler Regeln zur Anwendungsüberwachung und zugehöriger Aktionen. Die folgenden Anmerkungen gelten für alle Regeln:

- Kommentare werden mit geschweiften Klammern { } begrenzt.

- Die meisten Einträge können in Großbuchstaben, Kleinbuchstaben oder gemischt angegeben werden. Die Ausnahme ist der Anwendungsname, der als Argument für die Regel `applname` angegeben wird, bei dem die Groß-/Kleinschreibung zu beachten ist.
- Jede Regel endet mit einem Semikolon (;).

Erforderliche Regeln

Die folgenden Regeln geben die zu überwachende Datenbank und das Intervall an, in dem der Dämon nach jedem Durchlauf der Aktivitäten wieder aktiv wird. Jede dieser Regeln wird nur einmal in der Datei angegeben.

dbname

Der Name oder Aliasname der zu überwachenden Datenbank.

account *nnnn*

Abrechnungsdatensätze, die statistische Daten zur CPU-Auslastung für jede Verbindung enthalten, werden jeweils nach Ablauf des in Minuten angegebenen Intervalls geschrieben.

Anmerkung: Diese Option ist in der Windows-Umgebung nicht verfügbar.

Findet eine kurze Verbindungssitzung vollständig innerhalb eines Abrechnungsintervalls statt, so wird kein Protokollsatz geschrieben. Wenn Protokollsätze geschrieben werden, enthalten sie CPU-Statistiken, die Aufschluss über die CPU-Verwendung der Verbindung seit dem vorangegangenen Protokollsatz geben. Wird Governor gestoppt und erneut gestartet, wird die CPU-Verwendung möglicherweise in zwei Protokollsätzen aufgezeichnet. Diese können über die Anwendungs-IDs in den Protokollsätzen ermittelt werden.

interval

Das Intervall in Sekunden, nach dem der Dämon jeweils aktiv wird. Wenn Sie kein Intervall angeben, wird der Standardwert von 120 Sekunden verwendet.

Regeln, die Aktionen festlegen

Im Anschluss an die erforderlichen Regeln können Sie Regeln hinzufügen, die angeben, wie Anwendungen zu behandeln sind. Diese Regeln bestehen aus kleineren Komponenten, den so genannten Regelklauseln. Bei ihrer Verwendung müssen die Klauseln in einer bestimmten Reihenfolge in der Regelanweisung wie folgt angegeben werden:

1. **desc** (optional): Ein in Anführungszeichen eingeschlossener Kommentar über die Regel.
2. **time** (optional): Die Tageszeit, zu der die Regel ausgewertet wird.
3. **authid** (optional): Mindestens eine Berechtigungs-ID, unter der die Anwendung Anweisungen ausführt.
4. **applname** (optional): Der Name der ausführbaren Datei oder der Objektdatei, welche die Verbindung zur Datenbank herstellt. Dieser Name ist von der Groß-/Kleinschreibung abhängig. Der Anwendungsname muss in doppelte Anführungszeichen gesetzt werden, wenn er Leerzeichen enthält.
5. **setlimit**: Die Grenzwerte, die von Governor geprüft werden. Dies kann einer von mehreren Werten sein, wie zum Beispiel die CPU-Zeit, die Anzahl zurückgegebener Zeilen oder die inaktive Zeit.

6. **action** (optional): Die Aktion, die auszuführen ist, wenn ein Grenzwert erreicht wird. Wenn keine Aktion angegeben wird, verringert Governor die Priorität von Agenten, die für die Anwendung aktiv sind, um den Wert 10, wenn ein Grenzwert erreicht wird. Aktionen, die an der Datenbank ausgeführt werden können, sind zum Beispiel eine Verringerung der Agentenpriorität, eine erzwungene Trennung der Verbindung der Anwendung zur Datenbank oder eine Einstellung von Zeitplanoptionen für die Operationen der Anwendung.

Zur Definition einer Regel kombinieren Sie die Regelklauseln, wobei Sie jede Klausel in einer Regel nur einmal verwenden, und schließen jede Regel mit einem Semikolon ab, wie in den folgenden Beispielen zu sehen ist:

```
desc "Keine UOW darf länger als 1 Stunde dauern"  
setlimit uowtime 3600 action force;
```

```
desc "Beschränken der Nutzung von db2 CLP durch neuen Benutzer novice"  
authid novice  
applname db2bp.exe  
setlimit cpu 5 locks 100 rowssel 250;
```

Wenn mehrere Regeln auf eine Anwendung zutreffen, werden alle Regeln angewendet. Gewöhnlich wird die Aktion, die dem zuerst angetroffenen Grenzwert einer Regel zugeordnet ist, auch zuerst angewendet. Ausnahmen treten dann auf, wenn Sie den Wert -1 für eine Klausel in einer Regel angeben. In diesem Fall kann ein in der Klausel der nachfolgenden Regel angegebener Wert nur den zuvor in der *gleichen* Klausel angegebenen Wert überschreiben: die übrigen Klauseln in der vorhergehenden Regel bleiben weiterhin gültig. Eine Regel gibt in den Klauseln rowssel 100000 uowtime 3600 zum Beispiel an, dass die Priorität einer Anwendung verringert werden muss, wenn ihre abgelaufene Zeit 1 Stunde überschreitet oder die Anwendung mehr als 100 000 Zeilen auswählt. Eine nachfolgende Regel enthält die Klausel uowtime -1, um anzugeben, dass die gleiche Anwendung über unbegrenzte Zeit verfügen kann. Wenn in diesem Fall die Anwendung länger als 1 Stunde ausgeführt wird, wird ihre Priorität nicht geändert. Das heißt, die Klausel uowtime -1 überschreibt die Klausel uowtime 3600. Wenn die Anwendung jedoch über 100 000 Zeilen auswählt, wird ihre Priorität herabgesetzt, weil die Klausel rowssel 100000 weiterhin gültig ist.

Reihenfolge der Regelanwendung

Der Governor verarbeitet die Regeln in der Konfigurationsdatei vom Anfang der Datei nach unten. Wenn jedoch die Klausel **setlimit** einer späteren Regel weniger restriktiv ist als eine vorherige Regel, wird die restriktivere Regel immer noch angewendet. Zum Beispiel wird admin in der folgenden Konfigurationsdatei trotz der späteren Regel auf 5000 Zeilen begrenzt, da die erste Regel restriktiver ist.

```
desc "Erzwingen, dass alle 5000 oder mehr Zeilen auswählen"  
setlimit rowssel 5000 action force;
```

```
desc "Benutzer admin das Auswählen von weiteren Zeilen erlauben"  
authid admin  
setlimit rowssel 10000 action force;
```

Um sicherzustellen, dass eine weniger restriktive Regel eine restriktivere Regel außer Kraft zu setzen, die in der Datei weiter oben steht, können Sie die Option -1 angeben, um die vorherige Regel vor dem Anwenden der neuen Regel zu löschen. Beispielsweise begrenzt die Anfangsregel in der folgenden Konfigurationsdatei alle Benutzer auf 5000 Zeilen. Die zweite Regel löscht diese Begrenzung für admin und die dritte Regel setzt die Begrenzung für admin auf 10000 Zeilen zurück.

```
desc "Erzwingen, dass alle 5000 oder mehr Zeilen auswählen"  
setlimit rowssel 5000 action force;
```

```
desc "Begrenzung rowssel für admin löschen"  
authid admin  
setlimit rowssel -1;
```

```
desc "Nun die höhere Begrenzung rowssel für admin festlegen"  
authid admin  
setlimit rowssel 10000 action force;
```

Zugehörige Konzepte:

- „Beispiel für eine Governor-Konfigurationsdatei“ auf Seite 119
- „Elemente für Governor-Regeln“ auf Seite 114

Elemente für Governor-Regeln

Jede Regel in der Konfigurationsdatei von Governor wird aus Klauseln zusammengesetzt, welche die Bedingungen zur Anwendung der Regel und die Aktion angeben, die folgt, wenn die Regel als wahr ausgewertet wird. Die Klauseln müssen in der angezeigten Reihenfolge angegeben werden. In den Beschreibungen der Klauseln weisen geschweifte Klammern [] auf eine optionale Klausel hin.

Optionale Startelemente

[desc] Gibt eine Textbeschreibung der Regel an. Die Beschreibung muss in einfachen oder doppelten Anführungszeichen stehen.

[time] Gibt die Zeitspanne an, während der die Regel ausgewertet werden soll.

Die Zeitspanne muss im Format `time hh:mm hh:mm` angegeben werden, z. B. `time 8:00 18:00`. Wird diese Klausel nicht angegeben, ist die Regel 24 Stunden am Tag gültig.

[authid]

Gibt eine oder mehrere Berechtigungs-IDs (`authid`) an, mit denen die Anwendung ausgeführt wird. Mehrere Berechtigungs-IDs müssen durch ein Komma (,) voneinander getrennt werden, z. B. `authid gene, michael, james`. Wird diese Klausel in einer Regel nicht angegeben, gilt die Regel für alle Berechtigungs-IDs.

[applname]

Gibt den Namen der ausführbaren Datei (oder Objektdatei) an, welche die Verbindung zur Datenbank herstellt.

Mehrere Anwendungsnamen müssen durch ein Komma (,) voneinander getrennt werden, z. B. `applname db2bp, batch, geneprog`. Wird diese Klausel in einer Regel nicht angegeben, gilt die Regel für alle Anwendungsnamen.

Anmerkungen:

1. Bei Anwendungsnamen muss Groß-/Kleinschreibung beachtet werden.
2. Der Datenbankmanager schneidet alle Anwendungsnamen auf 20 Zeichen ab. Stellen Sie sicher, dass die zu prüfende Anwendung mit den ersten 20 Zeichen ihres Anwendungsnamens eindeutig identifiziert wird. Ist das nicht der Fall, wird möglicherweise unbeabsichtigt eine andere Anwendung überprüft.

Anwendungsnamen in der Governor-Konfigurationsdatei werden auf 20 Zeichen abgeschnitten, damit sie mit ihrer internen Darstellung übereinstimmen.

Grenzwertklauseln

setlimit

Gibt mindestens einen Grenzwert an, den Governor überprüfen soll. Die Grenzwerte dürfen nur den Wert -1 oder Werte größer als 0 annehmen (z. B. `cpu -1 locks 1000 rowssel 10000`). Sie müssen mindestens eine der Begrenzungen (`cpu`, `locks`, `rowssel`, `uowtime`) angeben. Jede nicht angegebene Begrenzung wird durch die jeweilige Regel nicht eingeschränkt. Governor kann folgende Begrenzungen überprüfen:

cpu *nnn*

Gibt die Anzahl der CPU-Sekunden an, die eine Anwendung nutzen kann. Wenn Sie den Wert -1 angeben, schränkt Governor die CPU-Nutzung der Anwendung nicht ein.

locks *nnn*

Gibt die Anzahl der Sperren an, die eine Anwendung halten kann. Wenn Sie den Wert -1 angeben, schränkt Governor die Anzahl der durch die Anwendung gehaltenen Sperren nicht ein.

rowssel *nnn*

Gibt die Anzahl von Zeilen an, die an die Anwendung zurückgegeben werden. Dieser Wert ist nur auf dem Koordinatorknoten ungleich Null. Wenn Sie den Wert -1 angeben, schränkt Governor die Anzahl der auswählbaren Zeilen nicht ein. Der Maximalwert, der für *nnn* angegeben werden kann, lautet 4 294 967 298.

uowtime *nnn*

Gibt die Zeitspanne in Sekunden an, die verstreichen kann, nachdem eine UOW (Unit of Work, Arbeitseinheit) (UOW) zum ersten Mal aktiv wird. Wenn Sie den Wert -1 angeben, wird die abgelaufene Zeit nicht eingeschränkt.

Anmerkung: Wenn Sie die API `sqlmon` (den Datenbanksystemmonitor-Schalter) zum Inaktivieren des Monitorschalters für die UOW oder des Monitorschalters für die Zeitmarke verwendet haben, wird die Fähigkeit von Governor beeinträchtigt, Anwendungen auf der Grundlage der abgelaufenen Zeit der UOW zu regeln. Governor verwendet das Überwachungsprogramm zum Sammeln von Systeminformationen. Wenn Sie die Schalter in der Konfigurationsdatei des Datenbankmanagers ausschalten, wird dieses Programm für die gesamte Instanz abgeschaltet. Dadurch erhält Governor diese Informationen nicht mehr.

idle *nnn*

Gibt die für eine Verbindung zulässige Leerlaufzeit in Sekunden an, bevor eine Aktion ausgeführt wird. Wenn Sie den Wert -1 angeben, wird die Leerlaufzeit der Verbindung nicht eingeschränkt.

rowsread *nnn*

Gibt die Anzahl der von einer Anwendung auswählbaren Zeilen an. Wenn Sie den Wert -1 angeben, kann die Anwendung eine

unbegrenzte Anzahl an Zeilen auswählen. Der Maximalwert, der für *nnn* angegeben werden kann, lautet 4 294 967 298.

Anmerkung: Diese Begrenzung ist nicht mit *rowssel* identisch. Der Unterschied besteht darin, dass sich *rowsread* auf die Anzahl der Zeilen bezieht, die gelesen werden mussten, um eine Ergebnismenge zurückgeben zu können. Die Anzahl der gelesenen Zeilen schließt Lesevorgänge der Katalogtabellen durch die Steuerkomponente ein und kann durch die Verwendung von Indizes verringert werden.

Aktionsklauseln

[action]

Gibt die Aktion an, die ausgeführt werden muss, wenn ein angegebener Grenzwert überschritten wird. Sie können folgende Aktionen angeben:

Anmerkung: Wenn ein Grenzwert überschritten wird und die Aktionsklausel nicht angegeben ist, verringert Governor die Priorität der für die Anwendung aktiven Agenten um 10.

nice nnn

Gibt eine Änderung der relativen Priorität der für die Anwendung aktiven Agenten an. Gültige Werte: -20 bis +20.

Dieser Parameter ist unter folgenden Voraussetzungen wirksam:

- Auf UNIX-Plattformen muss der Parameter *agentpri* des Datenbankmanagers den Standardwert aufweisen. Andernfalls überschreibt er die Prioritätsklausel.
- Auf Windows-Plattformen kann der Datenbankmanagerparameter *agentpri* gemeinsam mit der Aktion *priority* verwendet werden.

force Gibt an, dass der Agent, der die Anwendung bedient, zum Trennen der Verbindung gezwungen wird. (Setzt die Anweisung *FORCE APPLICATION* ab, um den koordinierenden Agenten zu beenden.)

Anmerkung: In Umgebungen mit partitionierten Datenbanken wird die Aktion *FORCE* nur ausgeführt, wenn der Governor-Dämon in der koordinierenden Datenbankpartition der Anwendung aktiv ist. Wenn also ein Governor-Dämon in Datenbankpartition A aktiv ist und ein Grenzwert für eine Anwendung überschritten wird, deren koordinierende Datenbankpartition die Datenbankpartition B ist, wird die Aktion *FORCE* ausgelassen.

schedule [class]

Durch eine Zeitzuweisung werden die Prioritäten der für die Anwendungen aktiven Agent verbessert, um die durchschnittlichen Antwortzeiten zu minimieren, ohne Anwendungen zu benachteiligen.

Governor wählt die Anwendungen mit den jeweils höchsten Werten für die Zeitzuweisung nach den folgenden drei Kriterien aus:

- Die Anwendung, welche die meisten Sperren hält

Diese Auswahl erfolgt mit dem Ziel, die Anzahl von Situationen zu verringern, in denen auf Sperren gewartet werden muss (lockwaits).

- Die älteste Anwendung
- Die Anwendung mit der kürzesten geschätzten verbleibenden Ausführungszeit

Diese Auswahl ist ein Versuch, möglichst viele kurzzeitig aktive Anweisungen während des Intervalls zum Abschluss zu bringen.

Die drei bei den jeweiligen Kriterien höchstplatzierten Anwendungen erhalten höhere Prioritäten als alle anderen Anwendungen. Das bedeutet, der höchstplatzierten Anwendung in jeder Kriteriumsgruppe wird die höchste Priorität, den nächsthöchsten Anwendungen die zweithöchste Priorität und den dritthöchsten Anwendungen die dritthöchste Priorität erteilt. Wenn eine einzelne Anwendung auf einem der drei höchsten Plätze in mehreren Kategorien rangiert, erhält sie die entsprechende Priorität für das Kriterium, bei dem sie den höchsten Rang innehat. Die nächsthöchste Anwendung erhält die nächsthöchste Priorität für das andere Kriterium. Wenn zum Beispiel Anwendung A die meisten Sperren hält, jedoch die drittkürzeste geschätzte verbleibende Ausführungszeit hat, erhält sie die höchste Priorität für das erste Kriterium. Die Anwendung, die mit der kürzesten geschätzten Ausführungszeit auf Platz vier rangiert erhält dadurch die dritthöchste Priorität für dieses zweite Kriterium.

Die Anwendungen, die von dieser Governor-Regel ausgewählt werden, werden in bis zu drei Klassen unterteilt. Für jede Klasse wählt Governor neun Anwendungen aus, welche nach den oben aufgeführten Kriterien jeweils die drei höchstplatzierten Anwendungen jeder Klasse sind. Wenn Sie die Klassenoption (class) angeben, werden alle Anwendungen, die von dieser Regel ausgewählt werden, als eine einzige Klasse betrachtet, wobei neun Anwendungen ausgewählt werden und wie oben beschrieben höhere Prioritäten erhalten.

Wenn eine Anwendung in mehreren Governor-Regeln ausgewählt wird, wird sie von der letzten Regel registriert, in der sie ausgewählt wird.

Anmerkung: Wenn Sie die API `sqlmon` (den Datenbanksystemmonitor-Schalter) zum Inaktivieren des Schalters der Anweisung verwendet haben, wird die Fähigkeit von Governor beeinträchtigt, Anwendungen auf der Grundlage der abgelaufenen Zeit der Anweisung zu überprüfen. Governor verwendet das Überwachungsprogramm zum Sammeln von Systeminformationen. Wenn Sie die Schalter in der Konfigurationsdatei des Datenbankmanagers ausschalten, wird dieses Programm für die gesamte Instanz abgeschaltet. Dadurch erhält Governor diese Informationen nicht mehr.

Eine Zeitzuweisungsaktion kann folgenden Zwecken dienen:

- Anwendungen in unterschiedlichen Gruppen erhalten Zeit zugewiesen, ohne dass diese Zeit gleichmäßig unter allen Anwendungen aufgeteilt wird.

Wenn beispielsweise 14 Anwendungen (3 kurze, 5 mittlere und 6 lange) gleichzeitig aktiv sind, haben möglicherweise alle eine schlechte Antwortzeit, weil sie die CPU-Zeit teilen. Der Datenbankadministrator kann zwei Gruppen einrichten, eine mit mittleren Anwendungen und eine mit langen Anwendungen. Mit Hilfe der Prioritäten kann Governor alle kurzen Anwendungen ausführen und sicherstellen, dass höchstens drei mittlere und drei lange Anwendungen gleichzeitig aktiv sind. Zu diesem Zweck enthält die Governor-Konfigurationsdatei eine Regel für mittlere und eine Regel für lange Anwendungen.

Das folgende Beispiel zeigt einen Abschnitt einer Governor-Konfigurationsdatei, der dies veranschaulicht:

```
desc "Mittellange Anwendungen in 1 Zeitzuweisungsklasse zusammenfassen"
applname medq1, medq2, medq3, medq4, medq5
setlimit cpu -1
action schedule class;
```

```
desc "Lange Anwendungen in 1 Zeitzuweisungsklasse zusammenfassen"
applname longq1, longq2, longq3, longq4, longq5, longq6
setlimit cpu -1
action schedule class;
```

- Unterschiedliche Benutzergruppen (z. B. Abteilungen in Unternehmen) erhalten gleiche Kriterien für die Vergabe von Prioritäten.

Wenn eine Gruppe eine Vielzahl von Anwendungen ausführt, kann der Administrator sicherstellen, dass andere Gruppen dennoch akzeptable Antwortzeiten für ihre Anwendungen erhalten. Sind zum Beispiel drei Abteilungen beteiligt (Finanzen, Lagerbestand und Planung), kann man alle Benutzer der Finanzabteilung in eine Gruppe legen, alle Benutzer aus dem Lager in eine andere und alle Planer in die dritte. Dann werden die Verarbeitungskapazitäten in etwa gleichmäßig unter den drei Abteilungen aufgeteilt. Das folgende Beispiel zeigt einen Abschnitt einer Governor-Konfigurationsdatei, der dies veranschaulicht:

```
desc "Benutzer der Finanzwesenabteilung in Klasse zusammenfassen"
authid tom, dick, harry, mo, larry, curly
setlimit cpu -1
action schedule class;
```

```
desc "Benutzer der Lagerabteilung in Klasse zusammenfassen"
authid pat, chris, jack, jill
setlimit cpu -1
action schedule class;
```

```
desc "Benutzer der Planungsabteilung in Klasse zusammenfassen"
authid tara, dianne, henrietta, maureen, linda, candy
setlimit cpu -1
action schedule class;
```

- Governor bindet alle Anwendungen in eine Zeitzuweisung ein.

Wenn die Option *class* nicht mit der Aktion angegeben wird, erstellt Governor eigene Klassen nach der Anzahl der aktiven Anwendungen, für die diese Aktion gilt, und ordnet die Anwen-

dungen in verschiedene Klassen ein. Dies erfolgt entsprechend der Aufwandsschätzung durch den DB2-Abfragecompiler für die Abfrage, die von der Anwendung ausgeführt wird. Der Administrator kann alle Anwendungen für die Zeitzuweisung (Scheduling) auswählen, indem er nicht angibt, welche Anwendungen auszuwählen sind. Die Klauseln *applname* und *authid* werden also nicht angegeben, und die Klausel *setlimit* verursacht keine Einschränkungen.

Anmerkung: Wenn ein Grenzwert überschritten wird und die Aktionsklausel nicht angegeben ist, verringert Governor die Priorität der für die Anwendung aktiven Agenten.

Zugehörige Konzepte:

- „Beispiel für eine Governor-Konfigurationsdatei“ auf Seite 119
- „Die Konfigurationsdatei von Governor“ auf Seite 111

Zugehörige Tasks:

- „Konfigurieren von Governor“ auf Seite 110

Beispiel für eine Governor-Konfigurationsdatei

Das folgende Beispiel zeigt eine Governor-Konfigurationsdatei, die verschiedene Regeln mit Aktionen definiert:

```
{ Einmal pro Sekunde aktiv werden, Datenbankname lautet ibmsampl,  
  Abrechnungsdatensätze alle 30 Minuten }  
interval 1; dbname ibmsampl; account 30;
```

```
desc "CPU-Einschränkungen gelten 24 Stunden pro Tag für alle"  
setlimit cpu 600 rowssel 1000000 rowsread 5000000;
```

```
desc "Keine UOW darf länger als 1 Stunde dauern"  
setlimit uowtime 3600 action force;
```

```
desc 'Verlangsamen einer Untermenge von Anwendungen'  
applname jointA, jointB, jointC, queryA  
setlimit cpu 3 locks 1000 rowssel 500 rowsread 5000;
```

```
desc "Governor soll diese 6 langen Anwendungen in 1 Prioritätenklasse einordnen"  
applname longq1, longq2, longq3, longq4, longq5, longq6  
setlimit cpu -1  
action schedule class;
```

```
desc "Zeitzuweisung für alle Anwendungen von der Planungsabteilung"  
authid planid1, planid2, planid3, planid4, planid5  
setlimit cpu -1  
action schedule;
```

```
desc "Einordnen aller CPU-Fresser in eine Klasse zur Verbrauchssteuerung"  
setlimit cpu 3600  
action schedule class;
```

```
desc "Beschränken der Nutzung von db2 CLP durch neuen Benutzer novice"  
authid novice  
applname db2bp.exe  
setlimit cpu 5 locks 100 rowssel 250;
```

```
desc "Zur Tagesarbeitszeit darf keine Anwendung länger als 10 Sek. aktiv sein"  
time 8:30 17:00 setlimit cpu 10 action force;
```

```
desc "Einige Benutzer, die mit Leistungsoptimierung befasst sind, dürfen  
einige Ihrer Anwendungen in der Mittagspause durchführen"  
time 12:00 13:00 authid ming, geoffrey, john, bill  
applname tpcc1, tpcc2, tpcA, tpvG setlimit cpu 600 rowssel 120000 action force;
```

```
desc "Einige Benutzer sollten nicht eingeschränkt werden -- Datenbankadministrator  
und einige andere. Da dies die letzte Angabe in der Datei ist,  
wird zuvor Angegebenes hierdurch überschrieben."  
authid gene, hershel, janet setlimit cpu -1 locks -1 rowssel -1 uowtime -1;
```

```
desc "Erhöhen der Priorität einer wichtigen Anwendung, so dass sie immer  
schnell verarbeitet wird"  
applname Vlapp setlimit cpu 1 locks 1 rowssel 1 action priority -20;
```

Zugehörige Konzepte:

- „Elemente für Governor-Regeln“ auf Seite 114
- „Die Konfigurationsdatei von Governor“ auf Seite 111

Zugehörige Tasks:

- „Konfigurieren von Governor“ auf Seite 110

Governor-Protokolldateien

Wenn ein Governor-Dämon eine Aktion ausführt, schreibt er einen Datensatz in seine Protokolldatei. Zu diesen Aktionen gehören:

- Erzwungene Beendigung einer Anwendung
- Lesen der Governor-Konfigurationsdatei
- Ändern einer Anwendungspriorität
- Feststellen einer Fehler- oder Warnbedingung
- Starten oder Stoppen

Jeder Governor-Dämon besitzt eine eigene Protokolldatei. Getrennte Protokolldateien vermeiden Engpässe durch Sperrungen von Dateien, die auftreten könnten, wenn viele Governor-Dämonen gleichzeitig in dieselbe Datei schreiben würden. Mit dem Dienstprogramm `db2govlg` können Sie die Protokolldateien zusammenfügen und Abfragen ausführen.

Die Protokolldateien werden im Unterverzeichnis `log` des Verzeichnisses `sql1ib` gespeichert. Für Windows-Betriebssysteme gilt jedoch die Ausnahme, dass sich das Unterverzeichnis `log` unterhalb des Instanzverzeichnisses befindet. Wenn Sie Governor mit dem Befehl `db2gov` starten, geben Sie den Basisnamen für die Protokolldatei an. Stellen Sie sicher, dass der Protokolldateiname den Datenbanknamen enthält, um die Protokolldateien in jeder Datenbankpartition unterscheiden zu können, in der Governor ausgeführt wird. Um in einer Umgebung mit partitionierten Datenbanken sicherzustellen, dass der Dateiname für jeden Governor eindeutig ist, wird die Partitionsnummer der Datenbank, in der der Governor-Dämon ausgeführt wird, automatisch an den Protokolldateinamen angefügt.

Datensatzformat der Protokolldatei

Ein Datensatz in der Protokolldatei hat folgendes Format:

Datum Uhrzeit Knoten-Nr. Satztyp Nachricht

Anmerkung: Das Format der Felder *Datum* und *Uhrzeit* ist yyyy-mm-tt hh.mm.ss. Sie können die Protokolldateien in jeder Datenbankpartition durch Sortieren nach diesem Feld zusammenfügen.

Das Feld *Knoten-Nr.* gibt die Nummer der Datenbankpartition an, in der Governor aktiv ist.

Das Feld *Satztyp* enthält unterschiedliche Werte, je nach Typ des in das Protokoll geschriebenen Eintrags. Folgende Werte können eingetragen werden:

- START: Governor wurde gestartet.
- STOP: Governor wurde gestoppt.
- FORCE: Eine Anwendung wurde zwangsweise getrennt.
- NICE: Die Priorität einer Anwendung wurde geändert.
- ERROR: Ein Fehler ist aufgetreten.
- WARNING: Eine Warnung ist aufgetreten.
- READCFG: Governor hat die Konfigurationsdatei gelesen.
- ACCOUNT: Die Abrechnungsstatistik der Anwendung.
- SCHEDGRP: Eine Änderung der Agentenpriorität ist aufgetreten.

Einige dieser Werte werden weiter unten näher beschrieben.

START

Der Datensatz START wird geschrieben, wenn der Governor gestartet wird. Er hat folgendes Format:

Datenbank = <datenbankname>

STOP Der Datensatz STOP wird geschrieben, wenn der Governor gestoppt wird. Er hat folgendes Format:

Datenbank = <datenbankname>

FORCE

Der Datensatz wird immer dann ausgeschrieben, wenn der Governor feststellt, dass eine Anwendung in der Konfigurationsdatei des Governor erzwungen werden soll, wie dies für eine Regel erforderlich ist. Der Datensatz FORCE hat folgendes Format:

<anwendungsname> <authentifizierungs_id> <anwendungs_id>
<koordinierende_partition> <konfigurationszeile>
<einschränkung_überschritten>

Dabei gilt:

<koordinierende_partition>

Gibt die Zahl der koordinierenden Datenbankpartition der Anwendung an.

<konfigurationszeile>

Gibt die Zeilennummer in der Governor-Konfigurationsdatei an, in der sich die Regel befindet, die das Erzwingen der Anwendung verursacht.

<einschränkung_überschritten>

Stellt Einzelangaben zur Verfügung, wie die Regel überschritten wurde. Dies können die folgenden Werte sein:

- CPU: Die gesamte USR CPU plus die SYS CPU-Zeit in Sekunden.
- Locks: Die Summe der Sperren, die von der Anwendung gehalten werden
- Rowsel: Die Summe der Zeilen, die von der Anwendung ausgewählt werden

- Rowsread: Die Summe der Zeilen, die von der Anwendung gelesen werden
- Idle: Zeit, in der die Anwendung inaktiv war
- ET: Zeit, die seit dem Start der aktuellen UOW (Unit of Work, Arbeitseinheit) der Anwendung vergangen ist (das uowtime-Zeitlimit wurde überschritten)

NICE Der Datensatz NICE wird geschrieben, wenn die Priorität einer Anwendung über eine Prioritätsaktion geändert wird, die in der Konfigurationsdatei des Governor angegeben wird. Der Datensatz NICE hat folgendes Format:

```
<anwendungsname> <authentifizierungs_id> <anwendungs_id> <nice_wert>
(<konfigurationszeile>) <einschränkung_überschritten>
```

Dabei gilt Folgendes:

<nice_wert>

Gibt das Inkrement (oder Dekrement) an, das für den Agentenprozess der Anwendung zum Prioritätswert gemacht wird.

<konfigurationszeile>

Gibt die Zeilennummer in der Konfigurationsdatei des Governor an, in der sich die Regel befindet, die das Ändern der Anwendungspriorität verursacht.

<einschränkung_überschritten>

Stellt Einzelangaben zur Verfügung, wie die Regel überschritten wurde. Dies können die folgenden Werte sein:

- CPU: Die gesamte USR CPU plus die SYS CPU-Zeit in Sekunden.
- Locks: Die Summe der Sperren, die von der Anwendung gehalten werden
- Rowssel: Die Summe der Zeilen, die von der Anwendung ausgewählt werden
- Rowsread: Die Summe der Zeilen, die von der Anwendung gelesen werden
- Idle: Zeit, in der die Anwendung inaktiv war
- ET: Zeit, die seit dem Start der aktuellen UOW der Anwendung vergangen ist (das uowtime-Zeitlimit wurde überschritten)

ERROR

Der Datensatz ERROR wird geschrieben, wenn der Governor-Dämon abgeschaltet werden muss.

WARNING

Der Datensatz WARNING wird in den folgenden Situationen in das Governor-Protokoll geschrieben:

- Die API sqlefrce wurde aufgerufen, um eine Anwendung zu erzwingen aber es wurde ein positiver SQLCODE-Wert zurückgegeben.
- Ein Momentaufnahmenaufruf gab einen positiven SQLCODE-Wert zurück, der nicht 1611 ("SQL1661 Es wurden keine Daten zurückgegeben") war.
- Ein Momentaufnahmenaufruf gab einen negativen SQLCODE-Wert zurück, der nicht -1224 ("SQL1224N Ein Datenbankagent konnte nicht für die Anforderung gestartet werden, oder er wurde aufgrund eines Systemabschlusses der Datenbank bzw. durch den Befehl FORCE been-

det") oder -1032 ("SQL1032N Der Befehl DB2START wurde nicht abgesetzt") war. Diese Rückkehrcodes treten auf, wenn eine zuvor aktive Instanz inaktiv wurde.

- In einer UNIX-Umgebung wurde ein Versuch unternommen, eine Signalaroutine zu installieren. Der Versuch ist fehlgeschlagen.

ACCOUNT

Der Datensatz ACCOUNT wird in den folgenden Situationen in das Governor-Protokoll geschrieben:

- Der Wert von agent_usr_cpu oder agent_sys_cpu für diese Anwendung wurde geändert seit der letzte Datensatz ACCOUNT für diese Anwendung geschrieben wurde.
- Es wird festgestellt, dass die Anwendung nicht mehr aktive ist.

Der Datensatz ACCOUNT hat folgendes Format:

<authentifizierungs_id> <anwendungs_id> <anwendungsname> <verbindungs-dauer> <agent_usr_cpu delta> <agent_sys_cpu delta>

SCHEDGRP

Der Datensatz SCHEDGRP wird immer unter den folgenden Umständen geschrieben:

- Die Anwendung wird einer Zeitplanungsgruppe hinzugefügt.
- Die Anwendung wird von einer Zeitplanungsgruppe in eine andere versetzt.

Der Datensatz SCHEDGRP hat folgendes Format:

<anwendungsname> <authentifizierungs_id> <anwendungs_id>
<konfigurationszeile> <einschränkung_überschritten>

Dabei gilt Folgendes:

<konfigurationszeile>

Gibt die Zeilennummer in der Governor-Konfigurationsdatei an, in der sich die Regel befindet, die das Terminieren der Anwendung verursacht.

<einschränkung_überschritten>

Stellt Einzelangaben zur Verfügung, wie die Regel überschritten wurde. Dies können die folgenden Werte sein:

- CPU: Die gesamte USR CPU plus die SYS CPU-Zeit in Sekunden.
- Locks: Die Summe der Sperren, die von der Anwendung gehalten werden
- Rowssel: Die Summe der Zeilen, die von der Anwendung ausgewählt werden
- Rowsread: Die Summe der Zeilen, die von der Anwendung gelesen werden
- Idle: Zeit, in der die Anwendung inaktiv war
- ET: Zeit, die seit dem Start der aktuellen UOW der Anwendung vergangen ist (das uowtime-Zeitlimit wurde überschritten)

Da standardisierte Werte geschrieben werden, können Sie die Protokolldateien nach unterschiedlichen Aktionstypen abfragen. Das Feld *Nachricht* enthält andere nicht standardisierte Informationen, die sich nach dem im Feld *Satztyp* angegebenen Wert ändern. Ein Eintrag des Typs FORCE oder NICE beispielsweise zeigt Informationen zur Anwendung im Feld *Nachricht* an, während ein Eintrag des Typs ERROR eine Fehlernachricht enthält.

Eine Beispielprotokolldatei könnte folgendermaßen aussehen:

```
1995-12-11 14.54.52 0 START      Database = TQTEST
1995-12-11 14.54.52 0 READCFG    Config = /u/db2instance/sqllib/tqtest.cfg
1995-12-11 14.54.53 0 ERROR      SQLMON Error: SQLCode = -1032
1995-12-11 14.54.54 0 ERROR      SQLMONSZ Error: SQLCode = -1032
```

Zugehörige Konzepte:

- „Abfragen von Governor-Protokolldateien“ auf Seite 124
- „Das Dienstprogramm Governor“ auf Seite 107

Abfragen von Governor-Protokolldateien

Jeder Governor-Dämon schreibt in eine eigene Protokolldatei. Mit dem Dienstprogramm `db2govlg` können Sie die Protokolldatei abfragen. Sie können die Protokolldateien einer einzigen oder aller Datenbankpartitionen, nach Datum und Uhrzeit sortiert, auflisten. Außerdem können Sie Abfragen auf der Grundlage des Protokollfelds *RecType* (Satztyp) durchführen. `db2govlg` hat folgende Syntax:

```
db2govlg protokolldatei [nodenum knotennummer] [rectype satztyp]
```

Abbildung 11. Syntax von `db2govlg`

Der Befehl hat die folgenden Parameter:

protokolldatei

Der Basisdateiname der Protokolldatei (oder Dateien), die Sie abfragen möchten

nodenum knotennummer

Die Knotennummer der Datenbankpartition, auf der Governor aktiv ist

rectype satztyp

Der Satztyp, die Sie abfragen möchten. Es gibt folgende Satztypen:

- START
- READCFG
- STOP
- FORCE
- NICE
- ERROR
- WARNING
- ACCOUNT

Für die Verwendung dieses Dienstprogramms gibt es keinerlei Berechtigungseinschränkungen. Dadurch können alle Benutzer abfragen, ob Governor ihre Anwendung geändert hat. Wenn Sie den Zugriff auf dieses Dienstprogramm beschränken möchten, können Sie die Gruppenberechtigungen für die Datei `db2govlg` ändern.

Zugehörige Konzepte:

- „Governor-Protokolldateien“ auf Seite 120
- „Das Dienstprogramm Governor“ auf Seite 107

Zugehörige Referenzen:

- „db2govlg - DB2 governor log query command“ in *Command Reference*

Kapitel 7. Maximieren der E/A-Effizienz

Reduzieren der E/A-Operationen durch Optimieren der SQL- und XQuery-Anweisungen

Angeben von Zeilenblockung zur Verringerung des Systemaufwands

Zeilenblockung verringert den Systemaufwand des Datenbankmanagers für Cursor durch Abrufen eines *Blocks* von Zeilen in einer einzigen Operation.

Anmerkung: Der Zeilenblock, den Sie angeben, ist eine Anzahl von Seiten im Speicher. Dabei handelt es sich nicht um einen Block einer mehrdimensionalen MDC-Tabelle (MDC - Mehrdimensionales Clustering), der wiederum einem EXTENTSIZE-Speicherbereich auf dem Datenträger zugeordnet ist.

Die Stufen der Zeilenblockung werden in den folgenden Argumenten für die Befehle BIND oder PREP angegeben:

UNAMBIG

Die Blockung wird für Lesezugriffscursor und für Cursor, die nicht mit „FOR UPDATE OF“ definiert sind, verwendet. Mehrdeutige Cursor werden als aktualisierbar betrachtet.

ALL Die Blockung wird für Lesezugriffscursor und für Cursor, die nicht mit „FOR UPDATE OF“ definiert sind, verwendet. Mehrdeutige Cursor werden wie Lesezugriffscursor behandelt.

NO Es wird keine Blockung für Cursor verwendet. Mehrdeutige Cursor werden wie Lesezugriffscursor behandelt.

Voraussetzungen:

Zwei Konfigurationsparameter des Datenbankmanagers müssen entsprechend definiert werden. Beide Werte werden als Anzahl von Seiten im Speicher definiert. Notieren Sie sich die Werte dieser Parameter zur Verwendung in Blockgrößenberechnungen.

- Der Konfigurationsparameter *aslheapsz* des Datenbankmanagers gibt die Größe des Zwischenspeichers der Anwendungsunterstützungsebene an.
- Der Konfigurationsparameter *rqrioblk* des Datenbankmanagers gibt die Größe des Kommunikationspuffers zwischen fernen Anwendungen und ihren Datenbankagenten auf dem Datenbankserver an.

Vorgehensweise:

Gehen Sie wie folgt vor, um die Zeilenblockung zu definieren:

1. Schätzen Sie mit Hilfe der Werte der Konfigurationsparameter *aslheapsz* und *rqrioblk* die Anzahl der Zeilen ab, die für jeden Block zurückgegeben werden. In den beiden folgenden Formeln steht *azl* jeweils für die Ausgabezeilenlänge.

- Verwenden Sie die folgende Formel für lokale Anwendungen:

$$\text{Zeilen pro Block} = \text{aslheapsz} * 4096 / \text{azl}$$

Die Anzahl von Byte pro Seite beträgt 4 096.

- Verwenden Sie die folgende Formel für ferne Anwendungen:

$$\text{Zeilen pro Block} = \text{rqrioblk} / \text{azl}$$

2. Zur Aktivierung der Zeilenblockung geben Sie ein entsprechendes Argument für die Option BLOCKING in den Befehlen PREP oder BIND an.

Wenn Sie keine Option BLOCKING angeben, wird standardmäßig der Zeilenblockungstyp UNAMBIG verwendet. Bei Verwendung des Befehlszeilenprozessors und von Call Level Interface ist der Standardtyp der Zeilenblockung ALL.

Anmerkung: Bei Verwendung der Klausel FETCH FIRST n ROWS ONLY oder OPTIMIZE FOR n ROWS in einer SELECT-Anweisung entspricht die Anzahl der Zeilen pro Block dem kleinsten der folgenden Werte:

- Der mit der oben angegebenen Formel berechnete Wert
- Der Wert von n in der Klausel FETCH FIRST
- Der Wert von n in der Klausel OPTIMIZE FOR

Zugehörige Referenzen:

- „aslheapsz - Zwischenspeichergröße für Anwendungsunterstützungsebene “ auf Seite 461
- „rqrioblk - E/A-Blockgröße für Clients “ auf Seite 464

Effiziente SELECT-Anweisungen

Da SQL eine flexible Abfragesprache höherer Ebene ist, haben Sie die Möglichkeit, mehrere verschiedene SELECT-Anweisungen zum Abrufen der gleichen Daten zu schreiben. Die Leistung kann jedoch für die verschiedenen Formen der Anweisung sowie für die verschiedenen Optimierungsklassen unterschiedlich sein.

Berücksichtigen die folgenden Richtlinien für SELECT-Anweisungen:

- Geben Sie nur Spalten an, die Sie benötigen. Obwohl es einfacher ist, alle Spalten mit einem Stern (*) anzugeben, führt dies zu überflüssigem Verarbeitungsaufwand und zur Rückgabe unnötiger Spalten.
- Verwenden Sie Vergleichselemente, die die Antwortmenge auf nur die Zeilen einschränken, die Sie benötigen.
- Wenn die Anzahl der Zeilen, die Sie benötigen, bedeutend kleiner ist als die Gesamtanzahl der Zeilen, die abgerufen werden könnten, geben Sie die Klausel OPTIMIZE FOR an. Diese Klausel wirkt sich sowohl auf die Auswahl der Zugriffspläne als auch auf die Anzahl der im Kommunikationspuffer geblockten Zeilen aus.
- Wenn die Anzahl der abzurufenden Zeilen klein ist, geben Sie nur die Klausel OPTIMIZE FOR k ROWS an. Die Klausel FETCH FIRST n ROWS ONLY ist nicht erforderlich. Wenn n jedoch groß ist und Sie die ersten k Zeilen schnell abrufen wollen und eine mögliche Verzögerung für die nachfolgenden k Zeilen in Kauf nehmen, geben Sie beide Klauseln an. Die Größe der Kommunikationspuffer wird in Abhängigkeit vom kleineren der Werte für n und k festgelegt. Das folgende Beispiel zeigt beide Klauseln:

```
SELECT EMPNAME, SALARY FROM EMPLOYEE
ORDER BY SALARY DESC
FETCH FIRST 100 ROWS ONLY
OPTIMIZE FOR 20 ROWS
```

- Zur Verwendung der Zeilenblockung geben Sie die Klausel FOR READ ONLY oder FOR FETCH ONLY an, um die Leistung zu verbessern. Dadurch wird

außerdem der gleichzeitige Zugriff verbessert, weil für die abgerufenen Spalten zu keiner Zeit exklusive Sperren aktiviert sind. Zusätzliches Umschreiben der Abfrage kann ebenfalls stattfinden. Durch die Angabe der Klausel FOR READ ONLY oder FOR FETCH ONLY sowie durch die BIND-Option BLOCKING ALL lässt sich die Leistung von Abfragen für Kurznamen in einem föderierten System in ähnlicher Weise verbessern.

- Geben Sie für Cursor, die durch positionierte Aktualisierungen aktualisiert werden, die Klausel FOR UPDATE OF an, um dem Datenbankmanager zu ermöglichen, gleich zu Anfang angemessenere Sperrstufen zu wählen und potenzielle Deadlocks zu vermeiden. Beachten Sie, dass FOR UPDATE-Cursor die Zeilenblockung nicht nutzen können.
- Für Cursor, die durch Aktualisierungen mit Suche aktualisiert werden, können Sie Deadlocks vermeiden und trotzdem die Zeilenblockung ermöglichen, indem Sie Sperren des Modus U durch die Klauseln FOR READ ONLY und USE AND KEEP UPDATE LOCKS für die betroffenen Zeilen erzwingen.
- Vermeiden Sie nach Möglichkeit Umsetzungen numerischer Datentypen. Beim Vergleichen von Werten ist es in der Regel effizienter, zwei Werte mit demselben Datentyp miteinander zu vergleichen. Wenn Umwandlungen erforderlich sind, können Ungenauigkeiten aufgrund begrenzter Präzision und Leistungseinbußen aufgrund von Umwandlungen, die zur Laufzeit ausgeführt werden müssen, die Folge sein.

Verwenden Sie, falls möglich, die folgenden Datentypen:

- Zeichen (CHAR) anstatt Zeichen variierender Länge (VARCHAR) für kurze Spalten
- Ganze Zahlen (INTEGER) anstatt Gleitkommazahlen (FLOAT) oder Dezimalzahlen (DECIMAL)
- Datum/Uhrzeit (Datetime) anstatt Zeichen
- Numerische Typen anstatt Zeichen
- Lassen Sie zur Verringerung der Wahrscheinlichkeit von Sortieroperationen Klauseln oder Operationen wie DISTINCT oder ORDER BY weg, wenn solche Operationen nicht erforderlich sind.
- Wählen Sie eine einzige Zeile aus, wenn Sie das Vorhandensein von Zeilen prüfen wollen. Öffnen Sie dazu einen Cursor und rufen Sie eine Zeile ab (FETCH) oder führen Sie eine Auswahl für eine Zeile (SELECT INTO) durch. Vergessen Sie nicht, auf den SQLCODE-Wert -811 zu prüfen, wenn mehr als eine Zeile gefunden wird.

Sofern Sie nicht wissen, dass die Tabelle sehr klein ist, vermeiden Sie die folgende Anweisung zur Prüfung auf einen Nichtnullwert:

```
SELECT COUNT(*) FROM TABLENAME
```

Das Zählen aller Zeilen wirkt sich bei großen Tabellen negativ auf die Leistung aus.

- Wenn das Aktualisierungsaufkommen gering ist und die Tabellen umfangreich sind, sollten Sie Indizes für Spalten definieren, die häufig in Vergleichselementen verwendet werden.
- Ziehen Sie die Verwendung einer IN-Liste in Betracht, wenn dieselbe Spalte in mehreren PREDICATE-Klauseln erscheint. Bei großen IN-Listen, die mit Hostvariablen verwendet werden, können Verarbeitungsschleifen für Untergruppen der Hostvariablen die Leistung verbessern.

Die folgenden Vorschläge gelten insbesondere für SELECT-Anweisungen, die auf mehrere Tabellen zugreifen.

- Verwenden Sie Joinvergleichselemente für den Join von Tabellen. Ein Joinvergleichselement ist ein Vergleich zwischen zwei Spalten verschiedener Tabellen in einem Join.
- Definieren Sie Indizes für die Spalten in dem Joinvergleichselement, um eine effizientere Verarbeitung des Joins zu ermöglichen. Indizes sind außerdem für UPDATE- und DELETE-Anweisungen förderlich, die SELECT-Anweisungen enthalten, die auf mehrere Tabellen zugreifen.
- Vermeiden Sie nach Möglichkeit Ausdrücke oder Klauseln OR mit Joinvergleichselementen, weil der Datenbankmanager einige Jointechniken nicht nutzen kann. Dies kann dazu führen, dass nicht die effizienteste Joinmethode gewählt wird.
- In einer Umgebung mit partitionierten Datenbanken stellen Sie nach Möglichkeit sicher, dass beide Tabellen, die verknüpft werden, über die Joinspalte partitioniert sind.

Stichprobendaten in SQL- und XQuery-Abfragen

Datenbanken wachsen so stark an und die Abfragen für diese Datenbanken werden so komplex, dass es häufig unpraktisch und manchmal auch unnötig ist, auf sämtliche für eine Abfrage relevanten Daten zuzugreifen. In einigen Fällen interessiert sich ein Benutzer für allgemeine Trends oder Muster, bei denen angenäherte Antworten innerhalb gewisser Fehlertoleranzen vollkommen ausreichen. Eine Möglichkeit, solche Abfragen zu beschleunigen, besteht darin, die Abfrage an einer zufälligen Stichprobe der Datenbank auszuführen. Mit DB2 können Sie eine effiziente Datenstichprobe in SQL- und XQuery-Abfragen erstellen, wodurch sich potenzielle Leistungsverbesserungen für umfangreiche Abfragen in mehreren Größenordnungen ergeben und gleichzeitig ein hoher Grad an Genauigkeit bewahrt werden kann.

Die gängigste Anwendung von Stichprobendaten betrifft Abfragen mit Spaltenfunktionen wie AVG, SUM und COUNT, bei denen vernünftig genaue Ergebniswerte aus einer Stichprobe der Daten gewonnen werden können. Die Stichprobenerstellung kann auch dazu dienen, eine zufällige Teilmenge der tatsächlichen Zeilen einer Tabelle zu Prüfungszwecken abzurufen oder Data Mining- und Analyseoperationen zu beschleunigen.

DB2 stellt zwei Methoden zur Stichprobenerstellung bereit: Stichproben auf Zeilenebene und Stichproben auf Blockebene.

Stichprobenentnahme auf Zeilenebene nach Bernoulli:

Die Bernoulli-Stichprobe auf Zeilenebene ruft eine Stichprobe von P Prozent der Tabellenzeilen mit Hilfe eines als Suchargument verwendbaren Vergleichselements ab, das jede Zeile mit einer Wahrscheinlichkeit von $P/100$ einschließt und sie mit einer Wahrscheinlichkeit von $1-P/100$ ausschließt.

Die Bernoulli-Stichprobe auf Zeilenebene generiert unabhängig von den Datenwerthäufungen (Clusterbildung) immer eine gültige, zufällige Stichprobe. Allerdings ist die Leistung dieser Stichprobenmethode sehr gering, wenn kein Index verfügbar ist, da jede Zeile abgerufen und das Vergleichselement der Stichprobe auf sie angewandt werden muss. Wenn kein Index vorhanden ist, ergeben sich keine E/A-Einsparungen gegenüber der Ausführung einer Abfrage ohne Stichprobenerstellung. Wenn ein Index vorhanden ist, wird eine bessere Leistung mit dieser Art der Stichprobe erzielt, weil das Vergleichselement der Stichprobe auf die

Satz-IDs (Zeilen-ID, RIDs) innerhalb der Indexblattseiten angewandt wird. Im Normalfall erfordert dies nur eine E/A-Operation pro ausgewählter RID und eine E/A-Operation pro Indexblattseite.

Stichprobenentnahme auf Systemseitenebene:

Stichprobe auf Systemseitenebene ist der Stichprobe auf Zeilenebene ähnlich, jedoch werden bei ihr Stichproben der Seiten und nicht der Zeilen erstellt. Eine Seite wird mit einer Wahrscheinlichkeit von $P/100$ in die Stichprobe eingeschlossen. Das Einschließen einer Seite bedeutet, dass alle Zeilen dieser Seite mit eingeschlossen werden.

Die Leistung der Stichprobenentnahme auf Systemseitenebene ist hervorragend, da nur eine E/A-Operation für jede Seite erforderlich ist, die in die Stichprobe eingeschlossen wird. Im Vergleich zu keiner Stichprobenentnahme verbessert die Stichprobe auf Seitenebene die Leistung um Größenordnungen. Allerdings ist die Genauigkeit kumulierter Schätzung bei Seitenstichproben tendenziell geringer als bei Zeilenstichproben. Diese Diskrepanz in der Genauigkeit wird am deutlichsten, wenn es viele Zeilen pro Block gibt oder die in der Abfrage angegebenen Zeilen einen hohen Grad der Clusterbildung innerhalb der Seiten aufweisen.

Die beste Stichprobenmethode für einen bestimmten Zweck wird durch die Zeitbegrenzung eines Benutzers sowie den gewünschten Grad an Genauigkeit festgelegt.

Angeben der Stichprobenmethode:

Zur Ausführung einer Abfrage an einer zufälligen Stichprobe von Daten aus einer Tabelle können Sie die Klausel `TABLESAMPLE` der Tabellenverweisklausel in einer SQL-Anweisung verwenden. Zur Angabe der Stichprobenmethode können Sie die Schlüsselwörter `BERNOULLI` oder `SYSTEM` verwenden.

Das Schlüsselwort `BERNOULLI` gibt an, dass eine Bernoulli-Stichprobe auf Zeilenebene auszuführen ist.

Das Schlüsselwort `SYSTEM` gibt an, dass eine Stichprobe auf Systemseitenebene auszuführen ist, sofern nicht das Optimierungsprogramm feststellt, dass es effizienter ist, stattdessen eine Bernoulli-Stichprobe auf Zeilenebene auszuführen.

Zugehörige Referenzen:

- „Subselect“ in *SQL Reference, Volume 1*

Optimieren der Sortierleistung

Da Abfragen häufig sortierte oder gruppierte Ergebnisse erfordern, werden häufig Sortierungen angefordert, und eine geeignete Konfiguration der Sortierspeicherbereiche spielt für eine gute Abfrageleistung eine wichtige Rolle. Sortieren ist in folgenden Fällen erforderlich:

- Es gibt keinen Index, um eine angeforderte Reihenfolge (z. B. durch eine Anweisung `SELECT` mit einer Klausel `ORDER BY`) der Daten herzustellen.
- Es gibt einen Index, aber Sortieren ist effizienter als der Zugriff über den Index.
- Ein Index wird erstellt.
- Ein Index wird gelöscht, wodurch eine Sortierung der Indexseitennummern verursacht wird.

Elemente mit Auswirkung auf das Sortieren

Die folgenden Faktoren wirken sich auf die Sortierleistung aus:

- Die Einstellungen für die folgenden Datenbankkonfigurationsparameter:
 - Die Zwischenspeichergröße für Sortierlisten (*sortheap*) gibt die Größe des Speichers an, der für jede Sortierung verwendet wird.
 - Der Schwellenwert für Sortierspeicher (*sheapthres*) und der Schwellenwert für Sortierspeicher für gemeinsame Sortiervorgänge (*sheapthres_shr*) steuern die Gesamtgröße des Speichers, der für das Sortieren in der gesamten Instanz für alle Sortiervorgänge verfügbar ist.
- Die Anzahl an Anweisungen in einer Auslastung, für die eine große Menge an Sortierungen erforderlich sind
- Vorhandene oder fehlende Indizes, die zur Vermeidung unnötiger Sortiervorgänge dienen könnten
- Verwendung von Anwendungslogik, die das Sortieraufkommen nicht minimiert
- Paralleles Sortieren, das die Leistung der Sortierungen erhöht, aber nur auftreten kann, wenn die Anweisung *partitionsinterne Parallelität* verwendet
- Ob die Sortierung *mit* oder *ohne* Überlauf stattfindet. Wenn die sortierten Daten nicht vollständig in den Sortierspeicher passen, das heißt in den Speicherblock, der jedes Mal zugeordnet wird, wenn eine Sortierung durchgeführt wird, laufen die Daten in temporäre Datenbanktabellen über. Sortierungen, die keinen Überlauf verursachen, zeigen stets eine bessere Leistung als solche, bei denen ein Überlauf auftritt.
- Ob die Ergebnisse der Sortierung *über eine Pipe* oder *nicht über eine Pipe* geleitete werden. Wenn sortierte Daten direkt zurückgegeben werden können, ohne dass eine temporäre Tabelle zum Speichern der endgültigen sortierten Liste von Daten erforderlich ist, handelt es sich um eine Sortiervorgang mit Piping (d. h. die Daten werden über eine Pipe zurückgegeben). Wenn die sortierten Daten über eine temporäre Tabelle zurückgegeben werden müssen, wird dies als Sortierung ohne Piping bezeichnet. Eine Sortierung mit Piping ist immer effizienter als eine Sortierung ohne Piping.

Beachten Sie außerdem, dass bei einer Sortierung mit Piping der Sortierzwischenspeicher nicht freigegeben wird, bevor die Anwendung den dieser Sortierung zugeordneten Cursor schließt. Eine Sortierung mit Piping kann weiter Speicher belegen, bis der Cursor geschlossen wird.

Im Allgemeinen sollte der für die gesamte Instanz verfügbare Sortierspeicher (*sheapthres*) so groß wie möglich sein, ohne übermäßiges Seitenauslagern (Paging) zu verursachen. Obwohl eine Sortierung vollständig im Sortierspeicher durchgeführt werden kann, kann dies zu einem übermäßigen Auslagern von Seiten führen. In diesem Fall geht der Vorteil eines großen Sortierzwischenspeichers verloren. Aus diesem Grund sollten Sie einen Betriebssystemmonitor verwenden, um alle Änderungen in der Auslagerung von Seiten durch das System zu verfolgen, wenn Sie die Konfigurationsparameter für das Sortieren anpassen.

Techniken zur Verwaltung der Sortierleistung

Ermitteln Sie bestimmte Anwendungen und Anweisungen, bei denen die Sortierung ein wesentliches Leistungsproblem darstellt:

1. Richten Sie Ereignismonitore (Event Monitors) auf Anwendungs- und Anweisungsebene ein, um Unterstützung bei der Ermittlung von Anwendungen mit der längsten Gesamtsortierzeit zu erhalten.

2. Ermitteln Sie innerhalb dieser Anwendungen die Anweisungen mit der längsten *Gesamtsortierzeit*.
Sie können auch die EXPLAIN-Tabellen durchsuchen, um die Abfragen mit Sortieroperationen zu ermitteln.
3. Verwenden Sie diese Anweisungen als Eingabe für den Designadvisor, der Indizes identifiziert und optional erstellt, um den Sortierbedarf zu reduzieren.

Verwenden Sie den Datenbanksystemmonitor und Vergleichstesttechniken zur Unterstützung bei der Einstellung der Konfigurationsparameter *sortheap* und *sheapthres*. Gehen Sie für jeden Datenbankmanager und die zugehörigen Datenbanken folgendermaßen vor:

1. Erstellen Sie eine repräsentative Auslastung und führen Sie sie aus.
2. Sammeln Sie für jede betroffene Datenbank Durchschnittswerte für die folgenden Leistungsvariablen über den Auslastungszeitraum der Vergleichstests:
 - Gesamter verwendeter Sortierspeicher
 - Aktive Sortiervorgänge
3. Setzen Sie den Parameter *sortheap* auf den Durchschnittswert für den *gesamten verwendeten Sortierspeicher* für jede Datenbank.

Anmerkung: Nachdem die DB2-Binärsortierung mit Teilschlüssel verbessert wurde, sodass auch nicht ganzzahlige Datentypschlüssel unterstützt werden, ist beim Sortieren langer Schlüssel zusätzlicher Speicher erforderlich. Wenn lange Schlüssel für Sortierungen verwendet werden, müssen Sie möglicherweise den Wert für den Konfigurationsparameter *sortheap* erhöhen.

4. Definieren Sie den Wert für *sheapthres*. Gehen Sie wie folgt vor, um eine angemessene Größe zu schätzen:
 - a. Stellen Sie fest, welche Datenbank in der Instanz über den größten Wert für *sortheap* verfügt.
 - b. Ermitteln Sie die durchschnittliche Größe des Sortierspeichers für diese Datenbank.
Wenn die Ermittlung des Durchschnittswerts zu aufwendig ist, verwenden Sie als Wert 80% des maximalen Sortierspeichers.
 - c. Setzen Sie den Wert für *sheapthres* auf die Durchschnittsanzahl der aktiven Sortiervorgänge multipliziert mit der oben berechneten Durchschnittsgröße des Sortierspeichers.
Dies ist die empfohlene Anfangseinstellung. Anschließend können Sie mit Hilfe von Vergleichstests diesen Wert optimieren.

Zugehörige Referenzen:

- „*sortheap* - Zwischenspeichergröße für Sortierlisten ” auf Seite 458
- „*sheapthres* - Schwellenwert für Sortierspeicher ” auf Seite 456
- „*sheapthres_shr* - Schwellenwert für Sortierspeicher für gemeinsame Sortiervorgänge ” auf Seite 446

Pufferpools und Vorablesezugriff

Pufferpoolverwaltung

Ein Pufferpool ist ein Speicher, der beim Lesen von Daten vom Datenträger oder beim Ändern von Daten zum Zwischenspeichern von Tabellen- und Indexdaten-seiten verwendet wird. Der Pufferpool verbessert die Leistung des Datenbanksys-

tems, indem er die Möglichkeit schafft, auf Daten im Hauptspeicher anstatt auf der Platte zuzugreifen. Da der Zugriff auf den Hauptspeicher wesentlich schneller als der Plattenzugriff arbeitet, ist die Leistung umso höher, je seltener der Datenbankmanager Daten von der Platte lesen bzw. auf die Platte schreiben muss. Aufgrund der Tatsache, dass die meisten Bearbeitungsschritte für Daten in Pufferpools erfolgen, ist die Konfiguration von Pufferpools der wichtigste Einzelbereich bei der Optimierung. Nur LOB-Daten und Langfelddaten werden nicht in einem Pufferpool bearbeitet.

Wenn eine Anwendung auf eine Zeile einer Tabelle zum ersten Mal zugreift, liest der Datenbankmanager die Seite, die die Zeile enthält, in den Pufferpool ein. Wenn in der Folge eine Anwendung Daten anfordert, sucht sie der Datenbankmanager im Pufferpool. Befinden sich die angeforderten Daten im Pufferpool, können sie ohne Plattenzugriff abgerufen werden, was eine höhere Geschwindigkeit ermöglicht.

Der Speicher für den Pufferpool wird zugeordnet, wenn eine Datenbank aktiviert wird oder wenn die erste Anwendung die Verbindung zur Datenbank herstellt. Pufferpools können auch erstellt, gelöscht oder in der Größe geändert werden, während der Datenbankmanager aktiv ist. Wenn Sie das Schlüsselwort IMMEDIATE in der Anweisung ALTER BUFFERPOOL zur Vergrößerung des Pufferpools verwenden, wird der Speicher zugeordnet, sobald Sie den Befehl eingegeben haben, sofern der Speicher verfügbar ist. Wenn der Speicher nicht verfügbar ist, findet die Änderung statt, wenn alle Anwendungen ihre Verbindungen getrennt haben und die Datenbank erneut aktiviert wird. Wenn Sie den Pufferpool verkleinern, wird der Speicher zum Zeitpunkt des Commits freigegeben. Wenn alle Anwendungen getrennt sind, wird der Pufferpoolspeicher freigegeben.

Anmerkung: Zur Verringerung der Notwendigkeit, den Wert des Datenbankkonfigurationsparameters *dbheap* zu erhöhen, wenn der Pufferpool vergrößert wird, wird beinahe der gesamte Pufferpoolspeicher, der auch Seitendeskriptoren, Pufferpoolsdeskriptoren und die Hashtabellen umfasst, dem gemeinsamen Datenbankspeicher entnommen und automatisch in der Größe angepasst.

Um sicherzustellen, dass in allen Situationen ein geeigneter Pufferpool verfügbar ist, erstellt DB2 kleine Systempufferpools, d. h. je einen mit der Seitengröße 4 KB, 8 KB, 16 KB und 32 KB. Die Größe der Pufferpools beträgt 16 Seiten. Diese Pufferpools sind für den Benutzer verdeckt. Sie treten weder in den Systemkatalogen noch in den Pufferpoolsystemdateien auf. Diese Pufferpools können von Ihnen auch nicht direkt verwendet oder geändert werden, sondern sie werden von DB2 in den folgenden Situationen genutzt:

- Wenn ein Pufferpool der erforderlichen Seitengröße nicht aktiv ist, weil nicht genügend Speicher zu seiner Erstellung verfügbar war, nachdem eine Anweisung CREATE BUFFERPOOL mit dem Schlüsselwort IMMEDIATE ausgeführt wurde.

Eine Nachricht wird in das Benachrichtigungsprotokoll für die Systemverwaltung geschrieben. Falls erforderlich, werden Tabellenbereiche einem Systempufferpool zugeordnet. Die Leistung könnte sich beträchtlich verringern.

- Wenn die normalen Pufferpools bei der Herstellung einer Datenbankverbindung nicht aktiviert werden können.

Dieses Problem hat wahrscheinlich eine ernsthafte Ursache, wie zum Beispiel, dass kein Speicher mehr verfügbar ist. Obwohl DB2 auf Grund der Systempufferpools den vollen Funktionsumfang bewahrt, verschlechtert sich die Leistung erheblich. Sie sollten dieses Problem unverzüglich untersuchen. Wenn die-

ser Fall eintritt, empfangen Sie eine Warnung und eine Nachricht wird in das Benachrichtigungsprotokoll für die Systemverwaltung geschrieben.

Seiten verbleiben im Pufferpool, bis die Datenbank gestoppt wird oder der von einer Seite belegte Speicherbereich im Pufferpool für eine andere Seite benötigt wird. Die folgenden Bedingungen bestimmen, welche Seite entfernt wird, um ein andere Seite einzulesen:

- Der letzte Verweis auf die Seite
- Die Wahrscheinlichkeit, dass der letzte Agent, der die Seite gelesen hat, erneut auf die Seite zugreift
- Der Typ der Daten auf der Seite
- Der Änderungsstatus der Seite, d. h. ob sie im Speicher geändert, jedoch noch nicht auf Platte geschrieben wurde (geänderte Seiten werden immer auf der Festplatte gespeichert, bevor sie im Pufferpool überschrieben werden)

Damit erneut auf Seiten im Speicher zugegriffen werden kann, werden geänderte Seiten nicht aus dem Pufferpool entfernt, nachdem sie auf die Festplatte geschrieben wurden, sofern der von ihnen belegte Speicherplatz nicht benötigt wird.

Wenn Sie einen Pufferpool erstellen, ist die Standardseitengröße die Größe, die bei der Datenbankerstellung angegeben wurde; sie können aber auch eine andere Seitengröße angeben. Da Seiten in einen Pufferpool nur eingelesen werden können, wenn die Seitengröße des Tabellenbereichs mit der Seitengröße des Pufferpools übereinstimmt, sollte die Seitengröße Ihrer Tabellenbereiche auch die Seitengröße sein, die Sie für Pufferpools angeben. Nach der Erstellung eines Pufferpools können Sie seine Seitengröße nicht mehr ändern. Sie müssen einen neuen Pufferpool mit einer anderen Seitengröße erstellen.

Mit dem Speichertracker, der über db2mtrk aufgerufen wird, können Sie die Menge an den Pufferpools zugeordnetem Datenbankspeicher anzeigen. In der folgenden Beispielausgabe db2mtrk wird ein vom Benutzer erstellter Pufferpool, der durch die Nummer "1" in der runden Klammer angegeben wird, sowie vier Systempufferpools, die durch die Seitengröße in der runden Klammer angegeben werden, erstellt:

```
> db2mtrk -d
Speicherüberwachung am 2005/10/24 um 12:47:26
```

Speicher für die Datenbank: XMLDB

utilh	pckcacheh	catcacheh	bph (1)	bph (S32K)	bph (S16K)	bph (S8K)
64,0 K	576,0 K	64,0 K	4,2 M	576,0 K	320,0 K	192,0 K
bph (S4K)	shsorth	lockh	dbh	other		
128,0 K	0	640,0 K	4,2 M	192,0 K		

Anmerkung: Auf 32-Bit-Plattformen mit Windows können Sie große Pufferpools erstellen, wenn Sie Address Windowing Extensions (AWE) oder Advanced Server und Data Center Server unter Windows aktiviert haben.

Zugehörige Konzepte:

- „Pufferpoolverwaltung von Datenseiten“ auf Seite 136
- „Illustration der Datenseitenverwaltung in Pufferpools“ auf Seite 138
- „Verwaltung mehrerer Datenbankpufferpools“ auf Seite 134
- „Hauptspeicherzuordnung in DB2“ auf Seite 286

Zugehörige Referenzen:

- „db2mtrk - Memory tracker command“ in *Command Reference*

Verwaltung mehrerer Datenbankpufferpools

Obwohl für jede Datenbank mindestens ein Pufferpool erforderlich ist, können Sie für eine einzige Datenbank, die Tabellenbereiche mit mehr als einer Seitengröße hat, mehrere Pufferpools erstellen, die alle eine andere Größe oder Seitengröße aufweisen. In Abhängigkeit von der Plattform verfügt jeder Pufferpool über eine Mindestgröße. Mit dem Befehl `ALTER BUFFERPOOL` können Sie die Größe eines Pufferpools ändern.

Eine neue Datenbank verfügt über einen Standardpufferpool mit dem Namen `IBM-DEFAULTBP` mit einer allgemeinen Größe, die von der Plattform festgelegt wird, und der Standardseitengröße, die auf der bei der Datenbankerstellung angegebenen Seitengröße basiert. Die Standardseitengröße wird als Informationsdatenbank-konfigurationsparameter `pagesize` gespeichert. Wenn Sie einen Tabellenbereich mit der Standardseitengröße erstellen und ihn keinem bestimmten Pufferpool zuordnen, wird der Tabellenbereich dem Standardpufferpool zugeordnet. Sie können die Größe des Standardpufferpools und seine Attribute ändern, jedoch können Sie ihn nicht löschen.

Seitengrößen für Pufferpools

Nach der Erstellung oder Migration einer Datenbank können Sie weitere Pufferpools erstellen. Sie können die Datenbank mit einer Standardseitengröße von 8 KB erstellen. Der Standardpufferpool wird dann mit der Standardseitengröße erstellt (in diesem Fall beträgt die Seitengröße 8 KB). Alternativ dazu können Sie einen Pufferpool mit einer Seitengröße von 8 KB sowie einen oder mehrere Tabellenbereiche mit derselben Seitengröße erstellen. Bei dieser Methode ist es nicht erforderlich, bei der Erstellung der Datenbank die Standardseitengröße von 4 KB zu ändern. Sie können einen Tabellenbereich keinem Pufferpool zuordnen, der eine andere Seitengröße verwendet.

Anmerkung: Wenn Sie einen Tabellenbereich mit einer Seitengröße über 4 KB (z. B. 8 KB, 16 KB oder 32 KB) erstellen, müssen Sie ihn einem Pufferpool zuordnen, der dieselbe Seitengröße verwendet. Wenn dieser Pufferpool momentan nicht aktiv ist, versucht DB2, den Tabellenbereich vorübergehend einem anderen aktiven Pufferpool mit der gleichen Seitengröße zuzuordnen, falls einer vorhanden ist, bzw. einem der Pufferpools, die DB2 standardmäßig erstellt, wenn der erste Client eine Verbindung zur Datenbank herstellt. Ist der ursprünglich angegebene Pufferpool beim nächsten Aktivieren der Datenbank aktiv, ordnet DB2 den Tabellenbereich zu diesem Pufferpool zu.

Wenn Sie einen Pufferpool erstellen, geben Sie die Größe des Pufferpools an. Zur späteren Vergrößerung oder Verkleinerung des Pufferpools verwenden Sie die Anweisung `ALTER BUFFERPOOL`.

In einer partitionierten Datenbankumgebung besitzt jeder Pufferpool für eine Datenbank in allen Datenbankpartitionen die gleiche Standarddefinition, sofern dies nicht in der Anweisung `CREATE BUFFERPOOL` anders angegeben oder die Größe des Pufferpools für eine bestimmte Datenbankpartition mit der Anweisung `ALTER BUFFERPOOL` geändert wurde.

Vorteile großer Pufferpools

Große Pufferpools bieten die folgenden Vorteile:

- Sie ermöglichen, dass häufig angeforderte Datenseiten im Pufferpool behalten werden können. Dadurch kann schneller auf sie zugegriffen werden. Durch weniger E/A-Operationen können E/A-Konkurrenzsituationen besser vermieden werden, so dass die Antwortzeiten verbessert und die für E/A-Operationen benötigten Prozessorressourcen reduziert werden.
- Sie bieten die Möglichkeit, höhere Transaktionsgeschwindigkeiten mit derselben Antwortzeit zu erzielen.
- Sie vermeiden Konkurrenzsituationen bei der Ein-/Ausgabe für häufig verwendete Plattenspeichereinheiten zum Beispiel für Katalogtabellen, häufig verwendete Benutzertabellen und Indizes. Auch Sortierungen durch Abfragen profitieren von verminderten E/A-Konkurrenzsituationen auf Plattenspeichereinheiten, die die temporären Tabellenbereiche enthalten.

Vorteile vieler Pufferpools

Wenn eine der folgenden Bedingungen auf Ihr System zutrifft, sollten Sie nur einen einzigen Pufferpool verwenden:

- Der gesamte Pufferspeicherbereich beträgt weniger als 10 000 4-KB-Seiten.
- Es stehen keine Fachleute mit Anwendungskenntnissen für die spezialisierte Optimierung zur Verfügung.
- Sie arbeiten auf einem Testsystem.

In allen anderen Fällen sollten Sie die Verwendung mehrerer Pufferpools aus folgenden Gründen in Betracht ziehen:

- Tabellenbereiche für temporäre Tabellen können einem getrennten Pufferpool zugeordnet werden, um eine bessere Leistung für Abfragen zu erzielen, die temporären Speicherbereich benötigen, insbesondere Abfragen mit zahlreichen Sortierungen.
- Wenn auf Daten wiederholt und schnell von vielen kurzen Anwendungen mit Aktualisierungstransaktionen zugegriffen werden muss, ziehen Sie in Betracht, den Tabellenbereich, der die Daten enthält, einem getrennten Pufferpool zuzuordnen. Wenn dieser Pufferpool eine geeignete Größe hat, ist die Wahrscheinlichkeit höher, dass die Seiten im Pufferpool gefunden werden. Dies trägt zur Verkürzung der Antwortzeiten und zur Reduzierung der Transaktionskosten bei.
- Sie können Daten in getrennten Pufferpools isolieren, um eine bevorzugte Verarbeitung bestimmter Anwendungen, Daten und Indizes zu erreichen. Zum Beispiel könnte es sinnvoll sein, Tabellen und Indizes, die häufig aktualisiert werden, in einen Pufferpool einzulesen, der von anderen Tabellen und Indizes, die zwar häufig abgefragt, aber nicht häufig aktualisiert werden, getrennt ist. Durch diese Änderung werden die Auswirkungen verringert, die häufige Aktualisierungen an der ersten Gruppe von Tabellen auf häufige Abfragen auf die zweite Gruppe von Tabellen haben.
- Sie können kleinere Pufferpools für die Daten verwenden, auf die Anwendungen zugreifen, die sehr selten ausgeführt werden, insbesondere wenn eine solche Anwendung einen sehr wahlfreien Zugriff auf eine sehr umfangreiche Tabelle benötigt. In diesem Fall brauchen die Daten nicht länger als für eine einzige Abfrage im Pufferpool behalten zu werden. Es ist günstiger, einen kleinen Pufferpool für diese Daten zu verwenden und den übrigen Speicher für andere Zwecke freizugeben, wie zum Beispiel für andere Pufferpools.

- Nach der Trennung der verschiedenen Aktivitäten und Daten in verschiedene Pufferpools können gute und relativ unaufwendige Daten zur Leistungsdiagnose aus den Statistiken und aus Tracefunktionen für Benutzeraktivitäten gewonnen werden.

Zuordnung von Pufferpoolspeicher beim Start

Wenn Sie einen Pufferpool mit dem Befehl `CREATE BUFFERPOOL` erstellen oder Pufferpools mit der Anweisung `ALTER BUFFERPOOL` ändern, muss der gesamte Speicherbereich, der von allen Pufferpools benötigt wird, dem Datenbankmanager zur Verfügung stehen, damit alle Pufferpools beim Starten der Datenbank zugeordnet werden können. Wenn Sie Pufferpools erstellen oder modifizieren, während der Datenbankmanager online ist, sollte zusätzlicher Speicherplatz im globalen Datenbankspeicher verfügbar sein. Wenn Sie das Schlüsselwort `IMMEDIATE` bei der Erstellung eines neuen Pufferpools bzw. bei der Vergrößerung eines vorhandenen Pufferpools angeben und der erforderliche Speicherplatz nicht verfügbar ist, führt der Datenbankmanager die Änderung bei der nächsten Aktivierung der Datenbank durch. Auf 32-Bit-Plattformen muss der Speicherplatz verfügbar sein und kann im globalen Datenbankspeicher reserviert werden, wie in den Detailinformationen zum Datenbankkonfigurationsparameter `database_memory` beschrieben wird.

Wenn dieser Speicher beim Start einer Datenbank nicht verfügbar ist, startet der Datenbankmanager nur mit Systempufferpools (einer pro Seitengröße) mit einer Minimalgröße von 16 Seiten; außerdem wird die Warnung `SQL1478W (SQLSTATE01626)` zurückgegeben. Die Datenbank setzt den Betrieb in diesem Status fort, bis ihre Konfiguration geändert wird und die Datenbank vollständig neu gestartet werden kann. Die Leistung ist möglicherweise beeinträchtigt. Der Datenbankmanager wird mit den minimal dimensionierten Werten nur deshalb gestartet, um Ihnen die Möglichkeit zu geben, eine Verbindung zur Datenbank herzustellen und die Pufferpoolgrößen anders zu konfigurieren bzw. andere wichtige Maßnahmen durchzuführen. Starten Sie die Datenbank neu, sobald Sie diese Maßnahmen durchgeführt haben. Lassen Sie die Datenbank nicht über längere Zeit in einem solchen Status arbeiten.

Damit die Datenbank nicht nur mit Systempufferpools gestartet wird, können Sie die Registrierdatenbankvariable `DB2_OVERRIDE_BPF` verwenden, um die aktuellen Pufferpoolgrößen zu überschreiben, damit der erforderliche Gesamtspeicher verfügbar ist.

Zugehörige Konzepte:

- „Pufferpoolverwaltung“ auf Seite 131

Pufferpoolverwaltung von Datenseiten

Seiten im Puffer können entweder *im Gebrauch* oder nicht, und sie können *benutzt* (geändert) oder *sauber* (ungeändert) sein:

- Seiten, die im Gebrauch sind, werden momentan gelesen oder aktualisiert. Während eine Seite durch einen Agenten im Gebrauch ist, kann sie von anderen Agenten oder Vorabsefunktionen in der Datenbank gelesen, jedoch nicht aktualisiert werden.
- Benutzte Seiten enthalten Daten, die geändert, jedoch noch nicht auf die Platte geschrieben wurden.
- Wenn eine Seite auf die Platte geschrieben wurde, ist sie sauber, verbleibt jedoch im Pufferpool, bis Platz für neue Seiten benötigt wird.

Agenten für Seitenlöschfunktionen

In einem entsprechend optimierten System schreiben meist Agenten für Seitenlöschfunktionen benutzte Seiten auf die Platte. Agenten für Seitenlöschfunktionen führen E/A-Operationen als Hintergrundprozesse aus und ermöglichen Anwendungen eine schnellere Ausführung, weil ihre Agenten die tatsächlichen Transaktionen ausführen können. Agenten für Seitenlöschfunktionen können auch als *asynchrone Seitenlöschfunktionen* oder *asynchrone Pufferschreibfunktionen* bezeichnet werden, da sie nicht mit der Arbeit anderer Agenten koordiniert werden und nur bei Bedarf aktiv werden.

Zur Verbesserung der Leistung für aktualisierungsintensive Auslastungen kann es nützlich sein, mehr Agenten für Seitenlöschfunktionen zu konfigurieren. Die Leistung kann sich erhöhen, wenn mehr Seitenlöschagenten verfügbar sind, um benutzte Seiten auf die Festplatte zu schreiben. Dies gilt insbesondere, wenn Momentaufnahmen offenbaren, dass eine erhebliche Anzahl synchroner Schreiboperationen für Daten- oder Indexseiten im Verhältnis zur Anzahl asynchroner Schreiboperationen für Daten- oder Indexseiten stattfindet.

Seitenbereinigung und schnelle Recovery

Wenn mehr Seiten auf die Festplatte geschrieben wurden, ist die Recovery der Datenbank nach einem Systemabsturz schneller, weil der Datenbankmanager größere Teile des Pufferpools von der Festplatte rekonstruieren kann, anstatt Transaktionen aus den Datenbankprotokolldateien erneut durchzuführen.

Die Größe des Protokolls, das während der Recovery gelesen werden muss, errechnet sich aus der Differenz der Positionen der folgenden Datensätze im Protokoll:

- Der zuletzt geschriebene Protokollsatz
- Der Protokollsatz, der die älteste Änderung an Daten im Pufferpool beschreibt

Die Standardfunktionsweise der Seitenlöschfunktionen sieht vor, dass eine Bereinigung von Seiten durchgeführt wird, wenn die Größe des Protokolls, das während einer Recovery wieder angewendet werden müsste, den folgenden Maximalwert überschreitet:

$$\text{logfilsiz} * \text{softmax}$$

Dabei gilt Folgendes:

- *logfilsiz* ist die Größe der Protokolldateien
- *softmax* ist der Prozentsatz der Protokolldateien, die nach einem Datenbankabsturz wiederherzustellen sind. Wenn der Parameter *softmax* zum Beispiel den Wert 250 hat, enthalten 2,5 Protokolldateien die Änderungen, die wiederhergestellt werden müssen, wenn ein Systemabsturz auftritt.

Ermitteln Sie zur Minimierung der Zeit, die bei einer Recovery auf das Lesen der Protokolle verwendet wird, mit Hilfe des Datenbanksystemmonitors die Häufigkeit, mit der Seitenlöschfunktionen aktiv werden. Das Monitorelement *pool_lsn_gap_clns* (für Pufferpoolprotokollbereich ausgelöste Seitenlöschfunktionen) des Systemmonitors stellt diese Information bereit, wenn Sie nicht die proaktive Seitenbereinigung für Ihre Datenbank aktiviert haben. Wenn Sie diese alternative Seitenbereinigungsfunktion aktiviert haben, sollte diese Situation nicht eintreten und das Monitorelement *pool_lsn_gap_clns* zeigt immer den Wert 0.

Mit Hilfe des Monitorelements *log_held_by_dirty_pages* können Sie bestimmen, ob die Seitenlöschfunktionen nicht ausreichend Seiten bereinigen, um die vom Benut-

zer festgelegten Bindungen für die Recovery zu erfüllen. Wenn das Monitorelement *log_held_by_dirty_pages* beständig einen erheblich größeren Wert als *logfilesiz * softmax* zeigt, sind entweder mehr Seitenlöschfunktionen erforderlich oder der Wert des Parameters *softmax* muss angepasst werden.

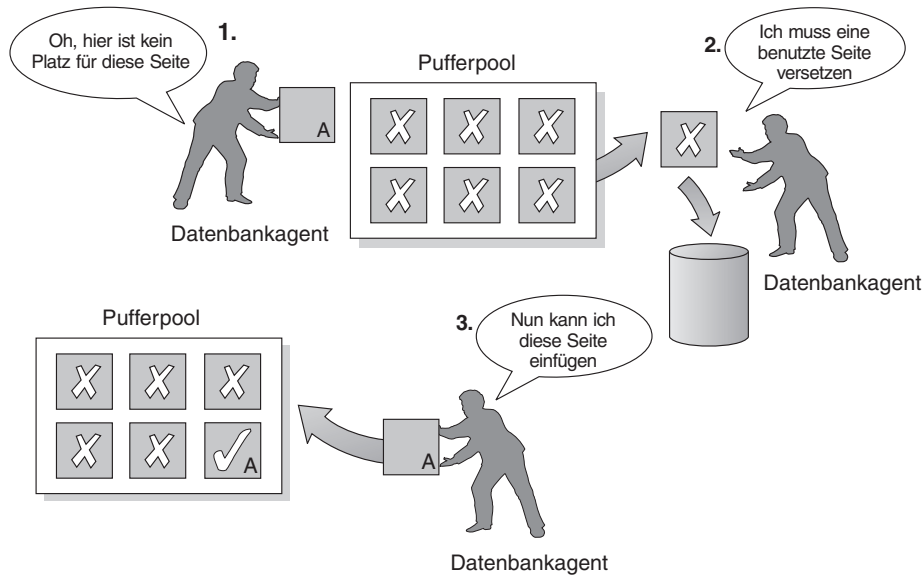
Zugehörige Konzepte:

- „Illustration der Datenseitenverwaltung in Pufferpools“ auf Seite 138

Illustration der Datenseitenverwaltung in Pufferpools

Die folgende Abbildung veranschaulicht, wie die Arbeit zur Verwaltung des Pufferpools zwischen den Agenten für Seitenlöschfunktionen und den Datenbankagenten aufgeteilt werden kann, im Vergleich zu der Situation, in der die Datenbankagenten die gesamte Ein-/Ausgabe selbst durchführen.

Ohne Seitenlöschfunktionen



Mit Seitenlöschfunktionen

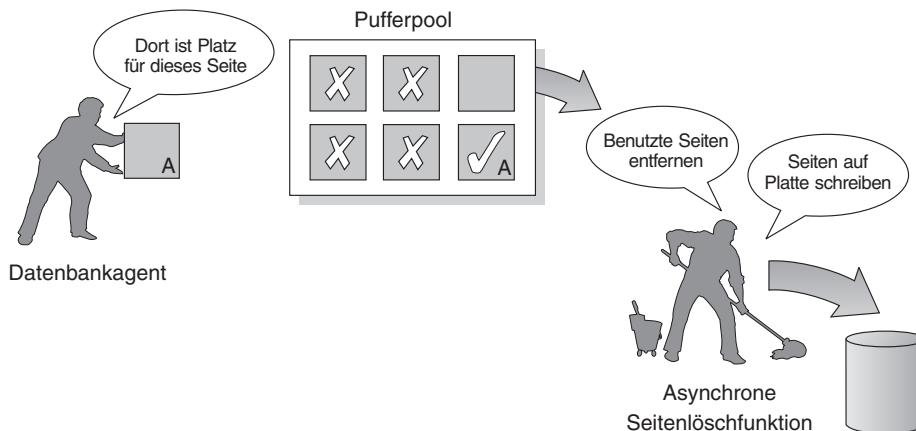


Abbildung 12. Asynchrone Seitenlöschfunktion. Geänderte Seiten werden auf Platte geschrieben.

Zugehörige Konzepte:

- „Pufferpoolverwaltung“ auf Seite 131
- „Pufferpoolverwaltung von Datenseiten“ auf Seite 136

Aktualisierungsverarbeitung

Wenn ein Agent eine Seite aktualisiert, arbeitet der Datenbankmanager mit dem folgenden Protokoll, um die für die Transaktion erforderliche E/A zu minimieren und die Wiederherstellbarkeit zu gewährleisten.

1. Die zu aktualisierende Seite wird fixiert und mit einer exklusiven Sperre belegt. In den Protokollpuffer wird ein Protokollsatz geschrieben, der beschreibt, wie die Änderung wiederholt und widerrufen werden kann. Während dieser Aktion wird auch eine Protokollfolgennummer (Log Sequence Number, LSN) empfangen und in den Kopfdaten der zu aktualisierenden Seite gespeichert.
2. Die Änderung wird an der Seite vorgenommen.
3. Die Sperre und die Fixierung der Seite werden aufgehoben.
Die Seite gilt als „benutzt“, da Änderungen an der Seite noch nicht auf Platte gespeichert wurden.
4. Der Protokollpuffer wird aktualisiert.
Sowohl die Daten im Protokollpuffer als auch die „benutzte“ Datenseite werden zwangsweise auf Platte geschrieben.

Zur Erzielung einer besseren Leistung werden diese E/A-Operationen verzögert, bis ein geeigneter Zeitpunkt eintritt, zum Beispiel eine Phase mit niedriger Systemauslastung, bis sie erforderlich sind, um die Wiederherstellbarkeit sicherzustellen, oder um die Dauer der Recovery zu begrenzen. Speziell wird eine „benutzte“ Seite zu folgenden Zeitpunkten gezwungenermaßen auf Platte geschrieben:

- Wenn ein anderer Agent sie auswählt.
- Wenn eine Seitenlöschfunktion die Seite aus einem der folgenden Gründe bearbeitet:
 - Ein anderer Agent wählt sie aus.
 - Der durch den Datenbankkonfigurationsparameter *chngpgs_thresh* angegebene Prozentsatz wird überschritten. Wenn dieser Wert überschritten wird, werden asynchrone Seitenlöschfunktionen aktiv und schreiben geänderte Seiten auf Platte.
Wenn die proaktive Seitenbereinigung aktiviert ist, ist dieser Wert irrelevant und löst keine Seitenlöschfunktionen aus.
 - Der durch den Datenbankkonfigurationsparameter *softmax* angegebene Prozentsatz wird überschritten. Wenn dieser Wert überschritten ist, werden asynchrone Seitenlöschfunktionen aktiv und schreiben geänderte Seiten auf Platte.
Wenn die proaktive Seitenbereinigung für die Datenbank aktiviert und die Anzahl von Seitenlöschfunktionen für die Datenbank richtig konfiguriert ist, sollte dieser Wert nie überschritten werden.
 - Die Anzahl sauberer Seiten auf der Vermeidungsliste fällt auf einen zu niedrigen Wert. Seitenlöschfunktionen reagieren auf diese Bedingung nur unter der Methode der proaktiven Seitenbereinigung.
 - Wenn benutzte Seiten zurzeit oder absehbar zu einer LSNGAP-Bedingung führen. Seitenlöschfunktionen reagieren auf diese Bedingung nur unter der Methode der proaktiven Seitenbereinigung.
- Wenn die Seite als Teil einer Tabelle aktualisiert wurde, für die die Klausel NOT LOGGED INITIALLY aufgerufen wird, und eine COMMIT-Anweisung abgesetzt

wird. Wenn die COMMIT-Anweisung ausgeführt wird, werden alle geänderten Seiten auf Platte geschrieben, um die Wiederherstellbarkeit sicherzustellen.

Zugehörige Konzepte:

- „Verringern des Protokollierungsaufwands zur Verbesserung der Abfrageleistung“ auf Seite 151
- „Client-/Serververarbeitungsmodell“ auf Seite 166

Zugehörige Referenzen:

- „softmax - Vor bedingtem Prüfpunkt zu schreibende Protokollsätze “ auf Seite 514
- „chngpgs_thresh - Schwellenwert für geänderte Seiten “ auf Seite 476

Proaktive Seitenbereinigung

Seit Version 8.1.4 steht eine alternative Methode zur Konfiguration der Seitenbereinigung in Ihrem System zur Verfügung. Diese alternative Methode unterscheidet sich insofern von der Standardfunktionsweise, als dass die Seitenlöschfunktionen sich proaktiver bei der Auswahl der benutzten (geänderten) Seiten verhalten, die zu einem gegebenen Zeitpunkt ausgelesen und auf die Platte geschrieben werden. Diese neue Methode der Seitenbereinigung unterscheidet sich von der Standardseitenbereinigung in zwei wesentlichen Punkten:

1. Die Seitenlöschfunktionen werden nicht durch den Konfigurationsparameter 'chngpgs_thresh' beeinflusst.

Bei dieser alternativen Methode der Seitenbereinigung reagieren Seitenlöschfunktionen nicht mehr auf den Wert des Konfigurationsparameters 'chngpgs_thresh'. Anstatt zu versuchen, einen Prozentsatz des Pufferpools sauber zu halten, stellt die alternative Methode der Seitenbereinigung einen Mechanismus bereit, durch den Agenten über die Position gut geeigneter Seiten, die gerade auf die Platte geschrieben wurden, informiert werden, so dass Agenten den Pufferpool nicht durchsuchen müssen, um geeignete Seiten zu finden. Wenn die Anzahl gut geeigneter Seiten unter einen akzeptablen Wert fällt, werden die Seitenlöschfunktionen ausgelöst, die daraufhin den gesamten Pufferpool durchsuchen und potenziell geeignete Seiten auf die Platte schreiben und die Agenten über die Positionen dieser Seiten informieren.

2. Seitenfunktionen reagieren nicht mehr auf Trigger bei LSN-Abstimmungsverlusten (LSN - Protokollfolgennummer), die von der Protokollfunktion abgesetzt werden.

Wenn die Größe des Protokollspeichers, der den Protokollsatz enthält, der die älteste Seite im Pufferpool aktualisiert hat, und die aktuelle Protokollposition den vom Parameter 'softmax' zugelassenen Wert überschreiten, wird diese Situation als LSN-Abstimmungsverlust (LSNGAP-Bedingung) der Datenbank bezeichnet. Wenn die Protokollfunktion unter der Standardmethode der Seitenbereinigung erkennt, dass ein LSN-Abstimmungsverlust aufgetreten ist, startet sie die Seitenlöschfunktionen, um alle Seiten, die zu dieser Situation beitragen, auf die Platte zu schreiben. Das heißt, sie schreibt die Seiten, die älter sind, als durch den Parameter 'softmax' zugelassen wird, auf die Platte. Seitenlöschfunktionen sind über einen gewissen Zeitraum, in dem kein LSN-Abstimmungsverlust auftritt, nicht aktiv. Wenn dann ein LSN-Abstimmungsverlust auftritt, werden die Seitenlöschfunktionen aktiviert und schreiben eine große Anzahl von Seiten auf die Platte, bevor sie wieder inaktiv werden. Dies kann zu einer Sättigung des E/A-Subsystems führen, die wiederum andere Agenten beeinträchtigen kann, die gerade Seiten lesen oder schreiben. Darüber hinaus ist es möglich, dass bis zu dem Zeitpunkt, zu dem ein LSN-Abstimmungsverlust aus-

gelöst wird, die Seitenlöschfunktionen die Seiten schon nicht mehr schnell genug bereinigen können und DB2 am Ende keinen Protokollspeicherbereich mehr zur Verfügung hat.

Die alternative Methode der Seitenbereinigung moduliert diese Funktionsweise, indem sie die gleiche Anzahl von Schreiboperationen über einen größeren Zeitraum verteilt. Die Löschfunktionen führen dies aus, indem sie proaktiv nicht nur Seiten bereinigen, die sich zurzeit in einer Situation mit LSN-Abstimmungsverlust befinden, sondern auch die Seiten, die in Anbetracht des aktuellen Aktivitätsvolumens bald in eine solche Situation kommen werden.

Zur Verwendung der neuen Methode der Seitenbereinigung setzen Sie die Registrierdatenbankvariable DB2_USE_ALTERNATE_PAGE_CLEANING auf den Wert ON.

Zugehörige Konzepte:

- „Pufferpoolverwaltung von Datenseiten“ auf Seite 136

Vorablesen von Daten in den Pufferpool

Vorablesen von Seiten bedeutet, dass eine oder mehrere Seiten von der Platte in der Erwartung abgerufen werden, dass sie von einer Anwendung angefordert werden. Durch das Vorablesen von Index- und Datenseiten in den Pufferpool kann die Leistung verbessert werden, indem die E/A-Wartezeit verringert wird. Darüber hinaus wird die Effizienz des Vorablesezugriffs durch parallele E/A-Operationen erhöht.

Es gibt zwei Kategorien des Vorablesezugriffs:

- **Sequenzieller Vorablesezugriff:** Ein Mechanismus, mit dem aufeinander folgende Seiten in den Pufferpool gelesen werden, bevor die Seiten von der Anwendung angefordert werden.
- **Vorablesezugriff über Listen:** Wird manchmal als *sequenzieller Vorablesezugriff über Listen* bezeichnet. Ist eine effiziente Methode zum Vorablesen von Datenseiten, die nicht aufeinander folgen.

Diese beiden Methoden zum Lesen von Datenseiten erfolgen zusätzlich zum normalen Lesen. Normale Lesevorgänge werden verwendet, wenn nur eine oder wenige aufeinander folgende Seiten abgerufen werden. Bei einem normalen Lesevorgang wird jeweils nur eine Seite von Daten übertragen.

Vorablesezugriff und partitionsinterne Parallelität

Das Vorablesen ist wichtig für die Leistung der partitionsinternen Parallelität, bei der mehrere Subagenten beim Durchsuchen eines Index oder einer Tabelle verwendet werden. Solche parallelen Suchoperationen ziehen einen größeren Datenverarbeitungsdurchsatz nach sich, wodurch höhere Vorablesegeschwindigkeiten erforderlich werden.

Der Nachteil durch ungeeignetes Vorablesen ist bei parallelen Suchoperationen höher als bei seriellen Suchoperationen. Wenn für eine serielle Suchoperation kein Vorablesezugriff stattfindet, arbeitet die Abfrage langsamer, weil der Agent immer auf E/A-Operationen warten muss. Wenn für eine parallele Suchoperation kein Vorablesezugriff stattfindet, müssen eventuell alle Subagenten warten, weil ein Subagent auf eine E/A-Operation wartet.

Aufgrund seiner Bedeutung wird der Vorablesezugriff bei partitionsinterner Parallelität intensiver durchgeführt. Die Funktion zur Sequenzerkennung toleriert grö-

ßere Lücken zwischen benachbarten Seiten, so dass die Seiten als sequenziell betrachtet werden können. Die Breite dieser Lücken erhöht sich mit der Anzahl der an der Suchoperation beteiligten Subagenten.

Zugehörige Konzepte:

- „Pufferpoolverwaltung“ auf Seite 131
- „Konfiguration von E/A-Servern für Vorablesezugriff und Parallelität“ auf Seite 146
- „Illustration des Vorablesens mit paralleler E/A“ auf Seite 147
- „Vorablesezugriff über Listen“ auf Seite 145
- „Sequenzielles Vorablesen“ auf Seite 142

Sequenzielles Vorablesen

Durch das Lesen mehrerer aufeinander folgender Seiten in den Pufferpool in einer einzigen E/A-Operation kann der Systemaufwand für Ihre Anwendung wesentlich reduziert werden. Darüber hinaus können mehrere parallele E/A-Operationen zum Lesen mehrerer Bereiche von Seiten in den Pufferpool bei der Verringerung der E/A-Wartezeiten helfen.

Der Vorablesezugriff beginnt, wenn der Datenbankmanager bestimmt, dass sequenzielle E/A-Operationen zweckmäßig sind und dass durch den Vorablesezugriff die Leistung verbessert werden könnte. In solchen Fällen wie Tabellensuchen und Sortierungen in Tabellen kann der Datenbankmanager leicht feststellen, dass durch den sequenziellen Vorablesezugriff die E/A-Leistung verbessert wird. In diesen Fällen startet der Datenbankmanager den sequenziellen Vorablesezugriff automatisch. Das folgende Beispiel zeigt eine Abfrage, die wahrscheinlich eine Tabellensuche erforderlich macht und für die deshalb der sequenzielle Vorablesezugriff die nahe liegende Methode wäre:

```
SELECT NAME FROM EMPLOYEE
```

Auswirkungen des PREFETCHSIZE-Werts für Tabellenbereiche

Zur Definition der Anzahl vorabgelesener Seiten für jeden Tabellenbereich können Sie die Klausel PREFETCHSIZE in den Anweisungen CREATE TABLESPACE oder ALTER TABLESPACE verwenden. Der von Ihnen definierte Wert wird in der Spalte PREFETCHSIZE der Systemkatalogtabelle SYSCAT.TABLESPACES gespeichert.

Es empfiehlt sich, den Wert für PREFETCHSIZE explizit als ein Vielfaches der Anzahl von Tabellenbereichscontainern, der Anzahl physischer Platten unter jedem Container (wenn eine RAID-Einheit eingesetzt wird) und des Werts für EXTENTSIZE für Ihren Tabellenbereich zu definieren. Der Wert für EXTENTSIZE gibt die Anzahl von Seiten an, die der Datenbankmanager in einen Container schreibt, bevor er einen anderen Container verwendet. Ist zum Beispiel für EXTENTSIZE ein Wert von 16 Seiten festgelegt und hat der Tabellenbereich zwei Container, könnten Sie den Wert für die Anzahl der vorab gelesenen Seiten (PREFETCHSIZE) auf den Wert 32 setzen. Wenn fünf physische Platten pro Container vorhanden sind, können Sie die Menge der vorab gelesenen Daten auf 160 Seiten setzen.

Der Datenbankmanager überwacht die Verwendung des Pufferpools, um sicherzustellen, dass durch den Vorablesezugriff keine Seiten aus dem Pufferpool entfernt werden, wenn eine andere UOW (Unit of Work, Arbeitseinheit) sie noch benötigt.

Zur Vermeidung von Problemen kann der Datenbankmanager die Anzahl der vorabgelesenen Seiten auf einen kleineren als den von Ihnen für den Tabellenbereich angegebenen Wert begrenzen.

Die Vorablesegröße (PREFETCHSIZE) kann erhebliche Auswirkungen auf die Leistung insbesondere für Suchoperationen in großen Tabellen haben. Verwenden Sie den Datenbanksystemmonitor und andere Systemmonitortools als Unterstützung bei der Optimierung des PREFETCHSIZE-Werts für Ihre Tabellenbereiche. Zum Beispiel könnten Sie folgende Arten von Informationen sammeln:

- Mit Überwachungsprogrammen für Ihr Betriebssystem können Sie feststellen, ob E/A-Wartezeiten für die Abfrage auftreten.
- Mit Hilfe des Datenelements *pool_async_data_reads* (asynchrone Leseoperationen für Pufferpooldaten), das vom Datenbanksystemmonitor bereitgestellt wird, können Sie feststellen, ob Vorablesezugriffe stattfinden.

Wenn E/A-Wartezeiten auftreten und für die Abfrage der Vorablesezugriff aktiv ist, könnten Sie den Wert für PREFETCHSIZE erhöhen. Wenn die Vorablesefunktion nicht die Ursache für die E/A-Wartezeiten ist, verbessert eine Erhöhung des Werts für PREFETCHSIZE die Leistung Ihrer Abfrage nicht.

Bei allen Arten des Vorablesezugriffs können mehrere E/A-Operationen parallel ausgeführt werden, wenn für PREFETCHSIZE ein Vielfaches des Werts für EXTENTSIZE für den Tabellenbereich angegeben ist und sich die durch EXTENTSIZE definierten Bereiche des Tabellenbereichs in separaten Containern befinden. Konfigurieren Sie die Container so, dass sie getrennte physische Einheiten verwenden, um eine bessere Leistung zu erzielen.

Sequenzerkennung

In einigen Fällen ist es nicht von vornherein offensichtlich, dass durch einen Vorablesezugriff eine Leistungsverbesserung erreicht werden kann. In diesen Fällen kann der Datenbankmanager die E/A-Operationen überwachen und den Vorablesezugriff aktivieren, wenn sequenzielles Lesen von Seiten auftritt. In solchen Fällen wird der Vorablesezugriff vom Datenbankmanager nach Zweckmäßigkeit aktiviert und inaktiviert. Diese Art des sequenziellen Vorablesens wird als *Sequenzerkennung* bezeichnet und wird für Index- und Datenseiten angewandt. Mit dem Konfigurationsparameter *seqdetect* können Sie steuern, ob der Datenbankmanager mit Sequenzerkennung arbeiten soll.

Wenn die Sequenzerkennung zum Beispiel aktiviert ist, könnte die folgende SQL-Anweisung von einem sequenziellen Vorablesezugriff profitieren:

```
SELECT NAME FROM EMPLOYEE  
WHERE EMPNO BETWEEN 100 AND 3000
```

In diesem Beispiel könnte das Optimierungsprogramm vielleicht begonnen haben, die Tabelle mit Hilfe eines Index für die Spalte EMPNO zu durchsuchen. Wenn die Tabelle eine hohe Clusterbildung in Bezug auf diesen Index aufweist, dann sind die Leseoperationen für die Datenseiten beinahe sequenziell und ein Vorablesezugriff könnte zu einer Leistungsverbesserung führen. Infolgedessen findet ein Vorablesezugriff auf die Datenseiten statt.

In diesem Beispiel könnte auch ein Vorablesezugriff auf die Indexseiten erfolgen. Wenn eine große Anzahl von Indexseiten untersucht werden muss und der Datenbankmanager erkennt, dass ein sequenzielles Lesen der Indexseiten stattfindet, wird ein Vorablesezugriff auf Indexseiten durchgeführt.

Zugehörige Konzepte:

- „Blockorientierte Pufferpools für besseren sequenziellen Vorablesezugriff“ auf Seite 144
- „Pufferpoolverwaltung“ auf Seite 131
- „Vorablesezugriff über Listen“ auf Seite 145
- „Vorablesen von Daten in den Pufferpool“ auf Seite 141

Blockorientierte Pufferpools für besseren sequenziellen Vorablesezugriff

Das Vorablesen von Seiten vom Plattenspeicher ist aufgrund des E/A-Systemaufwands teuer. Der Durchsatz kann beträchtlich verbessert werden, wenn sich die Verarbeitung mit den E/A-Operationen überlappen kann. Die meisten Plattformen stellen Hochleistungsmechanismen bereit, die zusammenhängende Seiten vom Plattenspeicher in nicht zusammenhängende Bereiche des Hauptspeichers einlesen. Diese Mechanismen werden gewöhnlich als gestreutes Lesen (engl. *scattered read*) oder *über einen Vektor definierte E/A* bezeichnet. Auf einigen Plattformen kann die Leistung dieser Mechanismen nicht mit E/A-Operationen mit großen Blockgrößen konkurrieren.

Standardmäßig sind Pufferpools seitenorientiert. Dies bedeutet, dass zusammenhängende Seiten auf dem Plattenspeicher in nicht zusammenhängende Seiten im Arbeitsspeicher vorab gelesen werden. Der sequenzielle Vorablesezugriff kann verbessert werden, wenn zusammenhängende Seiten vom Plattenspeicher in zusammenhängende Seiten in einem Pufferpool gelesen werden können.

Zu diesem Zweck können Sie blockorientierte Pufferpools erstellen. Ein blockorientierter Pufferpool besteht sowohl aus einem Seitenbereich als auch aus einem Blockbereich. Der Seitenbereich ist für nicht sequenzielle Vorableseoperationen erforderlich. Der Blockbereich besteht aus Blöcken, wobei jeder Block eine angegebene Anzahl zusammenhängender Seiten enthält, die als *Blockgröße* bezeichnet wird.

Die optimale Nutzung eines blockorientierten Pufferpools hängt von der angegebenen Blockgröße ab. Die Blockgröße ist die Granularität, mit der E/A-Server bei sequenziellen Vorablesezugriffen eine blockorientierte E/A-Operation in Betracht ziehen. Die EXTENTSIZE-Größe ist die Granularität, mit der Tabellenbereiche in Containern einheitenübergreifend gespeichert werden (Striping). Da mehrere Tabellenbereiche mit unterschiedlichen EXTENTSIZE-Werten an einen Pufferpool gebunden werden können, der mit der gleichen Blockgröße definiert ist, beachten Sie, in welcher Beziehung die EXTENTSIZE-Größe und die Blockgröße bei der effizienten Nutzung des Pufferpoolspeichers zueinander stehen. Pufferpoolspeicher kann unter folgenden Umständen verschwendet werden:

- Wenn der Wert für EXTENTSIZE, durch den die Größe für Vorableseanforderungen festgelegt wird, kleiner als der für den Pufferpool angegebene Wert für BLOCK_SIZE ist.
- Wenn sich einige Seiten, die durch eine Vorableseanforderung angefordert werden, bereits im Seitenbereich des Pufferpools befinden.

Der E/A-Server lässt einige verschwendete Seiten in jedem Pufferpoolblock zu. Wenn jedoch zu große Teile eines Blocks verschwendet würden, führt der E/A-Server ein nicht blockorientiertes Vorablesen in den Seitenbereich des Pufferpools durch. Dies führt nicht zu einer optimalen Leistung.

Zur Erzielung einer optimalen Leistung binden Sie Tabellenbereiche der gleichen EXTENTSIZE-Größe an einen Pufferpool mit einer Blockgröße, die der EXTENTSIZE-Größe der Tabellenbereiche entspricht. Eine gute Leistung lässt sich erreichen, wenn der Wert für EXTENTSIZE größer als die Blockgröße ist, jedoch nicht, wenn der Wert für EXTENTSIZE kleiner als die Blockgröße ist. Verwenden Sie die Anweisungen CREATE und ALTER BUFFERPOOL zur Erstellung eines blockorientierten Pufferpools.

Blockorientierte E/A und AWE-Unterstützung können von einem Pufferpool nicht gleichzeitig genutzt werden. Die AWE-Unterstützung erhält Vorrang vor der Unterstützung für blockorientierte E/A, wenn beide für einen Pufferpool angegeben sind. In diesem Fall wird die Unterstützung für blockorientierte E/A für den Pufferpool inaktiviert. Sie wird wieder aktiviert, wenn die AWE-Unterstützung inaktiviert wird.

Anmerkung: Blockorientierte Pufferpools sind für den sequenziellen Vorableszugriff gedacht. Wenn Ihre Anwendungen keinen sequenziellen Vorableszugriff nutzen, ist der Blockbereich im Pufferpool verschwendet.

Zugehörige Konzepte:

- „Pufferpoolverwaltung“ auf Seite 131
- „Vorablesen von Daten in den Pufferpool“ auf Seite 141
- „Sequenzielles Vorablesen“ auf Seite 142

Vorableszugriff über Listen

Der *Vorableszugriff über Listen* oder *sequenzielle Vorableszugriff über Listen* ist eine effiziente Methode, auf Daten zuzugreifen, auch wenn die benötigten Datenseiten nicht in aufeinander folgender Reihenfolge vorliegen. Der Vorableszugriff über Listen kann in Verbindung mit dem Zugriff über einen oder mehrere Indizes verwendet werden.

Wenn das Optimierungsprogramm einen Index zum Zugriff auf Zeilen verwendet, kann es das Lesen der Datenseiten verzögern, bis alle Zeilen-IDs (RIDs) aus dem Index empfangen wurden. Zum Beispiel könnte das Optimierungsprogramm eine Indexsuche durchführen, um die abzurufenden Zeilen und Datenseiten zu ermitteln, wenn zuvor der Index IX1 definiert wurde:

```
INDEX IX1:  NAME    ASC,
            DEPT    ASC,
            MGR      DESC,
            SALARY   DESC,
            YEARS    ASC
```

Betrachten Sie folgende Suchbedingungen:

```
WHERE NAME BETWEEN 'A' and 'I'
```

Wenn die Daten in Bezug auf diesen Index keine Clusterbildung aufweisen, enthält der Vorableszugriff über Listen einen Schritt zum Sortieren der durch die Indexsuche ermittelten Liste von Zeilen-IDs (RIDs).

Zugehörige Konzepte:

- „Pufferpoolverwaltung“ auf Seite 131
- „Vorablesen von Daten in den Pufferpool“ auf Seite 141
- „Sequenzielles Vorablesen“ auf Seite 142

Konfiguration von E/A-Servern für Vorablesezugriff und Parallelität

Zur Aktivierung des Vorablesezugriffs startet der Datenbankmanager zum Lesen von Datenseiten separate Steuerthreads, die als *E/A-Server* bezeichnet werden. Infolgedessen gliedert sich die Verarbeitung einer Abfrage in zwei parallele Aktivitäten: Datenverarbeitung (CPU) und E/A-Operationen für Datenseiten. Die E/A-Server warten auf Vorableseanforderungen aus der CPU-Verarbeitungsaktivität. Diese Vorableseanforderungen enthalten eine Beschreibung der E/A-Operationen, die zur Erfüllung der Abfrage benötigt werden. Die möglichen Vorablesemethoden bestimmen, wann und wie der Datenbankmanager die Vorableseanforderungen generiert.

Durch Konfigurieren einer ausreichenden Anzahl von E/A-Servern mit Hilfe des Konfigurationsparameters *num_ioservers* kann die Leistung von Abfragen, für die der Vorablesezugriff auf Daten verwendet werden kann, erheblich gesteigert werden. Setzen Sie den Wert des Parameters *num_ioservers* mindestens auf die Zahl der physischen Platten in der Datenbank, um die Möglichkeit für parallele E/A-Operationen zu maximieren.

Es ist besser, die Anzahl von E/A-Servern zu hoch als zu niedrig anzusetzen. Wenn Sie zusätzliche E/A-Server angeben, werden diese Server nicht verwendet, und ihre Speicherseiten werden ausgelagert. Infolgedessen wird die Leistung nicht beeinträchtigt. Jeder E/A-Serverprozess hat eine Nummer. Der Datenbankmanager verwendet immer den Prozess mit der niedrigsten Nummer, so dass einige der Prozesse mit höheren Nummern möglicherweise nie zum Einsatz kommen.

Bei der Schätzung, wie viele E/A-Server eventuell erforderlich sind, sollten Sie folgende Punkte beachten:

- Die Anzahl der Datenbankagenten, die Vorableseanforderungen an die E/A-Serverwarteschlange gleichzeitig schreiben könnten.
- Der höchste Grad, bis zu dem die E/A-Server parallel arbeiten können.

Sie sollten die Einstellung des Werts des Datenbankkonfigurationsparameters *num_ioservers* auf AUTOMATIC in Betracht ziehen, damit DB2 intelligente Werte auf der Basis der Systemkonfiguration auswählen kann.

Konfiguration für asynchrone E/A

Auf einigen Plattformen arbeitet DB2 mit asynchroner Ein-/Ausgabe (AIO), um die Leistung von Aktivitäten wie Seitenlöschfunktionen und Vorablesezugriffe zu verbessern. Asynchrone E/A (AIO) ist am effektivsten, wenn die Daten in den Containern über mehrere Platten verteilt sind. Die Leistung profitiert außerdem von einer Optimierung der AIO-Infrastruktur des zugrunde liegenden Betriebssystems.

Unter AIX können Sie zum Beispiel die AIO für das Betriebssystem optimieren. Wenn die asynchrone Ein-/Ausgabe entweder mit SMS-Containern oder mit DMS-Dateicontainern arbeitet, verwalten Betriebssystemprozesse, so genannte AIO-Server, die E/A-Operationen. Eine kleine Anzahl solcher Server kann den Vorteil der asynchronen Ein-/Ausgabe einschränken, indem sie die Anzahl von AIO-Anforderungen begrenzt. Verwenden Sie zur Konfiguration der Anzahl von AIO-Servern unter AIX die *smit*-AIO-Parameter *minservers* und *maxservers*.

Zugehörige Konzepte:

- „Illustration des Vorablesens mit paralleler E/A“ auf Seite 147
- „Verwaltung der parallelen Ein-/Ausgabe“ auf Seite 148

Zugehörige Referenzen:

- „num_ioservers - Anzahl von E/A-Servern “ auf Seite 480

Illustration des Vorablesens mit paralleler E/A

Die folgende Abbildung veranschaulicht, wie E/A-Server zum Vorablesen von Daten in den Pufferpool verwendet werden.

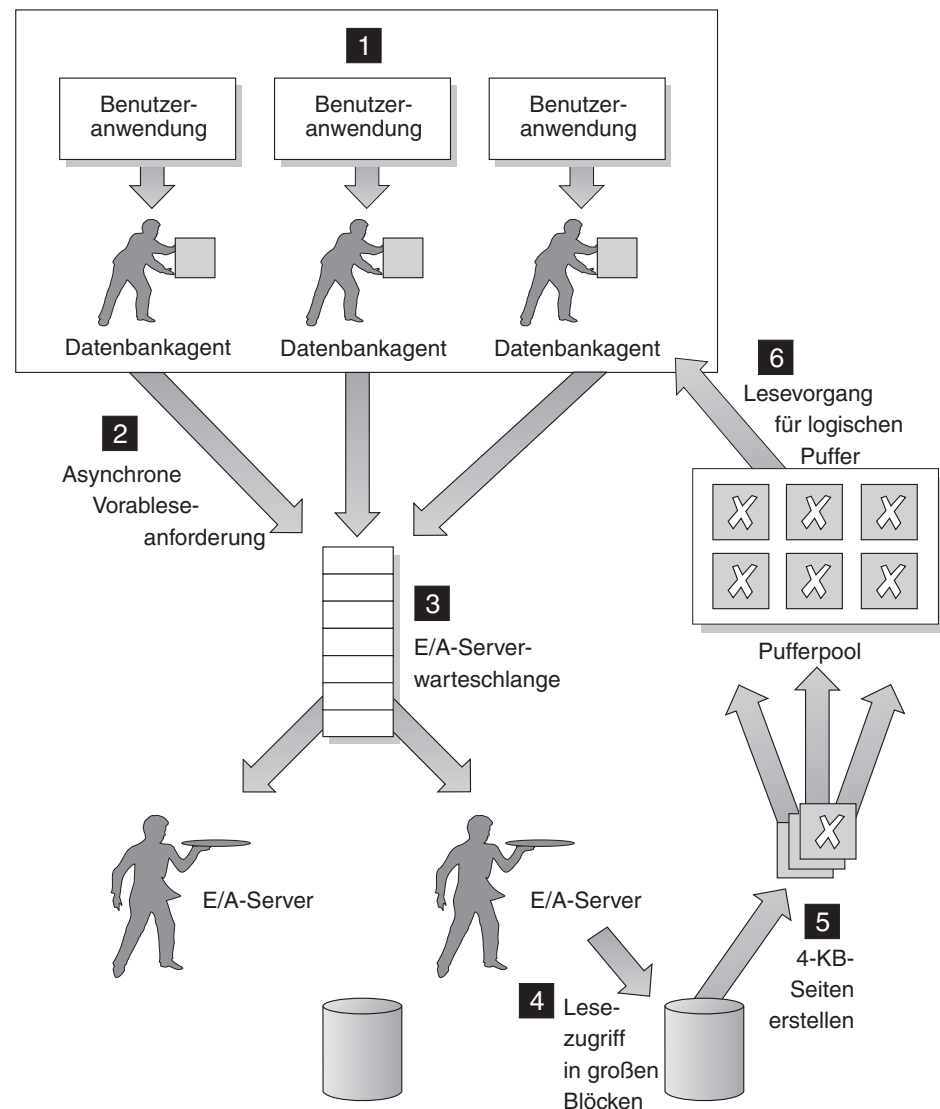


Abbildung 13. Vorablesen von Daten mit Hilfe von E/A-Servern

- 1** Die Benutzeranwendung übergibt die Anforderung an den Datenbankagenten, der der Benutzeranwendung vom Datenbankmanager zugeordnet wurde.
- 2, 3** Der Datenbankagent legt fest, dass ein Vorablesezugriff erfolgen soll, um

die angeforderten Daten abzurufen und so die Anforderung zu erfüllen, und schreibt eine Vorableseanforderung an die E/A-Serverwarteschlange.

4, 5

Der erste verfügbare E/A-Server liest die Vorableseanforderung aus der Warteschlange und liest die Daten aus dem Tabellenbereich in den Pufferpool. Die Anzahl von E/A-Servern, die Daten gleichzeitig aus einem Tabellenbereich abrufen können, hängt von der Anzahl der Vorleseanforderungen in der Warteschlange sowie von der Anzahl von E/A-Servern ab, die durch den Konfigurationsparameter *num_ioservers* der Datenbank konfiguriert sind.

6

Der Datenbankagent führt die erforderlichen Operationen an den Daten-seiten im Pufferpool durch und liefert das Ergebnis an die Benutzeranwendung zurück.

Zugehörige Konzepte:

- „Agenten in einer partitionierten Datenbank“ auf Seite 163
- „Verwaltung der parallelen Ein-/Ausgabe“ auf Seite 148
- „Konfiguration von E/A-Servern für Vorablesezugriff und Parallelität“ auf Seite 146
- „Vorablesezugriff über Listen“ auf Seite 145
- „Vorablesen von Daten in den Pufferpool“ auf Seite 141
- „Sequenzielles Vorablesen“ auf Seite 142

Verwaltung der parallelen Ein-/Ausgabe

Wenn mehrere Container für einen Tabellenbereich vorhanden sind, kann der Datenbankmanager die *parallele Ein-/Ausgabe* einleiten, bei der der Datenbankmanager mehrere E/A-Server zur Verarbeitung der E/A-Anforderungen einer einzelnen Abfrage verwendet. Jeder E/A-Server verarbeitet die E/A-Operationen für einen anderen Container, so dass mehrere Container parallel gelesen werden können. Die parallele Durchführung der E/A-Operationen kann zu bedeutenden Verbesserungen beim E/A-Durchsatz führen.

Obwohl ein getrennter E/A-Server die E/A-Operationen für jeden Container durchführen kann, ist die tatsächliche Anzahl der E/A-Server, die parallele E/A-Operationen durchführen können, auf die Anzahl der physischen Einheiten begrenzt, über die die angeforderten Daten verteilt sind. Aus diesem Grund benötigen Sie so viele E/A-Server, wie physische Einheiten vorhanden sind.

Die parallele Ein-/Ausgabe wird in folgenden Fällen unterschiedlich eingeleitet:

- **Sequenzieller Vorablesezugriff**

Beim sequenziellen Vorablesezugriff wird die parallele E/A initialisiert, wenn der Wert für PREFETCHSIZE (Menge der vorab gelesenen Daten) ein Vielfaches des Werts für EXTENTSIZE eines Tabellenbereichs ist. Jede Vorableseanforderung wird dann in viele kleine Anforderungen aufgeteilt, die sich an den Grenzen der durch EXTENTSIZE definierten Bereiche orientieren. Diese kleinen Anforderungen werden dann verschiedenen E/A-Servern zugeordnet.

- **Vorablesezugriff über Listen**

Beim Vorablesezugriff über Listen wird jede Liste von Seiten in Abhängigkeit von den Containern, in denen die Datenseiten gespeichert sind, in kleinere Listen unterteilt. Diese kleineren Listen werden dann verschiedenen E/A-Servern zugeordnet.

- **Backup und Restore von Datenbanken oder Tabellenbereichen**

Für Backups oder Restores von Daten ist die Anzahl paralleler E/A-Anforderungen gleich der Größe des Backup-Puffers dividiert durch den Wert von EXTENT-SIZE. Der Maximalwert ist gleich der Anzahl von Containern.

- **Restore einer Datenbank oder eines Tabellenbereichs**

Für den Restore von Daten werden parallele E/A-Anforderungen in gleicher Weise wie beim sequenziellen Vorabesezugriff eingeleitet und aufgeteilt. Die Daten werden nicht im Pufferpool wiederhergestellt, sondern direkt aus dem Restorepuffer auf die Platte versetzt.

- **LOAD**

Beim Laden von Daten können Sie den Grad der E/A-Parallelität mit der Option DISK_PARALLELISM des Befehls LOAD angeben. Wenn diese Option nicht angegeben wird, verwendet der Datenbankmanager einen Standardwert, der auf der kumulativen Anzahl von Tabellenbereichscontainern für alle Tabellenbereiche basiert, die der Tabelle zugeordnet sind.

Für eine optimale Leistung bei paralleler E/A sollten Sie folgende Voraussetzungen erfüllen:

- Es ist eine ausreichende Anzahl E/A-Server vorhanden. Geben Sie geringfügig mehr E/A-Server an als die Anzahl von Containern, die für alle Tabellenbereiche in der Datenbank verwendet werden.
- Die Werte für EXTENTSIZE und PREFETCHSIZE sind für den Tabellenbereich angemessen. Zur Vermeidung einer übermäßigen Nutzung des Pufferpools sollte der Wert für PREFETCHSIZE nicht zu groß sein. Eine ideale Größe ist ein Vielfaches des Werts für EXTENTSIZE, der Anzahl physischer Platten unter jedem Container (wenn eine RAID-Einheit eingesetzt wird) und der Anzahl der Tabellenbereichscontainer. Der Wert für EXTENTSIZE sollte relativ klein sein, wobei sich ein Wert zwischen 8 und 32 Seiten empfiehlt.
- Die Container befinden sich auf separaten physischen Laufwerken.
- Alle Container haben dieselbe Größe, um einen konsistenten Grad an Parallelität zu gewährleisten.

Wenn ein oder mehrere Container kleiner als die anderen sind, verringern sie das Potenzial für das optimierte parallele Vorablesen. Betrachten Sie die folgenden Beispiele:

- Wenn ein kleinerer Container gefüllt ist, werden weitere Daten in den übrigen Containern gespeichert, wodurch sich eine ungleichmäßige Auslastung der Container ergibt. Ungleichmäßig ausgelastete Container beeinträchtigen die Leistung des parallelen Vorablesens, da die Anzahl von Containern, aus denen Daten vorab gelesen werden können, eventuell kleiner ist als die Gesamtanzahl von Containern.
 - Wenn ein kleinerer Container zu einem späteren Zeitpunkt hinzugefügt wird und die Daten neu verteilt werden, enthält der kleinere Container weniger Daten als die anderen Container. Diese im Verhältnis zu den anderen Containern kleine Menge von Daten führt nicht zu einer Optimierung des parallelen Vorablesens.
 - Wenn ein Container größer ist und alle anderen Container vollständig gefüllt werden, wird dieser Container zum einzigen Container, in dem weitere Daten gespeichert werden. Beim Zugriff auf diese weiteren Daten kann der Datenbankmanager keinen parallelen Vorabesezugriff durchführen.
- Es ist eine angemessene E/A-Kapazität vorhanden, wenn partitionsinterne Parallelität verwendet wird. Auf SMP-Maschinen kann die partitionsinterne Parallelität die für eine Abfrage benötigte Zeit reduzieren, indem die Abfrage auf mehre-

ren Prozessoren ausgeführt wird. Es ist eine ausreichende E/A-Kapazität erforderlich, um jeden Prozessor auszulasten. In der Regel sind zusätzliche physische Laufwerke erforderlich, um diese E/A-Kapazität bereitzustellen.

Der Wert für PREFETCHSIZE muss höher sein, um ein Vorablesen mit höheren Geschwindigkeiten und eine effektive Nutzung der E/A-Kapazität zu ermöglichen.

Die Anzahl der erforderlichen physischen Laufwerke hängt von der Geschwindigkeit und der Kapazität der Laufwerke und des E/A-Busses sowie von der Geschwindigkeit der Prozessoren ab.

Zugehörige Konzepte:

- „Optimieren der Sortierleistung“ auf Seite 129
- „Konfiguration von E/A-Servern für Vorablesezugriff und Parallelität“ auf Seite 146
- „Illustration des Vorablesens mit paralleler E/A“ auf Seite 147

Verbessern der Einfügeleistung

Wenn SQL-Anweisungen mit Hilfe von INSERT neue Informationen in eine Tabelle einfügen, durch ein INSERT-Suchalgorithmus zunächst die FSCR-Sätze (Free Space Control Records), um eine Seite mit genügend Speicherplatz zu finden. Selbst wenn der FSCR jedoch angibt, dass auf einer bestimmten Seite genügend freier Speicherbereich vorhanden ist, kann dieser jedoch möglicherweise nicht verwendet werden, da er durch eine nicht festgeschriebene Anweisung DELETE von einer anderen Transaktion reserviert ist. Um sicherzustellen, dass nicht festgeschriebener freier Speicherplatz verwendbar ist, sollten Sie Transaktionen häufig durch COMMIT festschreiben.

Die Einstellung der Registrierdatenbankvariablen DB2MAXFSCRSEARCH bestimmt die Anzahl von FSCRs in einer Tabelle, die für eine INSERT-Anweisung durchsucht werden. Der Standardwert für diese Registrierdatenbankvariable ist fünf. Wenn in der angegebenen Anzahl von FSCRs kein Speicherplatz gefunden wird, wird der einzufügende Datensatz an das Ende der Tabelle angehängt. Zur Optimierung der Geschwindigkeit der Anweisung INSERT werden nachfolgende Datensätze ebenfalls an das Ende der Tabelle angehängt, bis zwei EXTENTSIZE-Speicherbereiche gefüllt sind. Wenn diese beiden Speicherbereiche gefüllt sind, setzt die nächste Anweisung INSERT die Suche bei dem FSCR fort, bei dem die letzte Suche beendet wurde.

Anmerkung: Setzen Sie die Registrierdatenbankvariable DB2MAXFSCRSEARCH auf einen kleinen Wert, um die INSERT-Geschwindigkeit, allerdings mit der möglichen Folge eines schnelleren Anwachsens von Tabellen, zu optimieren. Zur Optimierung der Wiederverwendung von Speicherplatz, möglicherweise zu Lasten der INSERT-Geschwindigkeit, setzen Sie die Registrierdatenbankvariable DB2MAXFSCRSEARCH auf einen höheren Wert.

Wenn alle FSCRs in der gesamten Tabelle auf diese Weise durchsucht wurden, werden die einzufügenden Datensätze ohne weiteres Suchen an das Ende angehängt. Es findet kein weiteres Suchen mit Hilfe der FSCRs mehr statt, sofern nicht Speicherplatz an einer Stelle in der Tabelle frei wird, wie zum Beispiel nach einer DELETE-Operation.

Es gibt zwei weitere INSERT-Algorithmusoptionen:

- Anhängemodus (APPEND MODE)

In diesem Modus werden neue Zeilen immer an das Ende der Tabelle angehängt. Es finden weder Such- noch Pflegeoperationen für FSCRs statt. Diese Option wird mit der Anweisung ALTER TABLE APPEND ON aktiviert und kann die Leistung für ausschließlich wachsende Tabellen wie Journale verbessern.

- Ein Clusterindex ist für die Tabelle definiert.

In diesem Fall versucht der Datenbankmanager, Datensätze auf derselben Seite wie andere Datensätze mit ähnliche Indexschlüsselwerten einzufügen. Wenn auf der betreffenden Seite kein Speicherbereich verfügbar ist, wird versucht, den Datensatz auf den umgebenden Seiten unterzubringen. Wenn dies keinen Erfolg hat, wird der oben beschriebene FSCR-Suchalgorithmus verwendet, jedoch mit dem Unterschied, dass der am schlechtesten passende Speicherplatz, nicht der erste passende gesucht wird. Durch dieses Verfahren werden normalerweise Seiten ausgewählt, die mehr freien Speicherbereich enthalten. Diese Methode richtet einen neuen Clustering-Bereich für Zeilen mit diesem Schlüsselwert ein.

Wenn Sie einen Clusterindex für eine Tabelle definieren, verwenden Sie die Anweisung ALTER TABLE... PCTFREE, bevor Sie Daten in die Tabelle laden oder die Tabelle reorganisieren. Die Klausel PCTFREE definiert einen Prozentsatz an freiem Speicherplatz, der auf der Datenseite der Tabelle nach dem Laden und Reorganisieren verbleiben soll. Dadurch steigt die Wahrscheinlichkeit, dass bei der Clusterindexoperation auf der richtigen Seite freier Speicherplatz gefunden wird.

Zugehörige Konzepte:

- „Tabellen- und Indexverwaltung für MDC-Tabellen“ auf Seite 387
- „Tabellen- und Indexverwaltung für Standardtabellen“ auf Seite 384
- „Aktualisierungsverarbeitung“ auf Seite 139

Verringern des Protokollierungsaufwands zur Verbesserung der Abfrageleistung

Alle Datenbanken pflegen Protokolldateien, die Datensätze über Datenbankänderungen aufzeichnen. Zwei Protokollierungsstrategien stehen zur Auswahl:

- Die Umlaufprotokollierung, bei der die Protokollsätze die Protokolldateien vollständig füllen und anschließend die ersten Protokollsätze in der ersten Protokolldatei überschreiben. Die überschriebenen Protokollsätze sind nicht wiederherstellbar.
- Behalten und Speichern der Protokollsätze, bei dem eine Protokolldatei archiviert wird, wenn sie mit Protokollsätzen gefüllt ist. Für weitere Protokollsätze werden neue Protokolldateien verfügbar gemacht. Die Speicherung der Protokolldateien ermöglicht eine **aktualisierende Recovery**. Die aktualisierende Recovery wendet Änderungen an der Datenbank auf der Basis der abgeschlossenen und im Protokoll aufgezeichneten UOWs (UOW = Unit of Work, Arbeitseinheit) bzw. Transaktionen erneut an. Sie können angeben, dass die aktualisierende Recovery bis zum Ende der Protokolle oder bis zu einem bestimmten Zeitpunkt vor dem Ende der Protokolle durchgeführt wird.

Unabhängig von der Protokollierungsstrategie werden alle Änderungen an regulären Daten- und Indexseiten in den Protokollpuffer geschrieben. Die Daten im Protokollpuffer werden durch den Protokollprozess auf die Platte geschrieben.

Unter folgenden Umständen muss die Abfrageverarbeitung darauf warten, dass Protokolldaten auf die Platte geschrieben werden:

- Bei einem Commit.
- Bevor die entsprechenden Datenseiten auf Platte geschrieben werden, da DB2 mit einer Vorabschreibprotokollierung (write ahead) arbeitet. Ein Vorteil der Vorabschreibprotokollierung besteht darin, dass bei Beendigung einer Transaktion durch Ausführen einer COMMIT-Anweisung nicht alle geänderten Daten und Indexseiten auf Platte geschrieben werden müssen.
- Bevor einige Änderungen an Metadaten vorgenommen werden, von denen sich die meisten aus der Ausführung von DDL-Anweisungen ergeben.
- Beim Schreiben von Protokollsätzen in den Protokollpuffer, wenn der Protokollpuffer voll ist.

DB2 verwaltet so das Schreiben von Protokolldaten auf Platte, um die Verarbeitungsverzögerung zu minimieren. In einer Umgebung, in der viele kurze gleichzeitige Transaktionen stattfinden, wird der größte Teil der Verarbeitungsverzögerung durch COMMIT-Anweisungen verursacht, die darauf warten müssen, dass Protokolldaten auf Platte geschrieben werden. Infolgedessen schreibt der Protokollprozess häufig kleine Mengen von Protokolldaten auf Platte, wobei eine zusätzliche Verzögerung durch den Systemaufwand für Protokoll-E/A entsteht. Um die Anwendungsantwortzeit gegen eine solche Protokollierungsverzögerung zu verbessern, setzen Sie den Datenbankkonfigurationsparameter *mincommit* auf einen höheren Wert als 1. Diese Einstellung kann zwar längere Verzögerungen für Commits aus einigen Anwendungen verursachen, jedoch können mehr Protokolldaten in einer Operation auf Platte geschrieben werden.

Änderungen an großen Objekten (LOBs) und LONG VARCHAR-Daten werden durch Erstellen einer Spiegelkopie für Speicherseiten (Shadow Paging) protokolliert. Änderungen an LOB-Spalten werden nur dann protokolliert, wenn Sie die Beibehaltung der Protokolldateien (log_retain) angeben und die LOB-Spalte in der Anweisung CREATE TABLE ohne die Klausel NOT LOGGED definiert wurde. Änderungen an den Zuordnungsseiten für LONG- oder LOB-Datentypen werden wie reguläre Datenseiten protokolliert.

Zugehörige Konzepte:

- „Client-/Serververarbeitungsmodell“ auf Seite 166
- „Aktualisierungsverarbeitung“ auf Seite 139

Zugehörige Referenzen:

- „mincommit - Anzahl der Gruppencommits “ auf Seite 512

Kapitel 8. Agentenverwaltung

Datenbankagenten

Für jede Datenbank, auf die eine Anwendung zugreift, werden verschiedene Prozesse oder Threads gestartet, um die verschiedenen Anwendungstasks durchzuführen. Zu diesen Tasks gehören das Protokollieren, die Kommunikation und der Vorabesezugriff.

Datenbankagenten sind EDU-Prozesse oder EDU-Threads (EDU - Engine Dispatchable Unit, zuteilbare Einheit der Steuerkomponente). Datenbankagenten erledigen die Arbeit im Datenbankmanager, die von Anwendungen angefordert wird. In UNIX-Umgebungen werden diese Agenten als Prozesse ausgeführt. Unter Intel-basierten Betriebssystemen wie Windows werden die Agenten als Threads ausgeführt.

Die maximale Anzahl von Anwendungsverbindungen wird durch den Konfigurationsparameter *max_connections* des Datenbankmanagers gesteuert. Die Arbeit jeder Anwendungsverbindung wird durch einen einzigen Verarbeitungsagenten koordiniert.

Ein *Verarbeitungsagent* führt Anwendungsanforderungen aus, ist aber nicht dauerhaft mit einer bestimmten Anwendung verbunden. Der Koordinatoragent verfügt über alle Informationen und Steuerblöcke, die zur Durchführung von Aktionen im Datenbankmanager erforderlich sind, die von der Anwendung angefordert wurden.

Es gibt vier Typen von Verarbeitungsagenten:

- Nicht zugeordnete Agenten
- Inaktive Agenten
- Aktive Koordinatoragenten
- Subagenten

Nicht zugeordnete Agenten

Dies ist die einfachste Form eines Verarbeitungsagenten. Er hat keine abgehende Verbindung und keine lokale Datenbankverbindung oder Instanzverbindung.

Inaktive Agenten

Ein inaktiver Agent ist ein Verarbeitungsagent, der sich nicht in einer aktiven Transaktion befindet, keine abgehende Verbindung hat und keine lokale Datenbank- oder Instanzverbindung hat. Inaktive Agenten sind verfügbar, um Arbeit für eine Anwendungsverbindung zu übernehmen.

Aktive Koordinatoragenten

Jeder Prozess oder Thread einer Clientanwendung hat einen einzelnen aktiven Agenten, der die Arbeit der Anwendung an der Datenbank koordiniert. Wenn der Koordinatoragent erstellt ist, führt er alle Datenbankanforderungen für die zugehörige Anwendung aus und kommuniziert mit anderen Agenten per Interprozesskommunikation (IPC) oder über Protokolle zur Fernverbindung. Jeder Agent arbeitet mit seinem eigenen privaten Speicher und benutzt Ressourcen des Datenbankmanagers und globale Datenbankressourcen, wie z. B. den Pufferpool, gemeinsam mit anderen

Agenten. Wenn eine Transaktion abgeschlossen wird, kann der aktive Koordinatoragent zu einem inaktiven Agenten werden.

Wenn ein Client die Verbindung zu einer Datenbank oder zu einer Instanz trennt, geschieht mit dem koordinierenden Agenten Folgendes:

- Er wird ein aktiver Agent. Wenn andere Verbindungen warten, wird der Verarbeitungsagent ein aktiver Koordinatoragent.
- Er wird freigegeben und als nicht zugeordnet markiert, wenn keine Verbindungen warten und die maximale Zahl von Poolagenten noch nicht erreicht wurde.
- Er wird beendet und der von ihm verwendete Speicher wird freigegeben, wenn keine Verbindungen warten und die maximale Zahl von Poolagenten erreicht wurde.

Subagenten

In Umgebungen mit partitionierten Datenbanken und Umgebungen mit aktivierter partitionsinterner Parallelität verteilt der Koordinatoragent die Datenbankanforderungen an Subagenten, die ihrerseits die Anforderungen für die Anwendung ausführen. Wenn der Koordinatoragent erstellt ist, verarbeitet er alle Datenbankanforderungen für seine Anwendung, indem er die Subagenten koordiniert, die die Anforderungen in der Datenbank ausführen.

Agenten, die keine Arbeit für Anwendungen ausführen und darauf warten, zugeordnet zu werden, gelten als nicht zugeordnete Agenten im Bereitschaftsmodus und befinden sich in einem *Agentenpool*. Diese Agenten sind für Anforderungen von Koordinatoragenten, die für Clientprogramme aktiv sind, oder für Subagenten, die für vorhandene Koordinatoragenten aktiv sind, verfügbar. Die Anzahl der verfügbaren Agenten ist vom Wert der Konfigurationsparameter *maxagents* und *num_poolagents* des Datenbankmanagers abhängig.

Wenn ein Agent seine Arbeit beendet, jedoch noch eine Verbindung zu einer Datenbank hat, wird er in den Agentenpool versetzt. Ungeachtet dessen, ob der Verbindungskonzentrator für die Datenbank aktiviert ist, wird ein Agent beendet, wenn er nicht innerhalb eines bestimmten Zeitraums zur Bedienung einer neuen Anforderung aktiviert wird oder wenn die aktuelle Anzahl aktiver und im Pool befindlicher Agenten größer als der Wert von *num_poolagents* ist.

Agenten aus dem Agentenpool (*num_poolagents*) werden als Koordinatoragenten für die folgenden Arten von Anwendungen wiederverwendet:

- Ferne TCP/IP-basierte Anwendungen
- Lokale Anwendungen auf UNIX-Betriebssystemen
- Lokale und ferne Anwendungen auf Windows-Betriebssystemen

Andere Arten ferner Anwendungen erstellen stets einen neuen Agenten. Falls kein Agent im Bereitschaftsmodus vorhanden ist, wenn ein Agent angefordert wird, wird ein neuer Agent dynamisch erstellt. Da die Erstellung eines neuen Agenten einen bestimmten Systemaufwand erfordert, ist die CONNECT- und ATTACH-Leistung besser, wenn ein nicht zugeordneter Agent im Bereitschaftsmodus für einen Client aktiviert werden kann.

Wenn ein Subagent für eine Anwendung aktiv ist, wird er dieser Anwendung *zugeordnet*. Nach Beendigung der angeforderten Operationen kann er in den Agentenpool versetzt werden, bleibt jedoch der ursprünglichen Anwendung zugeordnet.

Wenn die Anwendung eine weitere Operation anfordert, überprüft der Datenbankmanager zunächst den Agentenpool nach zugeordneten inaktiven Agenten, bevor er einen neuen Agenten erstellt.

Zugehörige Konzepte:

- „Agenten in einer partitionierten Datenbank“ auf Seite 163
- „Konfigurationsparameter mit Auswirkung auf die Anzahl von Agenten“ auf Seite 165
- „Verbesserungen des Verbindungskonzentrators für Clientverbindungen“ auf Seite 171
- „Verwaltung von Datenbankagenten“ auf Seite 163

Übersicht über die Architektur und Prozesse von DB2

Allgemeine Informationen zur Architektur und zu den Prozessen von DB2 können Ihnen beim Verständnis der ausführlichen Informationen zu bestimmten Themen behilflich sein.

Die folgende Abbildung zeigt eine allgemeine Übersicht über die Architektur und die Prozesse für IBM DB2 Version 9.1.

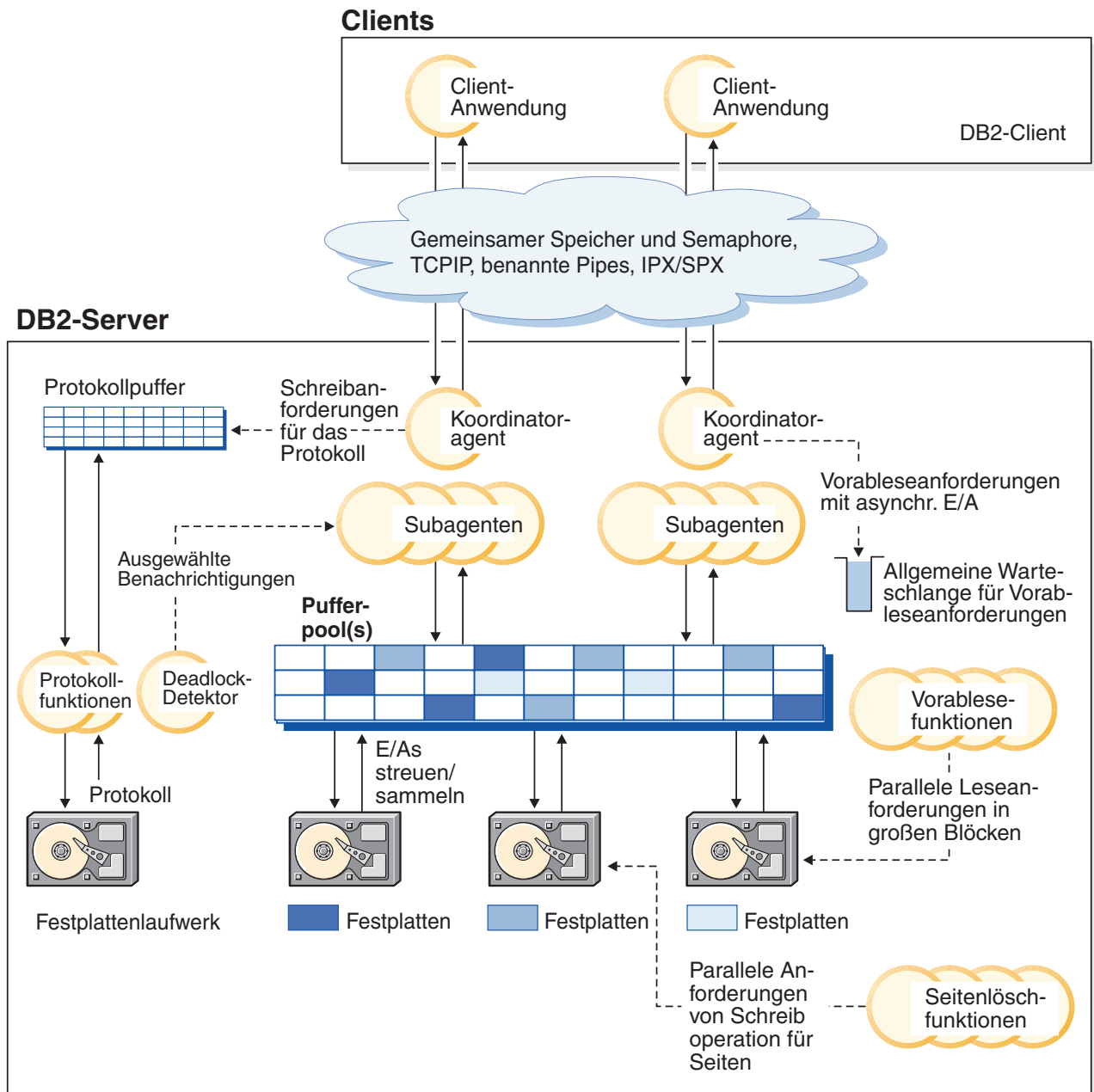


Abbildung 14. Architektur- und Prozessübersicht

Auf der Clientseite werden lokale und/oder ferne Anwendungen mit der Clientbibliothek von DB2 verbunden. Lokale Client kommunizieren über gemeinsamen Speicher und Semaphore, ferne Clients verwenden ein Protokoll wie benannte Pipes (Named Pipes - NPIPE), TCP/IP, NetBIOS oder SNA.

Auf der Serverseite werden die Aktivitäten durch zuteilbare Einheiten der Steuerkomponente, so genannte EDUs (Engine Dispatchable Units), gesteuert. In allen Abbildungen dieses Abschnitts werden EDUs als Kreise oder Gruppen von Kreisen dargestellt. EDUs werden als Threads in einem einzigen Prozess auf Windows-basierten Plattformen und als Prozesse unter UNIX implementiert. DB2-Agenten sind die gängigste Art von EDUs. Diese Agenten führen den größten Teil der SQL- und XQuery-Verarbeitung für Anwendungen aus. Vorablesefunktionen und Seitenlöschfunktionen sind weitere häufig genutzte Arten von EDUs.

Die Verarbeitung der Anforderungen einer Clientanwendung kann einer Gruppe von Subagenten übertragen werden. Mehrere Subagenten können zugeordnet werden, wenn die Maschine, auf der sich der Server befindet, mehrere Prozessoren hat oder Teil einer partitionierten Datenbank ist. In einer symmetrischen Multiprozessorumgebung (SMP) können mehrere SMP-Subagenten zum Beispiel die höhere Anzahl der Prozessoren ausnutzen.

Alle Agenten und Subagenten werden mit Hilfe eines Poolfunktionsalgorithmus verwaltet, der die Häufigkeit der Erstellung und Vernichtung von EDUs so gering wie möglich hält.

Pufferpools sind Bereiche des DatenbankserverSpeichers, in die Datenbankseiten mit Benutzertabellendaten, Indexdaten und Katalogdaten temporär eingelesen und dort modifiziert werden können. Pufferpools sind eine Schlüsseldeterminante der Datenbankleistung, weil der Zugriff auf Daten im Hauptspeicher wesentlich schneller als auf Daten auf dem Plattenspeicher erfolgen kann. Wenn größere Teile der Daten, die von Anwendungen benötigt werden, in einem Pufferpool vorhanden sind, ist für den Zugriff auf die Daten weniger Zeit erforderlich, als wenn sie auf einer Festplatte gesucht werden müssten.

Die Konfiguration der Pufferpools sowie der EDUs für Vorablesefunktionen und Seitenlöschfunktionen steuert, wie schnell auf Daten zugegriffen werden kann und wie rasch sie Anwendungen verfügbar gemacht werden können.

- **Vorablesefunktionen** rufen Daten von der Festplatte ab und lesen sie in den Pufferpool ein, bevor Anwendungen die Daten benötigen. Beispielsweise müssten Anwendungen, die große Volumina von Daten durchsuchen müssen, darauf warten, dass Daten vom Plattenspeicher in den Pufferpool gelesen werden, wenn es keine Vorablesefunktionen für Daten gäbe. Agenten der Anwendung senden asynchrone Vorableseanforderungen an eine allgemeine Vorablesewarteschlange. Wenn Vorablesefunktionen verfügbar werden, implementieren sie diese Anforderungen. Dabei verwenden sie Eingabeverfahren wie das Lesen großer Blöcke (Big-Block Read) oder gestreutes Lesen (Scatter Read), um die angeforderten Seiten vom Plattenspeicher in den Pufferpool zu laden. Wenn Sie mehrere Plattendatenträger zum Speichern der Datenbankdaten haben, können die Daten über die Plattendatenträger einheitenübergreifend gespeichert werden (Striping). Durch dieses einheitenübergreifende Lesen und Schreiben von Daten können Vorablesefunktionen mehrere Festplatten gleichzeitig zum Abrufen von Daten verwenden.
- **Seitenlöschfunktionen** speichern Daten aus dem Pufferpool zurück auf den Plattenspeicher. Seitenlöschfunktionen sind im Hintergrund aktive EDUs, die unabhängig von den Anwendungsagenten arbeiten. Sie suchen nach Seiten im Pufferpool, die nicht länger benötigt werden, und schreiben die Seiten auf den Plattenspeicher. Seitenlöschfunktionen stellen sicher, dass im Pufferpool Platz für Seiten ist, die von Vorablesefunktionen eingelesen werden.

Ohne diese unabhängigen EDUs für Vorablesezugriffe und Seitenlöschfunktionen müssten Anwendungsagenten die gesamten Lese- und Schreiboperationen für Daten zwischen dem Pufferpool und dem Plattenspeicher selbst durchführen.

Zugehörige Konzepte:

- „Verbesserungen des Verbindungskonzentrators für Clientverbindungen“ auf Seite 171
- „Deadlocks“ auf Seite 78
- „Vorablesen von Daten in den Pufferpool“ auf Seite 141

- „Verzeichnisse und Dateien einer Datenbank“ in *Systemverwaltung: Konzept*
- „Verringern des Protokollierungsaufwands zur Verbesserung der Abfrageleistung“ auf Seite 151
- „Client-/Serververarbeitungsmodell“ auf Seite 166
- „Aktualisierungsverarbeitung“ auf Seite 139

Zugehörige Referenzen:

- „max_connections - Maximale Anzahl von Clientverbindungen “ auf Seite 485
- „max_coordagents - Maximale Anzahl koordinierender Agenten “ auf Seite 485

Das DB2-Prozessmodell

Wenn Sie mit dem DB2-Prozessmodell vertraut sind, kann dieses Wissen Ihnen bei der Bestimmung des Fehlertyps behilflich sein, da Sie dann verstehen, wie der Datenbankmanager mit den zugehörigen Komponenten interagiert.

Das Prozessmodell, das von allen DB2-Servern verwendet wird, erleichtert die Kommunikation, die zwischen Datenbankservern und Clients und lokalen Anwendungen stattfindet. Es stellt zudem sicher, dass Datenbankanwendungen von Ressourcen wie Datenbanksteuerblöcken und kritischen Datenbankdateien isoliert werden.

UNIX-basierte Umgebungen arbeiten mit einer Architektur auf der Basis von **Systemprozessen**. Zum Beispiel werden die Listenerfunktionen für die DB2-Kommunikation als Systemprozesse erstellt. Intel-Betriebssysteme, wie z. B. Windows, arbeiten mit einer Architektur auf der Basis von **Threads**, um die Leistung zu optimieren. Sofern nicht ausdrücklich erwähnt, bezieht sich der Begriff *Prozess* in der vorliegenden Darstellung sowohl auf Prozesse als auch auf Threads. Einzelheiten zu den Unterschieden zwischen der Verwendung von Windows-Threads und UNIX-Prozessen werden später in diesem Abschnitt behandelt.

Für jede Datenbank, auf die zugegriffen wird, werden verschiedene Prozesse gestartet, um die verschiedenen Datenbankaufgaben, wie Vorablesen, Kommunikation und Protokollierung, wahrzunehmen.

Jeder Prozess einer Clientanwendung verfügt über einen einzigen **Koordinatoragenten**, der mit der Datenbank arbeitet. Ein Koordinatoragent arbeitet im Auftrag einer Anwendung und kommuniziert mit anderen Agenten durch Interprozesskommunikation (IPC) oder über Protokolle zur Fernkommunikation.

Die DB2-Architektur stellt eine **Firewall** bereit, sodass Anwendungen in einem anderen Adressraum als DB2 ausgeführt werden. Die Firewall schützt die Datenbank und den Datenbankmanager vor Anwendungen, gespeicherten Prozeduren und benutzerdefinierten Funktionen (UDFs). Eine Firewall wahrt die Integrität der Daten in den Datenbanken, weil sie verhindert, dass interne Puffer oder Dateien des Datenbankmanagers durch Programmierfehler in Anwendungen überschrieben werden. Die Firewall erhöht außerdem die Zuverlässigkeit, weil Anwendungsfehler den Datenbankmanager nicht zum Absturz bringen können.

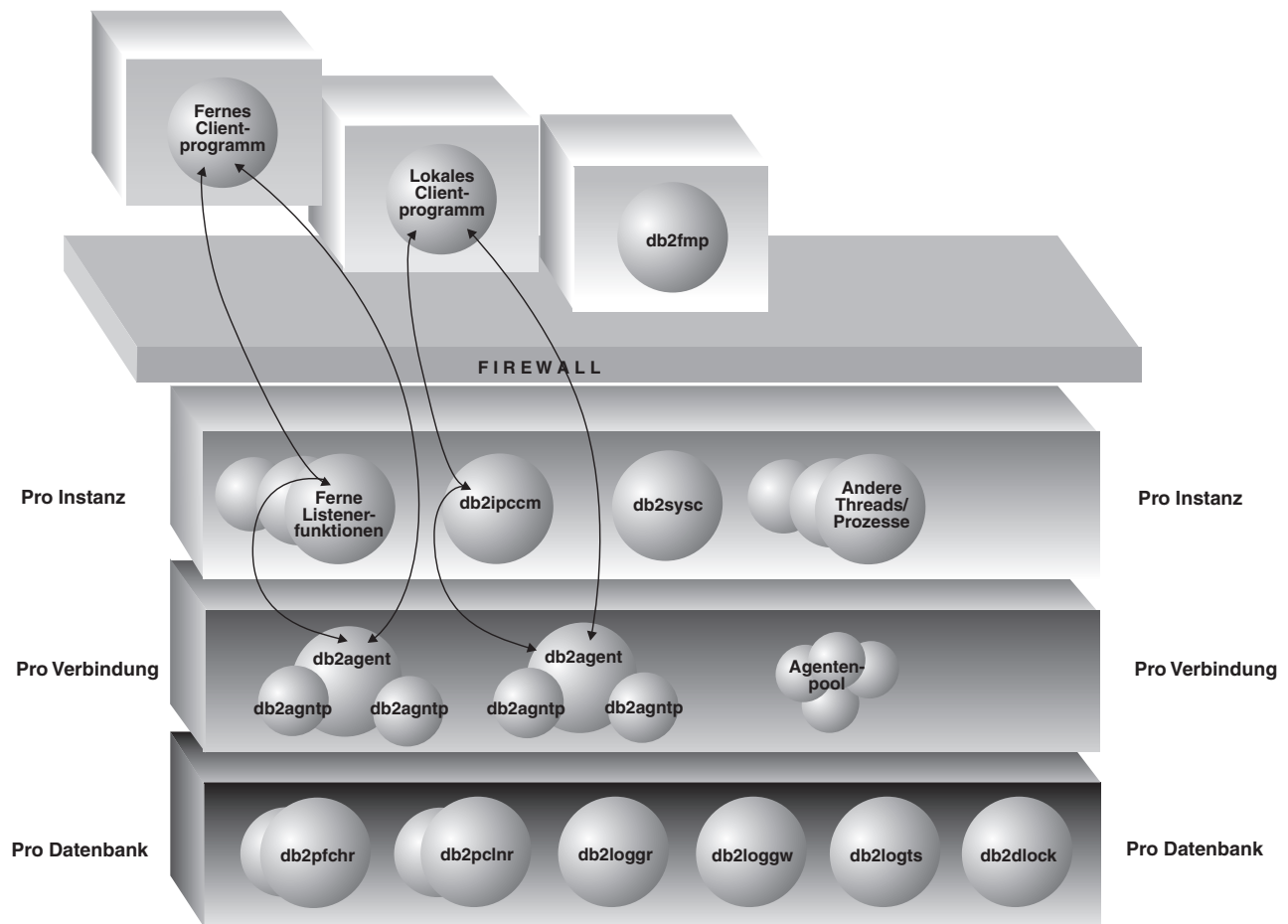


Abbildung 15. Prozessmodell für DB2-Systeme

Die folgende Liste enthält weitere Einzelheiten zu den in der Abbildung gezeigten Prozessen:

Clientprogramme:

Clientprogramme werden fern oder auf der gleichen Maschine wie der Datenbankserver ausgeführt. Sie stellen den ersten Kontakt mit der Datenbank über eine Listenerfunktion her. Anschließend wird ihnen ein Koordinatoragent (db2agent) zugeordnet.

Listenerfunktionen:

Clientprogramme stellen den einleitenden Kontakt mit kommunikationsbereiten Agenten her, die gestartet werden, wenn DB2 gestartet wird. Für jedes Kommunikationsprotokoll ist eine Listenerfunktion konfiguriert. Eine Listenerfunktion für Interprozesskommunikation (db2ipccm) ist für lokale Clientprogramme konfiguriert. Folgende Listenerfunktionen sind vorhanden:

- **db2ipccm** für lokale Clientverbindungen
- **db2tcpdm** für TCP/IP-Verbindungen
- **db2snacm** für APPC-Verbindungen
- **db2tcpdm** für TCP/IP-Anforderungen des Tools Discovery

Agenten:

Allen Verbindungsanforderungen von Clientanwendungen, d. h. lokalen und fernem, wird ein entsprechender Koordinatoragent (**db2agent**) zugeordnet. Wenn der Koordinatoragent erstellt ist, führt er alle Datenbankankforderungen im Auftrag der Anwendung aus.

In einigen Umgebungen, in denen der Konfigurationsparameter *intra_parallel* des Datenbankmanagers aktiviert ist, verteilt der Koordinatoragent die Datenbankankforderungen auf Subagenten (**db2agntp**). Diese Agenten führen die Anforderungen für die Anwendung aus. Wenn der Koordinatoragent erstellt ist, verarbeitet er alle Datenbankankforderungen für seine Anwendung, indem er die Subagenten (**db2agntp**) koordiniert, die Anforderungen in der Datenbank ausführen. Subagenten, die zwar einer Anwendung zugeordnet, aber momentan inaktiv sind, werden durch den Prozessnamen **db2agnta** identifiziert. Unabhängige Koordinatoragenten werden durch den Prozessnamen **db2agnti** identifiziert.

Ein Koordinatoragent kann folgende Verbindungen haben:

- Eine Verbindung (CONNECT) zur Datenbank mit einem Aliasnamen. Zum Beispiel ist "db2agent (DATA1)" mit dem Datenbankaliasnamen "DATA1" verbunden.
- Eine Verbindung (ATTACH) zu einer Instanz. Zum Beispiel ist "db2agent (user1)" mit der Instanz "user1" verbunden.

Ein weiterer Agententyp (**db2agnsc**) wird von DB2 erstellt, um bestimmte Operationen parallel auszuführen. Diese Agenten werden insbesondere bei Aktionen zur Datenbankrecovery verwendet.

Nicht zugeordnete Agenten befinden sich im Agentenpool. Diese Agenten sind für Anforderungen von Koordinatoragenten, die für Clientprogramme aktiv sind, oder für Anforderungen von Subagenten, die für vorhandene Koordinatoragenten aktiv sind, verfügbar. Die Anzahl der verfügbaren Agenten hängt vom Wert der Konfigurationsparameter *maxagents* und *num_poolagents* des Datenbankmanagers ab.

db2fmp:

Der Prozess im abgesicherten Modus. In seine Zuständigkeit fällt die Ausführung abgesicherter gespeicherter Prozeduren und benutzerdefinierter Funktionen außerhalb der Firewall. Der Prozess *db2fmp* ist immer ein separater Prozess, kann jedoch je nach den Arten der ausgeführten Routinen mehrere Threads (Multithreading) enthalten.

Threads/Prozesse der Datenbank:

Die folgende Liste enthält einige der wichtigen Threads/Prozesse, die von jeder Datenbank verwendet werden:

- **db2pfchr** für Vorablesefunktionen für Pufferpools.
- **db2pclnr** für Seitenlöschfunktionen für Pufferpools.
- **db2loggr** zur Behandlung von Protokolldateien für die Durchführung der Transaktionsverarbeitung und -recovery.
- **db2loggw** zum Schreiben von Protokollsätzen in Protokolldateien.

- **db2logts** zur Erfassung von Protokollinformationen darüber, welche Protokolle bei der Änderung eines Tabellenbereichs aktiv sind. Diese Informationen werden letztlich in der Datei DB2TSG.HIS im Datenbankverzeichnis aufgezeichnet. Sie dienen der Beschleunigung der aktualisierenden Recovery (Rollforward) von Tabellenbereichen.
- **db2dlock** zur Erkennung von Deadlocks. In einer Umgebung mit partitionierten Datenbanken wird ein zusätzlicher Prozess **db2glock** zur Koordinierung der Informationen verwendet, die aus den db2dlock-Prozessen der einzelnen Partitionen gesammelt werden. Der Prozess *db2glock* wird nur in der Katalogtabellenpartition ausgeführt.
- **db2taskd** zum Verteilen von im Hintergrund ausgeführten Datenbanktasks. Die Tasks werden durch **db2taskp**-Prozesse ausgeführt.
- **db2hadrp**, primärer HADR-Serverprozess.
- **db2hadrs**, HADR-Bereitschaftsserverprozess.
- **db2lfr** für Protokolldatei-Leseeinheiten, die einzelne Protokolldateien verarbeiten.
- **db2logts** zeichnet auf, welche Tabellenbereiche in den einzelnen Protokolldateien über Protokollsätze verfügen.
- **db2shred** verarbeitet einzelne Protokollsätze in Protokollseiten.
- **db2redom** für den Master für aktualisierende Wiederherstellung. Bei der Recovery werden Protokollsätze für die aktualisierende Wiederherstellung verarbeitet und Protokollsätze Worker für aktualisierende Wiederherstellung zugeordnet.
- **db2redow** für den Worker für aktualisierende Wiederherstellung. Bei der Recovery werden Protokollsätze für die aktualisierende Wiederherstellung bei der Anforderung des Masters für aktualisierende Wiederherstellung verarbeitet.
- **db2logmgr** für den Protokollmanager. Verwaltet Protokolldateien für eine wiederherstellbare Datenbank.
- Ereignismonitorprozesse werden wie folgt angegeben:
 - **db2evm%1%2 (%3)** Dabei kann %1 für Folgendes stehen:
 - **g** - globaler Dateiereignismonitor
 - **l** - lokaler Dateiereignismonitor
 - **t** - Tabellenereignismonitor
 - **gp** - globaler, über eine Pipe geleiteter Ereignismonitor
 - **lp** - lokaler, über eine Pipe geleiteter Ereignismonitor
 %2 kann für Folgendes stehen:
 - **i** - Koordinator
 - **p** - kein Koordinator
 und %3 steht für den Namen des Ereignismonitors.
- Backup- und Restoreprozesse werden wie folgt angegeben:
 - **db2bm.%1.%2** - Puffermanipulator für Backup und Restore und **db2med.%1.%2** - Mediencontroller für Backup und Restore. Dabei gilt Folgendes:
 - %1 - Die Prozessnummer des Agenten, der die Backup- bzw. Restoresitzung steuert.
 - %2 - Ein sequenzieller Wert, der zur eindeutigen Unterscheidung der (möglicherweise vielen) Prozesse verwendet wird, die zu einer bestimmten Backup- bzw. Restoresitzung gehören.

Beispiel: db2bm.13579.2 gibt den zweiten db2bm-Prozess an, der durch den db2agent-Prozess mit der PID 13579 gesteuert wird.

Threads und Prozesse des Datenbankservers:

Der Systemcontroller (**db2sysc**) muss vorhanden sein, damit der Datenbankserver funktioniert. Darüber hinaus können die folgenden Threads und Prozesse gestartet werden, um verschiedene Aufgaben zu übernehmen:

- **db2resync** ist der Resynchronisationsagent, der die globale Resynchronisationsliste (resync) durchsucht.
- **db2gds** ist der globale Dämonenstarter auf UNIX-basierten Systemen, der neue Prozesse startet.
- **db2wdog** ist der Wachprozess auf UNIX-basierten Systemen, der abnormale Beendigungen behandelt.
- **db2fcms** ist der FCM-Senderdämon.
- **db2fcmr** ist der FCM-Empfängerdämon.
- **db2pdbc** ist der parallele Systemcontroller, der Parallelanforderungen von fernen Knoten verarbeitet (nur in einer partitionierten Datenbankumgebung).
- **db2cart** dient zur Archivierung von Protokolldateien, wenn auf eine Datenbank zugegriffen wird, für die der Benutzerexit (USEREXIT) aktiviert ist.
- **db2fmtlg** dient zur Formatierung von Protokolldateien, wenn auf eine Datenbank zugegriffen wird, in deren Konfiguration LOGRETAIN aktiviert, USEREXIT jedoch inaktiviert ist.
- **db2panic** ist der Notfallagent, der dringende Anforderungen verarbeitet, nachdem Agentengrenzwerte auf einem bestimmten Knoten erreicht wurden (nur in einer partitionierten Datenbankumgebung).
- **db2srlst** verwaltet Listen mit Adressen für Systeme, wie z. B. DB2 für z/OS.
- **db2fmd** ist der Fehlermonitordämon.
- **db2disp** ist der Dispatcher für den Clientverbindungskonzentrator.
- **db2acd** ist der Autonomic Computing-Dämon, der den Host für die Dienstprogramme für den Diagnosemonitor und zur automatischen Pflege darstellt. Dieser Prozess wurde früher **db2hmon** genannt.

Unterschiede zwischen Windows und UNIX

DB2 für Windows unterscheidet sich von UNIX-basierten Umgebungen dadurch, dass die Datenbanksteuerkomponente mit mehreren Threads (Multithreading) und nicht mit mehreren Prozessen arbeitet. In Windows-Systemen ist jede zuteilbare Einheit auf der Agentenseite der Firewall ein Thread unter dem Prozess *db2sysc*. Dadurch kann die Datenbanksteuerkomponente dem Betriebssystem ermöglichen, Taskumschaltungen auf der Threadebene anstatt der Prozessebene durchzuführen. Für jede Datenbank, auf die zugegriffen wird, werden Threads gestartet, die die Datenbanktasks erledigen (z. B. den Vorabsezugriff).

Ein weiterer Unterschied besteht in der Behandlung abnormaler Beendigungen. In Windows-Systemen ist kein Wachprozess (watchdog) erforderlich, weil diese Systeme sicherstellen, dass die zugeordneten Ressourcen nach einer abnormalen Beendigung bereinigt werden. Daher gibt es auf Windows-Systemen kein Gegenstück zum Prozess *db2wdog*. Darüber hinaus ist auf Windows-Systemen auch kein Prozess oder Thread *db2gds* erforderlich, da diese Systeme über eigene Mechanismen zum Starten von Threads verfügen.

Zugehörige Konzepte:

- „FFDC - First Failure Data Capture“ in *Fehlerbehebung*
- „Ermitteln des Status aktiver Prozesse“ in *Fehlerbehebung*

Verwaltung von Datenbankagenten

Die meisten Anwendungen richten eine Eins-zu-eins-Beziehung zwischen der Zahl der verbundenen Anwendungen und der Zahl der Anwendungsanforderungen ein, die von der Datenbank verarbeitet werden können. Es kann jedoch sein, dass Sie aufgrund Ihrer Arbeitsumgebung eine Viele-zu-eins-Beziehung zwischen der Zahl der verbundenen Anwendungen und der Zahl der Anwendungsanforderungen benötigen, die verarbeitet werden können.

Die Möglichkeit, diese Faktoren separat zu steuern, wird durch Konfigurationsparameter des Datenbankmanagers bereitgestellt:

- Der Parameter *max_connections* gibt die Anzahl verbundener Anwendungen an.
- Der Parameter *max_coordagents* gibt die Anzahl von Anwendungsanforderungen an, die verarbeitet werden können.

Der Verbindungskonzentrator ist aktiviert, wenn der Wert des Parameters *max_connections* größer als der Wert des Parameters *max_coordagents* ist.

Da jeder aktive Koordinatoragent globale Ressourcen erfordert, steigt mit wachsender Zahl dieser Agenten auch die Wahrscheinlichkeit, dass die Obergrenze an verfügbaren globalen Datenbankressourcen erreicht wird. Um das Erreichen der Obergrenze der verfügbaren globalen Datenbankressourcen zu vermeiden, können Sie den Parameter *max_connections* auf einen höheren Wert als den Parameter *max_coordagents* setzen.

Zugehörige Konzepte:

- „Agenten in einer partitionierten Datenbank“ auf Seite 163
- „Konfigurationsparameter mit Auswirkung auf die Anzahl von Agenten“ auf Seite 165
- „Verbesserungen des Verbindungskonzentrators für Clientverbindungen“ auf Seite 171

Agenten in einer partitionierten Datenbank

In Umgebungen mit partitionierten Datenbanken und Umgebungen mit aktivierter partitionsinterner Parallelität verfügt jede Datenbankpartition (d. h. jeder Datenbankserver oder Knoten) über einen eigenen Pool von Agenten, aus dem Subagenten entnommen werden. Durch die Verwendung dieses Pools müssen Subagenten nicht jedes Mal erstellt und wieder gelöscht werden, wenn ein Subagent benötigt wird oder seine Arbeit beendet hat. Die Subagenten können als zugeordnete Agenten im Pool bleiben und vom Datenbankmanager für neue Anforderungen von der Anwendung, der sie zugeordnet sind, verwendet werden.

Anmerkung: Wenn der Verbindungskonzentrator aktiviert ist, werden Subagenten nicht unbedingt einer Anwendung zugeordnet.

In Umgebungen mit partitionierten Datenbanken und Umgebungen mit aktivierter partitionsinterner Parallelität besteht eine enge Beziehung zwischen dem Einfluss auf die Leistung und den Speicherbedarf innerhalb des Systems und der Optimierung des Agentenpools:

- Der Konfigurationsparameter des Datenbankmanagers für die Größe des Agentenpools (*num_poolagents*) betrifft die Gesamtanzahl von Subagenten, die zu Anwendungen in einer Datenbankpartition (auch als Knoten bezeichnet) zuge-

ordnet bleiben können. Wenn die Poolgröße zu klein ist und der Pool voll ist, wird ein Subagent aus der Zuordnung mit der Anwendung, für die er aktiv ist, gelöst und beendet. Da Subagenten ständig erstellt und Anwendungen erneut zugeordnet werden müssen, sinkt die Leistung.

Außerdem könnte bereits eine Anwendung den Pool mit zugeordneten Subagenten füllen, wenn der Wert für den Parameter *num_poolagents* zu klein ist. Wenn nun eine andere Anwendung einen neuen Subagenten benötigt und im zugeordneten Pool über keine Agenten verfügt, stoppt sie Subagenten aus den Agentenpools anderer Anwendungen und startet sie erneut. Dieses Verhalten stellt sicher, dass Ressourcen vollständig belegt werden.

- Wägen Sie die Vor- und Nachteile zu weniger Agenten gegen den Ressourcenbedarf zu vieler Agenten ab, die zu einem gegebenen Zeitpunkt aktiv sein können.

Wenn der Wert des Parameters *num_poolagents* zum Beispiel zu groß ist, werden zugeordnete Subagenten lange Zeit ungenutzt im Pool behalten und belegen Datenbankmanagerressourcen, die für andere Tasks nicht verfügbar sind.

Anmerkung: Wenn der Verbindungskonzentrator aktiviert ist, hat die Anzahl von Agenten, die durch den Parameter *num_poolagents* angegeben wird, nur empfehlenden Charakter. Es können sich zu einem gegebenen Zeitpunkt auch mehr Agenten im Agentenpool befinden.

Andere asynchrone Prozesse und Threads

Neben den Datenbankagenten werden auch noch andere asynchrone Aktivitäten des Datenbankmanagers als eigene Prozesse bzw. Threads ausgeführt. Dazu gehören:

- E/A-Server oder E/A-Vorabselektfunktionen der Datenbank
- Asynchrone Seitenlöschfunktionen der Datenbank
- Protokollfunktionen der Datenbank
- Deadlock-Detektoren für Datenbanken
- Ereignismonitore
- Übertragungs- und IPC-Empfangsprozesse (Listeners)
- Funktionen zum Neuausgleich von Daten in Tabellenbereichscontainern

Zugehörige Konzepte:

- „Konfigurationsparameter mit Auswirkung auf die Anzahl von Agenten“ auf Seite 165
- „Datenbankagenten“ auf Seite 153
- „Verwaltung von Datenbankagenten“ auf Seite 163
- „Konfiguration von E/A-Servern für Vorabselektzugriff und Parallelität“ auf Seite 146
- „Illustration des Vorablesens mit paralleler E/A“ auf Seite 147

Konfigurationsparameter mit Auswirkung auf die Anzahl von Agenten

Die folgenden Konfigurationsparameter des Datenbankmanagers legen fest, wie viele Datenbankagenten erstellt werden und wie sie verwaltet werden:

- Maximale Anzahl von Agenten (*maxagents*): Die Anzahl von Agenten, die gleichzeitig aktiv sein können. Dieser Wert bezieht sich auf die Gesamtanzahl von Agenten, die für alle Anwendungen aktiv sind, einschließlich Koordinatoragenten, Subagenten, inaktive Agenten und Agenten im Bereitschaftsmodus.
- Agentenpoolgröße (*num_poolagents*): Die Gesamtanzahl von Agenten, einschließlich aktiver Agenten und Agenten im Agentenpool, die im System verfügbar gehalten werden. Der Standardwert für diesen Parameter ist die Hälfte der durch *maxagents* angegebenen Anzahl.
- Anfangswert für die Anzahl von Agenten im Pool (*num_initagents*): Wenn der Datenbankmanager gestartet wird, wird ein Pool von Verarbeitungsagenten nach diesem Wert erstellt. Dadurch wird die Leistung für Erstabfragen erhöht. Die Verarbeitungsagenten beginnen alle als Agenten im Bereitschaftsmodus.
- Maximale Anzahl von Verbindungen (*max_connections*): Gibt die maximale Anzahl von Verbindungen zum Datenbankmanagersystem an, die in jeder Datenbankpartition zulässig sind.
- Maximale Anzahl koordinierender Agenten (*max_coordagents*): In Umgebungen mit partitionierten Datenbanken und Umgebungen mit aktivierter partitionsinterner Parallelität und aktiviertem Verbindungskonzentrator begrenzt dieser Wert die Anzahl koordinierender Agenten.
- Maximale Anzahl gleichzeitig aktiver Agenten (*maxcagents*): Mit diesem Wert wird die Anzahl von *Token* gesteuert, die der Datenbankmanager zulässt. Für jede Datenbanktransaktion (UOW = Unit of Work, Arbeitseinheit), die stattfindet, wenn ein Client mit einer Datenbank verbunden ist, muss ein koordinierender Agent die Berechtigung zur Verarbeitung vom Datenbankmanager einholen. Diese Berechtigung wird als *Verarbeitungstoken* bezeichnet. Der Datenbankmanager erlaubt nur Agenten mit einem Verarbeitungstoken, eine UOW an einer Datenbank auszuführen. Wenn kein Token verfügbar ist, muss der Agent warten, bis ein Token zur Verarbeitung der Transaktion verfügbar ist.

Dieser Parameter kann in Umgebungen nützlich sein, in denen der Leistungsbedarf in Zeiten hoher Auslastung die Kapazität der vorhandenen Systemressourcen für Hauptspeicher, CPU und Plattenspeicher übersteigt. In einer solchen Umgebung könnte zum Beispiel das Aus- und Einlagern von Seiten (Paging) eine Leistungsbeeinträchtigung in Spitzenlastzeiten zur Folge haben. Mit diesem Parameter können Sie die Auslastung steuern und den Leistungsabfall verhindern. Er kann sich jedoch auf den gleichzeitigen Zugriff, auf Wartezeiten oder auf beides auswirken.

Zugehörige Konzepte:

- „Agenten in einer partitionierten Datenbank“ auf Seite 163
- „Datenbankagenten“ auf Seite 153
- „Verwaltung von Datenbankagenten“ auf Seite 163
- „Verbindungskonzentrator“ in *DB2 Connect Benutzerhandbuch*

Client-/Serververarbeitungsmodell

Lokale und ferne Anwendungsprozesse können mit derselben Datenbank arbeiten. Eine ferne Anwendung initiiert eine Datenbankaktion von einer Maschine aus, die sich von der Datenbankmaschine entfernt befindet. Lokale Anwendungen sind auf der Servermaschine direkt mit der Datenbank verbunden.

Wie DB2 Clientverbindungen verwaltet, ist davon abhängig, ob der Verbindungskonzentrator aktiviert ist oder nicht. Der Verbindungskonzentrator ist aktiviert, wenn der Konfigurationsparameter *max_connections* des Datenbankmanagers auf einen höheren Wert als der Konfigurationsparameter *max_coordagents* gesetzt ist.

- Wenn der Verbindungskonzentrator inaktiviert ist, wird jeder Clientanwendung eine eindeutige EDU (Engine Dispatchable Unit) zugeordnet, die als *Koordinatoragent* bezeichnet wird und die die Verarbeitung für diese Anwendung koordiniert und mit der Anwendung kommuniziert.
- Wenn der Verbindungskonzentrator aktiviert ist, kann jeder Koordinatoragent viele Clientverbindungen jeweils einzeln verwalten und ggf. die anderen Arbeitsagenten zur Erledigung dieser Arbeit koordinieren. Für Internetanwendungen mit vielen relativ kurzfristigen Verbindungen oder ähnliche Anwendungen mit zahlreichen relativ kleinen Transaktionen verbessert der Verbindungskonzentrator die Leistung, indem er wesentlich mehr Clientanwendungen die Herstellung einer Verbindung ermöglicht. Darüber hinaus verringert er die Belastung der Systemressourcen durch jede einzelne Verbindung.

Alle Kreise in der folgenden Abbildung stellen EDUs (Engine Dispatchable Units, zuteilbare Einheiten der Steuerkomponente) dar, die auf UNIX-Plattformen als „Prozesse“ und auf Windows-Betriebssystemen als „Threads“ bezeichnet werden.

Bevor die Arbeit ausgeführt werden kann, die für die Anwendung in der Datenbank durchgeführt werden soll, muss zwischen einer Anwendung und dem Datenbankmanager ein Kommunikationsmittel eingerichtet werden.

Bei A1 in der folgenden Abbildung richtet ein lokaler Client eine Kommunikationsverbindung zunächst über eine *db2ipccm*-EDU für ein. Bei A2 arbeitet die *db2ipccm*-EDU mit einer *db2agent*-EDU, die zum Koordinatoragenten für die Anwendungsanforderungen vom lokalen Client wird. Der Koordinatoragent nimmt anschließend bei A3 mit der Clientanwendung Kontakt auf, um zwischen der Clientanwendung und dem Koordinator eine Kommunikation über gemeinsamen Speicher einzurichten. Die Anwendung auf dem lokalen Client wird bei A4 mit der Datenbank verbunden.

Bei B1 in der folgenden Abbildung stellt ein ferner Client eine Kommunikationsverbindung über eine *db2tcpcm*-EDU her. Wenn ein anderes Kommunikationsprotokoll ausgewählt wird, wird der entsprechende Kommunikationsmanager verwendet. Die *db2tcpcm*-EDU richtet eine TCP/IP-Kommunikation zwischen der Clientanwendung und der *db2tcpcm*-EDU ein. Dann arbeitet sie mit einer *db2agent*-EDU bei B2, die zum Koordinatoragenten für die Anwendung wird, und übergibt die Verbindung an diesen Agenten. Bei B3 nimmt der Koordinatoragent Kontakt mit der fernen Clientanwendung auf und wird mit der Datenbank verbunden.

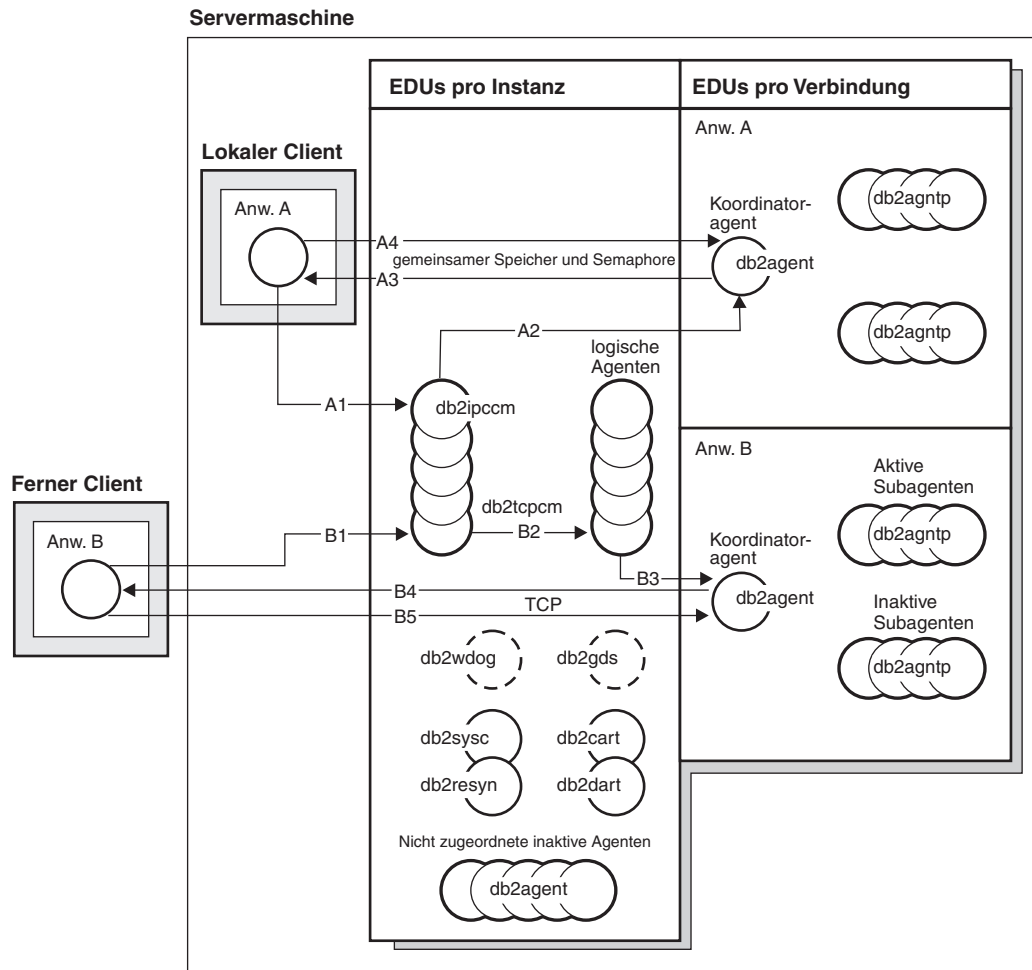


Abbildung 16. Übersicht über das Prozessmodell

Ferner sind folgende Details dieser Abbildung zu beachten:

- Arbeitsagenten führen Anwendungsanforderungen aus.
- Es gibt vier Typen von Arbeitsagenten: aktive Koordinatoragenten, aktive Subagenten, zugeordnete Subagenten und nicht zugeordnete Agenten in Bereitschaft.
- Jede Clientverbindung wird mit einem aktiven Koordinatoragenten verknüpft.
- In einer Umgebung mit partitionierten Datenbanken sowie in Umgebungen mit aktivierter partitionsinterner Parallelität verteilen die Koordinatoragenten Datenbankanforderungen an Subagenten (db2agntp). Die Subagenten führen die Anforderungen für die jeweilige Anwendung aus.
- Es gibt einen Agentenpool (db2agent), in dem inaktive Agenten zusammengefasst im Bereitschaftsmodus auf neue Arbeit warten.
- Andere EDUs verwalten Clientverbindungen, Protokolle, zweiphasige Commits, Backup- und Restore-Tasks sowie weitere Tasks.

Servermaschine

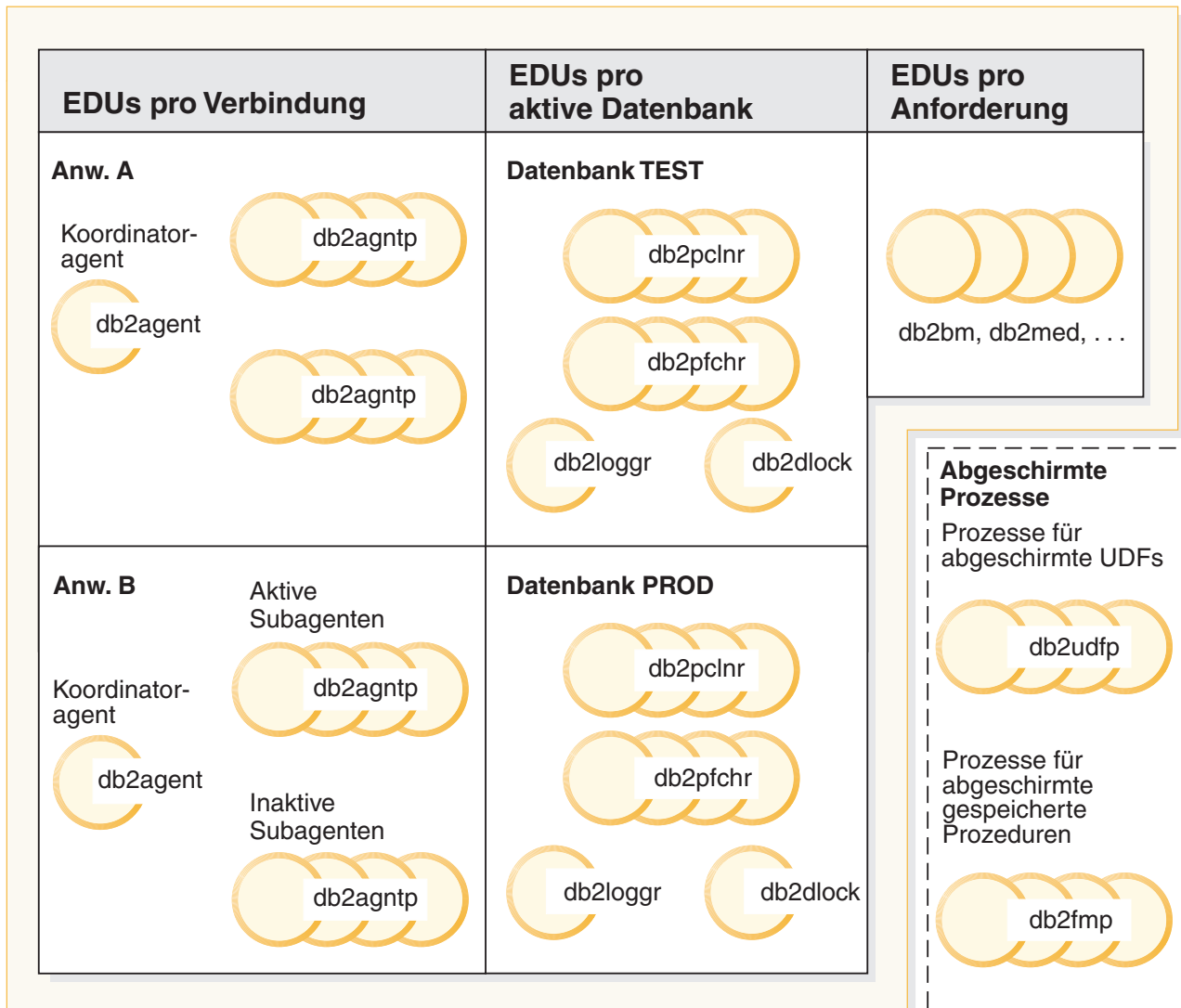


Abbildung 17. Prozessmodell, Teil 2

In dieser Abbildung werden zusätzliche EDUs (Engine Dispatchable Units, zuteilbare Einheiten der Steuerkomponente) gezeigt, die Teil der Servermaschinenumgebung sind. Alle aktiven Datenbanken verfügen über eigene gemeinsame Pools mit Vorablesefunktionen (db2pfchr) und Seitenlöschfunktionen (db2pclnr) sowie eigene Protokollfunktionen (db2loggr) und Deadlock-Detektoren (db2dlock).

Abgeschirmte benutzerdefinierte Funktionen (UDFs) und gespeicherte Prozeduren, die in der Abbildung nicht gezeigt werden, werden verwaltet, um den Aufwand zu minimieren, der mit ihrer Erstellung und Löschung verbunden ist. Der Standardwert für den Konfigurationsparameter *keepfenced* des Datenbankmanagers ist „YES“, wodurch der Prozess der gespeicherten Prozedur zur erneuten Verwendung beim nächsten Aufruf einer gespeicherten Prozedur verfügbar bleibt.

Anmerkung: Nicht abgeschirmte benutzerdefinierte Funktionen (UDFs) und gespeicherte Prozeduren werden zwecks besserer Leistung direkt im Adressraum des Agenten ausgeführt. Da diese UDFs und gespeicher-

ten Prozeduren jedoch über unbeschränkten Zugriff auf den Adressraum des Agenten verfügen, müssen sie vor ihrer Verwendung umfassend getestet werden.

Das Prozessmodell mit mehreren Datenbankpartitionen ist eine logische Erweiterung des Prozessmodells mit einer einzigen Datenbankpartition. Beide Betriebsarten werden durch eine gemeinsame Codebasis unterstützt. In der folgenden Abbildung werden die Ähnlichkeiten und Unterschiede zwischen dem Prozessmodell mit einer einzigen Datenbankpartition, wie in den beiden vorangehenden Abbildungen gezeigt, und dem Prozessmodell mit mehreren Datenbankpartitionen angezeigt.

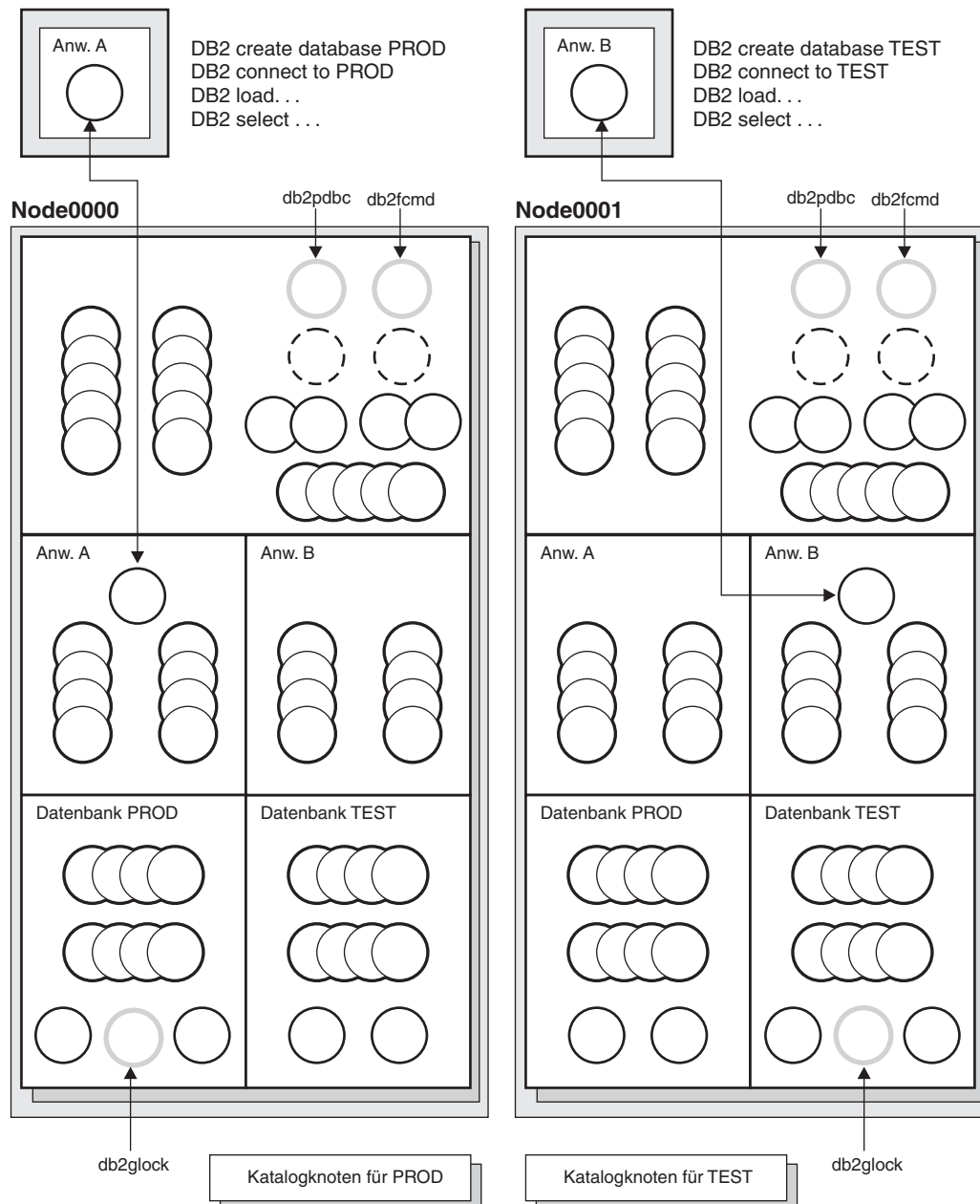


Abbildung 18. Prozessmodell und mehrere Datenbankpartitionen

Die meisten EDUs stimmen zwischen dem Prozessmodell mit einer einzigen Datenbankpartition und dem Prozessmodell mit mehreren Datenbankpartitionen überein.

In einer Umgebung mit mehreren Datenbankpartitionen (bzw. Knoten) ist eine der Datenbankpartitionen der Katalogknoten. Der Katalog speichert alle Information in Bezug auf Objekte in der Datenbank.

Wie in der obigen Abbildung gezeigt, wird der Katalog der PROD-Datenbank im Knoten Node0000 erstellt, weil Anwendung A die PROD-Datenbank auf diesem Knoten erstellt. Analog wird der Katalog für die TEST-Datenbank im Knoten Node0001 erstellt, weil Anwendung B die TEST-Datenbank auf diesem Knoten erstellt. Die Datenbanken sollten in verschiedenen Knoten erstellt werden, um so die zusätzlichen mit den Katalogen verbundenen Aktivitäten für jede Datenbank gleichmäßig auf die Knoten in der Systemumgebung zu verteilen.

Der Instanz sind zusätzliche EDUs (db2pdbc und db2fcmd) zugeordnet, die sich in jedem Knoten einer Datenbankumgebung mit mehreren Partitionen befinden. Diese EDUs werden für die Koordination von Anforderungen zwischen Datenbankpartitionen und zum Aktivieren des FCM (Fast Communications Manager) benötigt.

Dem Katalogknoten der Datenbank ist ebenfalls eine zusätzliche EDU (db2glock) zugeordnet. Diese EDU steuert globale Sperren für die Knoten, in denen sich die aktive Datenbank befindet.

Jede CONNECT-Verbindung von einer Anwendung wird durch eine Verbindung dargestellt, die einem Koordinatoragenten zur Verwaltung der Verbindung zugeordnet ist. Der *Koordinatoragent* ist der Agent, der mit der Anwendung kommuniziert, indem er Anforderungen empfängt und Antworten sendet. Er kann die Anforderung selbst erfüllen oder mehrere Subagenten zur Bearbeitung der Anforderung koordinieren. Die Datenbankpartition, in der sich der Koordinatoragent befindet, wird als *Koordinator-knoten* dieser Anwendung bezeichnet. Der Koordinator-knoten kann auch mit dem Befehl SET CLIENT CONNECT_NODE festgelegt werden.

Teile der Datenbankanforderungen von der Anwendung werden vom Koordinator-knoten an Subagenten der anderen Datenbankpartitionen gesendet. Alle Ergebnisse der anderen Datenbankpartitionen werden auf dem Koordinator-knoten zusammengeführt, bevor sie zurück zur Anwendung gesendet werden.

Die Datenbankpartition, in der der Befehl CREATE DATABASE ausgeführt wurde, wird „Katalogknoten“ der Datenbank genannt. In dieser Datenbankpartition werden die Katalogtabellen gespeichert. Normalerweise werden alle Benutzertabellen über eine Gruppe von Knoten verteilt.

Anmerkung: Eine beliebige Anzahl von Datenbankpartitionen kann zur Ausführung auf einer einzigen Maschine konfiguriert werden. Dies wird als Konfiguration mit „mehreren logischen Partitionen“ oder mit „mehreren logischen Knoten“ bezeichnet. Eine solche Konfiguration ist bei großen SMP-Maschinen (SMP - Symmetric Multiprocessor) mit einem sehr großen Hauptspeicher sehr nützlich. In dieser Umgebung kann die Kommunikation zwischen Datenbankpartitionen zur Verwendung von gemeinsam genutztem Speicher und Semaphors optimiert werden.

Zugehörige Konzepte:

- „Übersicht über die Architektur und Prozesse von DB2“ auf Seite 155
- „Verbesserungen des Verbindungskonzentrators für Clientverbindungen“ auf Seite 171
- „Verringern des Protokollierungsaufwands zur Verbesserung der Abfrageleistung“ auf Seite 151
- „Aktualisierungsverarbeitung“ auf Seite 139

Verbesserungen des Verbindungskonzentrators für Clientverbindungen

Für Internetanwendungen mit vielen relativ kurzfristigen Verbindungen oder ähnliche Arten von Anwendungen verbessert der Verbindungskonzentrator die Leistung, indem er es ermöglicht, wesentlich mehr Clientverbindungen effizient zu verarbeiten. Darüber hinaus verringert er die Speicherauslastung für jede Verbindung und senkt die Anzahl von Kontextumschaltungen.

Anmerkung: Der Verbindungskonzentrator ist aktiviert, wenn der Wert des Parameters *max_connections* größer als der Wert des Parameters *max_cordagents* ist.

In einer Umgebung, die viele gleichzeitige Benutzerverbindungen erfordert, können Sie den Verbindungskonzentrator aktivieren, um die Systemressourcen effizienter zu nutzen. Diese Funktion bietet Vorteile, die früher nur im Verbindungspooling von DB2 Connect zu finden waren. Sowohl das Verbindungspooling als auch der Verbindungskonzentrator werden im DB2 Connect Benutzerhandbuch beschrieben. Nach Herstellung der ersten Verbindung verringert der Verbindungskonzentrator die Zeit für die Verbindung zu einem Host. Wenn die Trennung einer Verbindung zu einem Host angefordert wird, wird die eingehende Verbindung gelöscht, während die abgehende Verbindung zum Host in einem Pool erhalten bleibt. Wenn eine neue Anforderung zur Herstellung einer Verbindung zu dem Host erfolgt, versucht DB2 eine vorhandene abgehende Verbindung aus dem Pool wiederzuverwenden.

Anmerkung: Wenn Anwendungen mit einem Verbindungspooling oder mit dem Verbindungskonzentrator arbeiten, erreichen Sie die beste Leistung, indem Sie die Parameter optimieren, die die Größe des Datenblocks steuern, der im Cache zwischengespeichert wird. Weitere Informationen finden Sie im DB2 Connect Benutzerhandbuch.

Mit DB2 Connect-Verbindungspools und dem Verbindungskonzentrator schließt der aktive Agent seine abgehende Verbindung nicht, nachdem der Client seine Verbindung getrennt hat, sondern er wird in den Agentenpool für die Anwendung versetzt, wo er zu einem *logischen Subagenten* wird, der von einem *logischen Koordinatoragenten* mit einer aktiven Verbindung zu dem fernen Host gesteuert wird.

Beim Einsatz des Verbindungspoolings ist DB2 Connect auf eingehende TCP/IP- und abgehende TCP/IP- und SNA-Verbindungen beschränkt. Bei Verwendung von SNA muss der Sicherheitstyp auf den Wert NONE (Keine) eingestellt sein, damit die Verbindung in den Pool gesetzt werden kann. Bei Verbindungspools werden diese Agenten im Bereitschaftsmodus als *inaktive Agenten* bezeichnet. Die Bezeichnung „Pool für inaktive Agenten“ ist mit „Pool für abgehende Verbindungen“ synonym.

Der Verbindungskonzentrator implementiert eine ähnliche Methode zur Beibehaltung inaktiver Agenten zur späteren Verwendung in einem anwendungsspezifischen Pool.

Verwendungsbeispiele:

1. Betrachten Sie eine ESE-Umgebung mit einer einzelnen Datenbankpartition, in der durchschnittlich 1000 Benutzer mit der Datenbank verbunden sind. Manchmal steigt die Anzahl gleichzeitiger Transaktion bis auf 200 an, wird jedoch nie höher als 250. Transaktionen sind kurz.

Für diese Auslastung definiert der Administrator die folgenden Konfigurationsparameter des Datenbankmanagers:

- *max_connections* wird auf 1000 gesetzt, um eine Unterstützung der durchschnittlichen Anzahl von Verbindungen sicherzustellen.
- *max_coordagents* wird auf 250 gesetzt, um die maximale Anzahl gleichzeitiger Transaktionen zu unterstützen.
- *maxagents* wird mit einem ausreichend hohen Wert definiert, um alle Koordinatoragenten und Subagenten (wenn zutreffend) zu unterstützen, die zur Ausführung von Transaktionen auf dem Knoten erforderlich sind.

Wenn *intra_parallel* den Wert OFF hat, wird *maxagents* auf den Wert 250 gesetzt, weil in einer solchen Umgebung keine Subagenten verfügbar sind. Wenn *intra_parallel* den Wert ON hat, sollte *maxagents* auf einen ausreichend hohen Wert gesetzt werden, um den Koordinatoragenten und die Subagenten zu berücksichtigen, die für jede Transaktion erforderlich sind, die auf Daten auf dem Knoten zugreift. Wenn zum Beispiel jede Transaktion vier Subagenten erfordert, sollte *maxagents* auf den Wert $(4+1) * 250$, also 1250, gesetzt werden. Zur weiteren Optimierung des Parameters *maxagents* erfassen Sie Monitormomentaufnahmen für den Datenbankmanager. Die erreichte obere Grenze für die Anzahl der Agenten gibt die geeignete Einstellung für *maxagents* an.

- *num_poolagents* wird abhängig vom Wert für *maxagents* auf einen Wert von mindestens 250 bzw. einen Wert bis maximal 1250 gesetzt, um sicherzustellen, dass genügend Datenbankagenten verfügbar sind, um ankommende Clientanforderungen ohne Aufwand zur Erstellung neuer Agenten verarbeiten zu können.

Allerdings sollte diese Anzahl gesenkt werden, um die Ressourcennutzung in Zeiten geringer Auslastung zu reduzieren. Wenn dieser Wert zu niedrig eingestellt wird, werden Agenten beendet anstatt in den Agentenpool versetzt zu werden, was eine Erstellung neuer Agenten erforderlich macht, bevor der Server eine durchschnittliche Auslastung verarbeiten kann.

- *num_initagents* wird auf den gleichen Wert wie *num_poolagents* gesetzt, da die Anzahl der Agenten bekannt ist, die aktiv sein sollten. Dies veranlasst den Datenbankmanager, eine angemessene Anzahl von Agenten beim Starten zu erstellen, anstatt sie vor der Verarbeitung einer Anforderung zu erstellen.

Die Fähigkeit der zugrunde liegenden Hardware, eine bestimmte Auslastung zu bewältigen, wird an dieser Stelle nicht behandelt. Wenn die zugrunde liegende Hardware keine Anzahl X von Agenten gleichzeitig verarbeiten kann, müssen Sie diese Anzahl auf den Maximalwert verringern, der von der Hardware unterstützt wird. Wenn zum Beispiel der Maximalwert bei 1500 Agenten liegt, begrenzt dieser Wert die maximale Anzahl gleichzeitiger Transaktionen, die verarbeitet werden können. Sie sollten diese Art leistungsrelevanter Einstellungen überwachen, da es nicht immer möglich ist, die exakte Anzahl von Anforderungen zu bestimmen, die zu einem gegebenen Zeitpunkt an andere Knoten gesendet werden.

2. In einem System, in dem die Auslastung auf ein Maximum von 100 gleichzeitigen Transaktionen und die gleiche Anzahl verbundener Benutzer wie in Beispiel 1 begrenzt werden muss, können Sie die Konfigurationsparameter des Datenbankmanagers wie folgt einstellen:

- *max_coordagents* auf 100
- *num_poolagents* auf 100

Mit diesen Einstellungen liegt die maximale Anzahl von Clients, die gleichzeitig Transaktionen ausführen können, bei 100. Wenn alle Clients ihre Verbindungen trennen, warten 100 Agenten darauf, neue Clientverbindungen zu bedienen. Sie sollten den Parameter *maxagents* unter Berücksichtigung der Art der Auslastung, der Einstellungen für die abfrageinterne Parallelität, der Anzahl der Datenbankpartitionen und der zugrunde liegenden Hardware einstellen.

3. Betrachten Sie als Nächstes eine ESE-Installation mit fünf Datenbankpartitionen, in der jede Datenbankpartition eine durchschnittliche Anzahl von 1000 Benutzerverbindungen hat, und die gleichzeitig ablaufenden Transaktionen eine Anzahl von 200, jedoch nie mehr als 250 Verbindungen erreichen. Die Datenbankkonfigurationsparameter sollten wie folgt definiert werden:

- *max_coordagents* wird auf 250 gesetzt, weil wie in Beispiel 1 höchstens 250 Clients gleichzeitig Transaktionen ausführen.
- *maxagents* wird auf 1500 gesetzt. Unter der Annahme, dass die Daten auf fünf Datenbankpartitionen verteilt sind, könnte jede Transaktion mindestens einen Teilbereich auf jedem Knoten im System ausführen. $((1 \text{ Koordinator-agent} + 5 \text{ Subagenten}) * 250 = 1500.)$
- *num_poolagents* wird auf 1200 gesetzt. (Unter der Annahme einer Durchschnittszahl von 200 gleichzeitigen Transaktionen mit einem Maximum von 250 ergibt sich als Anzahl von Agenten, die durchschnittlich erforderlich sind: $(1+5)*200 = 1200.)$
- *num_initagents* wird wie in Beispiel 1 auf den gleichen Wert wie *num_poolagents* gesetzt.

4. In einem System, für das Sie den Verbindungskonzentrator nicht aktivieren wollen, jedoch 250 gleichzeitig verbundene Benutzer zulassen wollen, stellen Sie die Konfigurationsparameter des Datenbankmanagers wie folgt ein:

- *max_connections* auf 250
- *max_coordagents* auf 250

Zugehörige Konzepte:

- „Datenbankagenten“ auf Seite 153
- „Übersicht über die Architektur und Prozesse von DB2“ auf Seite 155
- „Verwaltung von Datenbankagenten“ auf Seite 163

Kapitel 9. Entwurf für den optimalen Datenzugriff

Der SQL- und XQuery-Compilerprozess

Der SQL- und XQuery-Compiler führt mehrere Schritte aus, um einen Zugriffsplan zu erstellen, der ausgeführt werden kann. Diese Schritte werden in der folgenden Abbildung dargestellt und in den anschließenden Abschnitten beschrieben. Beachten Sie, dass einige Schritte nur für Abfragen in einer föderierten Datenbank durchgeführt werden.

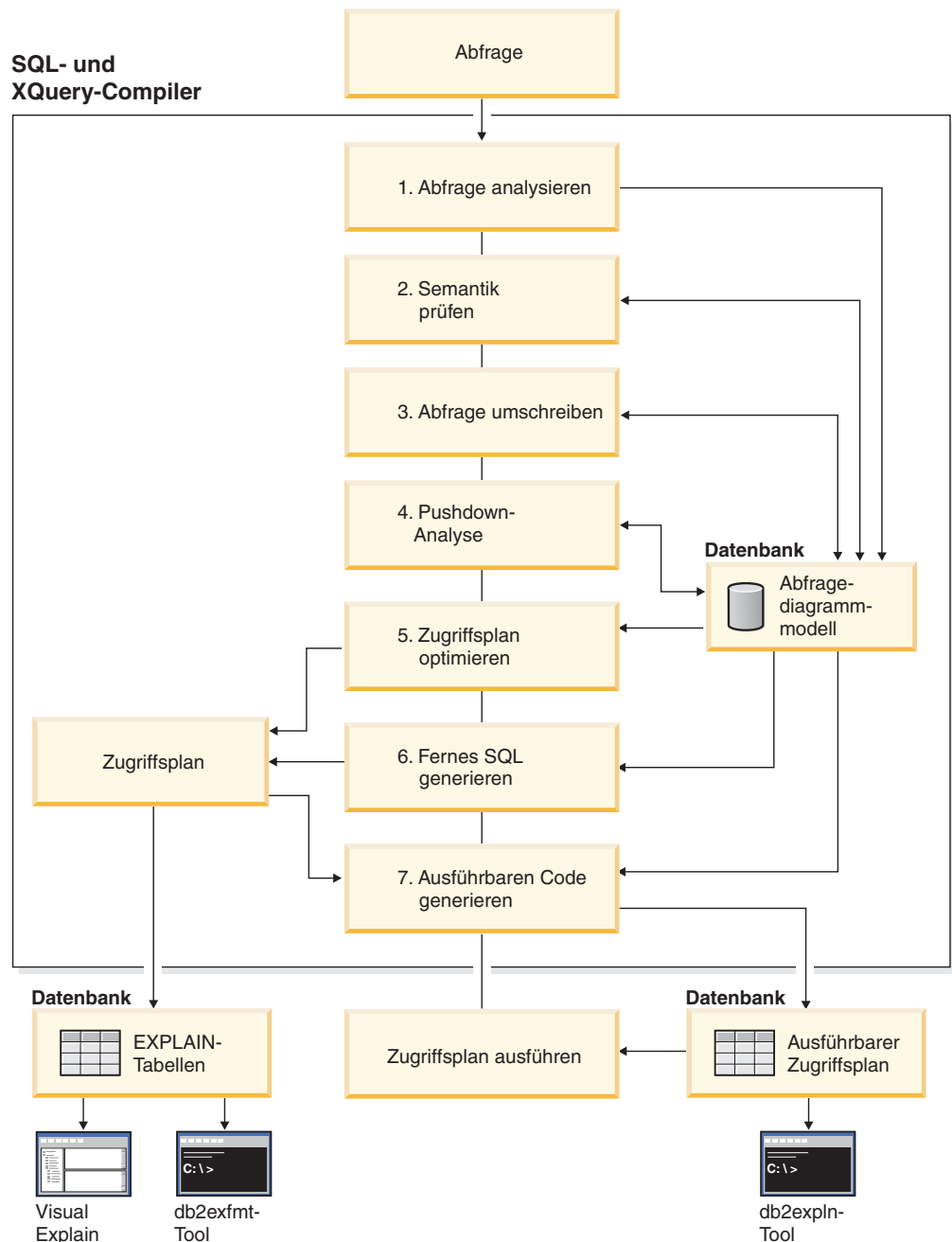


Abbildung 19. Vom SQL- und XQuery-Compiler ausgeführte Schritte

Abfragediagrammmodell

Das *Abfragediagrammmodell* ist eine interne, im Speicher befindliche Datenbank, welche die Abfrage im Verlauf der in den folgenden Schritten beschriebenen Verarbeitung darstellt:

1. Abfrage analysieren

Der SQL- und XQuery-Compiler analysiert die Abfrage, um die Gültigkeit der Syntax zu überprüfen. Wenn Syntaxfehler festgestellt werden, stoppt der Abfragecompiler die Verarbeitung und gibt den entsprechenden Fehler an die

Anwendung zurück, die die Abfrage übergeben hat. Wenn die Analyse abgeschlossen ist, wird eine interne Darstellung der Abfrage erstellt und im Abfragediagrammmodell gespeichert.

2. Semantik prüfen

Der Compiler stellt sicher, dass keine Inkonsistenzen zwischen Teilen der Anweisung bestehen. Ein einfaches Beispiel für eine Semantikprüfung ist die Überprüfung durch den Compiler, ob eine für die Skalarfunktion YEAR angegebene Spalte mit dem Datentyp DATETIME (Datum/Uhrzeit) definiert wurde. Der Compiler fügt außerdem die Bedingungssemantik in das Abfragediagrammmodell ein, zu der die Auswirkungen der referenziellen Integritätsbedingungen, der Prüfungen auf Integritätsbedingungen in Tabellen, der Trigger und der Sichten gehören. Das Abfragediagrammmodell enthält die gesamte Semantik der Abfrage, einschließlich der Abfrageblöcke, Unterabfragen, Korrelationen, abgeleiteten Tabellen, Ausdrücke, Datentypen, Datentypumsetzungen, Codepageumsetzungen und der Verteilungsschlüssel.

3. Abfrage umschreiben

Der Compiler verwendet die im Abfragediagrammmodell gespeicherte globale Semantik, um die Abfrage in eine Form umzusetzen, die leichter optimiert werden kann, und speichert das Ergebnis im Abfragediagrammmodell.

Zum Beispiel kann der Compiler ein Vergleichselement verschieben und somit die Ebene ändern, auf der es angewandt wird, um dadurch potenziell die Abfrageleistung zu erhöhen. Diese Art der Verschiebung einer Operation wird als allgemeine Vergleichselementverschiebung (*General Predicate Pushdown*) bezeichnet. In einer Umgebung mit partitionierten Datenbanken sind die folgenden Abfrageoperationen etwas rechenintensiver:

- Spaltenberechnungen (Aggregation)
- Umverteilen von Zeilen
- Korrelierte Unterabfragen, d. h. Unterabfragen, die einen Verweis auf eine Spalte einer Tabelle enthalten, die sich außerhalb der Unterabfrage befindet

Für einige Abfragen in einer Umgebung mit partitionierten Datenbanken kann eine Dekorrelierung im Rahmen des Umschreibens der Abfrage erfolgen.

4. Pushdown-Analyse (förderierte Datenbanken)

Die Hauptfunktion dieses Schrittes ist, dem Optimierungsprogramm eine Empfehlung zu liefern, ob eine Operation an eine Datenquelle verschoben und fern ausgewertet werden kann, was als *Pushdown* bezeichnet wird. Diese Art von Pushdown-Aktivität ist für Datenquellenabfragen spezifisch und bildet eine Erweiterung zu den allgemeinen Operationen der Vergleichselementverschiebung.

Dieser Schritt wird nur ausgeführt, wenn Abfragen für förderierte Datenbanken ausgeführt werden.

5. Zugriffsplan optimieren

Mit dem Abfragediagrammmodell als Eingabe generiert die Optimierungskomponente des Compilers zahlreiche alternative Ausführungspläne zur Erfüllung der Abfrage. Das Optimierungsprogramm schätzt den Ausführungsaufwand mit Hilfe der Statistiken für Tabellen, Indizes, Spalten und Funktionen für jeden der verschiedenen Pläne ab. Dann wählt es den Plan mit dem geringsten geschätzten Ausführungsaufwand aus. Das Optimierungsprogramm verwendet das Abfragediagrammmodell, um die Abfragesemantik zu analysieren und Informationen zu einer Vielzahl von Faktoren, einschließlich Indizes, Basistabellen, abgeleitete Tabellen, Unterabfragen, Korrelationen und Rekursion, zu erhalten.

Das Optimierungsprogramm kann außerdem eine andere Art von Verschiebeoperation (Pushdown) in Betracht ziehen, nämlich für *Spaltenberechnungen und Sortierungen*. Die Leistung kann erhöht werden, wenn die Auswertung dieser Operationen an die Komponente der Datenverwaltungsservices (Data Management Services) verschoben werden kann.

Das Optimierungsprogramm berücksichtigt auch, ob es Pufferpools verschiedener Größen gibt, wenn es die Auswahl der Seitengröße festlegt. Der Faktor, dass die Umgebung eine partitionierte Datenbank enthält, wird ebenso berücksichtigt wie die Möglichkeit, den ausgewählten Plan für eine potenzielle abfrageinterne Parallelität in einer symmetrischen Multiprozessorumgebung (SMP-Umgebung) auszulegen. Diese Informationen werden vom Optimierungsprogramm zur Auswahl des am besten geeigneten Zugriffsplans für die Abfrage verwendet.

Die Ausgabe dieses Schritts des Compilers ist ein Zugriffsplan. Dieser Zugriffsplan stellt die Informationen bereit, die in den EXPLAIN-Tabellen erfasst werden. Die Informationen, die zur Generierung des Zugriffsplans dienen, können mit einer Momentaufnahme der EXPLAIN-Einrichtung erfasst werden.

6. Generierung von fernem SQL (förderierte Datenbanken)

Der endgültige Plan, der vom Optimierungsprogramm gewählt wird, kann aus einer Reihe von Schritten bestehen, die an einer fernen Datenquelle ausgeführt werden. Für Operationen, die an der jeweiligen Datenquelle ausgeführt werden, erstellt der Schritt der Generierung von fernem SQL eine effiziente SQL-Anweisung, die auf der SQL-Version der Datenquelle beruht.

7. „Ausführbaren“ Code generieren

Im letzten Schritt verwendet der Compiler den Zugriffsplan und das Abfragediagrammmodell, um einen ausführbaren Zugriffsplan oder Abschnitt für die Abfrage zu erstellen. Bei dieser Generierung des Codes werden Informationen des Abfragediagrammmodells verwendet, um eine Wiederholung der Ausführung von Ausdrücken zu vermeiden, die für eine Abfrage nur einmal berechnet werden müssen. Diese Art der Optimierung ist beispielsweise für Umwandlungen von Codepages und die Verwendung von Hostvariablen möglich.

Damit eine Abfrageoptimierung bzw. -reoptimierung für statische und dynamische SQL- und XQuery-Anweisungen, die Hostvariablen, Sonderregister oder Parametermarken enthalten, ausgeführt werden kann, binden Sie das Paket mit der Bindeoption REOPT. Wenn diese Bindeoption verwendet wird, wird der Zugriffspfad für eine SQL- oder XQuery-Anweisung, die zu diesem Paket gehört und Hostvariablen, Parametermarken oder Sonderregister enthält, mit den Werten dieser Variablen und nicht mit Standardschätzwerten, die der Compiler auswählt, optimiert. Diese Optimierung erfolgt bei der Ausführung der Abfrage, wenn die Werte verfügbar sind.

Informationen über die Zugriffspläne für statische SQL- und XQuery-Anweisungen werden in den Systemkatalogtabellen gespeichert. Wenn das Paket ausgeführt wird, verwendet der Datenbankmanager die in den Systemkatalogtabellen gespeicherten Informationen, um festzulegen, wie auf die Daten zugegriffen werden soll, und Ergebnisse für eine Abfrage bereitzustellen. Diese Informationen werden vom Tool *db2expln* verwendet.

Anmerkung: Führen Sie RUNSTATS in angemessenen Intervallen für Tabellen aus, die häufig geändert werden. Das Optimierungsprogramm benötigt aktuelle Statistikinformationen zu den Tabellen und ihren Daten, um die effizientesten Zugriffspläne zu generieren. Führen Sie einen Rebind für Ihre Anwendung durch, um die aktualisierten Statistiken zu nutzen. Wenn das Dienstprogramm RUNSTATS nicht ausgeführt wird oder das Optimierungsprogramm annimmt, dass RUNSTATS

für leere oder fast leere Tabellen ausgeführt wurde, kann das Optimierungsprogramm entweder Standardwerte verwenden oder versuchen, bestimmte Statistikdaten mit Hilfe der Anzahl der Dateiseiten (FPAGES) abzuleiten, die zum Speichern der Tabelle auf dem Plattenspeicher verwendet werden. Die Gesamtanzahl belegter Blöcke wird in der Spalte ACTIVE_BLOCKS gespeichert.

Zugehörige Konzepte:

- „Advantages of deferred binding“ in *Developing Embedded SQL Applications*
- „MQTs (MQT - Materialized Query Table, gespeicherte Abfragetabelle)“ auf Seite 11
- „Datenzugriffsmethoden“ auf Seite 192
- „Auswirkungen des Sortierens und Gruppierens“ auf Seite 202
- „Richtlinien zur Analyse, wo eine Abfrage für föderierte Datenbanken ausgewertet wird“ auf Seite 249
- „Joins“ auf Seite 220
- „Optimierungsstrategien für partitionsinterne Parallelität“ auf Seite 208
- „Optimierungsstrategien für MDC-Tabellen“ auf Seite 19
- „Vergleichselementtypologie und Zugriffspläne“ auf Seite 200
- „Methoden zum Umschreiben von Abfragen und Beispiele“ auf Seite 236

Zugehörige Referenzen:

- „Einschränkungen für den nativen XML-Datenspeicher“ in *XML-Handbuch*

Auswählen von Optimierungsklassen

Die Einstellung der Optimierungsklasse kann, insbesondere zu folgenden Zwecken, einige der Vorteile bringen, die durch die explizite Angabe von Optimierungstechniken erzielt werden:

- Verwalten sehr kleiner Datenbanken oder sehr einfacher dynamischer Abfragen
- Kompilieren mit beschränkten Speicherkapazitäten auf dem Datenbankserver
- Verkürzen der Abfragekompilierzeit, zum Beispiel für PREPARE

Die meisten Anweisungen lassen sich in angemessener Weise bei einem sinnvollen Einsatz von Ressourcen unter Verwendung der Standardoptimierungsklasse 5 optimieren. Bei einer bestimmten Optimierungsklasse werden die Kompilierzeit und der Bedarf an Ressourcen in erster Linie durch die Komplexität der Abfrage, insbesondere durch die Anzahl der Joins und Unterabfragen, beeinflusst. Allerdings werden die Kompilierzeit und der Ressourcenbedarf auch vom Grad der durchgeführten Optimierung beeinflusst.

Die Abfrageoptimierungsklassen 1, 2, 3, 5 und 7 sind alle für allgemeine Zwecke geeignet. Ziehen Sie die Klasse 0 nur in Betracht, wenn Sie die Kompilierzeit von Abfragen weiter verringern müssen und wissen, dass die SQL- und XQuery-Anweisungen extrem einfach sind.

Tipp: Zur Analyse von Abfragen mit langer Laufzeit können Sie die Abfragen mit db2batch ausführen, um herauszufinden, wie viel Zeit auf die Kompilierung und wie viel Zeit auf die Ausführung verwendet wird. Wenn die Kompilierung mehr Zeit erfordert, verringern Sie die Optimierungsklasse. Wenn die Ausführung mehr Zeit erfordert, ziehen Sie eine höhere Optimierungsklasse in Betracht.

Beachten Sie bei der Auswahl einer Optimierungsklasse die folgenden allgemeinen Richtlinien:

- Beginnen Sie mit der Standardoptimierungsklasse für Abfragen: Klasse 5.
- Zur Verwendung einer anderen als die Standardklasse, versuchen Sie zunächst die Klasse 1, 2 oder 3. Die Klassen 0, 1 und 2 arbeiten mit dem Algorithmus der schnellen Joinaufzählung (Greedy Join Enumeration).
- Verwenden Sie die Optimierungsklasse 1 oder 2, wenn Sie viele Tabellen mit vielen Joinvergleichselementen für dieselbe Spalte haben und die Dauer der Kompilierung von Bedeutung ist.
- Verwenden Sie eine niedrige Optimierungsklasse (0 oder 1) für Abfragen, die sehr kurze Laufzeiten von unter einer Sekunde haben. Solche Abfragen sind häufig durch folgende Merkmale gekennzeichnet:
 - Zugriff auf eine oder nur einige wenige Tabellen
 - Abruf nur einer oder einiger weniger Zeilen
 - Verwendung vollständig qualifizierter eindeutiger Indizes

OLTP-Transaktionen (Online-Transaktionsverarbeitung) sind gute Beispiele für diese Art von Abfrage.

- Verwenden Sie eine höhere Optimierungsklasse (3, 5 oder 7) für Abfragen mit längerer Laufzeit, die mehr als 30 Sekunden benötigen.
- Klasse 3 und höhere Klassen arbeiten mit dem Algorithmus zur dynamisch programmierten Joinaufzählung (Dynamic Programming Join Enumeration). Dieser Algorithmus berücksichtigt wesentlich mehr alternative Pläne und kann, insbesondere bei steigender Tabellenzahl, deutlich mehr Kompilierungszeit als die Klassen 0, 1 und 2 erfordern.
- Verwenden Sie die Optimierungsklasse 9 nur, wenn Sie spezielle, über das normale Maß hinausgehende Optimierungsanforderungen für eine Abfrage haben.

Für komplexe Abfragen können andere Grade an Optimierung erforderlich sein, um den besten Zugriffsplan auszuwählen. Ziehen Sie höhere Optimierungsklassen für Abfragen mit den folgenden Merkmalen in Betracht:

- Zugriff auf große Tabellen
- Eine große Anzahl von Vergleichselementen
- Zahlreiche Unterabfragen
- Zahlreiche Joins
- Zahlreiche Mengenoperatoren wie UNION und INTERSECT
- Zahlreiche die Vergleichselemente erfüllende Zeilen
- Operationen GROUP BY und HAVING
- Verschachtelte Tabellenausdrücke
- Eine große Anzahl von Sichten

Abfragen zur Entscheidungshilfe oder Abfragen für Monatsberichte aus vollständig normalisierten Datenbanken sind gute Beispiele für komplexe Abfragen, für die zumindest die Standardoptimierungsklasse verwendet werden sollte.

Verwenden Sie höhere Optimierungsklassen für SQL- und XQuery-Anweisungen, die von einem Abfragegenerator erstellt wurden. Viele Abfragegeneratoren erstellen ineffiziente Abfragen. Ineffizient geschriebene Abfragen, einschließlich der von einem Abfragegenerator erstellten, können eine zusätzliche Optimierung erforderlich machen, um einen guten Zugriffsplan auszuwählen. Durch Verwendung der Abfrageoptimierungsklasse 2 oder einer höheren Abfrageoptimierungsklasse können solche SQL- und XQuery-Abfragen verbessert werden.

Anmerkung: In einer Abfrage für föderierte Datenbanken gilt die Optimierungsklasse nicht für das ferne Optimierungsprogramm.

Zugehörige Konzepte:

- „Durchführen von Vergleichstests“ auf Seite 409
- „Konfigurationsparameter mit Einfluss auf die Abfrageoptimierung“ auf Seite 186
- „Optimierungsstrategien für partitionsinterne Parallelität“ auf Seite 208
- „Optimierungsstrategien für MDC-Tabellen“ auf Seite 19

Zugehörige Tasks:

- „Einstellen der Optimierungsklasse“ auf Seite 184

Zugehörige Referenzen:

- „Optimierungsklassen“ auf Seite 181

Optimierungsklassen

Wenn Sie eine SQL- oder XQuery-Abfrage kompilieren, können Sie eine Optimierungsklasse angeben, die bestimmt, wie das Optimierungsprogramm den effizientesten Zugriffsplan für diese Abfrage auswählt. Die Optimierungsklassen werden nach Anzahl und Typ der Optimierungsstrategien unterschieden, die in der Abfragekompilierung berücksichtigt werden. Obwohl Sie einerseits die Optimierungstechniken einzeln angeben können, um die Laufleistung für die Abfrage zu verbessern, werden andererseits um so mehr Zeit und Systemressourcen für die Abfragekompilierung benötigt, je mehr Optimierungstechniken Sie angeben.

Bei der Kompilierung einer SQL- oder XQuery-Abfrage können Sie eine der folgenden Optimierungsklassen angeben:

- 0 - Diese Klasse weist das Optimierungsprogramm an, nur eine Minimaloptimierung zur Generierung eines Zugriffsplans durchzuführen. Diese Optimierungsklasse ist durch folgende Merkmale gekennzeichnet:
- Statistiken über ungleichmäßige Verteilungen von Datenwerten werden vom Optimierungsprogramm nicht berücksichtigt.
 - Nur Grundregeln für das Umschreiben der Abfragen werden angewandt.
 - Schnelle Joinaufzählung findet statt.
 - Nur die Zugriffsmethoden durch Joins mit Verschachtelungsschleife und Indexsuchen sind möglich.
 - Ein Vorablesezugriff wird in den generierten Zugriffsmethoden nicht verwendet.
 - Die Strategie des Sternjoins wird nicht berücksichtigt.

Diese Klasse sollte nur unter Umständen verwendet werden, unter denen der Systemaufwand zur Kompilierung der Abfrage so gering wie möglich gehalten werden muss. Die Abfrageoptimierungsklasse 0 eignet sich für eine Anwendung, die insgesamt aus sehr einfachen dynamischen SQL- oder XQuery-Anweisungen besteht, die auf korrekt indexierte Tabellen zugreifen.

- 1 - Diese Optimierungsklasse ist durch folgende Merkmale gekennzeichnet:
- Statistiken über ungleichmäßige Verteilungen von Datenwerten werden vom Optimierungsprogramm nicht berücksichtigt.

- Nur eine Teilmenge der Regeln für das Umschreiben von Abfragen wird angewandt.
- Schnelle Joinaufzählung findet statt.
- Ein Vorabsezugriff wird in den generierten Zugriffsmethoden nicht verwendet.

Die Optimierungsklasse 1 ist der Klasse 0 ähnlich, abgesehen davon, dass Mischjoins und Tabellensuchen ebenfalls verfügbar sind.

- 2 - Diese Klasse weist das Optimierungsprogramm an, einen Optimierungsgrad zu verwenden, der den der Klasse 1 deutlich übertrifft, und gleichzeitig den Kompilierungsaufwand für komplexe Abfragen wesentlich geringer als bei den Klassen ab 3 aufwärts zu halten. Diese Optimierungsklasse ist durch folgende Merkmale gekennzeichnet:

- Alle verfügbaren Statistiken, einschließlich der Statistiken zur Häufigkeit und zu Quantilen ungleichmäßiger Verteilungen, werden verwendet.
- Alle Regeln für das Umschreiben von Abfragen, einschließlich der Weiterleitung von Abfragen an MQTs (MQT = Materialized Query Table, gespeicherte Abfragetabelle), werden berücksichtigt, außer den Regeln, die sehr rechenintensiv sind und nur in seltenen Fällen zur Anwendung kommen.
- Schnelle Joinaufzählung wird verwendet.
- Viele verschiedene Zugriffsmethoden werden berücksichtigt, einschließlich des Vorabsezugriffs über Listen und der Weiterleitung an MQTs.
- Die Strategie des Sternjoins wird gegebenenfalls berücksichtigt.

Die Optimierungsklasse 2 ist der Klasse 5 ähnlich, sie verwendet jedoch schnelle Joinaufzählung und nicht dynamische programmierte Joinaufzählung. Diese Klasse hat den höchsten Optimierungsgrad aller Klassen, die mit dem Algorithmus für schnelle Joinaufzählung arbeiten, der für komplexe Abfragen weniger Alternativen berücksichtigt und dadurch einen geringeren Kompilierungsaufwand erfordert als die Klassen ab 3 aufwärts. Klasse 2 empfiehlt sich für sehr komplexe Abfragen in einer Umgebung zur Entscheidungshilfe oder mit analytischer Onlineverarbeitung (OLAP). In solchen Umgebungen werden spezifische Abfragen nur selten exakt wiederholt, so dass ein Zugriffsplan mit hoher Wahrscheinlichkeit nicht bis zur nächsten Ausführung der Abfrage im Cache erhalten bleibt.

- 3 - Diese Klasse fordert einen moderaten Optimierungsgrad an. Diese Klasse kommt den Merkmalen der Abfrageoptimierung von DB2 für MVS/ESA, OS/390 oder z/OS am nächsten. Diese Optimierungsklasse ist durch folgende Merkmale gekennzeichnet:

- Statistiken über ungleichmäßige Verteilungen von Datenwerten, die die Häufigkeit von Werten erfassen, werden verwendet, wenn sie verfügbar sind.
- Die meisten Regeln zum Umschreiben von Abfragen, einschließlich der Umsetzungen von Unterabfragen in Joins, werden angewandt.
- Dynamisch programmierte Joinaufzählung (Dynamic Programming Join Enumeration) wie folgt:
 - Eingeschränkte Verwendung zusammengesetzter innerer Tabellen
 - Eingeschränkte Verwendung kartesischer Produkte für Sternschemata, für die Suchtabellen erforderlich sind
- Viele verschiedene Zugriffsmethoden werden berücksichtigt, einschließlich des Vorabsezugriffs über Listen, logischer Joins von Indizes über AND (Index ANDing) und Sternjoins.

Diese Klasse eignet sich für eine große Bandbreite von Anwendungen. Diese Klasse verbessert Zugriffspläne für Abfragen mit vier und mehr Joins. Es ist jedoch möglich, dass der vom Optimierungsprogramm gewählte Plan nicht so gut ist wie einer, der mit der Standard-optimierungsklasse gewählt würde.

- 5 - Diese Klasse weist das Optimierungsprogramm an, einen bedeutenden Grad an Optimierung bei der Generierung eines Zugriffsplans durchzuführen. Diese Optimierungsklasse ist durch folgende Merkmale gekennzeichnet:
- Alle verfügbaren Statistiken, einschließlich der Verteilungsstatistiken zur Häufigkeit und zu Quantilen, werden verwendet.
 - Alle Regeln für das Umschreiben von Abfragen, einschließlich der Weiterleitung von Abfragen an MQTs, werden berücksichtigt, außer den Regeln, die sehr rechenintensiv sind und nur in seltenen Fällen zur Anwendung kommen.
 - Dynamisch programmierte Joinaufzählung (Dynamic Programming Join Enumeration) wie folgt:
 - Eingeschränkte Verwendung zusammengesetzter innerer Tabellen
 - Eingeschränkte Verwendung kartesischer Produkte für Sternschemata, für die Suchtabellen erforderlich sind
 - Viele verschiedene Zugriffsmethoden werden berücksichtigt, einschließlich des Vorabsezugriffs über Listen, logischer Joins von Indizes über AND und der Weiterleitung an MQTs.

Stellt das Optimierungsprogramm fest, dass die zusätzlichen Ressourcen und die Verarbeitungszeit für komplexe dynamische SQL- oder XQuery-Abfragen nicht gewährleistet sind, wird die Optimierung reduziert. Das Ausmaß oder der Umfang der Reduzierung hängt von der Maschinengröße und der Anzahl der Vergleichselemente ab.

Reduziert das Abfrageoptimierungsprogramm den Grad an Abfrageoptimierung, wendet es weiterhin alle Regeln für das Umschreiben von Abfragen an, die normalerweise angewandt würden. Es verwendet jedoch die schnelle Joinaufzählung und reduziert die Anzahl der Zugriffsplan-kombinationen, die in Erwägung gezogen werden.

Die Abfrageoptimierungsklasse 5 ist hervorragend für eine gemischte Umgebung geeignet, in der sowohl Transaktionen als auch komplexe Abfragen ausgeführt werden. Diese Optimierungsklasse wurde zur Verwendung der wertvollsten Abfragetransformationen und anderer Optimierungstechniken für Abfragen in einer effizienten Weise entwickelt.

- 7 - Diese Klasse weist das Optimierungsprogramm an, einen bedeutenden Grad an Optimierung bei der Generierung eines Zugriffsplans durchzuführen. Sie entspricht der Abfrageoptimierungsklasse 5, jedoch reduziert sie nicht den Umfang der Abfrageoptimierung für komplexe dynamische SQL- oder XQuery-Abfragen.
- 9 - Diese Klasse weist das Optimierungsprogramm an, alle verfügbaren Optimierungstechniken anzuwenden. Dazu gehören:
- Alle verfügbaren Statistiken
 - Alle Regeln für das Umschreiben von Abfragen
 - Alle Möglichkeiten für Joinaufzählungen, einschließlich kartesischer Produkte und uneingeschränkter zusammengesetzter innerer Tabellen
 - Alle Zugriffsmethoden

Diese Klasse kann die Anzahl der möglichen Zugriffspläne, die vom Optimierungsprogramm ausgewertet werden, erheblich vergrößern. Diese Klasse kann verwendet werden, um festzustellen, ob eine umfassendere Optimierung zur Generierung eines besseren Zugriffsplans für sehr komplexe und zeitintensive Abfragen auf große Tabellen führen würde. Überprüfen Sie anhand von EXPLAIN-Daten und Leistungswerten, ob ein besserer Plan gefunden wurde.

Zugehörige Konzepte:

- „Auswählen von Optimierungsklassen“ auf Seite 179
- „Optimierungsstrategien für partitionsinterne Parallelität“ auf Seite 208
- „Optimierungsstrategien für MDC-Tabellen“ auf Seite 19
- „Generierung von fernem SQL und globale Optimierung in föderierten Datenbanken“ auf Seite 253

Zugehörige Tasks:

- „Einstellen der Optimierungsklasse“ auf Seite 184

Einstellen der Optimierungsklasse

Berücksichtigen Sie bei der Angabe einer Optimierungsstufe, ob eine Abfrage statische oder dynamische SQL- und XQuery-Anweisungen verwendet und ob die gleiche dynamische Abfrage wiederholt ausgeführt wird. Für statische SQL- und XQuery-Anweisungen werden die Kompilierzeit für Abfragen und die Ressourcen nur einmal aufgewendet, und der ausgegebene Plan kann mehrfach verwendet werden. Im Allgemeinen gilt, dass für SQL- und XQuery-Anweisungen stets die Standardoptimierungsklasse verwendet werden sollte. Da dynamische Anweisungen bei der Ausführung gebunden und ausgeführt werden, müssen Sie überlegen, ob der Aufwand für eine zusätzliche Optimierung der dynamischen Anweisungen die allgemeine Leistung verbessert. Wenn allerdings dieselbe dynamische SQL- oder XQuery-Anweisung wiederholt ausgeführt wird, wird der ausgewählte Zugriffsplan im Cache zwischengespeichert. Solche Anweisungen können dieselben Optimierungsstufen wie statische SQL- und XQuery-Anweisungen verwenden.

Wenn Sie annehmen, dass für eine Abfrage eine weitere Optimierung von Vorteil wäre, Sie aber nicht sicher sind oder Bedenken hinsichtlich der Kompilierzeit oder des Ressourcenbedarfs haben, können Sie einige Vergleichstests (Benchmarktests) durchführen.

Vorgehensweise:

Führen Sie folgende Schritte aus, um eine Abfrageoptimierungsklasse anzugeben:

1. Analysieren Sie die Leistungsfaktoren entweder informell oder wie folgt mit formalen Tests:
 - Für **dynamische** Abfrageanweisungen sollte bei Tests die durchschnittliche Laufzeit für die Anweisung verglichen werden. Schätzen Sie eine durchschnittliche Laufzeit mit Hilfe folgender Formel ab:

$$\frac{\text{Kompilierzeit} + \text{Summe der Ausführungszeiten aller Iterationen}}{\text{Anzahl der Iterationen}}$$

In dieser Formel entspricht die Anzahl der Iterationen der Häufigkeit, mit der die Abfrageanweisung Ihrer Schätzung nach jedes Mal, wenn sie kompiliert wird, ausgeführt werden könnte.

Anmerkung: Nach der Erstkompilierung werden dynamische SQL- und XQuery-Anweisungen erneut kompiliert, wenn eine Änderung an der Umgebung dies erfordert. Wenn sich die Umgebung nicht ändert, nachdem eine Abfrageanweisung im Cache zwischengespeichert wurde, muss sie nicht neu kompiliert zu werden, da nachfolgende PREPARE-Anweisungen die zwischengespeicherte Anweisung wiederverwenden.

- Vergleichen Sie für **statische** SQL- und XQuery-Anweisungen die Laufzeiten der Anweisungen.

Obwohl es vielleicht auch interessant wäre, die Kompilierzeit statischer SQL- und XQuery-Anweisungen zu kennen, ist die Gesamtzeit aus Kompilier- und Laufzeit der Anweisung in einem sinnvollen Kontext nur wenig aussagekräftig. Beim Vergleich der Gesamtzeit wird die Tatsache außer Acht gelassen, dass eine statische Abfrageanweisung nach jedem Binden viele Male ausgeführt werden kann und dass sie in der Regel nicht während der Laufzeit gebunden wird.

2. Geben Sie die Optimierungsklasse wie folgt an:

- **Dynamische** SQL- und XQuery-Anweisungen verwenden die Optimierungsklasse, die im Sonderregister CURRENT QUERY OPTIMIZATION angegeben ist, das mit der SQL-Anweisung SET definiert wird. Beispielsweise setzt die folgende Anweisung die Optimierungsklasse auf 1:

```
SET CURRENT QUERY OPTIMIZATION = 1
```

Um sicherzustellen, dass eine dynamische SQL- oder XQuery-Anweisung immer dieselbe Optimierungsklasse verwendet, kann eine Anweisung SET in das Anwendungsprogramm integriert werden.

Wenn das Register CURRENT QUERY OPTIMIZATION nicht definiert ist, werden die dynamischen Anweisungen mit der Standardoptimierungsklasse für Abfragen gebunden. Der Standardwert für dynamische und statische Abfragen wird durch den Wert des Datenbankkonfigurationsparameters *dft_queryopt* festgelegt. Der Standardwert für diesen Parameter ist Klasse 5. Die Standardwerte für die Bindeoption und das Sonderregister werden ebenfalls aus dem Datenbankkonfigurationsparameter *dft_queryopt* gelesen.

- **Statische** SQL- und XQuery-Anweisungen verwenden die in den Befehlen PREP und BIND angegebene Optimierungsklasse. In der Spalte QUERYOPT in der Katalogtabelle SYSCAT.PACKAGES wird die zum Binden des Pakets verwendete Optimierungsklasse aufgezeichnet. Wenn das Paket erneut implizit oder mit dem Befehl REBIND PACKAGE gebunden wird, wird dieselbe Optimierungsklasse für die statischen Abfrageanweisungen verwendet. Verwenden Sie zum Ändern der Optimierungsklasse für solche statischen SQL- und XQuery-Anweisungen den Befehl BIND. Wenn Sie keine Optimierungsklasse angeben, verwendet DB2 die vom Datenbankkonfigurationsparameter *dft_queryopt* angegebene Standardoptimierungsklasse.

Zugehörige Konzepte:

- „Auswählen von Optimierungsklassen“ auf Seite 179

Zugehörige Referenzen:

- „Optimierungsklassen“ auf Seite 181

Konfigurationsparameter mit Einfluss auf die Abfrageoptimierung

Verschiedene Konfigurationsparameter wirken sich auf die Auswahl des Zugriffsplans durch den SQL- oder XQuery-Compiler aus. Viele von ihnen gelten für eine Umgebung mit Einzelpartitionsdatenbanken, während einige nur für eine Umgebung mit partitionierten Datenbanken gelten. In einer Umgebung mit partitionierten Datenbanken sollten die Werte, die für die einzelnen Parameter verwendet werden, für alle Datenbankpartitionen gleich sein.

Anmerkung: Wenn Sie einen Konfigurationsparameter dynamisch ändern, liest das Optimierungsprogramm die geänderten Parameterwerte möglicherweise nicht sofort, da vielleicht noch ältere Zugriffspläne im Paketcache vorhanden sind. Führen Sie zum Zurücksetzen des Paketcache den Befehl `FLUSH PACKAGE CACHE` aus.

Wenn in einem föderierten System die Mehrzahl der Abfragen auf Kurznamen zugreift, sollten Sie die Art der Abfragen, die Sie senden, auswerten, bevor Sie Ihre Umgebung ändern. Zum Beispiel speichert der Pufferpool in einer föderierten Datenbank keine Seiten aus Datenquellen im Cache zwischen. Datenquellen sind die Datenbankverwaltungssysteme (DBMSs) und Daten in einem föderierten System. Aus diesem Grund kann eine Erhöhung der Puffergröße nicht garantieren, dass das Optimierungsprogramm weitere Alternativen für Zugriffspläne in Betracht zieht, wenn es einen Zugriffsplan für Abfragen wählt, die Kurznamen enthalten. Allerdings stellt das Optimierungsprogramm möglicherweise fest, dass eine lokale Speicherung von Datenquellentabellen die Methode mit dem geringsten Aufwand oder ein erforderlicher Schritt für eine Sortieroperation ist. In diesem Fall kann eine Vergrößerung der für IBM DB2 Version 9.1 verfügbaren Ressourcen die Leistung verbessern.

Die folgenden Konfigurationsparameter bzw. Faktoren wirken sich auf die Auswahl des Zugriffsplans durch den SQL- oder XQuery-Compiler aus:

- Die Größe der Pufferpools, die Sie angegeben haben, als Sie sie erstellt oder geändert haben

Bei der Auswahl des Zugriffsplans bezieht das Optimierungsprogramm den Ein-/Ausgabeaufwand für das Laden von Seiten von der Platte in den Pufferpool in die Kalkulation mit ein und schätzt die Anzahl der erforderlichen E/A-Operationen zur Erfüllung der Abfrage ab. Die Schätzung schließt eine Voraussetzung über die Nutzung des Pufferpools mit ein, da zum Lesen von Zeilen einer Seite, die sich bereits im Pufferpool befindet, keine weiteren physischen E/A-Operationen anfallen.

Das Optimierungsprogramm berücksichtigt den Wert der Spalte *npages* in den Systemkatalogtabellen `BUFFERPOOLS` und, in Umgebungen mit partitionierten Datenbanken, in den Systemkatalogtabellen `BUFFERPOOLDBPARTITION`.

Der Ein-/Ausgabeaufwand für das Lesen der Tabellen kann sich auf folgende Bereiche auswirken:

- Wie zwei Tabellen verknüpft werden.
- Ob ein Index ohne Clusterbildung zum Lesen der Daten verwendet wird.

- Grad der Parallelität (`dft_degree`)

Der Konfigurationsparameter `dft_degree` gibt die Parallelität durch Bereitstellen eines Standardwerts für das Sonderregister `CURRENT DEGREE` und Bindeoption `DEGREE` an. Der Wert 1 bedeutet keine partitionsinterne Parallelität. Der Wert -1 bedeutet, dass das Optimierungsprogramm den Grad der partitionsinternen Parallelität anhand der Anzahl von Prozessoren und der Art der Abfrage bestimmt.

- Standardabfrageoptimierungsklasse (dft_queryopt)

Obwohl Sie beim Kompilieren von SQL- oder XQuery-Abfragen eine Abfrageoptimierungsklasse angeben können, ist es vielleicht sinnvoll, einen Standardoptimierungsgrad zu definieren.

Anmerkung: Eine partitionsinterne Parallelverarbeitung findet nur statt, wenn Sie sie durch das Definieren des Datenbankkonfigurationsparameters *intra_parallel* aktivieren.

- Durchschnittliche Anzahl aktiver Anwendungen (avg_appls)

Mit Hilfe des Parameters *avg_appls* versucht das Optimierungsprogramm zu ermitteln, wie viel vom Pufferpool während der Ausführung für den ausgewählten Zugriffsplan verfügbar ist. Höhere Werte für diesen Parameter können das Optimierungsprogramm dahin gehend beeinflussen, dass es Zugriffspläne auswählt, der mit dem Pufferpool etwas sparsamer umgehen. Wenn Sie den Wert 1 angeben, geht das Optimierungsprogramm davon aus, dass der gesamte Pufferpool für die Anwendung verfügbar ist.

- Zwischenspeicher für Sortierlisten (sortheap)

Wenn die zu sortierenden Zeilen mehr als den im Zwischenspeicher für Sortierlisten verfügbaren Speicherbereich in Anspruch nehmen, werden mehrere Sortierarbeitsgänge durchgeführt, wobei in jedem Arbeitsgang eine Untermenge der Gesamtmenge von Zeilen sortiert wird. Jeder Arbeitsgang des Sortiervorgangs wird in einer temporären Tabelle im Pufferpool gespeichert, die eventuell auf den Datenträger geschrieben wird. Wenn alle Arbeitsgänge des Sortiervorgangs abgeschlossen sind, werden die sortierten Untermengen zu einer einzigen sortierten Menge von Zeilen zusammengefügt. Eine Sortierung wird als „über eine Pipe geleitet (piped)“ betrachtet, wenn sie keine temporäre Tabelle zur Speicherung der endgültigen, sortierten Liste von Daten erforderlich macht. Das heißt, dass die Ergebnisse der Sortierung in einem einzigen sequenziellen Zugriff gelesen werden können. Über eine Pipe geleitete Sortierungen führen zu einer besseren Leistung als nicht über eine Pipe geleitete und werden daher, wenn möglich, verwendet.

Bei der Auswahl eines Zugriffsplans schätzt das Optimierungsprogramm den Aufwand der Sortieroperationen, einschließlich der Möglichkeiten, eine Sortierung über eine Pipe zu leiten, folgendermaßen ab:

- Abschätzen der Menge der zu sortierenden Daten
- Bestimmen mit Hilfe des Parameters *sortheap*, ob genügend Speicherbereich für die Sortierung mit Pipe zur Verfügung steht.

- Maximaler Speicher für Sperrenliste (locklist) und maximale Anzahl Sperren pro Anwendung (maxlocks)

Wenn die Isolationsstufe **RR** (Wiederholtes Lesen) verwendet wird, berücksichtigt das Optimierungsprogramm die Werte der Parameter *locklist* und *maxlocks*, um zu bestimmen, ob Sperren auf Zeilenebene möglicherweise durch Sperreneskulation in eine Sperre auf Tabellenebene umgewandelt werden. Wenn das Optimierungsprogramm eine Sperreneskulation für einen Tabellenzugriff vorausieht, wählt es für den Zugriffsplan eine Sperre auf Tabellenebene und vermeidet den Systemaufwand, der mit einer Sperreneskulation während der Ausführung der Abfrage verbunden wäre.

- CPU-Geschwindigkeit (cpuspeed)

Der Parameter für die CPU-Geschwindigkeit wird vom Optimierungsprogramm zur Abschätzung des Aufwands für bestimmte Operationen verwendet. Die Schätzwerte zum CPU-Aufwand und zu verschiedenen E/A-Aufwänden helfen bei der Auswahl des besten Zugriffsplans für eine Abfrage.

Die CPU-Geschwindigkeit eines Systems kann die Auswahl des Zugriffsplans wesentlich beeinflussen. Dieser Konfigurationsparameter wird bei der Installation oder Migration der Datenbank automatisch auf einen geeigneten Wert gesetzt. Sie sollten diesen Parameter nicht anpassen, es sei denn, Sie wollen eine Produktionsumgebung auf einem Testsystem modellieren oder die Auswirkungen einer Änderung der Hardware testen. Mit Hilfe dieses Parameters können Sie eine andere Hardwareumgebung modellieren, um die Zugriffspläne zu ermitteln, die für die andere Umgebung ausgewählt würden. Wenn DB2 den Wert dieses automatischen Konfigurationsparameters neu berechnen soll, setzen Sie den Parameter auf den Wert -1.

- Anweisungszwischenspeicher (*stmthep*)

Die Größe des Anweisungszwischenspeichers hat zwar keinen Einfluss darauf, welchen Zugriffspfad das Optimierungsprogramm auswählt, kann sich jedoch auf den Grad der Optimierung auswirken, der für komplexe SQL- oder XQuery-Anweisungen ausgeführt wird.

Wenn der Wert für den Parameter *stmthep* nicht groß genug ist, empfangen Sie möglicherweise eine Warnung, die angibt, dass nicht genügend Speicher zur Verarbeitung der Anweisung zur Verfügung steht. Zum Beispiel kann der SQL-CODE +437 (SQLSTATE 01602) angeben, dass der Optimierungsgrad, der zur Kompilierung einer Anweisung verwendet wurde, geringer war als der Grad, den Sie angefordert haben.

- Maximaler Grad der Parallelität bei Abfragen (*max_querydegree*)

Wenn der Parameter *max_querydegree* den Wert „ANY“ hat, wählt das Optimierungsprogramm den Grad der Parallelität für die Anwendung aus. Ist ein anderer Wert als „ANY“ definiert, legt der vom Benutzer angegebene Wert den Grad der Parallelität für die Anwendung fest.

- Kommunikationsbandbreite (*comm_bandwidth*)

Die Kommunikationsbandbreite wird vom Optimierungsprogramm verwendet, um den Zugriffsplan zu bestimmen. Das Optimierungsprogramm verwendet den Wert dieses Parameters, um den Aufwand zur Durchführung für bestimmte Operationen zwischen den Datenbankpartitionsservern in einer Umgebung mit partitionierten Datenbanken abzuschätzen.

- Zwischenspeichergröße für Anwendungen (*applheapsz*)

Umfangreiche Schemata erfordern ausreichend Speicher im Anwendungszwischenspeicher. Es ist empfehlenswert, den Anwendungszwischenspeicher wie folgt zu vergrößern:

```
db2 update db cfg using applheapsz 1000
```

Zugehörige Konzepte:

- „Datenzugriffsmethoden“ auf Seite 192
- „Optimierungsstrategien für partitionsinterne Parallelität“ auf Seite 208
- „Optimierungsstrategien für MDC-Tabellen“ auf Seite 19
- „Der SQL- und XQuery-Compilerprozess“ auf Seite 175

Zugehörige Referenzen:

- „avg_appls - Durchschnittliche Anzahl aktiver Anwendungen “ auf Seite 484
- „comm_bandwidth - Kommunikationsbandbreite “ auf Seite 566
- „cpuspeed - CPU-Geschwindigkeit “ auf Seite 567
- „dft_degree - Standardgrad der Parallelität “ auf Seite 541
- „maxlocks - Maximale Anzahl Sperren pro Anwendung “ auf Seite 474

- „max_querydegree - Maximaler Grad der Parallelität bei Abfragen ” auf Seite 559
- „locklist - Maximaler Speicher für Sperrenliste ” auf Seite 438
- „sortheap - Zwischenspeichergröße für Sortierlisten ” auf Seite 458
- „stmtheap - Größe des Anweisungszwischenspeichers ” auf Seite 460

Auswirkung von Datenbankpartitionsgruppen auf die Abfrageoptimierung

In Umgebungen mit partitionierten Datenbanken erkennt das Optimierungsprogramm die Kollokation von Tabellen und nutzt sie bei der Bestimmung des besten Zugriffsplans für eine Abfrage. Wenn Tabellen häufig in Joinabfragen einbezogen werden, sollten sie auf Datenbankpartitionen in einer Umgebung mit partitionierten Datenbanken aufgeteilt werden, sodass sich die Zeilen aus jeder Tabelle, die verknüpft wird, in der gleichen Datenbankpartition befinden. Während der Joinoperation wird durch die Kollokation der Daten aus beiden verknüpften Tabellen eine Verschiebung der Daten von einer Datenbankpartition in die andere vermieden. Speichern Sie beide Tabellen in derselben Datenbankpartitionsgruppe, um sicherzustellen, dass die Daten aus den Tabellen durch Kollokation zusammengefasst werden.

In einer Umgebung mit partitionierten Datenbanken wird durch das Verteilen von Daten auf mehrere Datenbankpartitionen, abhängig von der Größe der Tabelle, die geschätzte Dauer (bzw. der Aufwand) zur Ausführung einer Abfrage verringert. Die Anzahl der Tabellen, die Größe der Tabellen, die Speicherposition der Daten in diesen Tabellen und die Art der Abfrage (d. h., ob ein Join erforderlich ist) wirken sich alle auf den Aufwand für die Abfrage aus.

Zugehörige Konzepte:

- „Joinmethoden in Umgebungen mit partitionierten Datenbanken” auf Seite 231
- „Joinstrategien in partitionierten Datenbanken” auf Seite 229
- „Hinzufügen von Datenbankpartitionen in einer Umgebung mit partitionierten Datenbanken” in *Systemverwaltung: Implementierung*

Auswirkung von Tabellenbereichen auf die Abfrageoptimierung

Bestimmte Merkmale der verwendeten Tabellenbereiche können die Auswahl des Zugriffsplans durch den Abfragecompiler beeinflussen:

- Kenndaten der Container

Containerkenndaten können sich wesentlich auf den Ein-/Ausgabeaufwand auswirken, der mit der Abfrageausführung verbunden ist. Bei der Auswahl eines Zugriffsplans berücksichtigt das Abfrageoptimierungsprogramm diesen Ein-/Ausgabeaufwand, einschließlich aller Unterschiede in den Aufwänden für die Zugriffe auf Daten verschiedener Tabellenbereiche. Zwei Spalten im Systemkatalog SYSCAT.TABLESPACES werden vom Optimierungsprogramm zur Abschätzung der E/A-Aufwände für den Zugriff auf Daten in einem Tabellenbereich herangezogen:

- Die Spalte OVERHEAD, die einen Schätzwert in Millisekunden für die Zeit enthält, die der Container benötigt, bevor irgendwelche Daten in den Speicher gelesen werden. In diesen Wert fließen der Aufwand für den E/A-Controller des Containers und die Latenzzeit der Platte, zur der auch die Suchzeit der Platte gehört, mit ein.

Mit Hilfe der folgenden Formel lässt sich der Systemaufwand (OVERHEAD) abschätzen:

$$\text{OVERHEAD} = \text{durchschnittliche Suchzeit in Millisekunden} + (0,5 * \text{rotationsbedingte Latenzzeit})$$

Dabei gilt Folgendes:

- 0,5 stellt den durchschnittlichen Aufwand für eine halbe Umdrehung (Rotation) dar.
- Die rotationsbedingte Latenzzeit für jede vollständige Umdrehung wird wie folgt in Millisekunden berechnet:

$$(1 / \text{Umdrehung pro Minute}) * 60 * 1000$$

Dabei sind die Berechnungsschritte wie folgt:

- Dividieren durch die Umdrehungen pro Minute, um die Minuten pro Umdrehung zu erhalten
- Multiplizieren mit 60, um die Sekunden pro Umdrehung zu erhalten
- Multiplizieren mit 1000, um die Millisekunden pro Umdrehung zu erhalten

Als Beispiel sei eine Plattengeschwindigkeit von 7 200 Umdrehungen pro Minute angenommen. Mit Hilfe der Umdrehungs-Latenz-Formel ergäbe sich folgender Wert:

$$(1 / 7200) * 60 * 1000 = 8,328 \text{ Millisekunden}$$

Das Ergebnis kann anschließend in der Berechnung des Schätzwerts für OVERHEAD bei einer angenommenen durchschnittlichen Suchzeit von 11 Millisekunden verwendet werden:

$$\begin{aligned} \text{OVERHEAD} &= 11 + (0,5 * 8,328) \\ &= 15,164 \end{aligned}$$

Der geschätzte OVERHEAD-Wert läge in diesem Fall bei ca. 15 Millisekunden.

- Die Spalte TRANSFERRATE, die einen Schätzwert in Millisekunden für die Zeit enthält, die zum Einlesen einer Datenseite in den Hauptspeicher benötigt wird.

Wenn jeder Tabellenbereichscontainer eine einzelne physische Platte ist, können Sie mit Hilfe der folgenden Formel den Übertragungsaufwand pro Seite in Millisekunden abschätzen:

$$\text{TRANSFERRATE} = (1 / \text{spec\_rate}) * 1000 / 1\,024\,000 * \text{seitengröße}$$

Dabei gilt Folgendes:

- spec_rate ist die Übertragungsgeschwindigkeit in MB pro Sekunde, die für die Platte angegeben wird.
- Dividieren durch spec_rate, um die Sekunden pro MB zu erhalten
- Multiplizieren mit 1000, um die Millisekunden pro MB zu erhalten
- Dividieren durch 1 024 000, um die Millisekunden pro Byte zu erhalten
- Multiplizieren mit der Seitengröße in Byte (z. B. 4 096 Byte für eine 4-KB-Seite), um die Millisekunden pro Seite zu erhalten

Als Beispiel sei eine angegebene Übertragungsgeschwindigkeit (spec_rate) von 3 MB pro Sekunde angenommen. Daraus ergäbe sich folgende Berechnung:

$$\begin{aligned} \text{TRANSFERRATE} &= (1 / 3) * 1000 / 1024000 * 4096 \\ &= 1,333248 \end{aligned}$$

Der geschätzte TRANSFERRATE-Wert läge hier bei ca. 1,3 Millisekunden pro Seite.

Wenn es sich bei den Tabellenbereichscontainern nicht um einzelne physische Platten, sondern um Platteneinheiten (Disk-Arrays, z. B. RAID) handelt, sind zusätzliche Punkte bei der Abschätzung des zu verwendenden Werts für TRANSFERRATE beachten. Wenn die Platteneinheit relativ klein ist, können Sie den Wert für spec_rate mit der Anzahl Platten multiplizieren, da anzunehmen ist, dass der Engpass auf Plattenebene liegt.

Ist die Anzahl der Platten in der Einheit, die den Container bildet, jedoch groß, liegt der Engpass vielleicht nicht auf Plattenebene, sondern bei einer der E/A-Subsystemkomponenten wie Einheitencontroller, E/A-Busse oder dem Systembus. In diesem Fall kann nicht angenommen werden, dass der E/A-Durchsatz das Produkt aus spec_rate und der Anzahl der Platten ist. Stattdessen muss die tatsächliche E/A-Geschwindigkeit während einer sequenziellen Tabellensuche in MB gemessen werden. Zum Beispiel könnte eine Tabellensuche mit einer Anweisung wie `select count(*) from große_tabelle` durchgeführt werden, die mehrere MB umfasst. Dividieren Sie das Ergebnis durch die Anzahl der Container, die den Tabellenbereich bilden, in dem große_tabelle gespeichert ist. Setzen Sie das Ergebnis für spec_rate in die oben angegebene Formel ein. Zum Beispiel würde eine gemessene sequenzielle E/A-Geschwindigkeit von 100 MB beim Durchsuchen einer Tabelle in einem Tabellenbereich mit vier Containern einen Wert von 25 MB pro Container bzw. einen TRANSFERRATE-Wert von $(1/25) * 1000 / 1024000 * 4096 = 0,16$ Millisekunden pro Seite bedeuten.

Jeder der einem Tabellenbereich zugeordneten Container kann sich auf einer anderen physischen Platte befinden. Um die besten Ergebnisse erzielen zu können, sollten alle physischen Platten, die für einen bestimmten Tabellenbereich verwendet werden, über die gleichen Werte für OVERHEAD und TRANSFERRATE verfügen. Wenn diese Merkmale nicht übereinstimmen, sollten Sie bei der Einstellung der Werte für OVERHEAD und TRANSFERRATE jeweils den Mittelwert verwenden.

Medienspezifische Werte für diese Spalten können Sie den technischen Daten zur Hardware entnehmen oder durch Experimentieren ermitteln. Diese Werte können in den Anweisungen CREATE TABLESPACE und ALTER TABLESPACE angegeben werden.

Experimentieren wird in der oben erwähnten Umgebung, in der eine Platteneinheit als Container eingesetzt wird, besonders wichtig. Es empfiehlt sich, eine einfache Abfrage, die Daten versetzt, zu erstellen und sie in Verbindung mit einem plattformspezifischen Messprogramm auszuführen. Anschließend können Sie die Abfrage mit anderen Containerkonfigurationen innerhalb des Tabellenbereichs wiederholen. Mit Hilfe der Anweisungen CREATE und ALTER TABLESPACE können Sie die Art und Weise ändern, wie Daten in der Umgebung übertragen werden.

Die durch diese beiden Werte bereitgestellten Informationen zum Ein-/Ausgabeaufwand können das Optimierungsprogramm in mehrfacher Weise beeinflussen, zum Beispiel dahin gehend, ob ein Index für den Zugriff auf die Daten zu verwenden ist und welche Tabellen als innere und äußere Tabellen in einem Join auszuwählen sind.

- **Vorablesezugriff**

Bei der Kalkulation des Ein-/Ausgabeaufwands für den Zugriff auf Daten in einem Tabellenbereich berücksichtigt das Optimierungsprogramm auch die potenziellen Auswirkungen, die das Vorablesen von Daten- und Indexseiten von Platten auf die Leistung der Abfrage haben kann. Der Vorablesezugriff auf

Daten- und Indexseiten kann den Aufwand und die Wartezeit verringern, die mit dem Einlesen der Daten in den Pufferpool verbunden sind.

Das Optimierungsprogramm verwendet die Informationen der Spalten PREFETCHSIZE und EXTENTSIZE im Systemkatalog SYSCAT.TABLESPACES, um die Menge der durch den Vorabesezugriff gelesenen Daten für einen Tabellenbereich abzuschätzen.

- Der Wert für EXTENTSIZE kann nur bei der Erstellung eines Tabellenbereichs (z. B. mit der Anweisung CREATE TABLESPACE) festgelegt werden. Der Standardwert für EXTENTSIZE beträgt 32 Seiten (von jeweils 4 KB) und ist in der Regel ausreichend.
- Der Wert für PREFETCHSIZE kann sowohl bei der Erstellung als auch bei der Änderung des Tabellenbereichs mit der Anweisung ALTER TABLESPACE festgelegt werden. Der Standardwert für PREFETCHSIZE wird durch den Wert des Datenbankkonfigurationsparameters DFT_PREFETCH_SZ festgelegt. Lesen Sie die Empfehlungen zur Einstellung dieses Parameters und nehmen Sie Änderungen nach Bedarf vor, um das Versetzen von Daten zu verbessern.

Das folgende Beispiel zeigt die Syntax zur Änderung der Merkmale des Tabellenbereichs RESOURCE:

```
ALTER TABLESPACE RESOURCE
  PREFETCHSIZE 64
  OVERHEAD      19.3
  TRANSFERRATE 0.9
```

Ziehen Sie nach dem Vornehmen von Änderungen an den Tabellenbereichen in Betracht, einen Rebind für Ihre Anwendungen durchzuführen und das Dienstprogramm RUNSTATS zum Erfassen der neuesten Statistikdaten zu Indizes auszuführen, um sicherzustellen, dass die besten Zugriffspläne verwendet werden.

Zugehörige Konzepte:

- „Von der Datenbank verwalteter Speicherbereich (DMS)“ in *Systemverwaltung: Konzept*
- „Katalogstatistiktabellen“ auf Seite 330
- „Der SQL- und XQuery-Compilerprozess“ auf Seite 175

Optimieren von Zugriffsplänen

Datenzugriffsmethoden

Bei der Kompilierung einer SQL- oder XQuery-Anweisung schätzt das Abfrageoptimierungsprogramm den Ausführungsaufwand der verschiedenen Methoden ab, die die Anforderung erfüllen würden. Auf der Grundlage dieser Schätzungen wählt das Optimierungsprogramm einen optimalen Zugriffsplan aus. Ein *Zugriffsplan* gibt die Reihenfolge von Operationen an, die erforderlich sind, um eine SQL- oder XQuery-Anweisung auszuführen. Wenn ein Anwendungsprogramm gebunden wird, wird ein *Paket* erstellt. Dieses Paket enthält Zugriffspläne für alle statischen SQL- und XQuery-Anweisungen in dem entsprechenden Anwendungsprogramm. Die Zugriffspläne für dynamische SQL- und XQuery-Anweisungen werden zum Zeitpunkt der Ausführung der Anwendung erstellt.

Es gibt zwei Methoden für den Zugriff auf Daten in einer Tabelle:

- Sequenzielles Durchsuchen der gesamten Tabelle

- Lokalisieren bestimmter Tabellenzeilen durch einen vorherigen Zugriff auf einen Index für die Tabelle

Zur Erstellung der von der Abfrage angeforderten Ergebnisse werden Zeilen in Abhängigkeit von den Bedingungen des Vergleichselements ausgewählt, das in der Regel in einer WHERE-Klausel angegeben wird. Die ausgewählten Zeilen in den Tabellen, auf die zugegriffen wird, werden verknüpft, um die Ergebnismenge zu erstellen, und die Ergebnismenge wird unter Umständen weiter verarbeitet, indem die Ausgabe gruppiert oder sortiert wird.

Zugehörige Konzepte:

- „Datenzugriff über Indexsuchen“ auf Seite 193
- „Indexzugriff und Clusterverhältnisse“ auf Seite 199
- „Der SQL- und XQuery-Compilerprozess“ auf Seite 175
- „Arten des Indexzugriffs“ auf Seite 196

Datenzugriff über Indexsuchen

Als *Indexsuche* wird der Vorgang bezeichnet, bei dem der Datenbankmanager zu folgenden Zwecken auf einen Index zugreift:

- Eingrenzen der Menge der den Bedingungen entsprechenden Zeilen (durch Durchsuchen der Zeilen in einem bestimmten Bereich des Index) vor dem Zugriff auf die Basistabelle. Der *Suchbereich* des Index (der Start- und der Stoppunkt der Suche) wird durch die Werte in der Abfrage festgelegt, mit denen Indexspalten verglichen werden.
- Sortieren der Ausgabe.
- Direktes Abrufen der angeforderten Spaltendaten. Wenn sich alle angeforderten Daten im Index befinden, braucht kein Zugriff auf die Basistabelle stattzufinden. Dies wird als *reiner Indexzugriff* bezeichnet.

Wenn Indizes mit der Option ALLOW REVERSE SCANS erstellt wurden, können Indexsuchen auch in entgegengesetzter Richtung zu der Richtung durchgeführt werden, in der die Indizes definiert wurden.

Anmerkung: Das Optimierungsprogramm wählt eine Tabellensuche, wenn kein entsprechender Index erstellt wurde oder eine Indexsuche aufwendiger wäre. Eine Indexsuche kann aufwendiger sein, wenn die Tabelle klein ist, das Indexclusterverhältnis niedrig ist oder die Abfrage die meisten der Tabellenzeilen anfordert. Ob der Zugriffsplan eine Tabellensuche oder eine Indexsuche verwendet, lässt sich mit Hilfe der EXPLAIN-Einrichtung feststellen.

Indexsuchen zur Begrenzung eines Bereichs

Zur Bestimmung, ob ein Index für eine bestimmte Abfrage verwendet werden kann, wertet das Optimierungsprogramm jede Spalte des Index beginnend bei der ersten Spalte aus, um zu überprüfen, ob sie zur Erfüllung von Gleichheitsvergleichselementen und anderen Vergleichselementen in der WHERE-Klausel verwendet werden kann. Ein *Vergleichselement* ist ein Element einer Suchbedingung in einer Klausel WHERE, das eine Vergleichsoperation definiert oder impliziert. Vergleichselemente können zur Begrenzung des Bereichs einer Indexsuche zu folgenden Zwecken verwendet werden:

- Testen auf Gleichheit mit einer Konstante, einer Hostvariablen, einem Ausdruck, der zu einer Konstanten ausgewertet wird, oder einem Schlüsselwort.

- Testen auf „IS NULL“ oder „IS NOT NULL“.
- Testen auf Gleichheit mit einer einfachen Unterabfrage, d. h. mit einer Unterabfrage, die weder ANY, ALL oder SOME noch einen Verweis über eine korrelierte Spalte auf den unmittelbar übergeordneten Abfrageblock (d. h. auf die SELECT-Anweisung, für die diese Unterabfrage ein Subselect ist) enthält.
- Testen auf strenge oder einschließende Ungleichheit.

Die folgenden Beispiele veranschaulichen, wann ein Index zur Begrenzung eines Bereichs verwendet werden könnte:

- Betrachten Sie einen Index mit der folgenden Definition:

```
INDEX IX1:  NAME   ASC,
            DEPT   ASC,
            MGR    DESC,
            SALARY DESC,
            YEARS  ASC
```

In diesem Fall könnten die folgenden Vergleichselemente zur Begrenzung des Bereichs bei der Suche im Index IX1 verwendet werden:

```
WHERE NAME   = :hv1
AND  DEPT    = :hv2
```

oder

```
WHERE MGR    = :hv1
AND  NAME    = :hv2
AND  DEPT    = :hv3
```

Beachten Sie, dass in der zweiten WHERE-Klausel die Vergleichselemente nicht in derselben Reihenfolge angegeben werden müssen, in der die Schlüsselspalten im Index erscheinen. In den Beispielen werden zwar Hostvariablen (hv = Host-variable) verwendet, jedoch hätten andere Variablen wie Parametermarken, Ausdrücke oder Konstanten dieselbe Wirkung.

- Betrachten Sie einen einzelnen Index, der mit dem Parameter ALLOW REVERSE SCANS erstellt wurde. Solche Indizes unterstützen Suchoperationen in der Richtung, die bei der Erstellung des Index definiert wurde, und Suchoperationen in entgegengesetzter oder umgekehrter Richtung. Die Anweisung könnte etwa wie folgt aussehen:

```
CREATE INDEX iname ON tname (cname DESC) ALLOW REVERSE SCANS
```

In diesem Fall wird der Index (iname) nach den absteigenden (DESCending) Werten der Spalte cname gebildet. Durch Zulassen von umgekehrten Suchoperationen kann eine Suche in aufsteigender Folge durchgeführt werden, obwohl der Index für die Spalte für Suchoperationen in absteigender Folge definiert ist. Die tatsächliche Verwendung des Index in beiden Richtungen wird nicht von Ihnen, sondern vom Optimierungsprogramm bei der Erstellung und Auswahl von Zugriffsplänen gesteuert.

In der folgenden Klausel WHERE würden nur die Vergleichselemente für NAME und DEPT verwendet, um den Bereich der Indexsuche einzugrenzen, jedoch nicht die Vergleichselemente für SALARY und YEARS:

```
WHERE NAME   = :hv1
AND  DEPT    = :hv2
AND  SALARY  = :hv4
AND  YEARS   = :hv5
```

Der Grund dafür ist der, dass es eine Schlüsselspalte (MGR) gibt, die diese Spalten von den ersten beiden Indexschlüsselspalten trennt, so dass die Reihenfolge nicht gewährleistet wäre. Wenn aber der Bereich einmal durch die Vergleichselemente

NAME = :hv1 und DEPT = :hv2 festgelegt ist, können die verbleibenden Vergleichselemente an den verbleibenden Indexschlüsselspalten ausgewertet werden.

Indexsuchen zum Testen von Ungleichheit

Bestimmte Ungleichheitsvergleichselemente können den Bereich einer Indexsuche begrenzen. Es gibt zwei Typen von Ungleichheitsvergleichselementen:

- Vergleichselemente für strenge Ungleichheit

Die Operatoren für strenge (ausschließende) Ungleichheit, die für bereichsbegrenzende Vergleichselemente verwendet werden, sind > (größer als) und < (kleiner als).

Nur eine Spalte mit Vergleichselementen für strenge Ungleichheit wird zur Begrenzung des Bereichs für eine Indexsuche berücksichtigt. Im folgenden Beispiel können die Vergleichselemente für die Spalten NAME und DEPT verwendet werden, um den Bereich einzugrenzen, aber das Vergleichselement für die Spalte MGR nicht.

```
WHERE NAME = :hv1
AND DEPT > :hv2
AND DEPT < :hv3
AND MGR < :hv4
```

- Vergleichselemente für einschließende Ungleichheit

Die folgenden Operatoren für einschließende Ungleichheit können für Vergleichselemente verwendet werden, die einen Bereich begrenzen:

- >= und <=
- BETWEEN
- LIKE

Zur Begrenzung eines Bereichs für eine Indexsuche werden mehrere Spalten mit Vergleichselementen einschließender Ungleichheit berücksichtigt. Im folgenden Beispiel können alle Vergleichselemente zur Eingrenzung des Bereichs der Indexsuche verwendet werden:

```
WHERE NAME = :hv1
AND DEPT >= :hv2
AND DEPT <= :hv3
AND MGR <= :hv4
```

Zur weiteren Verdeutlichung dieses Beispiels seien die folgenden Werte angenommen: :hv2 = 404, :hv3 = 406 und :hv4 = 12345. Der Datenbankmanager durchsucht den Index für die Abteilungen (DEPT) 404 und 405 ganz, bricht aber die Suche in Abteilung 406 ab, wenn er auf den ersten Manager trifft, der eine Personalnummer (Spalte MGR) über dem Wert 12345 hat.

Indexsuchen zum Sortieren von Daten

Wenn die Abfrage eine sortierte Ausgabe erfordert, kann ein Index zum Sortieren der Daten verwendet werden, wenn die ordnenden Spalten nacheinander, angefangen bei der ersten Indexschlüsselspalte, im Index vertreten sind. Das Ordnen oder Sortieren kann aus Operationen wie ORDER BY, DISTINCT, GROUP BY, Unterabfrage mit „= ANY“, Unterabfrage mit „> ALL“, Unterabfrage mit „> ALL“, INTERSECT bzw. EXCEPT sowie UNION resultieren. Eine Ausnahme ist der Fall, wenn die Indexschlüsselspalten auf Gleichheit mit „konstanten Werten“ verglichen werden, das heißt mit einem beliebigen Ausdruck, der zu einer Konstanten ausgewertet wird. In diesem Fall können die ordnenden Spalten andere als die ersten Indexschlüsselspalten sein.

Betrachten Sie die folgende Abfrage:

```
WHERE NAME = 'JONES'
AND DEPT = 'D93'
ORDER BY MGR
```

Für diese Abfrage könnte der Index verwendet werden, um die Zeilen zu sortieren, da die Spalten NAME und DEPT immer dieselben Werte haben und so geordnet sind. Das heißt, die gezeigten Klauseln WHERE und ORDER BY sind äquivalent mit folgenden Klauseln:

```
WHERE NAME = 'JONES'
AND DEPT = 'D93'
ORDER BY NAME, DEPT, MGR
```

Ein eindeutiger Index kann auch zum Verkürzen einer Sortieranforderung verwendet werden. Betrachten Sie die folgende Indexdefinition und die ORDER BY-Klausel:

```
UNIQUE INDEX IX0: PROJNO ASC
SELECT PROJNO, PROJNAME, DEPTNO
FROM PROJECT
ORDER BY PROJNO, PROJNAME
```

Eine zusätzliche Sortierung über die Spalte PROJNAME ist nicht erforderlich, da der Index IX0 bereits sicherstellt, dass die Werte der Spalte PROJNO eindeutig sind. Diese Eindeutigkeit sorgt dafür, dass es nur einen Wert für PROJNAME für jeden Wert von PROJNO gibt.

Zugehörige Konzepte:

- „Datenzugriffsmethoden“ auf Seite 192
- „Indexzugriff und Clusterverhältnisse“ auf Seite 199
- „Indexstruktur“ auf Seite 31
- „Arten des Indexzugriffs“ auf Seite 196

Arten des Indexzugriffs

In einigen Fällen kann das Optimierungsprogramm feststellen, dass alle Daten, die eine Abfrage aus einer Tabelle anfordert, aus einem Index für die Tabelle abgerufen werden können. In anderen Fällen kann das Optimierungsprogramm auch mehrere Indizes für den Zugriff auf Tabellen verwenden. Im Fall von Bereichsclustertabellen kann der Zugriff auf Daten über einen „virtuellen“ Index erfolgen, der die Positionen von Datensätzen berechnet.

Reiner Indexzugriff

In einigen Fällen können alle angeforderten Daten aus dem Index abgerufen werden, ohne auf die Tabelle zuzugreifen. Dies wird als *reiner Indexzugriff* bezeichnet.

Betrachten Sie die folgende Indexdefinition zur Illustration des reinen Indexzugriffs:

```
INDEX IX1: NAME    ASC,
           DEPT    ASC,
           MGR     DESC,
           SALARY  DESC,
           YEARS   ASC
```

Die folgende Abfrage kann nur durch den Zugriff auf den Index ohne Lesen der Basistabelle erfüllt werden:

```
SELECT NAME, DEPT, MGR, SALARY
FROM EMPLOYEE
WHERE NAME = 'SMITH'
```

Häufig sind erforderliche Spalten jedoch nicht im Index vertreten. Um die Daten für diese Spalten abzurufen, müssen Tabellenzeilen gelesen werden. Damit das Optimierungsprogramm die Möglichkeit hat, einen reinen Indexzugriff zu wählen, erstellen Sie einen eindeutigen Index mit INCLUDE-Spalten. Betrachten Sie zum Beispiel die folgende Indexdefinition:

```
CREATE UNIQUE INDEX IX1 ON EMPLOYEE
(NAME ASC)
INCLUDE (DEPT, MGR, SALARY, YEARS)
```

Dieser Index gewährleistet die Eindeutigkeit der Datenwerte in der Spalte NAME und speichert und pflegt darüber hinaus Daten für die Spalten DEPT, MGR, SALARY und YEARS. Dadurch kann die folgende Abfrage nur durch einen Zugriff auf den Index erfüllt werden:

```
SELECT NAME, DEPT, MGR, SALARY
FROM EMPLOYEE
WHERE NAME='SMITH'
```

Wenn Sie das Hinzufügen von INCLUDE-Spalten für einen Index in Betracht ziehen, müssen Sie jedoch abwägen, ob der zusätzliche Speicherplatzbedarf und der Pflegeaufwand gerechtfertigt sind. Wenn Abfragen, die durch einen solchen Index erfüllt werden können, nur selten ausgeführt werden, ist der Zusatzaufwand möglicherweise nicht gerechtfertigt.

Zugriff auf mehrere Indizes

Das Optimierungsprogramm kann entscheiden, mehrere Indizes für dieselbe Tabelle zu durchsuchen, um die Vergleichselemente einer WHERE-Klausel zu erfüllen. Betrachten Sie zum Beispiel die beiden folgenden Indexdefinitionen:

```
INDEX IX2:  DEPT    ASC
INDEX IX3:  JOB     ASC,
            YEARS   ASC
```

Die folgenden Vergleichselemente können durch die Verwendung der beiden Indizes erfüllt werden:

```
WHERE DEPT = :hv1
      OR (JOB   = :hv2
          AND YEARS >= :hv3)
```

Das Durchsuchen des Index IX2 liefert eine Liste von Zeilen-IDs (RIDs), die das Vergleichselement DEPT = :hv1 erfüllen. Das Durchsuchen des Index IX3 liefert eine Liste der RIDs, die das Vergleichselement JOB = :hv2 AND YEARS >= :hv3 erfüllen. Diese beiden Listen von RIDs werden kombiniert und doppelte Werte entfernt, bevor auf die Tabelle zugegriffen wird. Diese Methode wird als *logischer OR-Join von Indizes* (Index ORing) bezeichnet.

Der OR-Join von Indizes kann auch für Vergleichselemente mit der Klausel IN wie in folgendem Beispiel verwendet werden:

```
WHERE DEPT IN (:hv1, :hv2, :hv3)
```

Während der Zweck des OR-Joins von Indizes in der Eliminierung doppelter RIDs liegt, besteht das Ziel des *logischen AND-Joins von Indizes* (Index ANDing) darin, gemeinsame RIDs zu finden. Der AND-Join von Indizes kann auftreten, wenn Anwendungen mehrere Indizes für entsprechende Spalten innerhalb derselben

Tabelle erstellen und eine Abfrage mit mehreren AND-Vergleichselementen für die Tabelle ausgeführt wird. Mehrere Indexsuchen für alle mit Indizes versehenen Spalten in einer solchen Abfrage ergeben Werte, mit denen in einem Hashverfahren Bitzuordnungen erstellt werden. Die zweite Bitzuordnung wird zur Prüfung der ersten Bitzuordnung verwendet, um die gesuchten Zeilen zu generieren, die zur Erstellung der endgültigen zurückzugebenden Datenmenge abgerufen werden.

Betrachten Sie zum Beispiel die beiden folgenden Indexdefinitionen:

```
INDEX IX4: SALARY  ASC
INDEX IX5: COMM   ASC
```

Die folgenden Vergleichselemente könnten mit Hilfe dieser beiden Indizes aufgelöst werden:

```
WHERE SALARY BETWEEN 20000 AND 30000
      AND COMM BETWEEN 1000 AND 3000
```

In diesem Beispiel generiert das Durchsuchen des Index IX4 eine Bitzuordnung, die das Vergleichselement SALARY BETWEEN 20000 AND 30000 erfüllt. Das Durchsuchen von IX5 und Prüfen gegen die Bitzuordnung für IX4 liefert eine Liste von RIDs, die beide Vergleichselemente erfüllen. Dies wird als „dynamischer AND-Join über Bitzuordnungen“ bezeichnet. Diese Methode wird nur angewandt, sofern die Tabelle ausreichend Kardinalität hat und die Spalten genügend Werte in dem gesuchten Bereich oder genügend Duplizität haben, wenn Gleichheitsvergleichselemente verwendet werden.

Zur Umsetzung der Leistungsvorteile dynamischer Bitzuordnungen beim Durchsuchen mehrerer Indizes kann es notwendig sein, den Wert des Konfigurationsparameters für die Größe des Sortierspeichers (*sortheap*) der Datenbank und des Konfigurationsparameters für den Schwellenwert für Sortierspeicher (*sheapthres*) des Datenbankmanagers zu ändern.

Bei Verwendung dynamischer Bitzuordnungen in Zugriffsplänen wird zusätzlicher Sortierspeicher benötigt. Wenn der Wert von *sheapthres* relativ nahe am Wert von *sortheap* liegt (d. h. weniger als ein Faktor von 2- oder 3-mal pro gleichzeitiger Abfrage), steht dynamischen Bitzuordnungen mit Zugriff über mehrere Indizes wesentlich weniger Speicher zur Verfügung als das Optimierungsprogramm veranschlagt hat. Die Lösung besteht darin, den Wert von *sheapthres* relativ zu *sortheap* zu erhöhen.

Anmerkung: Das Optimierungsprogramm verwendet beim Zugriff auf eine einzelne Tabelle keine Kombination aus AND- und OR-Joins von Indizes.

Indexzugriff in Bereichsclustertabellen

Im Gegensatz zu Standardtabellen benötigt eine Bereichsclustertabelle keinen physischen Index, der einer Zeile einen Schlüsselwert wie in einem traditionellen B-Baumstrukturindex zuordnet. Stattdessen greift er auf die sequenzielle Anordnung des Spaltenwertebereichs zurück und verwendet eine Zuordnungsfunktion, um die Position einer bestimmten Zeile in einer Tabelle zu generieren. Im einfachsten Beispiel einer solchen Zuordnung ist der erste Schlüsselwert im Bereich die erste Zeile in der Tabelle, der zweite Wert im Bereich die zweite Zeile in der Tabelle usw.

Das Optimierungsprogramm verwendet das Merkmal des Bereichsclustering der Tabelle, um Zugriffspläne auf der Basis eines perfekt angeordneten Index zu gene-

rieren, der lediglich den Aufwand zur Berechnung der Bereichsclusterfunktion erfordert. Das Clustering von Zeilen innerhalb der Tabelle ist gewährleistet, da Bereichsclustertabellen ihre Reihenfolge nach den ursprünglichen Schlüsselwerten beibehalten.

Zugehörige Konzepte:

- „Datenzugriffsmethoden“ auf Seite 192
- „Datenzugriff über Indexsuchen“ auf Seite 193
- „Indexzugriff und Clusterverhältnisse“ auf Seite 199
- „Verwenden von relationalen Indizes zur Leistungsverbesserung“ auf Seite 29

Indexzugriff und Clusterverhältnisse

Bei der Auswahl des Zugriffsplans schätzt das Optimierungsprogramm die Anzahl der E/A-Operationen ab, die zum Einlesen der benötigten Seiten von der Platte in den Pufferpool erforderlich sind. Diese Schätzung schließt eine Voraussage über die Nutzung des Pufferpools mit ein, da zum Lesen von Zeilen einer Seite, die sich bereits im Pufferpool befindet, keine weiteren E/A-Operationen anfallen.

Für Indexsuchen wird das Optimierungsprogramm durch Informationen aus den Systemkatalogtabellen (SYSCAT.INDEXES) bei der Abschätzung des Ein-/Ausgabeaufwands zum Lesen von Datenseiten in den Pufferpool unterstützt. Dabei werden Informationen aus den folgenden Spalten der Tabelle SYSCAT.INDEXES verwendet:

- Die Informationen der Spalte CLUSTERRATIO geben den Grad an, zu dem die Tabellendaten in Relation zu diesem Index in Clustern zusammengefasst sind (Clusterbildung). Je höher der Wert, desto besser sind die Zeilen in der Reihenfolge des Indexschlüssels geordnet. Wenn Tabellenzeilen nahe an der Reihenfolge des Indexschlüssels vorliegen, können Zeilen von einer Datenseite gelesen werden, während sich die Seite im Puffer befindet. Wenn der Wert dieser Spalte -1 ist, verwendet das Optimierungsprogramm die Informationen der Spalten PAGE_FETCH_PAIRS und CLUSTERFACTOR, wenn diese verfügbar sind.
- Die Spalte PAGE_FETCH_PAIRS enthält Paare von Zahlen, die zusammen mit CLUSTERFACTOR-Informationen jeweils ein Modell für die Anzahl der E/A-Operationen zum Lesen der Datenseiten in Pufferpools verschiedener Größen angeben. Für diese Spalten werden Daten nur erfasst, wenn Sie das Dienstprogramm RUNSTATS für den Index mit der Klausel DETAILED ausführen.

Wenn keine Statistiken zur Clusterbildung verfügbar sind, verwendet das Optimierungsprogramm Standardwerte, die von einem geringen Grad an Clusterbildung der Daten bezüglich des Index ausgehen.

Der Grad, zu dem die Daten in Bezug auf einen Index in Clustern zusammengefasst sind, kann bedeutende Auswirkungen auf die Leistung haben, so dass einer der Indizes für eine Tabelle auf einem Grad nahe an 100% Clusterbildung gehalten werden sollte.

Im Allgemeinen kann nur ein Index eine 100-prozentige Clusterbildung aufweisen. Eine Ausnahme bilden nur solche Fälle, in denen die Schlüssel eines anderen Index eine Obermenge der Schlüssel des Clusterindex sind oder in denen es eine De-Facto-Korrelation zwischen den Schlüsselspalten der beiden Indizes gibt.

Bei der Reorganisation einer Tabelle können Sie einen Index angeben, über den die Zeilen in Clustern angeordnet werden und versucht wird, diese Clusterbildung

bei der Verarbeitung von Einfügungen beizubehalten. Da Aktualisierungen und Einfügungen die Clusterbildung in Bezug auf den Index verringern können, müssen Sie die Tabelle eventuell in regelmäßigen Abständen reorganisieren. Um die Häufigkeit der Reorganisation einer Tabelle zu verringern, die aufgrund von INSERT-, UPDATE- und DELETE-Anweisungen häufig geändert wird, können Sie beim Ändern einer Tabelle (ALTER TABLE) den Parameter PCTFREE verwenden. Dieser Parameter ermöglicht es, weitere Einfügungen so in die vorhandenen Daten einzugliedern, dass die Clusterbildung erhalten bleibt.

Zugehörige Konzepte:

- „Tipps zur Leistung von relationalen Indizes“ auf Seite 35
- „Arten des Indexzugriffs“ auf Seite 196

Vergleichselementtypologie und Zugriffspläne

Eine Benutzeranwendung fordert eine Reihe von Zeilen aus der Datenbank mit Hilfe einer Abfrageanweisung an, in der Qualifikationsmerkmale für die gewünschten Zeilen angegeben werden, die als Ergebnismenge zurückgegeben werden sollen. Diese Qualifikationsbedingungen stehen in der Regel in der WHERE-Klausel der Abfrage. Solche Qualifikationsbedingungen werden als *Vergleichselemente* bezeichnet. Vergleichselemente können in vier Kategorien zusammengefasst werden, die dadurch bestimmt sind, wie und wann das jeweilige Vergleichselement im Auswertungsprozess verwendet wird. Die Kategorien werden im Folgenden in der Reihenfolge von der höchsten bis zur niedrigsten Leistung geordnet aufgelistet:

1. Bereichsbegrenzende Vergleichselemente
2. Bei Indexsuchen als Suchargument verwendbare Vergleichselemente (Index SARGable)
3. Bei Datensuchen als Suchargument verwendbare Vergleichselemente (Data SARGable)
4. Restvergleichselemente

Anmerkung: Die Bezeichnung *SARGable* ist aus dem Begriff *search argument* abgeleitet.

Die folgende Tabelle enthält eine Übersicht über die Kategorien von Vergleichselementen. In den nachfolgenden Abschnitten werden die einzelnen Kategorien eingehender beschrieben.

Tabelle 49. Zusammenfassung der Merkmale der Vergleichselementkategorien

Merkmal	Vergleichselementkategorie			
	Bereichsbegrenzend	Indexsuchargument	Datensuchargument	Restvergleichselement
Verringern der Index-Ein-/Ausgabe	Ja	Nein	Nein	Nein
Verringern der Datenseitenein-/ausgabe	Ja	Ja	Nein	Nein
Verringern der Anzahl intern übergebener Zeilen	Ja	Ja	Ja	Nein

Tabelle 49. Zusammenfassung der Merkmale der Vergleichselementkategorien (Forts.)

Merkmal	Vergleichselementkategorie			
	Bereichs- begrenzend	Indexsuch- argument	Datensuch- argument	Restvergleichs- element
Verringern der Anzahl der den Bedingungen entsprechenden Zeilen	Ja	Ja	Ja	Ja

Bereichsbegrenzende und bei Indexsuchen als Suchargumente verwendbare Vergleichselemente

Bereichsbegrenzende Vergleichselemente begrenzen den Bereich einer Indexsuche. Sie geben Start- und Stoppschlüsselwerte für die Indexsuche an. Bei Indexsuchen als Suchargument verwendbare Vergleichselemente können nicht den Bereich einer Suche begrenzen, aber sie können mit Hilfe des Index ausgewertet werden, da die im Vergleichselement verwendeten Spalten Teil des Indexschlüssels sind. Betrachten Sie zum Beispiel den folgenden Index:

```
INDEX IX1: NAME    ASC,
           DEPT    ASC,
           MGR     DESC,
           SALARY  DESC,
           YEARS   ASC
```

Betrachten Sie dazu eine Abfrage, welche die folgende WHERE-Klausel enthält:

```
WHERE NAME = :hv1
AND   DEPT = :hv2
AND   YEARS > :hv5
```

Die ersten beiden Vergleichselemente (NAME = :hv1, DEPT = :hv2) sind bereichsbegrenzende Vergleichselemente, während YEARS > :hv5 ein bei Indexsuchen als Suchargument verwendbares Vergleichselement ist.

Das Optimierungsprogramm verwendet die Indexdaten bei der Auswertung dieser Vergleichselemente, anstatt die Basistabelle zu lesen. Diese in Indizes als Suchargumente verwendbaren (*Index SARGable*) Vergleichselemente verringern die Anzahl der Zeilen, die aus der Tabelle gelesen werden müssen, sie beeinflussen jedoch nicht die Anzahl von Indexseiten, auf die zugegriffen wird.

Vergleichselemente für XML-Daten, die in XMLEXISTS- und XMLTABLE-Ausdrücken auftreten, werden ebenfalls von Suchvorgängen des Datenoperators XSCAN unterstützt. Einige dieser Vergleichselemente werden auch von Indexbereichsuchen unterstützt.

Bei Datensuchen als Suchargument verwendbare Vergleichselemente

Vergleichselemente, die nicht vom Indexmanager ausgewertet werden können, sondern nur von den Datenverwaltungsservices, werden als bei Datensuchen verwendbare (*Data SARGable*) Vergleichselemente bezeichnet. Für solche Vergleichselemente ist in der Regel ein Zugriff auf einzelne Zeilen einer Tabelle erforderlich. Bei Bedarf rufen die Datenverwaltungsservices die zur Auswertung des Vergleichselements benötigten Spalten und andere Spalten ab, um die Spalten für die SELECT-Liste, die nicht aus dem Index abgerufen werden konnten, zur Verfügung zu stellen.

Betrachten Sie zum Beispiel einen einzelnen Index, der für die Tabelle PROJECT definiert ist:

```
INDEX IX0: PROJNO ASC
```

Für die folgende Abfrage wird dann das Vergleichselement DEPTNO = 'D11' als bei Datensuchen als Suchargument verwendbar betrachtet.

```
SELECT PROJNO, PROJNAME, RESPEMP
FROM PROJECT
WHERE DEPTNO = 'D11'
ORDER BY PROJNO
```

Restvergleichselemente

Restvergleichselemente erfordern mehr E/A-Aufwand als der Zugriff auf eine Tabelle. Sie können folgende Merkmale aufweisen:

- Sie verwenden korrelierte Unterabfragen.
- Sie verwenden quantifizierte Unterabfragen mit den Klauseln ANY, ALL, SOME oder IN.
- Sie lesen LONG VARCHAR- oder LOB-Daten, die in einer von der Tabelle getrennten Datei gespeichert sind.

Solche Vergleichselemente werden von den Services für relationale Daten (Relational Data Services) ausgewertet.

Manchmal müssen Vergleichselemente, die nur auf den Index angewandt wurden, erneut angewandt werden, wenn auf die Datenseite zugegriffen wird. Zum Beispiel wenden Zugriffspläne mit OR-Joins oder AND-Joins von Indizes die Vergleichselemente immer ein weiteres Mal als Restvergleichselemente an, wenn auf die Datenseite zugegriffen wird.

Zugehörige Konzepte:

- „Der SQL- und XQuery-Compilerprozess“ auf Seite 175

Auswirkungen des Sortierens und Gruppierens

Wenn das Optimierungsprogramm einen Zugriffsplan auswählt, kalkuliert es die Auswirkungen einer Sortierung von Daten auf die Leistung mit ein. Sortieroperationen werden durchgeführt, wenn kein Index die angeforderte Reihenfolge der abgerufenen Daten herstellen kann. Eine Sortierung kann auch erfolgen, wenn das Optimierungsprogramm feststellt, dass eine Sortierung weniger aufwendig als eine Indexsuche ist. Das Optimierungsprogramm sortiert Daten auf eine der folgenden Arten:

- Weitergeben der Sortierergebnisse über eine Pipe (Piping), wenn die Abfrage ausgeführt wird
- Interne Behandlung der Sortierung innerhalb des Datenbankmanagers

Vergleich von Sortierungen mit und ohne Piping

Wenn die endgültige, sortierte Liste von Daten in einem einzigen sequenziellen Vorgang gelesen werden kann, können die Ergebnisse über eine *Pipe* geleitet werden (Piping). Mit der Piping-Methode lassen sich die Ergebnisse der Sortierung schneller übertragen als mit Methoden ohne Piping. Das Optimierungsprogramm wählt, wenn möglich, die Pipe zur Übergabe der Sortierergebnisse.

Ungeachtet dessen, ob eine Sortierung mit oder ohne Piping erfolgt, hängt die für die Sortierung benötigte Zeit von einer Reihe von Faktoren ab, wie zum Beispiel der Anzahl der zu sortierenden Zeilen, der Größe des Sortierschlüssels und der Zeilenlänge. Wenn die zu sortierenden Zeilen mehr als den im Zwischenspeicher für Sortierlisten verfügbaren Speicherbereich in Anspruch nehmen, werden mehrere Sortierarbeitsgänge durchgeführt, wobei in jedem Arbeitsgang eine Untermenge der Gesamtmenge von Zeilen sortiert wird. Jeder Arbeitsgang des Sortiervorgangs wird in einer temporären Tabelle im Pufferpool gespeichert. Falls im Pufferpool nicht genügend Platz vorhanden ist, können Seiten aus dieser temporären Tabelle auf die Platte geschrieben werden. Wenn alle Arbeitsgänge des Sortiervorgangs abgeschlossen sind, müssen die sortierten Untermengen zu einer einzigen sortierten Menge von Zeilen zusammengefügt werden. Wenn die Sortierung über eine Pipe geleitet wird, werden die Zeilen beim Zusammenfügen direkt an die Services für relationale Daten (Relational Data Services) weitergegeben.

Pushdown von Gruppier- und Sortieroperatoren

In einigen Fällen kann sich das Optimierungsprogramm entscheiden, einen Sortiervorgang oder eine Spaltenberechnung (Aggregation) von der Komponente der Services für relationale Daten an die Komponente der Datenverwaltungsservices zu verschieben („Pushdown“). Eine solche Verschiebung dieser Operationen verbessert die Leistung, da nun die Datenverwaltungsservices Daten direkt an eine Sortier- oder Spaltenberechnungsroutine übergeben können. Ohne diese Verschiebung übergeben die Datenverwaltungsservices diese Daten zunächst an die Services für relationale Daten, die anschließend wiederum mit den Sortier- bzw. Spaltenberechnungsroutinen kommunizieren. Die folgende Abfrage beispielsweise kann von dieser Art der Optimierung profitieren:

```
SELECT WORKDEPT, AVG(SALARY) AS AVG_DEPT_SALARY
FROM EMPLOYEE
GROUP BY WORKDEPT
```

Gruppierungsoperationen in Sortierungen

Wenn das Sortieren dazu dient, die erforderliche Reihenfolge für eine Operation GROUP BY herzustellen, kann das Optimierungsprogramm einige oder alle Spaltenberechnungen (Aggregation) für GROUP BY während des Sortierens durchführen. Dies ist vorteilhaft, wenn die Anzahl der Zeilen in jeder Gruppe sehr groß ist. Es wird sogar noch vorteilhafter, wenn die Durchführung eines Teils der Gruppierung während des Sortierens die Notwendigkeit, dass die Sortierung einen Überlauf auf die Festplatte verursacht, verringert oder ausschließt.

Eine Spaltenberechnung in einer Sortierung erfordert unter Umständen alle der drei folgenden Phasen der Spaltenberechnung, um sicherzustellen, dass die richtigen Ergebnisse zurückgegeben werden.

1. Die erste Phase der Spaltenberechnung, die partielle Spaltenberechnung, errechnet die Ergebniswerte, bis der Sortierspeicher voll ist. Bei der partiellen Spaltenberechnung werden nicht berechnete Daten entgegengenommen und partielle Ergebniswerte erstellt. Wenn der Sortierspeicher voll ist, läuft der Rest der Daten auf die Festplatte über, einschließlich aller partiellen Spaltenberechnungsergebnisse, die im aktuellen Sortierspeicher berechnet wurden. Nach dem Zurücksetzen des Sortierspeichers werden neue Spaltenberechnungen gestartet.
2. Die zweite Phase der Spaltenberechnung, die Zwischenberechnung, nimmt alle übergelaufenen Sortierdurchläufe und setzt die Spaltenberechnung für die Gruppierschlüssel fort. Die Spaltenberechnung kann nicht ausgeführt werden, weil die Gruppierschlüsselspalten eine Untergruppe der Verteilungsschlüsselspalten sind. Die Zwischenberechnung stellt aus vorhandenen partiellen Spalten-

berechnungen neue partielle Spaltenberechnungen. Diese Phase findet nicht immer statt. Sie wird sowohl für partitionsinterne als auch für partitionsübergreifende Parallelität verwendet. Bei der partitionsinternen Parallelität ist die Gruppierung beendet, wenn ein globaler Gruppierungsschlüssel verfügbar ist. Bei partitionsübergreifender Parallelität findet sie statt, wenn der Gruppierungsschlüssel eine Untergruppe des Verteilungsschlüssels ist, der Gruppen auf Datenbankpartitionen verteilt, und so eine Neuverteilung zur Beendigung der Spaltenberechnung erforderlich macht. Ein ähnlicher Fall liegt vor, wenn bei partitionsinterner Parallelität jeder Agent seine übergelaufenen Sortierdurchgänge zusammengefügt hat, bevor auf einen einzigen Agenten reduziert wird, um die Spaltenberechnung zu beenden.

3. Die letzte Phase der Spaltenberechnung, die Endberechnung, verwendet alle partiellen Berechnungsergebnisse und erstellt die endgültige Spaltenberechnung. Dieser Schritt findet immer in einem Operator GROUP BY statt. Das Sortieren kann die Spaltenberechnung nicht bis zu Ende durchführen, weil es keine Garantie gibt, dass die Sortierung nicht geteilt wird. Die Komplettberechnung nimmt nicht berechnete Daten auf und produziert Endergebnisse. Wenn die Verteilung die Verwendung dieser Methode nicht verhindert, wird die Methode der Spaltenberechnung in der Regel zur Gruppierung von Daten verwendet, die bereits in der richtigen Reihenfolge vorliegen.

Zugehörige Konzepte:

- „Optimieren der Sortierleistung“ auf Seite 129

Zugehörige Referenzen:

- „sheapthres - Schwellenwert für Sortierspeicher “ auf Seite 456
- „sortheap - Zwischenspeichergroße für Sortierlisten “ auf Seite 458

Berechnen von Gruppierungsschlüsselkardinalitäten mit Index- und Spaltengruppenstatistikdaten

Wenn eine Abfrage eine bestimmte Gruppierung von Daten erfordert, muss das Optimierungsprogramm die Anzahl der unterschiedlichen Gruppierungen, d. h. die *Gruppierungsschlüsselkardinalitäten* ('grouping keycard') berechnen. Eine Gruppierungsanforderung kann aus solchen Operationen wie GROUP BY oder DISTINCT resultieren.

Betrachten Sie die folgende Abfrage:

```
SELECT DEPTNO, YEARS, AVG(SALARY)
  FROM EMPLOYEE
 GROUP BY DEPTNO, MGR, YEAR_HIRED
```

Ohne Index- oder Spaltengruppenstatistiken entspricht die vom Optimierungsprogramm geschätzte Anzahl von Gruppierungen (in diesem Fall auch die Anzahl der zurückgegebenen Zeilen) dem Produkt aus den Anzahlen unterschiedlicher Werte der Spalten DEPTNO, MGR und YEAR_HIRED. Dieser Schätzwert basiert auf der Annahme, dass die Gruppierungsschlüsselspalten unabhängig sind. Diese Annahme kann jedoch abwegig sein, wenn jeder Manager genau eine Abteilung leitet. Darüber hinaus ist es unwahrscheinlich, dass jede Abteilung jedes Jahr Mitarbeiter angestellt hat. Daher könnte das Produkt der unterschiedlichen Werte der Spalten DEPTNO, MGR und YEAR_HIRED als Schätzwert für die tatsächliche Anzahl der Gruppen mit unterschiedlichen Werten zu hoch sein.

Betrachten Sie nun einen Index mit der folgenden Definition:

```
INDEX IX1: DEPTNO, MGR, YEAR_HIRED
```

In diesem Fall liefert der FULLKEYCARD-Wert von IX1 dem Optimierungsprogramm die genaue Anzahl der unterschiedlichen Gruppierungen für die obige Abfrage.

Betrachten Sie eine weitere Indexdefinition:

```
INDEX IX2: DEPTNO, MGR, YEAR_HIRED, COMM
```

Der Index IX2 könnte ebenfalls helfen, die Gruppierungsschlüsselkardinalität zu berechnen, da der Wert FIRST3KEYCARD angibt, wie viele unterschiedliche Gruppen sich aus (DEPTNO,MGR,YEAR_HIRED) ergeben.

Neben den Indexstatistikdaten (FIRST2KEYCARD, FIRST3KEYCARD, FIRST4KEYCARD und FULLKEYCARD) könnten auch Spaltengruppenstatistiken in ähnlicher Weise vom Optimierungsprogramm zur Berechnung der Gruppierungsschlüsselkardinalität genutzt werden. Spaltengruppenstatistikdaten, die für die Spalten DEPTNO, MGR und YEAR_HIRED erfasst werden, bieten den gleichen Nutzen wie die oben gezeigten Indizes IX1 und IX2:

```
RUNSTATS ON TABLE EMPLOYEE ON COLUMNS ((DEPTNO, MGR, YEAR_HIRED))
```

Beachten Sie, dass, wenn die Gruppierungsschlüssel aus fünf und mehr Spalten bestehen, eine Erfassung von Spaltengruppenstatistiken möglicherweise vorzuziehen ist. Dies liegt daran, dass das Dienstprogramm RUNSTATS nur Statistiken für die ersten vier Spalten sowie für alle Indexschlüsselspalten eines bestimmten Index erfasst.

Zugehörige Konzepte:

- „Spaltenkorrelation für mehrere Vergleichselemente“ auf Seite 227
- „Anzeigen der für die Optimierung relevanten Statistikdaten“ auf Seite 213

Zugehörige Tasks:

- „Erfassen von Verteilungsstatistiken für bestimmte Spalten“ auf Seite 324

Verbessern der Leistung durch Binden mit REOPT

SQL- oder XQuery-Abfragen können bei der Ausführung eine schlechte Leistung zeigen, wenn die Werte, die für die Eingabevariablen, wie zum Beispiel Parametermarken, Hostvariablen und Sonderregister, verwendet werden, außerhalb des vorhersagbaren Bereichs der Schätzwerte für Standardfilterfaktoren liegen. Standardfilterfaktoren, die in Szenarios verwendet werden, in denen der tatsächliche Datenwert unbekannt ist, sind Schätzwerte für die Anzahl von Zeilen, die tatsächlich zur Laufzeit zurückgegeben werden, wenn der tatsächliche Datenwert verwendet wird.

Die Bindeoption REOPT gibt an, ob DB2 einen Zugriffspfad unter Verwendung von Werten für Hostvariablen, Parametermarken und Sonderregistern optimieren soll. REOPT-Werte werden durch die folgenden Argumente in den Befehlen BIND, PREP und REBIND angegeben:

REOPT NONE

Der Zugriffspfad für eine gegebene SQL- bzw. XQuery-Anweisung, in der Hostvariablen, Parametermarken oder Sonderregister enthalten sind, wird nicht mit realen Werten für diese Variablen optimiert. Stattdessen werden

die Standardschätzwerte für diese Variablen verwendet. Ein solcher Plan wird in den Cache gestellt und nachfolgend verwendet. Dies ist das Standardverhalten.

REOPT ONCE

Der Zugriffspfad für eine gegebene SQL- bzw. XQuery-Anweisung wird mit den realen Werten der Hostvariablen, Parametermarken oder Sonderregister optimiert, wenn die Abfrage zum ersten Mal ausgeführt wird. Ein solcher Plan wird in den Cache gestellt und nachfolgend verwendet.

REOPT ALWAYS

Der Zugriffspfad für eine gegebene SQL- bzw. XQuery-Anweisung wird immer kompiliert und mit den bei der jeweiligen Ausführung bekannten Werten der Hostvariablen, Parametermarken oder Sonderregister optimiert.

Zugehörige Konzepte:

- „Performance improvements when using REOPT option of the BIND command“ in *Developing Embedded SQL Applications*

Replizierte MQTs in partitionierten Datenbankumgebungen

Replizierte MQTs (MQT - Materialized Query Table, gespeicherte Abfragetabelle) verbessern die Leistung häufig ausgeführter Joins in einer partitionierten Datenbankumgebung, indem sie der Datenbank die Möglichkeit geben, vorberechnete Werte der Tabellendaten zu pflegen.

Betrachten Sie ein Beispiel einer Abfrage und einer replizierten MQT. Dazu gelten folgende Annahmen:

- Die Tabelle SALES (Verkauf) befindet sich im Mehrpartitionstabellenbereich REGIONTABLESPACE und wird über die Spalte REGION unterteilt.
- Die Tabellen EMPLOYEE (Mitarbeiter) und DEPARTMENT (Abteilung) sind in einer Datenbankpartitionsgruppe mit Einzelpartition.

Erstellen Sie eine replizierte MQT auf der Basis der Informationen in der Tabelle EMPLOYEE.

```
CREATE TABLE R_EMPLOYEE
AS (
    SELECT EMPNO, FIRSTME, MIDINIT, LASTNAME, WORKDEPT
    FROM EMPLOYEE
)
DATA INITIALLY DEFERRED REFRESH IMMEDIATE
IN REGIONTABLESPACE
REPLICATED;
```

Führen Sie zur Aktualisierung der replizierten MQT die folgende Anweisung aus:

```
REFRESH TABLE R_EMPLOYEE;
```

Anmerkung: Nach der Verwendung der Anweisung REFRESH sollten Sie RUNSTATS für die replizierte Tabelle in gleicher Weise wie für die anderen Tabellen ausführen.

Im folgenden Beispiel werden der Verkauf nach Mitarbeiter, die Summe für die Abteilung und die Gesamtsumme berechnet:

```

SELECT d.mgrno, e.empno, SUM(s.sales)
FROM   department AS d, employee AS e, sales AS s
WHERE  s.sales_person = e.lastname
      AND e.workdept = d.deptno
GROUP BY ROLLUP(d.mgrno, e.empno)
ORDER BY d.mgrno, e.empno;

```

Anstatt der Tabelle EMPLOYEE, die nur in einer Datenbankpartition ist, verwendet der Datenbankmanager die Tabelle R_EMPLOYEE, die in jede der Datenbankpartitionen repliziert wird, in der die Tabelle SALES gespeichert ist. Der Leistungsvorteil ergibt sich daraus, dass die Mitarbeiterinformationen zur Berechnung des Joins nicht zu jeder Datenbankpartition über das Netzwerk gesendet werden müssen.

Replizierte MQTs in zusammengefassten Joins

Replizierte MQTs können auch bei der Kollokation von Joins helfen. Wenn beispielsweise ein Sternschema eine große Fakttablette enthält, die sich über zwanzig Knoten erstreckt, sind die Joins zwischen der Fakttablette und den Dimensionstabellen am effizientesten, wenn diese Tabellen durch Kollokation zusammengefasst werden. Wenn sich alle Tabellen in derselben Datenbankpartitionsgruppe befinden, ist höchstens eine Dimensionstabelle korrekt für einen zusammengefassten Join (d. h. Join von durch Kollokation zusammengefassten Tabellen) partitioniert. Die anderen Dimensionstabellen können nicht in einem zusammengefassten Join verwendet werden, weil die Joinspalten der Fakttablette nicht dem Verteilungsschlüssel der Fakttablette entsprechen.

Betrachten Sie zum Beispiel eine Tabelle mit dem Namen FACT (C1, C2, C3, ...), die über Spalte C1 unterteilt ist, und eine Tabelle mit dem Namen DIM1 (C1, dim1a, dim1b, ...), die über C1 unterteilt ist, und eine Tabelle DIM2 (C2, dim2a, dim2b, ...), die über C2 unterteilt ist, usw.

In diesem Fall können Sie erkennen, dass der Join zwischen FACT und DIM1 perfekt ist, da das Vergleichselement DIM1.C1 = FACT.C1 durch Kollokation zusammengefasst vorliegt. Beide Tabellen werden über die Spalte C1 unterteilt.

Der Join mit DIM2 mit dem Vergleichselement WHERE DIM2.C2 = FACT.C2 kann jedoch nicht durch Kollokation zusammengefasst werden, da FACT über die Spalte C1 unterteilt ist und nicht über die Spalte C2. In diesem Fall kann es sinnvoll sein, die Tabelle DIM2 in der Datenbankpartitionsgruppe der Fakttablette zu replizieren, so dass der Join lokal in jeder Datenbankpartition erfolgen kann.

Anmerkung: Diese Erörterung replizierter MQTs bezieht sich auf datenbankinterne Replikation. Datenbankübergreifende Replikation erfordert Subskriptionen, Steuertabellen und Daten in verschiedenen Datenbanken und auf verschiedenen Betriebssystemen.

Wenn Sie eine replizierte MQT erstellen, kann die Quellentabelle eine Tabelle auf einem einzelnen Knoten oder eine Tabelle auf mehreren Knoten in einer Datenbankpartitionsgruppe sein. In den meisten Fällen ist die replizierte Tabelle klein und kann in einer Datenbankpartitionsgruppe mit einem Knoten untergebracht werden. Sie können die zu replizierende Datenmenge einschränken, indem Sie nur einen Teil der Spalten der Tabelle angeben, die Zeilenzahl mit Hilfe der verwendeten Vergleichselemente begrenzen oder beide Methoden anwenden. Für das Funktionieren replizierter MQTs ist die Option zur Datenerfassung nicht erforderlich.

Eine replizierte MQT kann auch in einer Datenbankpartitionsgruppe mit mehreren Knoten erstellt werden, so dass Kopien der Quellentabelle in allen Datenbankpartitionen erstellt werden. Joins zwischen einer großen Fakttable und den Dimensionstabellen finden in dieser Art von Umgebung mit größerer Wahrscheinlichkeit lokal statt, als wenn die Quellentabelle an alle Datenbankpartitionen per Broadcast übertragen werden muss.

Indizes für replizierte Tabellen werden nicht automatisch erstellt. Sie können Indizes erstellen, die sich von denen für die Quellentabelle unterscheiden. Zur Vermeidung von Verletzungen von Integritätsbedingungen, die für die Quellentabellen nicht vorhanden sind, können Sie jedoch keine eindeutigen Indizes erstellen oder Integritätsbedingungen für die replizierten Tabellen definieren. Integritätsbedingungen sind auch dann nicht zulässig, wenn die gleichen Integritätsbedingungen für die Quellentabelle gelten.

Auf replizierte Tabellen kann in einer Abfrage direkt verwiesen werden, jedoch können Sie nicht das Vergleichselement NODENUMBER() mit einer replizierten Tabelle verwenden, um die Tabellendaten in einer bestimmten Partition abzurufen.

Verwenden Sie die EXPLAIN-Einrichtung, um festzustellen, ob eine replizierte MQT von dem Zugriffsplan für eine Abfrage genutzt wurde. Ob der vom Optimierungsprogramm ausgewählte Zugriffsplan die replizierte MQT verwendet, hängt von den Informationen ab, die verknüpft werden müssen. Die replizierte MQT könnte zum Beispiel dann nicht verwendet werden, wenn das Optimierungsprogramm feststellt, dass es weniger aufwendig wäre, die ursprüngliche Quellentabelle an die anderen Datenbankpartitionen der Datenbankpartitionsgruppe per Broadcast rundzusenden.

Zugehörige Konzepte:

- „Joins“ auf Seite 220

Optimierungsstrategien für partitionsinterne Parallelität

Das Optimierungsprogramm kann einen Zugriffsplan wählen, der eine Abfrage parallel innerhalb einer Datenbankpartition ausführt, wenn ein Grad von Parallelität bei der Kompilierung der SQL-Anweisung angegeben wird.

Während der Ausführung werden mehrere Datenbankagenten, so genannte Subagenten, zur Ausführung der Abfrage erstellt. Die Anzahl von Subagenten ist kleiner oder gleich dem Grad von Parallelität, der bei der Kompilierung der SQL-Anweisung angegeben wurde.

Zur Parallelisierung unterteilt das Optimierungsprogramm den Zugriffsplan in Teile, die jeweils von einem Subagenten, sowie in einen Teil, der vom koordinierenden Agenten ausgeführt wird. Die Subagenten übergeben Daten über Tabellwarteschlangen an den koordinierenden Agenten oder andere Subagenten. In einer Umgebung mit partitionierten Datenbanken können Subagenten Daten an Subagenten in anderen Datenbankpartitionen senden oder von ihnen empfangen.

Strategien zur partitionsinternen Parallelsuche

Tabellensuchen und Indexsuchen können parallel in derselben Tabelle oder demselben Index ausgeführt werden. Für parallele Tabellensuchen wird die Tabelle in Seiten- oder Zeilenbereiche unterteilt. Je ein Bereich von Seiten oder Zeilen wird

einem Subagenten zugewiesen. Ein Subagent durchsucht den ihm zugewiesenen Bereich und erhält einen anderen Bereich zugewiesen, wenn er mit dem Durchsuchen des aktuellen Bereichs fertig ist.

Für parallele Indexsuchen wird der Index in Bereiche von Datensätzen entsprechend den Indexschlüsselwerten und der Anzahl von Indexeinträgen für einen Schlüsselwert unterteilt. Die parallele Indexsuche wird wie die parallele Tabellensuche mit Subagenten durchgeführt, denen jeweils ein Bereich von Datensätzen zugewiesen wird. Einem Subagenten wird ein neuer Bereich zugewiesen, wenn er die Suche im aktuellen Bereich beendet hat.

Das Optimierungsprogramm legt die Sucheinheit (entweder Seite oder Zeile) sowie die Suchgranularität fest.

Bei Parallelsuchen wird die Arbeit gleichmäßig unter den Subagenten verteilt. Der Zweck einer Parallelsuche besteht darin, die Belastung unter den Subagenten ausgewogen zu verteilen und sie gleichmäßig auszulasten. Wenn die Anzahl aktiver Subagenten gleich der Anzahl verfügbarer Prozessoren ist und die Platten nicht mit E/A-Anforderungen überlastet sind, werden die Ressourcen der Maschine effektiv genutzt.

Andere Zugriffsplanstrategien können eine unausgewogene Datenverteilung bei der Ausführung der Abfrage verursachen. Das Optimierungsprogramm wählt Parallelstrategien aus, die für eine ausgewogene Datenverteilung unter den Subagenten sorgen.

Strategien zur partitionsinternen parallelen Sortierung

Das Optimierungsprogramm kann eine der folgenden parallelen Sortierstrategien auswählen:

- **Reihumsortierung**

Dies kann auch als *Umverteilungssortieren* bezeichnet werden. Diese Methode nutzt den gemeinsamen Speicher effizient, um die Daten so gleichmäßig wie möglich auf alle Subagenten zu verteilen. Zur Realisierung der gleichmäßigen Verteilung wird eine Art Reihumverteilungsalgorithmus verwendet. Zunächst wird ein Sortiervorgang für jeden Subagenten erstellt. Während der Einfügephase fügen Subagenten Zeilen reihum in jeden der einzelnen Sortiervorgänge ein, um eine gleichmäßigere Datenverteilung zu erzielen.

- **Partitionierte Sortierung**

Dies ist dem reihumverteilten Sortieren insofern ähnlich, als dass für jeden Subagenten ein Sortiervorgang erstellt wird. Die Subagenten wenden eine Hashfunktion auf die Sortierspalten an, um festzulegen, in welchen Sortiervorgang eine Zeile einzufügen ist. Wenn beispielsweise die innere und die äußere Tabelle eines Mischjoins an einem partitionierten Sortiervorgang beteiligt sind, kann ein Subagent mit Hilfe eines Mischjoins die entsprechenden Teile verknüpfen und parallel ausgeführt werden.

- **Replizierte Sortierung**

Diese Art der Sortierung wird verwendet, wenn jeder Subagent die gesamte Ausgabe der Sortierung benötigt. Es wird nur ein Sortiervorgang erstellt. Beim Einfügen von Zeilen in den Sortiervorgang werden die Subagenten synchronisiert. Nach Beendigung der Sortierung liest jeder Subagent das gesamte Sortierringesult. Wenn die Anzahl von Zeilen gering ist, kann diese Art der Sortierung zum erneuten Ausgleich des Datenstroms verwendet werden.

- **Gemeinsame Sortierung**

Diese Art der Sortierung entspricht der replizierten Sortierung, abgesehen davon, dass die Subagenten eine parallele Suche über das Sortierergebnis öffnen, um die Daten unter den Subagenten ähnlich wie bei einer Reihumsortierung zu verteilen.

Partitionsinterne temporäre Paralleltabellen

Subagenten können kooperieren, um eine temporäre Tabelle durch Einfügen von Zeilen in dieselbe Tabelle zu erstellen. Eine solche Tabelle ist eine gemeinsame temporäre Tabelle. Die Subagenten können private oder parallele Suchoperationen über die gemeinsam genutzte temporäre Tabelle öffnen; dies ist davon abhängig, ob der Datenstrom repliziert oder geteilt werden muss.

Strategien zur partitionsinternen parallelen Spaltenberechnung

Spaltenberechnungen (Aggregation) können von Subagenten parallel durchgeführt werden. Eine Spaltenberechnung setzt voraus, dass die Daten nach den Gruppierungsspalten geordnet sind. Wenn ein Subagent sicher sein kann, dass er alle Zeilen für eine Reihe von Gruppierungsspaltenwerten erhält, kann er eine vollständige Spaltenberechnung (Aggregation) durchführen. Dies ist möglich, wenn der Strom bereits aufgrund einer früheren partitionierten Sortierung über die Gruppierungsspalten geteilt ist.

Andernfalls kann der Subagent eine partielle Spaltenberechnung durchführen und eine andere Strategie zur Vervollständigung der Spaltenberechnung anwenden. Einige dieser Strategien sind:

- Senden der teilweise berechneten Daten an den Koordinatoragenten über eine Tabellenwarteschlange für Mischjoins. Der Koordinatoragent vervollständigt die Spaltenberechnung.
- Einfügen der teilweise berechneten Daten in eine partitionierte Sortierung. Die Sortierung wird über die Gruppierungsspalten geteilt und stellt sicher, dass alle Zeilen für eine Reihe von Gruppierungsspalten in einer Sortierpartition enthalten sind.
- Wenn der Strom zum Ausgleichen der Verarbeitung repliziert werden muss, können die teilweise berechneten Daten in eine replizierte Sortierung eingefügt werden. Die einzelnen Subagenten vervollständigen die Spaltenberechnung über die replizierte Sortierung und erhalten eine identische Kopie des Ergebnisses der Spaltenberechnung.

Strategien zu partitionsinternen parallelen Joins

Joinoperationen können von Subagenten parallel durchgeführt werden. Parallele Joinsstrategien werden durch die Merkmale des Datenstroms festgelegt.

Ein Join kann parallelisiert werden, indem der Datenstrom nach der inneren und äußeren Tabelle des Joins partitioniert, repliziert oder beides wird. Zum Beispiel kann ein Join mit Verschachtelungsschleife parallelisiert werden, wenn der äußere Datenstrom für eine Parallelsuche partitioniert ist und der innere Datenstrom von jedem Subagenten unabhängig neu ausgewertet wird. Ein Mischjoin kann parallelisiert werden, wenn der innere und der äußere Datenstrom für partitionierte Sortierungen nach ihren Werten partitioniert sind.

Zugehörige Konzepte:

- „Optimierungsstrategien für MDC-Tabellen“ auf Seite 19

Statistische Sichten

Statistische Sichten geben dem Optimierungsprogramm die Möglichkeit, präzisere Kardinalitätsschätzwerte zu berechnen. Eine Kardinalitätsschätzung ist der Prozess, bei dem das Optimierungsprogramm mit Hilfe von Statistikdaten die Größe von Abfrageteilergebnissen nach der Anwendung von Vergleichselementen oder nach einer Spaltenberechnung bestimmt. Die Genauigkeit der Kardinalitätsschätzungen hängt von den Vergleichselementen und den verfügbaren Statistikdaten ab. Es sind Statistikdaten verfügbar, welche die Verteilung von Datenwerten innerhalb einer Spalte darstellen und dadurch die Kardinalitätsschätzung verbessern können, wenn die Datenwerte ungleichmäßig verteilt sind. Darüber hinaus sind Statistikdaten verfügbar, welche die Anzahl der unterschiedlichen Werte in einer Gruppe von Spalten angeben. Durch diese Informationen lassen sich bessere Kardinalitätsschätzungen erzielen, wenn zwischen Spalten eine statistische Korrelation besteht. Sehr häufig allerdings können diese Statistikdaten komplexere Beziehungen möglicherweise nicht darstellen. Dies gilt zum Beispiel für den Filtereffekt von Vergleichselementen oder von Spaltenberechnungen mit korrelierten und ungleichmäßig verteilten Attributen (z. B. *make* = 'Honda' AND *model* = 'Accord'), für Vergleiche mit Ausdrucksergebnissen (z. B. *price* > MSRP + Dealer_markup), für Beziehungen, die mehrere Tabellen mit einbeziehen (z. B. *product.name* = 'Alloy wheels' AND *product.key* = *sales.product_key*), sowie für alle anderen Vergleichselemente oder Spaltenberechnungen mit unabhängigen Attributen und einfachen Vergleichsoperationen.

Statistische Sichten sind Sichten mit zugehörigen Statistikdaten, die zur Verbesserung von Kardinalitätsschätzungen für Abfragen genutzt werden können, bei denen sich die Sichtdefinition mit der Abfragedefinition überschneidet. Dies ist insofern eine leistungsfähige Funktionalität, als dass sie das Optimierungsprogramm mit präzisen Statistikdaten ausstattet, die zur Bestimmung von Kardinalitätsschätzwerten für Abfragen mit komplexen Gruppen von (möglicherweise korrelierten) Vergleichselementen verwendet werden können, die sich auf eine oder auch mehrere Tabellen beziehen.

Eine statistische Sicht braucht in einer Abfrage, die sie optimiert, nicht direkt angegeben zu werden. In den meisten Fällen können die Statistikdaten der Sicht genutzt werden, wenn sich die Definition der Sicht mit der Definition der Abfrage überschneidet. Voraussetzung für die Nutzung dieser neuen Funktionalität ist, dass eine Sicht mit Hilfe der Anweisung ALTER VIEW für die Optimierung aktiviert und die Systemkatalogtabellen mit Statistikdaten für diese Sicht gefüllt werden.

Zugehörige Konzepte:

- „MQTs (MQT - Materialized Query Table, gespeicherte Abfragetabelle)“ auf Seite 11
- „Szenario: Verbessern der Kardinalitätsschätzung mit Hilfe statistischer Sichten“ auf Seite 214

Zugehörige Tasks:

- „Verwenden statistischer Sichten“ auf Seite 212

Zugehörige Referenzen:

- „RUNSTATS command“ in *Command Reference*

Verwenden statistischer Sichten

Eine Sicht muss für die Optimierung aktiviert werden, bevor die auf sie bezogenen Statistikdaten zur Optimierung einer Abfrage herangezogen werden können. Eine Sicht, die für die Optimierung aktiviert ist, wird als *statistische Sicht* bezeichnet. Eine Sicht, die keine statistische Sicht ist, wird als für die Optimierung inaktiviert bezeichnet. Der Begriff *reguläre Sicht* bezieht sich auf eine Sicht, die für die Optimierung inaktiviert ist. Nach der Erstellung ist eine Sicht für die Optimierung zunächst inaktiviert.

Voraussetzungen:

- Zur Aktivierung einer Sicht für die Optimierung müssen Sie das Zugriffsrecht ALTER sowohl für die Sicht als auch für die Tabelle besitzen, für die die Sicht definiert ist.
- Zum Aufrufen des Dienstprogramms RUNSTATS für eine Sicht benötigen Sie eine der folgenden Berechtigungen:
 - Berechtigung SYSADM
 - Berechtigung SYSCTRL
 - Berechtigung SYSMAINT
 - Berechtigung DBADM
 - Zugriffsrecht CONTROL für die Sicht

Darüber hinaus benötigen Sie die geeigneten Zugriffsrechte für den Zugriff auf Zeilen der Sicht. Insbesondere benötigen Sie für jede Tabelle, jede Sicht oder jeden Kurznamen, auf den in der Sichtdefinition verwiesen wird, eines der folgenden Zugriffsrechte (bzw. Berechtigungen):

- Berechtigung SYSADM
- Berechtigung DBADM
- Zugriffsrecht CONTROL
- Zugriffsrecht SELECT

Einschränkungen:

Eine Sicht kann nicht für die Optimierung aktiviert werden, wenn eine der folgenden Bedingungen zutrifft:

- Die Sicht verweist direkt oder indirekt auf eine MQT (Materialized Query Table, gespeicherte Abfragetabelle). (Eine MQT oder statistische Sicht kann auf eine statistische Sicht verweisen.)
- Es handelt sich um eine funktionsunfähige Sicht.
- Es handelt sich eine typisierte Sicht.
- Eine weitere Sichtänderung wird in der gleichen Anweisung ALTER VIEW vorgenommen.

Wenn auf die Definition einer Sicht, die zur Aktivierung der Optimierung geändert wird, eine der folgenden Bedingungen zutrifft, wird die Anweisung ALTER VIEW ENABLE OPTIMIZATION zwar erfolgreich mit einer Warnung ausgeführt, jedoch werden die Statistikdaten vom Optimierungsprogramm nicht genutzt:

- Sie enthält Spaltenberechnungen (Aggregation) oder DISTINCT-Operationen.
- Sie enthält Gruppenoperationen mit UNION, EXCEPT oder INTERSECT.
- Sie enthält skalare Aggregatfunktionen (OLAP-Funktionen).

Vorgehensweise:

1. Aktivieren Sie die Sicht für die Optimierung. Eine Sicht kann mit Hilfe der neuen Klausel `ENABLE OPTIMIZATION` in der Anweisung `ALTER VIEW` für die Optimierung aktiviert werden. Eine Sicht, die für die Optimierung aktiviert wurde, kann nachfolgend mit der Klausel `DISABLE OPTIMIZATION` in der Anweisung `ALTER VIEW` für die Optimierung inaktiviert werden. Zum Beispiel würden Sie zur Aktivierung der Sicht `meinesicht` für die Optimierung, die folgende Anweisung eingeben:

```
ALTER VIEW meinesicht ENABLE QUERY OPTIMIZATION
```

Für eine Sicht, die für die Optimierung aktiviert ist, enthält die Spalte `PROPERTY` des entsprechenden `SYSTABLES`-Eintrags den Wert 'Y' an Zeichenposition 13. Bei einer Sicht, die für die Optimierung inaktiviert ist, enthält die Spalte `PROPERTY` des entsprechenden `SYSTABLES`-Eintrags eine leere Stelle an Zeichenposition 13.

2. Führen Sie das Dienstprogramm `RUNSTATS` aus. Um zum Beispiel Statistikdaten für die Sicht `meinesicht` zu erfassen, geben Sie den folgenden Befehl ein:

```
RUNSTATS ON TABLE db2dba.meinesicht
```

Wenn Sie bei der Erfassung der Sichtstatistiken statistische Stichproben, einschließlich Verteilungstatistiken, für 10 % der Zeilen durch eine Stichprobenentnahme auf Zeilenebene erfassen möchten, geben Sie den folgenden Befehl ein:

```
RUNSTATS ON TABLE db2dba.meinesicht WITH DISTRIBUTION TABLESAMPLE BERNOULLI (10)
```

Anmerkung: Vor DB2 Version 9.1 bewirkte die Ausführung von `RUNSTATS` für eine statistische Sicht nur eine Vorbereitung der Katalogstatistiktabellen für manuelle Aktualisierungen und keine tatsächliche Erfassung von Statistikdaten. In DB2 Version 9.1 bewirkt die Ausführung von `RUNSTATS` für eine statistische Sicht die Erfassung von Statistikdaten. Dies bedeutet, dass das Dienstprogramm `RUNSTATS` möglicherweise wesentlich mehr Zeit zur Ausführung benötigt als früher, je nachdem, wie viel Daten von der Sicht zurückgegeben werden.

3. Aktualisierungen an Statistikdaten einer Sicht können zu Änderungen an den Plänen für Abfragen führen, deren Definition sich mit der Sichtdefinition überschneidet. Wenn solche Abfragen Teil statischer SQL-Pakete sind, müssen diese Pakete erneut gebunden (`REBIND`) werden, um die Vorteile der Pläne zu nutzen, die sich aus den neuen Statistiken ergeben.

Zugehörige Konzepte:

- „Statistische Sichten“ auf Seite 211
- „Anzeigen der für die Optimierung relevanten Statistikdaten“ auf Seite 213

Zugehörige Referenzen:

- „`RUNSTATS` command“ in *Command Reference*

Anzeigen der für die Optimierung relevanten Statistikdaten

Nur Statistikdaten, welche die Datenverteilung der Abfrage kennzeichnen, die eine statistische Sicht definiert, zum Beispiel `CARD` und `COLCARD`, werden vom Optimierungsprogramm für die Optimierung in Betracht gezogen. Die folgenden Statistikdaten, die sich auf Sichtdatensätze beziehen, können erfasst und durch das Optimierungsprogramm ausgewertet werden.

Tabellenstatistiken (SYSCAT.TABLES, SYSSTAT.TABLES)

- CARD - Die Anzahl von Zeilen im Ergebnis für eine Sicht

Spaltenstatistiken (SYSCAT.COLUMNS, SYSSTAT.COLUMNS)

- COLCARD - Die Anzahl unterschiedlicher Werte einer Spalte im Ergebnis für eine Sicht
- AVGCOLLEN - Durchschnittliche Länge von Spalten im Ergebnis für eine Sicht
- HIGH2KEY - Zweithöchster Wert einer Spalte im Ergebnis für eine Sicht
- LOW2KEY - Zweitniedrigster Wert einer Spalte im Ergebnis für eine Sicht
- NUMNULLS - Anzahl der NULL-Werte in einer Spalte des Ergebnisses für eine Sicht
- SUB_COUNT - Durchschnittliche Anzahl von Unterelementen in einer Spalte des Ergebnisses für eine Sicht
- SUB_DELIM_LENGTH - Durchschnittliche Länge der einzelnen Begrenzer, die die Subelemente voneinander trennen

Spaltenverteilungsstatistiken (SYSCAT.COLDIST, SYSSTAT.COLDIST)

- DISTCOUNT - Wenn TYPE den Wert Q hat, die Anzahl unterschiedlicher Werte, die kleiner oder gleich dem Statistikwert COLVALUE sind
- SEQNO - Die Stelle in der Häufigkeitsrangfolge einer Folgennummer, die als Hilfe zur eindeutigen Bestimmung der Zeile in der Tabelle verwendet werden kann
- COLVALUE - Datenwert, für den Statistikdaten zur Häufigkeit oder zu Quantilwerten erfasst werden
- VALCOUNT - Häufigkeit, mit der der Datenwert in der Spalte einer Sicht auftritt, oder bei Quantilwerten die Anzahl der Werte, die kleiner oder gleich dem Datenwert (COLVALUE) sind

Statistikdaten, die keine Datenverteilung beschreiben, wie NPAGES und FPAGES, können zwar erfasst werden, werden jedoch vom Optimierungsprogramm ignoriert.

Zugehörige Konzepte:

- „Katalogstatistiken“ auf Seite 319
- „Statistische Sichten“ auf Seite 211
- „Unterelementstatistiken“ auf Seite 340

Zugehörige Referenzen:

- „RUNSTATS command“ in *Command Reference*

Szenario: Verbessern der Kardinalitätsschätzung mit Hilfe statistischer Sichten

In einem Data Warehouse ändern sich die Informationen von Fakttabellen in der Regel recht dynamisch, während die Daten von Dimensionstabellen eher statischen Charakter haben. Daher können Attributdaten für Dimensionen häufig positiv oder negativ mit Attributdaten von Fakttabellen korreliert sein. Die gegenwärtig für das Optimierungsprogramm verfügbaren traditionellen Statistikdaten für Basistabellen geben dem Optimierungsprogramm keine Möglichkeit, tabellenübergreifende Beziehungen zu erkennen. Verteilungsstatistiken zu Spalten und Tabellen für Statistische Sichten (und MQTs) können dazu genutzt werden, die erforderlichen Informationen für das Optimierungsprogramm bereitzustellen, um derartige Kardinalitätsschätzfehler zu korrigieren.

Betrachten Sie die folgende Abfrage, die den Jahresumsatz für Golfschläger berechnet, die im Monat Juli jedes Jahres verkauft wurden:

```
SELECT sum(f.sales_price), d2.year
FROM product d1, period d2, daily_sales f
WHERE d1.prodkey = f.prodkey
      AND d2.perkey = f.perkey
      AND d1.item_desc = 'golf club'
      AND d2.month = 'JUL'
GROUP BY d2.year
```

Ein Abfrageausführungsplan mit Sternjoin kann eine gute Wahl für diese Abfrage sein, sofern das Optimierungsprogramm feststellen kann, ob die einfache Gleichheitsverknüpfung (Semi-Join) von PRODUCT und DAILY_SALES oder die einfache Gleichheitsverknüpfung von PERIOD und DAILY_SALES die höhere Selektivität erzielt. Um einen effizienten Sternjoinplan generieren zu können, muss das Optimierungsprogramm in der Lage sein, die selektivste Semi-Join-Verknüpfung für den äußeren Part der logischen Indexverknüpfungsoperation über AND auszuwählen.

Data Warehouses enthalten häufig Datensätze für Produkte, die nicht länger in den Geschäftsregalen verfügbar sind. Dies kann dazu führen, dass die Verteilung von PRODUCT-Spalten nach dem Join völlig anders aussieht als ihre Verteilung vor dem Join. Da das Optimierungsprogramm mangels besserer Informationen die Selektivität lokaler Vergleichselemente allein auf der Grundlage von Statistiken zu den Basistabellen bestimmt, schätzt es die Selektivität des Vergleichselements *item_desc* = 'golf club' möglicherweise viel zu hoch ein.

Wenn Golfschläger historisch betrachtet zum Beispiel 1 % der hergestellten Produkte darstellen, jedoch jetzt 20 % der aktuellen Umsätze ausmachen, würde das Optimierungsprogramm mit hoher Wahrscheinlichkeit die Selektivität des Vergleichselements *item_desc* = 'golf club' überschätzen, da keine Statistikdaten vorhanden sind, die die Verteilung von *item_desc* nach dem Join beschreiben. Und wenn außerdem die Umsätze in allen zwölf Monaten gleichermaßen wahrscheinlich sind, läge die Selektivität des Vergleichselements *month* = 'JUL' bei ca. 8 %. Daher würde der Schätzfehler für die Selektivität des Vergleichselements *item_desc* = 'golf club' das Optimierungsprogramm fälschlicherweise dazu veranlassen, die scheinbar selektivere Semi-Join-Operation zwischen PRODUCT und DAILY_SALES als äußeren Part der logischen AND-Verknüpfung der Indizes des Sternjoinplans auszuführen.

Im folgenden Beispiel wird die Bereitstellung statistischer Sichten zur Lösung dieses Typs von Problem schrittweise veranschaulicht.

Das folgende Beispiel zeigt eine Datenbank aus einem typischen Data Warehouse, in dem STORE, CUSTOMER, PRODUCT, PROMOTION und PERIOD die Dimensionstabellen sind und DAILY_SALES die Fakttable ist.

Tabelle 50. STORE (63 Zeilen)

Spalte	storekey	store_number	city	Status	district	...
Attribut	integer	char(2)	char(20)	char(5)	char(14)	...
	not null					
	Primärschlüssel					

Tabelle 51. CUSTOMER (1.000.000 Zeilen)

Spalte	custkey	name	address	age	gender	...
Attribut	integer	char(30)	char(40)	smallint	char(1)	...
	not null					
	Primärschlüssel					

Tabelle 52. PRODUCT (19.450 Zeilen)

Spalte	prodkey	category	item_desc	price	cost	...
Attribut	integer	integer	char(30)	decimal(11)	decimal(11)	...
	not null					
	Primärschlüssel					

Tabelle 53. PROMOTION (35 Zeilen)

Spalte	promokey	promotype	promodesc	promovalue	...
Attribut	integer	integer	char(30)	decimal(5)	...
	not null				
	Primärschlüssel				

Tabelle 54. PERIOD (2922 Zeilen)

Spalte	perkey	calendar_date	month	period	year	...
Attribut	integer	date	char(3)	smallint	smallint	...
	not null					
	Primärschlüssel					

Tabelle 55. DAILY_SALES (754.069.426 Zeilen)

Spalte	storekey	custkey	prodkey	promokey	perkey	sales_price	...
Attribut	integer	integer	integer	integer	integer	decimal(11)	...

Nehmen Sie an, die Unternehmensmanager möchten herausfinden, ob Kunden ein Produkt noch einmal kaufen, wenn ihnen ein Rabatt für das gleiche Produkt bei einem weiteren Besuch angeboten wird. Nehmen Sie außerdem an, dass die Studie nur für das Geschäft (STORE) '01' durchgeführt wird, das 18 Niederlassungen im Land hat. Tabelle 56 zeigt die verschiedenen Kategorien von Verkaufsförderungs-typen (Promotionstypen) mit den Prozentsätzen ihres jeweiligen Anteils an allen Verkaufsfördermaßnahmen.

Tabelle 56. PROMOTION (35 Zeilen)

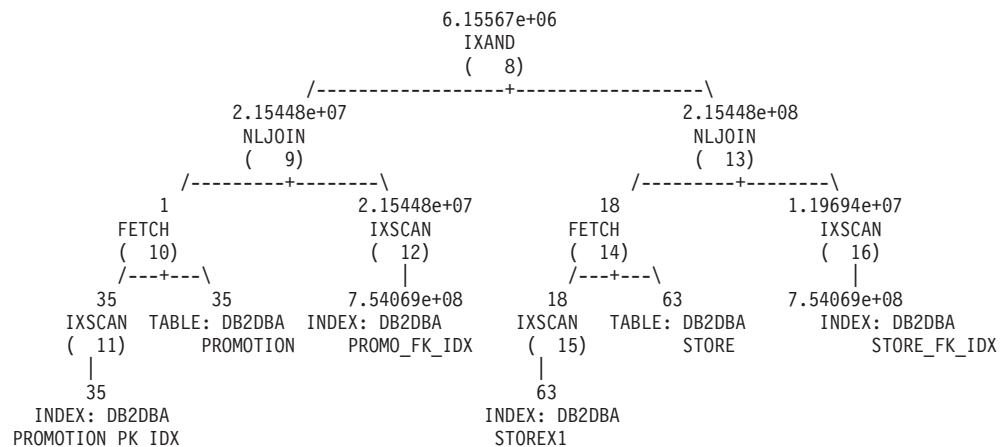
promotype	promodesc	COUNT(promotype)	percentage of promotions
1	Return customers	1	2.86%
2	Coupon	15	42.86%
3	Advertisement	5	14.29%
4	Manager's special	3	8.57%
5	Overstocked items	4	11.43%
6	End aisle display	7	20.00%

Die Tabelle zeigt, dass nur 2,86 % aller 35 Typen von angebotenen Verkaufsförderungsmaßnahmen Rabatte für wiederkehrende Kunden (Return customers) sind ($1 \div 35 \approx 0,0286$).

Nun wird die folgende Abfrage ausgeführt, die die Anzahl 12.889.514 zurückgibt:

```
SELECT count(*)
FROM store d1, promotion d2, daily_sales f
WHERE d1.storekey = f.storekey
      AND d2.promokey = f.promokey
      AND d1.store_number = '01'
      AND d2.promotype = 1
```

Diese Abfrage wird nach dem folgenden, vom Optimierungsprogramm generierten Plan ausgeführt. Bei jedem Knoten dieses Diagramms stellt der obere numerische Wert die Kardinalitätsschätzung, die zweite Zeile den Operatortyp und die Nummer in Klammern die Operator-ID dar.



Beim Join mit Verschachtelungsschleife (NLJOIN) Nummer 9 schätzt das Optimierungsprogramm, dass rund 2,86 % des Produktverkaufs auf Kunden zurückzuführen ist, die zurückkommen, um die gleichen Produkte zu einem reduzierten Preis zu kaufen ($2,15448e+07 \div 7,54069e+08 \approx 0,0286$). Beachten Sie, dass dies vor und nach dem Join der Tabelle PROMOTION mit der Tabelle DAILY_SALES der gleiche Prozentsatz ist. In Tabelle 57 sind die Kardinalitätsschätzwerte und ihre Prozentsätze (Filtereffekt) vor und nach dem Join zusammengefasst.

Tabelle 57. Geschätzte Kardinalitäten vor und nach dem Join mit der Tabelle DAILY_SALES

	Vor dem Join		Nach dem Join	
Vergleichselemente	Anzahl	Prozentsatz qualifizierter Zeilen	Anzahl	Prozentsatz qualifizierter Zeilen
store_number = '01'	18	28,57 %	2,15448e+08	28,57 %
promotype=1	1	2,86 %	2,15448e+07	2,86 %

Da die Wahrscheinlichkeit von *promotype* = 1 geringer ist als die von *store_number* = '01', wählt das Optimierungsprogramm die Semi-Join-Verknüpfung zwischen den Tabellen PROMOTION und DAILY_SALES als äußeren Teil der logischen AND-Verknüpfung von Indizes des Sternjoinplans. Dies führt zu einer geschätzten Anzahl von ca. 6.155.670 Produkten, die über den Verkaufsförderungstyp 1 verkauft wurden. Diese inkorrekte Kardinalitätsschätzung liegt um den Faktor 2,09 unter der tatsächlichen Anzahl ($12.889.514 \div 6.155.670 \approx 2,09$).

Was ist die Ursache dafür, dass das Optimierungsprogramm die Anzahl der durch die beiden Vergleichselemente qualifizierten Datensätze nur auf die Hälfte der tatsächlichen Anzahl schätzt? Das Geschäft '01' repräsentiert ca. 28,57 % aller Geschäfte. Was wäre, wenn andere Geschäfte mehr Umsätze hätten als Geschäft '01' (weniger als 28,57 %)? Oder wenn das Geschäft '01' tatsächlich die meisten Produktumsätze erzielt hätte (mehr als 28,57 %)? In ähnlicher Weise können auch die 2,86 % der durch Verkaufsförderungstyp 1 verkauften Produkte in Tabelle 57 auf Seite 217 irreführend sein. Der tatsächliche Prozentsatz in der Tabelle DAILY_SALES könnte sehr wohl ein anderer Wert sein als der projektierte.

Statistische Sichten können dem Optimierungsprogramm helfen, die Schätzwerte zu korrigieren und Über- bzw. Unterschätzungen zu verringern. Zunächst müssen zwei statistische Sichten erstellt werden, welche die beiden Semi-Join-Verknüpfungen in der obigen Abfrage darstellen. Die erste statistische Sicht stellt die Verteilungsdaten von Geschäften für alle täglichen Umsätze bereit. Die zweite statistische Sicht stellt die Verteilung von Verkaufsförderungstypen für alle täglichen Umsätze dar. Beachten Sie, dass diese statistischen Sichten die Verteilungsdaten für eine beliebige Geschäftsnummer bzw. einen beliebigen Verkaufsförderungstyp bereitstellen kann. In diesem Beispiel wird eine Stichprobenrate von 10 % verwendet, um die Datensätze aus der Tabelle DAILY_SALES für die jeweilige Sicht abzurufen und in globalen temporären Tabellen zu speichern. Anschließend werden diese Tabelle abgefragt, um die erforderlichen Statistikdaten zur Aktualisierung der beiden statistischen Sichten zu sammeln.

1. Erstellen Sie eine Sicht, die den Join der Tabelle STORE mit der Tabelle DAILY_SALES darstellt.

```
CREATE view sv_store_dailysales as
(SELECT s.*
 FROM store s, daily_sales ds
 WHERE s.storekey = ds.storekey)
```

2. Erstellen Sie eine Sicht, die den Join der Tabelle PROMOTION mit der Tabelle DAILY_SALES darstellt.

```
CREATE view sv_promotion_dailysales as
(SELECT p.*
 FROM promotion.p, daily_sales ds
 WHERE p.promokey = ds.promokey)
```

3. Machen Sie die Sichten zu statistischen Sichten, indem Sie die Abfrageoptimierung für sie aktivieren:

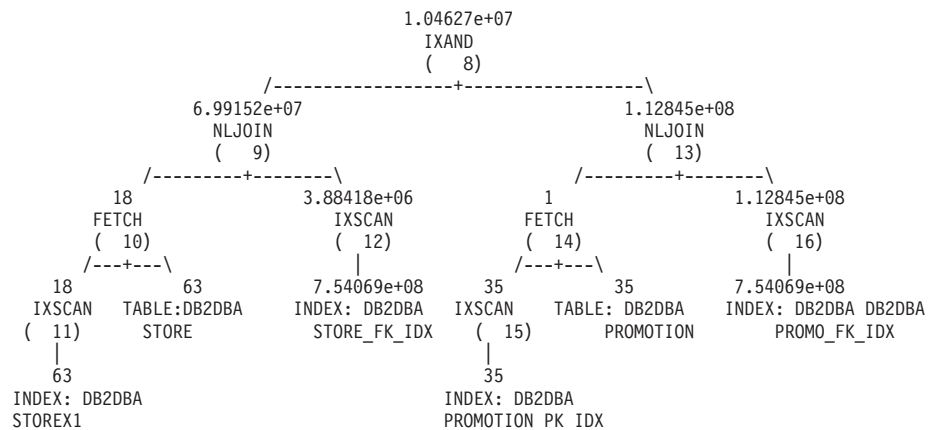
- ALTER VIEW sv_store_dailysales ENABLE QUERY OPTIMIZATION
- ALTER VIEW sv_promotion_dailysales ENABLE QUERY OPTIMIZATION

4. Führen Sie das Dienstprogramm RUNSTATS aus, um Statistikdaten für die Sichten zu erfassen:

- RUNSTATS on table db2dba.sv_store_dailysales WITH DISTRIBUTION
- RUNSTATS on table db2dba.sv_promotion_dailysales WITH DISTRIBUTION

5. Führen Sie die Abfrage erneut aus, damit sie erneut optimiert werden kann. Bei der Reoptimierung gleicht das Optimierungsprogramm die Sicht SV_STORE_DAILYSALES und die Sicht SV_PROMOTION_DAILYSALES mit der Abfrage ab und verwendet die Statistikdaten für die Sichten, um die Kardinalitätsschätzung der Semi-Join-Verknüpfungen zwischen der Fakttable und den Dimensionstabellen anzupassen. Dadurch wird es veranlasst, die Reihenfolge der Semi-Join-Verknüpfungen, die ohne diese Statistiken ausgewählt wurden, umzukehren.

Der neue Plan sieht folgendermaßen aus:



Beachten Sie, dass jetzt die Semi-Join-Verknüpfung zwischen den Tabellen STORE und DAILY_SALES im äußeren Part des Plans zur logischen Verknüpfung der Indizes über AND ausgeführt wird. Dies ist darauf zurückzuführen, dass die beiden statistischen Sichten dem Optimierungsprogramm im Wesentlichen mitteilen, dass das Vergleichselement 'store_number = '01'' mehr Zeilen herausfiltert als das Vergleichselement 'promotype = 1'. In Tabelle 58 sind die Schätzungen zur Kardinalität und die Prozentsätze (Filtereffekt) zusammengefasst, die vor und nach dem Join für die einzelnen Semi-Join-Verknüpfungen ermittelt wurden.

Beachten Sie, dass jetzt die Semi-Join-Verknüpfung zwischen den Tabellen STORE und DAILY_SALES im äußeren Part des Plans zur logischen Verknüpfung der Indizes über AND ausgeführt wird. Dies ist darauf zurückzuführen, dass die beiden statistischen Sichten dem Optimierungsprogramm im Wesentlichen mitteilen, dass das Vergleichselement 'store_number = '01'' mehr Zeilen herausfiltert als das Vergleichselement 'promotype = 1'. Hier schätzt das Optimierungsprogramm die Anzahl der verkauften Produkte auf ca. 10.462.700. Dieser Schätzwert liegt um den Faktor 1,23 ($12.889.514 \div 10.462.700 \approx 1,23$) unter der tatsächlichen Anzahl, was eine erhebliche Verbesserung gegenüber dem Schätzwert in Tabelle 57 auf Seite 217 darstellt. In Tabelle 58 sind die Kardinalitätsschätzwerte und ihre Prozentsätze (der Filtereffekt) für jedes Vergleichselement vor und nach dem Join zusammengefasst.

Tabelle 58. Geschätzte Kardinalitäten vor und nach dem Join mit der Tabelle DAILY_SALES

Vergleichs- elemente	Vor dem Join		Nach dem Join <i>Ohne statistische Sichten</i>		Nach dem Join <i>Mit statistischen Sichten</i>	
	Anzahl	Prozentsatz qualifizierter Zeilen	Anzahl	Prozentsatz qualifizierter Zeilen	Anzahl	Prozentsatz qualifizierter Zeilen
store_number = '01'	18	28,57 %	2,15448e+08	28,57 %	6,99152e+07	9,27 %
promotype=1	1	2,86 %	2,15448e+07	2,86 %	1,12845e+08	14,96 %

Jetzt schätzt das Optimierungsprogramm die Anzahl der verkauften Produkte auf ca. 10.462.700. Dieser Schätzwert weicht um den Faktor 1,23 von der tatsächlichen Anzahl ab, was eine erhebliche Verbesserung gegenüber dem Schätzwert ohne statistische Sichten ist.

Zugehörige Konzepte:

- „Erfassen von Statistiken an einer Stichprobe der Tabellendaten“ auf Seite 329

- „Statistische Sichten“ auf Seite 211
- „Strategien zur Auswahl optimaler Joins“ auf Seite 224

Zugehörige Tasks:

- „Erfassen von Katalogstatistiken“ auf Seite 322
- „Verwenden statistischer Sichten“ auf Seite 212

Zugehörige Referenzen:

- „RUNSTATS command“ in *Command Reference*

Joins

Joins

Ein *Join* ist der Prozess der Kombination von Informationen aus mindestens zwei Tabellen auf der Grundlage eines gemeinsamen Geltungsbereichs der Informationen. Zeilen aus der einen Tabelle werden mit Zeilen aus einer anderen Tabelle paarweise verbunden, wenn die Informationen in den entsprechenden Zeilen die Joinbedingung erfüllen.

Betrachten Sie zum Beispiel die beiden folgenden Tabellen:

Tabelle1		Tabelle2	
PROJ	PROJ_ID	PROJ_ID	NAME
A	1	1	Sam
B	2	3	Joe
C	3	4	Mary
D	4	1	Sue
		2	Mike

Für den Join von Tabelle1 und Tabelle2 an den Stellen, an denen die ID-Spalten die gleichen Werte haben, verwenden Sie die folgende SQL-Anweisung:

```
SELECT PROJ, x.PROJ_ID, NAME
FROM TABELLE1 x, TABELLE2 y
WHERE x.PROJ_ID = y.PROJ_ID
```

Diese Anweisung liefert die folgende Menge von Ergebniszeilen:

PROJ	PROJ_ID	NAME
A	1	Sam
A	1	Sue
B	2	Mike
C	3	Joe
D	4	Mary

Abhängig davon, ob ein Joinvergleichselement vorhanden ist und welche Verarbeitungsaufwände anhand der Tabellen- und Indexstatistiken ermittelt werden, wählt das Optimierungsprogramm eine der folgenden Joinmethoden aus:

- Join mit Verschachtelungsschleife (Nested-Loop Join)
- Mischjoin (Merge Join)
- Hash-Join

Beim Join zweier Tabellen wird die eine Tabelle als äußere Tabelle und die andere als innere Tabelle ausgewählt. Auf die äußere Tabelle wird zuerst zugegriffen, und sie wird nur einmal durchsucht. Ob die innere Tabelle mehrere Male durchsucht wird, hängt von der Art des Joins und den vorhandenen Indizes ab. Auch wenn in einer Abfrage mehr als zwei Tabellen verknüpft werden, verknüpft das Optimierungsprogramm jeweils nur zwei Tabellen gleichzeitig. Falls erforderlich, werden temporäre Tabellen zum Speichern von Zwischenergebnissen erstellt.

Sie können explizite Joinoperatoren wie INNER oder LEFT OUTER JOIN angeben, um festzulegen, wie Tabellen im Join verwendet werden. Bevor Sie jedoch eine Abfrage auf diese Art ändern, sollten Sie das Optimierungsprogramm ermitteln lassen, wie die Tabellen zu verknüpfen sind. Anschließend analysieren Sie die Abfrageleistung und entscheiden, ob Joinoperatoren hinzugefügt werden sollten.

Zugehörige Konzepte:

- „Joininformationen“ auf Seite 728
- „Joinmethoden“ auf Seite 221
- „Joinmethoden in Umgebungen mit partitionierten Datenbanken“ auf Seite 231
- „Joinstrategien in partitionierten Datenbanken“ auf Seite 229

Joinmethoden

Das Optimierungsprogramm kann eine von drei Grundstrategien wählen, wenn Abfragen einen Join von Tabellen erfordern.

- Join mit Verschachtelungsschleife (Nested-Loop Join)
- Mischjoin (Merge Join)
- Hash-Join

Diese Methoden werden in den folgenden Abschnitten beschrieben.

Join mit Verschachtelungsschleife

Ein Join mit Verschachtelungsschleife (Nested-Loop Join) wird auf eine der beiden folgenden Arten ausgeführt:

- Durchsuchen der inneren Tabelle für jede Zeile der äußeren Tabelle, auf die zugegriffen wird

Betrachten Sie zum Beispiel die Spalte A in den Tabellen T1 und T2 mit folgenden Werten:

Äußere Tabelle T1: Spalte A	Innere Tabelle T2: Spalte A
2	3
3	2
3	2
	3
	1

Zur Durchführung eines Joins mit Verschachtelungsschleife geht der Datenbankmanager in folgenden Schritten vor:

1. Lesen der ersten Zeile aus T1. Der Wert für A ist „2“.
2. Durchsuchen von T2, bis ein übereinstimmender Wert („2“) gefunden wird, und anschließender Join der beiden Zeilen.
3. Durchsuchen von T2, bis der nächste übereinstimmende Wert („2“) gefunden wird, und anschließender Join der beiden Zeilen.

4. Durchsuchen von T2 bis zum Ende der Tabelle.
 5. Zurückkehren zu T1 und Lesen der nächsten Zeile („3“).
 6. Durchsuchen von T2 von der ersten Zeile an, bis ein übereinstimmender Wert („3“) gefunden wird, und anschließender Join der beiden Zeilen.
 7. Durchsuchen von T2, bis der nächste übereinstimmende Wert („3“) gefunden wird, und anschließender Join der beiden Zeilen.
 8. Durchsuchen von T2 bis zum Ende der Tabelle.
 9. Zurückkehren zu T1 und Lesen der nächsten Zeile („3“).
 10. Durchsuchen von T2 wie vorher und Join aller übereinstimmenden Zeilen („3“).
- Durchführen einer Indexsuche für die innere Tabelle für jede Zeile der äußeren Tabelle, auf die zugegriffen wird

Diese Methode kann für die angegebenen Vergleichselemente verwendet werden, wenn es ein Vergleichselement der folgenden Form gibt:

```
ausdr(äußere_tabelle.spalte) relop innere_tabelle.spalte
```

Dabei ist relop ein relativer Operator (z. B. =, >, >=, < oder <=) und ausdr ist ein gültiger Ausdruck für die äußere Tabelle. Betrachten Sie die folgenden Beispiele:

```
OUTER.C1 + OUTER.C2 <= INNER.C1
```

```
OUTER.C4 < INNER.C3
```

Diese Methode könnte die Anzahl der Zeilen, auf die in der inneren Tabelle für jeden Zugriff auf die äußere Tabelle zugegriffen wird, wesentlich verringern, obwohl dies von einer Reihe von Faktoren abhängig ist, zu denen auch die Selektivität des Joinvergleichselements zählt.

Bei der Beurteilung eines Joins mit Verschachtelungsschleife entscheidet das Optimierungsprogramm auch, ob die äußere Tabelle sortiert wird oder nicht, bevor der Join durchgeführt wird. Durch Sortieren der äußeren Tabelle nach den Werten der Joinspalten kann die Anzahl der Leseoperationen zum Zugriff auf Seiten auf der Platte für die innere Tabelle verringert werden, da es wahrscheinlicher wird, dass sich die Seiten bereits im Pufferpool befinden. Wenn der Join einen Index mit einem hohen Grad der Clusterbildung verwendet, um auf die innere Tabelle zuzugreifen, und wenn die äußere Tabelle sortiert wurde, kann die Anzahl von Indexseiten, auf die zugegriffen wird, minimiert werden.

Wenn darüber hinaus das Optimierungsprogramm erwartet, dass durch den Join eine spätere Sortierung aufwendiger wird, kann es entscheiden, die Sortierung vor dem Join durchzuführen. Eine spätere Sortierung könnte erforderlich sein, um die Klauseln GROUP BY, DISTINCT, ORDER BY oder einen Mischjoin zu unterstützen.

Mischjoin

Ein Mischjoin (engl. Merge Join, Merge Scan Join oder Sort Merge Join) erfordert ein Vergleichselement der Form `tabelle1.spalte = tabelle2.spalte`. Ein solches Vergleichselement wird als *Equijoin-Vergleichselement* bezeichnet. Ein Mischjoin erfordert entweder über einen Indexzugriff oder durch eine Sortierung geordnete Eingaben für die Joinspalten. Ein Mischjoin kann nicht verwendet werden, wenn die Joinspalte eine LONG-Feldspalte oder eine LOB-Spalte ist.

Bei einem Mischjoin werden die Tabellen, die verknüpft werden, gleichzeitig durchsucht. Die äußere Tabelle des Mischjoins wird nur einmal durchsucht. Die innere Tabelle wird auch nur einmal durchsucht, sofern in der äußeren Tabelle keine sich wiederholenden Werte auftreten. Wenn wiederholte Werte auftreten,

wird eine Gruppe von Zeilen der inneren Tabelle eventuell noch einmal durchsucht. Betrachten Sie zum Beispiel die Spalte A in den Tabellen T1 und T2 mit folgenden Werten:

Äußere Tabelle T1: Spalte A

2
3
3

Innere Tabelle T2: Spalte A

1
2
2
3
3

Zur Durchführung eines Mischjoins geht der Datenbankmanager in folgenden Schritten vor:

1. Lesen der ersten Zeile aus T1. Der Wert für A ist „2“.
2. Durchsuchen von T2, bis ein übereinstimmender Wert gefunden wird, und anschließender Join der beiden Zeilen.
3. Fortsetzen des Durchsuchens von T2, solange die Spalten übereinstimmen, und dabei Join der Zeilen.
4. Wenn der Wert „3“ in T2 gelesen wird, Zurückkehren zu T1 und Lesen der nächsten Zeile.
5. Der nächste Wert in T1 ist „3“, der mit T2 übereinstimmt, also Join der Spalten.
6. Fortsetzen des Durchsuchens von T2, solange die Spalten übereinstimmen, und dabei Join der Zeilen.
7. Erreichen des Endes von T2.
8. Zurückkehren zu T1, um die nächste Zeile zu lesen. Beachten Sie, dass der nächste Wert in T1 derselbe ist wie der vorige Wert aus T1, so dass T2 noch einmal, angefangen vom ersten Wert „3“ in T2, durchsucht wird. Der Datenbankmanager speichert diese Position.

Hash-Join

Eine Hash-Join erfordert ein oder mehrere Vergleichselemente der Form $\text{tabelle1.spalteX} = \text{tabelle2.spalteY}$, wobei die Spaltentypen übereinstimmen müssen. Spalten des Typs CHAR müssen die gleiche Länge aufweisen. Bei Spalten des Typs DECIMAL muss die Genauigkeit und die Anzahl der Kommastellen übereinstimmen. Der Spaltentyp kann keine LONG-Feldspalte oder LOB-Spalte sein.

Zunächst wird die als INNER designierte Tabelle (innere Tabelle) durchsucht, wobei die Zeilen in Speicherpuffer kopiert werden, die dem Sortierzwischenspeicher entnommen werden, der wiederum durch den Datenbankkonfigurationsparameter *sorthheap* definiert ist. Die Speicherpuffer werden auf der Grundlage eines Hashwerts, der auf den Spalten der Joinvergleichselemente berechnet wird, in Abschnitte unterteilt. Wenn die Größe der inneren Tabelle (INNER) die Größe des Sortierspeichers überschreitet, werden Puffer aus ausgewählten Abschnitten in temporäre Tabellen geschrieben.

Wenn die innere Tabelle verarbeitet wurde, wird die zweite, als OUTER designierte Tabelle (äußere Tabelle) durchsucht und ihre Zeilen werden mit den Zeilen aus der inneren Tabelle (INNER) abgeglichen, indem zuerst der Hashwert, der für die Spalten der Joinvergleichselemente errechnet wurde, verglichen wird. Wenn der Hashwert für die OUTER-Zeilenspalte mit dem Hashwert der INNER-Zeilenspalte übereinstimmt, werden die tatsächlichen Werte der Joinelementspalten verglichen.

Zeilen der äußeren Tabelle (OUTER), die Teilen der Tabelle entsprechen, die nicht in eine temporäre Tabelle geschrieben wurden, werden sofort mit den Zeilen der inneren Tabelle im Speicher abgeglichen. Wenn der entsprechende Teil der inneren Tabelle in eine temporäre Tabelle geschrieben wurde, wird die OUTER-Zeile ebenfalls in eine temporäre Tabelle geschrieben. Schließlich werden abgeglichene Paare von Tabellenteilen aus den temporären Tabellen gelesen, die Hashwerte ihrer Zeilen abgeglichen und die Joinvergleichselemente geprüft.

Zur Erzielung der vollen Leistungsvorteile des Hash-Joins müssen Sie möglicherweise den Wert des Konfigurationsparameters *sortheap* der Datenbank und des Konfigurationsparameters *sheapthres* des Datenbankmanagers ändern.

Zur Erzielung der vollen Leistungsvorteile des Hash-Joins müssen Sie möglicherweise den Wert des Konfigurationsparameters *sortheap* der Datenbank und des Konfigurationsparameters *sheapthres* des Datenbankmanagers ändern.

Die Leistung des Hash-Joins ist am besten, wenn Sie Hashschleifen und Überläufe auf den Plattenspeicher vermeiden können. Zur Optimierung der Leistung von Hash-Joins schätzen Sie die maximale Speichergröße ab, die für *sheapthres* verfügbar ist, und optimieren dann den Parameter **sortheap**. Erhöhen Sie den Wert dieses Parameters, bis Sie möglichst viele Hashschleifen und Überläufe auf den Plattenspeicher vermeiden, jedoch nicht den durch den Parameter *sheapthres* definierten Grenzwert erreichen.

Die Erhöhung des Werts für *sortheap* sollte außerdem die Leistung von Abfragen verbessern, die mehrere Sortierungen erfordern.

Zugehörige Konzepte:

- „Joininformationen“ auf Seite 728
- „Joinmethoden in Umgebungen mit partitionierten Datenbanken“ auf Seite 231
- „Joinstrategien in partitionierten Datenbanken“ auf Seite 229
- „Joins“ auf Seite 220

Zugehörige Referenzen:

- „sortheap - Zwischenspeichergröße für Sortierlisten “ auf Seite 458
- „sheapthres - Schwellenwert für Sortierspeicher “ auf Seite 456

Strategien zur Auswahl optimaler Joins

Das Optimierungsprogramm nutzt verschiedene Methoden zur Auswahl einer optimalen Joinstrategie für eine Abfrage. Zu diesen Methoden zählen die folgenden Suchstrategien, die durch die Optimierungsklasse der Abfrage bestimmt werden:

- Schnelle Joinaufzählung (Greedy Join Enumeration)
 - Effizient im Hinblick auf Speicherbedarf und Zeit.
 - Aufzählung in einer Richtung, d. h., wenn eine Joinmethode für zwei Tabellen ausgewählt ist, wird sie im Laufe der weiteren Optimierung nicht mehr geändert.
 - Eventuell wird nicht der optimale Zugriffsplan gewählt, wenn viele Tabellen verknüpft werden. Wenn eine Abfrage nur zwei oder drei Tabellen verknüpft, ist der Zugriffsplan, der durch die schnelle Joinaufzählung ausgewählt wird, derselbe wie der Zugriffsplan, der durch die dynamisch programmierte Joinaufzählung (Dynamic Programming Join Enumeration) ausgewählt wird. Dies

gilt insbesondere dann, wenn die Abfrage viele entweder explizit angegebene oder implizit durch die Anwendung der Transitivität generierte Joinvergleichselemente für dieselbe Spalte hat.

- Dynamisch programmierte Joinaufzählung (Dynamic Programming Join Enumeration)
 - Der Bedarf an Speicherplatz und Zeit wächst exponentiell mit zunehmender Anzahl der verknüpften Tabellen.
 - Effiziente und erschöpfende Suche nach dem besten Zugriffsplan.
 - Ähnlich der Strategie, die von DB2 für OS/390 oder z/OS verwendet wird.

Der Algorithmus für die Joinaufzählung spielt die entscheidende Rolle bei der Bestimmung der Anzahl von Zugriffsplankombinationen, die das Optimierungsprogramm prüft.

Sternschemajoins

Die Tabellen, auf die in einer Abfrage verwiesen wird, sind fast immer durch Joinvergleichselemente miteinander verbunden. Wenn zwei Tabellen ohne Joinvergleichselement verknüpft werden, wird das kartesische Produkt der beiden Tabellen gebildet. Bei einem kartesischen Produkt wird jede ausgewählte Zeile der ersten Tabelle mit jeder ausgewählten Zeile der zweiten Tabelle verknüpft. Das Ergebnis ist eine Tabelle, die aus dem Kreuzprodukt in der Größe der beiden Tabellen besteht und in der Regel sehr groß ist. Da ein solcher Plan wahrscheinlich keine gute Leistung zulässt, vermeidet das Optimierungsprogramm sogar die Aufwandsabschätzung für einen Plan dieser Art.

Die einzigen Ausnahmen bilden die Fälle, in denen die Optimierungsklasse auf 9 gesetzt wird oder der Sonderfall eines Sternschemas vorliegt. Ein *Sternschema* (engl. star schema) enthält eine zentrale Tabelle, die als Fakttablette bezeichnet wird, und andere Tabellen, die als Dimensionstabellen bezeichnet werden. Die Dimensionstabellen haben, ungeachtet der Abfrage, jeweils nur einen einzigen Join, über den sie mit der Fakttablette verbunden sind. Jede Dimensionstabelle enthält zusätzliche Werte, die die Informationen über eine bestimmte Spalte in der Fakttablette erweitern. Eine typische Abfrage besteht aus mehreren lokalen Vergleichselementen, die auf Werte in den Dimensionstabellen verweisen, und enthält Joinvergleichselemente, die die Dimensionstabellen mit der Fakttablette verbinden. Für solche Abfragen kann es vorteilhaft sein, das kartesische Produkt mehrerer kleiner Dimensionstabellen zu berechnen und erst anschließend auf die umfangreiche Fakttablette zuzugreifen. Diese Technik ist dann von Vorteil, wenn mehrere Joinvergleichselemente einem mehrspaltigen Index entsprechen.

DB2 kann Abfragen erkennen, die für Datenbanken durchgeführt werden, die mit Sternschemata aufgebaut sind und über mindestens zwei Dimensionstabellen verfügen; außerdem kann es den Suchbereich vergrößern, um mögliche Pläne zur Berechnung des kartesischen Produkts von Dimensionstabellen zu berücksichtigen. Wenn der Plan mit den kartesischen Produkten den niedrigsten geschätzten Aufwand verursacht, wird er vom Optimierungsprogramm ausgewählt.

Die oben behandelte Strategie des Sternschemajoins basiert auf der Annahme, dass Primärschlüsselindizes im Join verwendet werden. Eine andere Situation liegt vor, wenn Fremdschlüsselindizes verwendet werden. Wenn die Fremdschlüsselspalten in der Fakttablette einspaltige Indizes sind und es eine relativ hohe Selektivität über alle Dimensionstabellen hinweg gibt, könnte die folgende Methode des Sternjoins zur Anwendung kommen:

1. Verarbeiten jeder Dimensionstabelle wie folgt:

- Durchführen eines Semi-Joins zwischen der Dimensionstabelle und dem Fremdschlüsselindex für die Fakttable
 - Dynamisches Erstellen einer Bitzuordnung durch ein Hashverfahren für die Werte der Zeilen-IDs (RID)
2. Anwenden von AND-Vergleichselementen auf die vorige Bitzuordnung für jede Bitzuordnung
 3. Feststellen der verbliebenen RIDs, nachdem die letzte Bitzuordnung verarbeitet wurde
 4. Optionales Sortieren dieser RIDs
 5. Abrufen einer Zeile aus der Basistabelle
 6. Erneuter Join der Fakttable mit jeder der zugehörigen Dimensionstabellen, dabei Zugreifen auf die Spalten in Dimensionstabellen, die für die SELECT-Klausel benötigt werden
 7. Erneutes Anwenden der Restvergleichselemente

Diese Methode erfordert keine mehrspaltigen Indizes. Explizite referenzielle Integritätsbedingungen zwischen Fakttable und Dimensionstabellen sind nicht erforderlich, obwohl die Beziehung zwischen Fakttable und Dimensionstabellen doch auf diese Weise realisiert werden sollte.

Die Bitzuordnungen, die von Sternjointechniken dynamisch erstellt und verwendet werden, erfordern Sortierspeicher, dessen Größe durch den Datenbankkonfigurationsparameter *sortheap* definiert wird.

Zusammengesetzte Tabellen

Wenn das Ergebnis des Joins zweier Tabellen eine neue Tabelle ist, die in diesem Fall als *zusammengesetzte* Tabelle bezeichnet wird, wird diese Tabelle in der Regel zur äußeren Tabelle eines weiteren Joins mit einer anderen inneren Tabelle. Ein solcher Join wird als „Composite-Outer-Join“ (zusammengesetzter Outer Join) bezeichnet. In einigen Fällen, besonders bei Verwendung der schnellen Joinaufzählung (Greedy Join Enumeration), ist es sinnvoll, das Ergebnis des Joins zweier Tabellen zur inneren Tabelle eines späteren Joins zu machen. Wenn die innere Tabelle eines Joins aus dem Ergebnis des Joins zweier oder mehrerer Tabellen besteht, wird ein solcher Plan als „Composite-Inner-Join“ (zusammengesetzter Inner Join) bezeichnet. Betrachten Sie zum Beispiel die folgende Abfrage:

```
SELECT COUNT(*)
FROM T1, T2, T3, T4
WHERE T1.A = T2.A AND T3.A = T4.A AND T2.Z = T3.Z
```

Hier könnte es von Vorteil sein, die Tabellen T1 und T2 zu verknüpfen (T1xT2), anschließend die Tabellen T3 und T4 zu verknüpfen (T3xT4) und schließlich das Ergebnis des ersten Joins als äußere Tabelle und das Ergebnis des zweiten Joins als innere Tabelle auszuwählen. Im daraus resultierenden Plan ((T1xT2) x (T3xT4)) ist das Ergebnis des Joins (T3xT4) eine zusammengesetzte innere Tabelle. Abhängig von der Abfrageoptimierungsklasse belegt das Optimierungsprogramm die maximale Anzahl von Tabellen, die als innere Tabelle eines Joins verwendet werden können, mit unterschiedlichen Einschränkungen. Composite-Inner-Joins sind bei den Optimierungsklassen 5, 7 und 9 zulässig.

Zugehörige Konzepte:

- „Joinmethoden“ auf Seite 221
- „Joinmethoden in Umgebungen mit partitionierten Datenbanken“ auf Seite 231
- „Joinstrategien in partitionierten Datenbanken“ auf Seite 229
- „Joins“ auf Seite 220

Spaltenkorrelation für mehrere Vergleichselemente

Ihre Anwendungen enthalten möglicherweise Abfragen, die mit Joins so konstruiert sind, dass zwei Tabellen über mehr als ein Joinvergleichselement verknüpft werden. Dies ist nicht ungewöhnlich, wenn Abfragen die Beziehungen zwischen ähnlichen zusammengehörigen Spalten in verschiedenen Tabellen bestimmen müssen.

Betrachten Sie zum Beispiel einen Hersteller, der Produkte aus Rohmaterialien verschiedener Farben, Elastizitäten und Qualitäten produziert. Das fertige Produkt hat dieselbe Farbe und Elastizität wie das Rohmaterial, aus dem es hergestellt ist. Der Hersteller führt die folgende Abfrage aus:

```
SELECT PRODUCT.NAME, RAWMATERIAL.QUALITY FROM PRODUCT, RAWMATERIAL
WHERE PRODUCT.COLOR      = RAWMATERIAL.COLOR
AND PRODUCT.ELASTICITY   = RAWMATERIAL.ELASTICITY
```

Diese Abfrage liefert die Namen und die Qualität der Rohmaterialien aller Produkte. Zwei Vergleichselemente werden für den Join verwendet:

```
PRODUCT.COLOR      = RAWMATERIAL.COLOR
PRODUCT.ELASTICITY = RAWMATERIAL.ELASTICITY
```

Wenn das Optimierungsprogramm einen Plan zur Ausführung dieser Abfrage auswählt, berechnet es die Selektivität jedes der beiden Vergleichselemente. Dabei geht es davon aus, dass sie unabhängig sind, das heißt, dass alle Variationen von Elastizität für jede Farbe vorkommen und umgekehrt für jede Elastizitätsstufe Rohmaterial in jeder Farbe verfügbar ist. Dann schätzt es die Gesamtselektivität des Vergleichselementpaares ab, indem es auf Katalogstatistikdaten für jede Tabelle auf der Basis der Anzahl der Elastizitätsstufen und der Anzahl verschiedener Farben zurückgreift. Ausgehend von diesem Schätzwert kann es beispielsweise einen Join mit Verschachtelungsschleife (Nested Loop Join) einem Mischjoin (Merge Join) vorziehen oder umgekehrt.

Es kann jedoch vorkommen, dass diese beiden Vergleichselemente nicht unabhängig sind. Zum Beispiel könnte es der Fall sein, dass die hochelastischen Materialien nur in wenigen Farben verfügbar sind und die sehr wenig elastischen Materialien nur in einigen anderen Farben verfügbar sind, die sich von den Farben der elastischen Materialien unterscheiden. In diesem Fall eliminiert die kombinierte Selektivität der Vergleichselemente weniger Zeilen, so dass die Abfrage mehr Zeilen liefert. Betrachten Sie den extremen Fall, dass es nur eine Stufe von Elastizität für jede Farbe gibt und umgekehrt. Hier könnte jedes der beiden Vergleichselemente logisch ganz weggelassen werden, da es vom jeweils anderen impliziert wird. Das Optimierungsprogramm wählt vielleicht nicht mehr den besten Plan. Zum Beispiel könnte es einen Plan mit Joins mit Verschachtelungsschleife wählen, obwohl ein Mischjoin schneller wäre.

Das DB2-Optimierungsprogramm versucht, eine Korrelation von Joinvergleichselementen zu ermitteln und zu kompensieren, wenn Sie einen Index für diese Spalten definieren oder wenn Sie Gruppenspaltenstatistiken für die entsprechenden Spalten erfassen und pflegen.

In dem obigen Beispiel mit Farben und Elastizitäten von Materialien könnten Sie einen oder beide der folgenden Indizes definieren:

```
IX1 PRODUCT.COLOR, PRODUCT.ELASTICITY
```

oder

oder beide.

Damit das Optimierungsprogramm die Korrelation erkennen kann, dürfen die Nicht-INCLUDE-Spalten dieses Index nur die korrelierten Spalten sein. Der Index kann darüber hinaus auch INCLUDE-Spalten enthalten, um Abfragen nur über Indexsuchen erfüllen zu können. Wenn es mehr als zwei korrelierte Spalten in Join-vergleichselementen gibt, stellen Sie sicher, dass Sie den Index so definieren, dass er alle diese Spalten abdeckt.

Eine Bedingung, die erfüllt sein muss, damit das Optimierungsprogramm die Kardinalität eines Indexschlüssels in Betracht zieht, um eine Korrelation zu erkennen, besteht darin, dass die Anzahl der unterschiedlichen Werte in jeder Spalte für jede Spalte aus der zu vergleichenden Tabelle höher sein muss. Nehmen Sie zum Beispiel an, Sie haben die Indizes IX1 und IX2 wie oben gezeigt definiert. Wenn die Anzahl der unterschiedlichen Werte der Spalte PRODUCT.COLOR kleiner ist als die Anzahl der unterschiedlichen Werte der Spalte RAWMATERIAL.COLOR und die Anzahl der unterschiedlichen Werte der Spalte PRODUCT.ELASTICITY kleiner ist als die der Spalte RAWMATERIAL.ELASTICITY, dann verwendet das Optimierungsprogramm den Index IX2 zur Erkennung der Korrelation. Ein Vergleich der Spaltenkardinalitäten legt nahe, dass die unterschiedlichen Werte in den PRODUCT-Spalten sehr wahrscheinlich, wenn auch nicht mit absoluter Sicherheit, in den unterschiedlichen Werten der RAWMATERIAL-Spalten enthalten sind. Um die Wahrscheinlichkeit, dass ein Wertebereich einen anderen voraussetzt, noch weiter zu erhärten, kann das Optimierungsprogramm außerdem die HIGH2KEY- und LOW2KEY-Statistikdaten für diese Indexspalten zu Rate ziehen.

Stellen Sie nach der Erstellung der geeigneten Indizes sicher, dass die Statistikdaten für Tabellen präzise und aktuell sind.

Das Optimierungsprogramm verwendet die Informationen in den Spalten FIRST-nKEYCARD und FULLKEYCARD der Statistiktabellen für eindeutige Indizes, um Fälle von Korrelation zu erkennen und die errechneten kombinierten Selektivitäten der korrelierten Vergleichselemente dynamisch anzupassen, um so eine realistischere Schätzung der Größe und des Aufwands des Joins zu erhalten.

Alternativ können Spaltengruppenstatistiken für eine Gruppe von Spalten gesammelt werden. Im obigen Elastizitätsbeispiel könnten Sie Statistiken für die Spalten PRODUCT.COLOR, PRODUCT.ELASTICITY und/oder RAWMATERIAL.COLOR, RAWMATERIAL.ELASTICITY sammeln.

Spaltengruppenstatistiken werden mit der Option ON COLUMNS des Dienstprogramms RUNSTATS erfasst. Wenn Sie zum Beispiel die Spaltengruppenstatistiken für PRODUCT.COLOR und PRODUCT.ELASTICITY sammeln wollen, führen Sie den folgenden RUNSTATS-Befehl aus:

```
RUNSTATS ON TABLE product ON COLUMNS ((color, elasticity))
```

Korrelation einfacher Gleichheitsvergleichselemente

Zusätzlich zur Korrelation von Joinvergleichselementen verwaltet das Optimierungsprogramm auch die Korrelation mit einfachen Gleichheitsvergleichselementen des Typs SPALTE =. Betrachten Sie zum Beispiel eine Tabelle mit verschiedenen Typen von Fahrzeugen vor, die jeweils mit MARKE (d. h. Hersteller), MODELL, JAHR, FARBE und AUSFÜHRUNG (z. B. Limousine, Kombi, Sport- oder Nutzfahrzeug) eingetragen sind. Da die weitaus meisten Hersteller die gleichen Stan-

dardfarben für jedes ihrer Modelle und jede ihrer Ausführungen Jahr für Jahr liefern, sind die Vergleichselemente für die Spalte FARBE wahrscheinlich von denen für die Spalten MARKE, MODELL, AUSFÜHRUNG oder JAHR unabhängig. Die Vergleichselemente MARKE und MODELL sind jedoch in keinem Fall unabhängig voneinander, da nur jeweils ein Fahrzeughersteller ein Modell mit einem bestimmten Namen produziert. Identische Modellnamen, die von zwei oder mehr Fahrzeugherstellern verwendet werden, sind sehr unwahrscheinlich und mit Sicherheit von den Fahrzeugherstellern nicht gewollt.

Wenn für die beiden Spalten MARKE und MODELL ein Index vorhanden ist oder Spaltengruppenstatistiken erfasst sind, verwendet das Optimierungsprogramm die Statistikdaten über diesen Index oder die Spalten, um die kombinierte Zahl an unterschiedlichen Werten festzustellen und die Selektivitäts- oder Kardinalitätsschätzung für die Korrelation zwischen diesen beiden Spalten anzupassen. Wenn die Vergleichselemente lokale Gleichheitsvergleichselemente sind, benötigt das Optimierungsprogramm keinen eindeutigen Index, um die Anpassung vorzunehmen.

Zugehörige Konzepte:

- „Methoden zum Umschreiben von Abfragen und Beispiele“ auf Seite 236
- „Der SQL- und XQuery-Compilerprozess“ auf Seite 175
- „Berechnen von Gruppierungsschlüsselkardinalitäten mit Index- und Spaltengruppenstatistikdaten“ auf Seite 204

Zugehörige Tasks:

- „Erfassen von Verteilungsstatistiken für bestimmte Spalten“ auf Seite 324

Zugehörige Referenzen:

- „RUNSTATS command“ in *Command Reference*

Joinstrategien in partitionierten Datenbanken

In mancher Hinsicht unterscheiden sich die Joinstrategien in einer Umgebung mit partitionierten Datenbanken von denen in einer Umgebung mit nicht partitionierten Datenbanken. Zur Verbesserung der Leistung können zusätzliche Techniken auf die Standardjoinmethoden angewandt werden.

Ein Aspekt bei Tabellen, die häufig an Joins in einer Umgebung mit partitionierten Datenbanken beteiligt sind, ist die Tabellenkollokation. Die Tabellenkollokation stellt eine Methode in einer Umgebung mit partitionierten Datenbanken dar, mit der Daten aus einer Tabelle mit Hilfe eines gleichen Verteilungsschlüssels zusammen mit den Daten aus einer anderen Tabelle in derselben Datenbankpartition angeordnet werden. Wenn die Daten einmal in dieser Weise zusammengefasst sind, können zu verknüpfende Daten an einer Abfrage beteiligt sein, ohne im Rahmen der Aktivitäten für die Abfrage von einer Datenbankpartition in eine andere übertragen werden zu müssen. Nur die Ergebnismenge eines Joins wird an den Koordinatorknoten übergeben.

Tabellenwarteschlangen

In den Beschreibungen von Jointechniken in einer Umgebung mit partitionierten Datenbanken wird die folgende Terminologie verwendet:

- **Tabellenwarteschlange**
Ein Mechanismus zur Übertragung von Zeilen zwischen Datenbankpartitionen oder auch zwischen Prozessoren in einer Datenbank mit einer Einzelpartition.
- **Gezielt übertragene Tabellenwarteschlange**
Eine Tabellenwarteschlange, in der Zeilen durch ein Hashverfahren gezielt an eine der empfangenden Datenbankpartitionen geleitet werden.
- **Broadcast-Tabellenwarteschlange**
Eine Tabellenwarteschlange, in der Zeilen ohne Hashverfahren an alle empfangenden Datenbankpartitionen gesendet werden.

Eine Tabellenwarteschlange wird zu folgenden Zwecken verwendet:

- Übertragen von Tabellendaten von einer Datenbankpartition zu einer anderen bei Verwendung partitionsübergreifender Parallelität
- Übertragen von Tabellendaten innerhalb einer Datenbankpartition bei Verwendung partitionsinterner Parallelität
- Übertragen von Tabellendaten innerhalb einer Datenbankpartition bei Verwendung einer einzigen Datenbankpartition

Jede Tabellenwarteschlange überträgt die Daten in eine einzige Richtung. Der Compiler entscheidet, wo Tabellenwarteschlangen erforderlich sind, und nimmt sie in den Plan auf. Bei der Ausführung des Plans werden die Tabellenwarteschlangen durch die Verbindungen zwischen den Datenbankpartitionen initiiert. Die Tabellenwarteschlangen werden am Ende der Prozesse wieder geschlossen.

Es gibt verschiedene Arten von Tabellenwarteschlangen:

- *Asynchrone Tabellenwarteschlangen.* Diese Tabellenwarteschlangen werden als asynchron bezeichnet, weil sie Zeilen vor einer FETCH-Anweisung, die durch eine Anwendung abgesetzt wird, vorab lesen. Wenn eine FETCH-Anweisung abgesetzt wird, wird die Zeile aus der Tabellenwarteschlange abgerufen.
Asynchrone Tabellenwarteschlangen werden verwendet, wenn Sie die Klausel FOR FETCH ONLY in der SELECT-Anweisung verwenden. Wenn Sie Zeilen nur mit FETCH abrufen, ist eine asynchrone Tabellenwarteschlange schneller.
- *Synchrone Tabellenwarteschlangen.* Diese Tabellenwarteschlangen werden als synchron bezeichnet, weil sie für jede FETCH-Anweisung, die durch eine Anwendung abgesetzt wird, eine Zeile lesen. In jeder Datenbankpartition wird der Cursor in die nächste Zeile gesetzt, die aus der jeweiligen Datenbankpartition zu lesen ist.
Synchrone Tabellenwarteschlangen werden verwendet, wenn Sie die Klausel FOR FETCH ONLY in der SELECT-Anweisung nicht angeben. In einer Umgebung mit partitionierten Datenbanken verwendet der Datenbankmanager synchrone Tabellenwarteschlangen, wenn Zeilen aktualisiert werden.
- *Tabellenwarteschlangen für Mischjoin.*
Bei dieser Art von Tabellenwarteschlange wird die Reihenfolge gewahrt.
- *Reguläre Tabellenwarteschlangen.*
Diese Tabellenwarteschlangen sind nicht für Mischjoins geeignet. Bei ihnen bleibt die Reihenfolge nicht erhalten.
- *Empfangende Tabellenwarteschlangen.*
Diese Tabellenwarteschlangen werden mit korrelierten Unterabfragen verwendet. Die Korrelationswerte werden an die Unterabfrage übergeben, und die Ergebnisse mit Hilfe dieser Art von Tabellenwarteschlange an den übergeordneten Abfrageblock zurückgeliefert.

Zugehörige Konzepte:

- „Joinmethoden“ auf Seite 221
- „Joinmethoden in Umgebungen mit partitionierten Datenbanken“ auf Seite 231
- „Joins“ auf Seite 220

Joinmethoden in Umgebungen mit partitionierten Datenbanken

Die folgenden Abbildungen veranschaulichen Joinmethoden in einer Umgebung mit partitionierten Datenbanken.

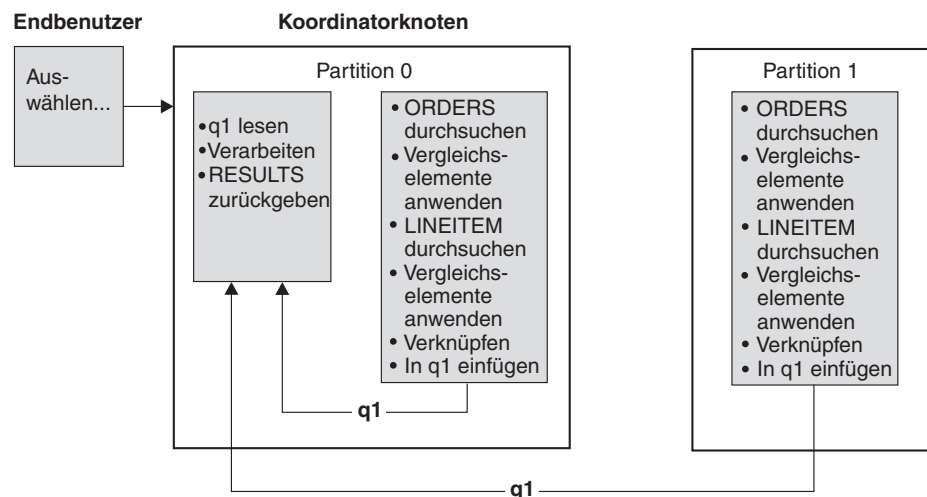
Anmerkung: In den Abbildungen beziehen sich q1, q2 und q3 (für „Queue“) auf die Tabellenwarteschlangen in den Beispielen. Die Tabellen, auf die zugegriffen wird, sind für den Zweck dieser Beispielszenarios auf zwei Datenbankpartitionen verteilt. Die Pfeile zeigen die Richtung an, in der die Tabellenwarteschlangen übertragen bzw. gesendet werden. Der Koordinatorknoten entspricht der Datenbankpartition 0.

Zusammengefasste Joins

Ein zusammengefasster Join findet lokal in der Datenbankpartition statt, in der sich die Daten befinden. Die Datenbankpartition sendet die Daten an die anderen Datenbankpartitionen, wenn der Join abgeschlossen ist. Damit das Optimierungsprogramm einen zusammengefassten Join in Erwägung ziehen kann, müssen die verknüpften Tabellen durch Kollokation zusammengefasst sein, und alle Paare mit dem entsprechenden Verteilungsschlüssel müssen Teil der Gleichheitsvergleichselemente sein.

Die folgende Abbildung zeigt ein Beispiel.

Anmerkung: Replizierte MQTs erhöhen die Wahrscheinlichkeit zusammengefasster Joins.



Die beiden Tabellen LINEITEM und ORDERS werden über die Spalte ORDERKEY partitioniert. Der Join erfolgt lokal in jeder Datenbankpartition. In diesem Beispiel wird folgendes Vergleichselement für den Join angenommen: ORDERS.ORDERKEY = LINEITEM.ORDERKEY.

Abbildung 20. Beispiel für einen zusammengefassten Join

Broadcast-Outer-Table-Joins

Broadcast-Outer-Table-Joins (Joins mit rundgesendeter äußerer Tabelle) sind eine parallele Joinstrategie, die angewandt werden kann, wenn es zwischen den zu verknüpfenden Tabellen keine Equijoin-Vergleichselemente gibt. Sie kann außerdem in solchen Fällen verwendet werden, in denen sie die Methode mit dem geringsten Aufwand darstellt. Zum Beispiel könnte ein Broadcast-Outer-Table-Join stattfinden, wenn eine sehr umfangreiche und eine sehr kleine Tabelle am Join beteiligt sind, von denen keine auf die Spalten, auf die die Joinvergleichselemente angewandt werden, verteilt ist. Anstatt beide Tabellen zu teilen, kann es günstiger sein, die kleinere Tabelle an alle Partitionen mit der größeren Tabelle per Broadcast rundzusenden. Die folgenden Abbildungen geben ein Beispiel.

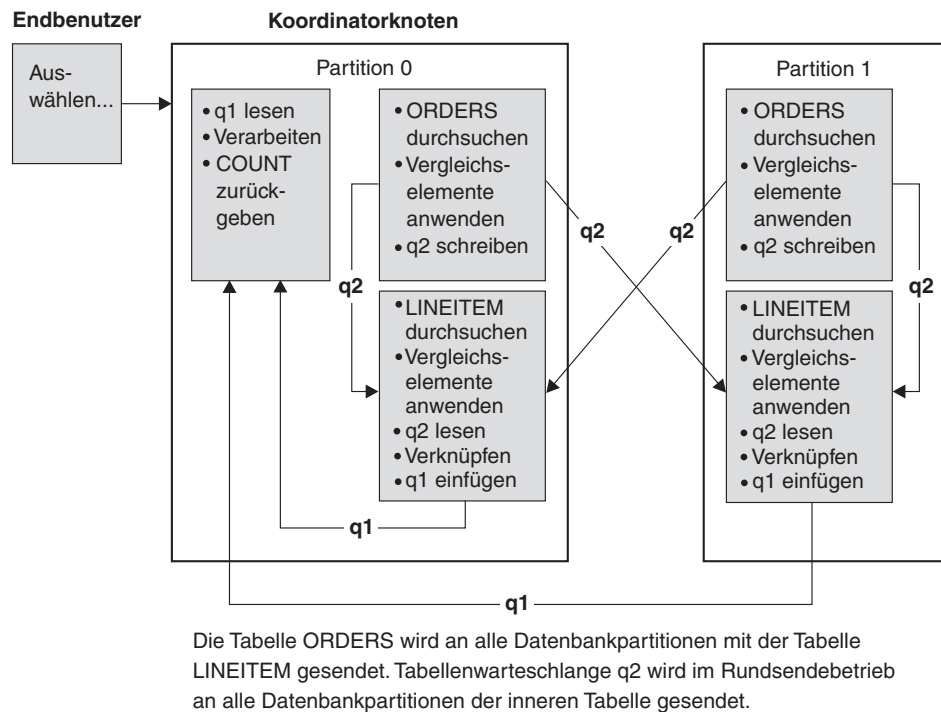
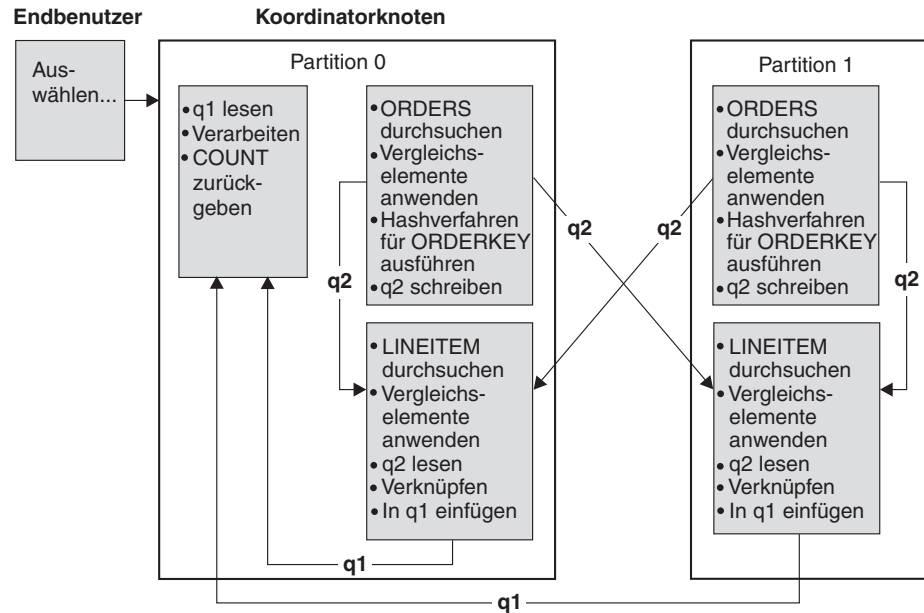


Abbildung 21. Beispiel für einen Broadcast-Outer-Table-Join

Directed-Outer-Table-Joins

Bei der Joinstrategie mit gezielt übertragener äußerer Tabelle (Directed-Outer-Table-Join) wird jede Zeile der äußeren Tabelle an einen Teil der inneren Tabelle entsprechend den Teilungsattributen der inneren Tabelle gesendet. Der Join erfolgt in dieser Datenbankpartition. Die folgende Abbildung zeigt ein Beispiel.

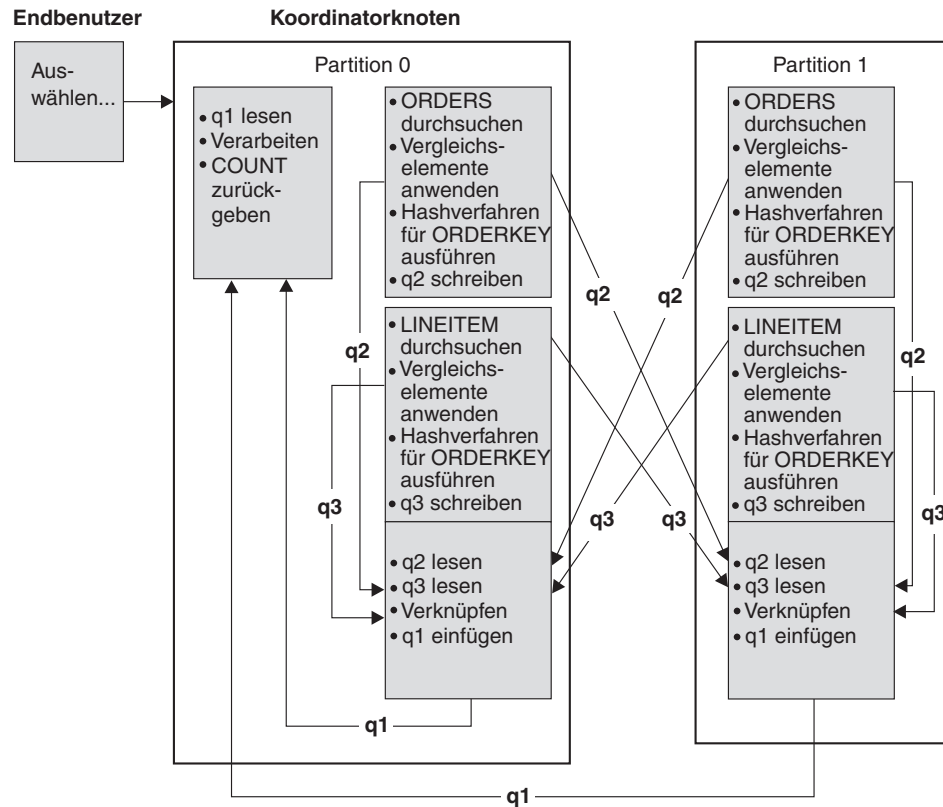


Die Tabelle LINEITEM wird über die Spalte ORDERKEY partitioniert.
 Die Tabelle ORDERS wird über eine andere Spalte partitioniert.
 Für die Tabelle ORDERS wird das Hashverfahren ausgeführt, und sie wird an die richtige Datenbankpartition der Tabelle LINEITEM gesendet.
 In diesem Beispiel wird folgendes Vergleichselement für den Join angenommen:
 ORDERS.ORDERKEY = LINEITEM.ORDERKEY.

Abbildung 22. Beispiel für einen Directed-Outer-Table-Join

Directed-Inner-Table- und Directed-Outer-Table-Joins

Bei der Joinstrategie mit gezielt übertragener innerer und äußerer Tabelle (Directed-Inner-Table- und Directed-Outer-Table-Join) werden Zeilen sowohl der äußeren als auch der inneren Tabelle gezielt an eine Gruppe von Datenbankpartitionen entsprechend den Werten der Joinspalten übertragen. Der Join erfolgt in diesen Datenbankpartitionen. Die folgende Abbildung zeigt ein Beispiel.

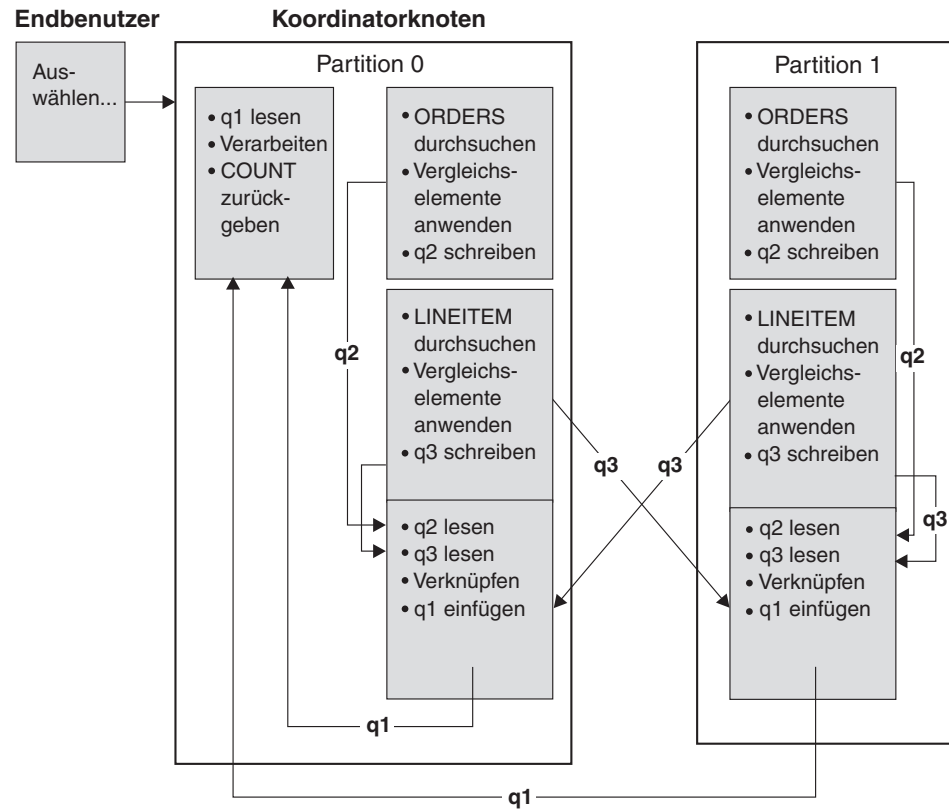


Es wird keine Tabelle über die Spalte ORDERKEY partitioniert.
Für beide Tabellen wird das Hashverfahren ausgeführt, und sie werden
an neue Datenbankpartitionen gesendet und dort verknüpft.
Beide Tabellenwarteschlangen q2 und q3 werden übertragen.
In diesem Beispiel wird folgendes Vergleichselement für den
Join angenommen:
ORDERS.ORDERKEY = LINEITEM.ORDERKEY

Abbildung 23. Beispiel für einen Directed-Outer-Table-Join

Broadcast-Inner-Table-Joins

Bei der Joinstrategie mit rundgesendeter innerer Tabelle (Broadcast-Inner-Table-Join) wird die innere Tabelle per Broadcast an alle Datenbankpartitionen der äußeren Jointabelle gesendet. Die folgende Abbildung zeigt ein Beispiel.

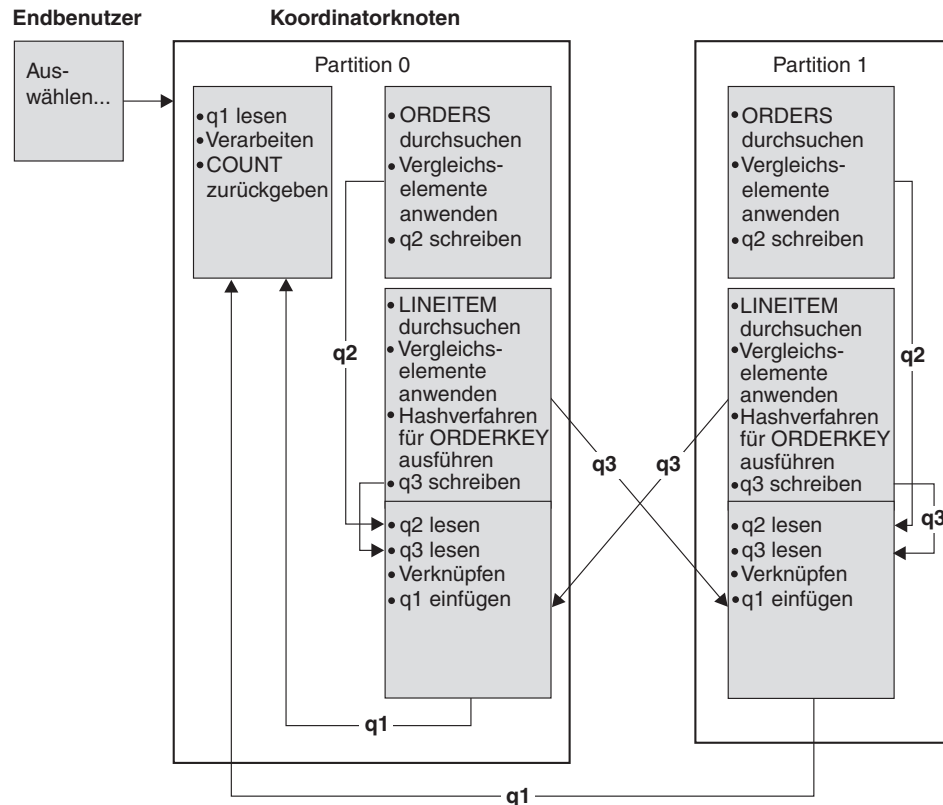


Die Tabelle LINEITEM wird an alle Datenbankpartitionen mit der Tabelle ORDERS gesendet. Tabellenwarteschlange q3 wird im Rundsendebetrieb an alle Datenbankpartitionen der äußeren Tabelle gesendet.

Abbildung 24. Beispiel für einen Broadcast-Inner-Table-Join

Directed-Inner-Table-Joins

Bei der Joinstrategie mit gezielt übertragener innerer Tabelle (Directed-Inner-Table-Join) wird jede Zeile der inneren Tabelle an eine Datenbankpartition der äußeren Tabelle entsprechend den Teilungsattributen der äußeren Tabelle übertragen. Der Join erfolgt in dieser Datenbankpartition. Die folgende Abbildung zeigt ein Beispiel.



Die Tabelle ORDERS wird über die Spalte ORDERKEY partitioniert. Die Tabelle LINEITEM wird über eine andere Spalte partitioniert. Für die Tabelle LINEITEM wird das Hashverfahren ausgeführt, und sie wird an die richtige Datenbankpartition der Tabelle ORDERS gesendet. In diesem Beispiel wird folgendes Vergleichselement für den Join angenommen:
ORDERS.ORDERKEY = LINEITEM.ORDERKEY.

Abbildung 25. Beispiel für einen Directed-Inner-Table-Join

Zugehörige Konzepte:

- „Joinmethoden“ auf Seite 221
- „Joinstrategien in partitionierten Datenbanken“ auf Seite 229
- „Joins“ auf Seite 220

Umschreiben von Abfragen

Methoden zum Umschreiben von Abfragen und Beispiele

Während der Phase des Umschreibens der Abfrage setzt der Abfragecompiler SQL- und XQuery-Anweisungen in Formate um, die leichter optimiert werden und infolgedessen die möglichen Zugriffspfade verbessern können. Das Umschreiben von Abfragen ist besonders wichtig für komplexe Abfragen, zu denen auch Abfragen mit zahlreichen Unterabfragen oder Joins zählen. Tools zur Generierung von Abfragen erstellen häufig diese sehr komplexen Arten von Abfragen.

Zur Beeinflussung der Anzahl von Regeln für das Umschreiben von Abfragen, die auf eine SQL- oder XQuery-Anweisung angewendet werden, ändern Sie die Optimierungsklasse. Einige der Ergebnisse dieses Umschreibens der Abfrage können Sie mit Hilfe der EXPLAIN-Einrichtung oder über Visual Explain anzeigen.

Abfragen können durch die folgenden drei Hauptmethoden umgeschrieben werden:

- **Zusammenfügen von Operationen**

Zur Generierung der Abfrage in ein Format, das möglichst wenige Operationen, insbesondere SELECT-Operationen, aufweist, schreibt der SQL- und XQuery-Compiler Abfragen um, damit Abfrageoperationen zusammengefügt werden können. Die folgenden Beispiele zeigen einige der Operationen, die zusammengefügt werden können:

- Beispiel - Zusammenfügen von Sichten

Eine SELECT-Anweisung, die Sichten verwendet, kann die Joinfolge der Tabelle einschränken und zudem überflüssige Joins von Tabellen nach sich ziehen. Wenn die Sichten während des Umschreibens der Abfrage zusammengefügt werden, können diese Einschränkungen aufgehoben werden.

- Beispiel - Umsetzungen von Unterabfragen in Joins

Wenn eine SELECT-Anweisung eine Unterabfrage enthält, kann die Auswahl der Reihenfolgeverarbeitung der Tabelle eingeschränkt sein.

- Beispiel - Eliminierung überflüssiger Joins

Während des Umschreibens der Abfrage können überflüssige Joins entfernt werden, um die SELECT-Anweisung zu vereinfachen.

- Beispiel - Gemeinsame Spaltenberechnungen

Wenn eine Abfrage verschiedene Funktionen verwendet, kann durch Umschreiben die Anzahl der erforderlichen Berechnungen reduziert werden.

- **Verschieben von Operationen**

Zur Generierung der Abfrage mit der kleinstmöglichen Anzahl an Operationen und Vergleichselementen schreibt der Compiler Abfragen um, um Abfrageoperationen zu verschieben. Die folgenden Beispiele zeigen einige Operationen, die verschoben werden können:

- Beispiel - Eliminierung von DISTINCT

Während des Umschreibens einer Abfrage kann das Optimierungsprogramm den Zeitpunkt verschieben, zu dem die DISTINCT-Operation durchgeführt wird, um den Aufwand für diese Operation zu verringern. In dem bereitgestellten erweiterten Beispiel wird die DISTINCT-Operation vollständig entfernt.

- Beispiel - Allgemeine Verschiebung von Vergleichselementen (Pushdown)

Während des Umschreibens von Abfragen kann das Optimierungsprogramm die Reihenfolge der angewandten Vergleichselemente ändern, so dass die Vergleichselemente, die die Auswahl in größerem Maße einschränken, zum frühestmöglichen Zeitpunkt angewandt werden.

- Beispiel - Dekorrelierung

In einer Umgebung mit partitionierten Datenbanken erfordert das Verschieben von Ergebnismengen zwischen den Datenbankpartitionen hohen Aufwand. Den Umfang des Broadcastbetriebs (Rundsenden) an andere Datenbankpartitionen oder die Anzahl der Broadcastvorgänge zu reduzieren, oder beides, gehört zu den Zielen des Umschreibens von Abfragen.

- **Übersetzen von Vergleichselementen**

Der SQL- und XQuery-Compiler schreibt Abfragen um, damit vorhandene Vergleichselemente für eine bestimmte Abfrage in ein optimiertes Format umgesetzt werden. Die folgenden Beispiele zeigen einige der Vergleichselemente, die übersetzt werden können:

- Beispiel - Hinzufügen implizierter Vergleichselemente

Während des Umschreibens können Vergleichselemente der Abfrage hinzugefügt werden, um dem Optimierungsprogramm die Möglichkeit zu geben, weitere Tabellenjoins bei der Auswahl des günstigsten Zugriffsplans für die Abfrage in Betracht zu ziehen.

– Beispiel - Transformation von OR zu IN

Während des Umschreibens einer Abfrage kann für einen effizienteren Zugriffsplan ein Vergleichselement OR in ein Vergleichselement IN umgesetzt werden. Der SQL- und XQuery-Compiler kann auch ein Vergleichselement IN in ein Vergleichselement OR umsetzen, wenn diese Transformation einen günstigeren Zugriffsplan generieren würde.

Zugehörige Konzepte:

- „Spaltenkorrelation für mehrere Vergleichselemente“ auf Seite 227
- „Beispiel für das Umschreiben durch den Compiler: Eliminierung von DISTINCT“ auf Seite 240
- „Beispiel für das Umschreiben durch den Compiler: implizierte Vergleichselemente“ auf Seite 242
- „Beispiel für das Umschreiben durch den Compiler: Zusammenfügen von Sichten“ auf Seite 238

Beispiel für das Umschreiben durch den Compiler: Zusammenfügen von Sichten

Nehmen Sie zum Beispiel an, Sie haben Zugriff auf die beiden folgenden Sichten der Tabelle EMPLOYEE, von denen die eine die Mitarbeiter mit einer hohen Ausbildungsstufe (EDLEVEL) und die andere die Mitarbeiter mit einem Gehalt (SALARY) über 35.000 Dollar zeigt:

```
CREATE VIEW EMP_EDUCATION (EMPNO, FIRSTNAME, LASTNAME, EDLEVEL) AS
SELECT EMPNO, FIRSTNAME, LASTNAME, EDLEVEL
FROM EMPLOYEE
WHERE EDLEVEL > 17
CREATE VIEW EMP_SALARIES (EMPNO, FIRSTNAME, LASTNAME, SALARY) AS
SELECT EMPNO, FIRSTNAME, LASTNAME, SALARY
FROM EMPLOYEE
WHERE SALARY > 35000
```

Jetzt wird beispielsweise die folgende Abfrage ausgeführt, um die Mitarbeiter, die eine hohe Ausbildungsstufe haben und deren Gehalt über 35.000 Dollar liegt, aufzulisten:

```
SELECT E1.EMPNO, E1.FIRSTNAME, E1.LASTNAME, E1.EDLEVEL, E2.SALARY
FROM EMP_EDUCATION E1,
     EMP_SALARIES E2
WHERE E1.EMPNO = E2.EMPNO
```

Während der Phase des Umschreibens könnten diese beiden Sichten zusammengefügt werden, um folgende Abfrage zu erstellen:

```
SELECT E1.EMPNO, E1.FIRSTNAME, E1.LASTNAME, E1.EDLEVEL, E2.SALARY
FROM EMPLOYEE E1,
     EMPLOYEE E2
WHERE E1.EMPNO = E2.EMPNO
AND E1.EDLEVEL > 17
AND E2.SALARY > 35000
```

Durch das Zusammenfügen der Anweisungen SELECT der beiden Sichten mit der vom Benutzer geschriebenen Anweisung SELECT kann das Optimierungsprogramm mehr Möglichkeiten bei der Auswahl des Zugriffsplans in Betracht ziehen.

Darüber hinaus kann die Abfrage noch weiter umgeschrieben werden, wenn die beiden zusammengeführten Sichten dieselbe Basistabelle verwenden.

Beispiel - Umsetzungen von Unterabfragen in Joins

Angenommen, der SQL- und XQuery-Compiler verfügt über eine Abfrage mit einer Unterabfrage wie die folgende:

```
SELECT EMPNO, FIRSTNAME, LASTNAME, PHONENO
FROM EMPLOYEE
WHERE WORKDEPT IN
    (SELECT DEPTNO
     FROM DEPARTMENT
     WHERE DEPTNAME = 'OPERATIONS')
```

Diese Unterabfrage wird vom SQL-Compiler in eine Joinabfrage der folgenden Form umgewandelt:

```
SELECT DISTINCT EMPNO, FIRSTNAME, LASTNAME, PHONENO
FROM EMPLOYEE EMP,
     DEPARTMENT DEPT
WHERE EMP.WORKDEPT = DEPT.DEPTNO
     AND DEPT.DEPTNAME = 'OPERATIONS'
```

Im Allgemeinen ist ein Join in der Ausführung wesentlich effektiver als eine Unterabfrage.

Beispiel - Eliminierung überflüssiger Joins

Manche geschriebenen oder generierten Abfragen enthalten unnötige Joins. Abfragen wie die folgende könnten auch während des Umschreibens einer Abfrage generiert werden.

```
SELECT E1.EMPNO, E1.FIRSTNAME, E1.LASTNAME, E1.EDLEVEL, E2.SALARY
FROM EMPLOYEE E1,
     EMPLOYEE E2
WHERE E1.EMPNO = E2.EMPNO
     AND E1.EDLEVEL > 17
     AND E2.SALARY > 35000
```

In dieser Abfrage kann der SQL- und XQuery-Compiler den Join eliminieren und die Abfrage wie folgt vereinfachen:

```
SELECT EMPNO, FIRSTNAME, LASTNAME, EDLEVEL, SALARY
FROM EMPLOYEE
WHERE EDLEVEL > 17
     AND SALARY > 35000
```

Im folgenden Beispiel wird davon ausgegangen, dass zwischen den Beispieltabellen EMPLOYEE und DEPARTMENT eine referenzielle Integritätsbedingung über die Abteilung (WORKDEPT/DEPTNO) vorhanden ist. Zuerst wird eine Sicht erstellt.

```
CREATE VIEW PEPLVIEW
AS SELECT FIRSTNAME, LASTNAME, SALARY, DEPTNO, DEPTNAME, MGRNO
   FROM EMPLOYEE E DEPARTMENT D
   WHERE E.WORKDEPT = D.DEPTNO
```

Eine Abfrage wie die folgende:

```
SELECT LASTNAME, SALARY
FROM PEPLVIEW
```

wird dann geändert in:

```
SELECT LASTNAME, SALARY
FROM EMPLOYEE
WHERE WORKDEPT NOT NULL
```

Beachten Sie bei dieser Situation, dass Benutzer die Abfrage eventuell nicht umschreiben können, selbst wenn sie wüssten, dass dies möglich ist, weil sie keinen Zugriff auf die zugrunde liegenden Tabellen haben. Sie haben eventuell nur Zugriff auf die oben gezeigte Sicht. Daher muss diese Art von Optimierung im Datenbankmanager ausgeführt werden.

Redundanz in Joins mit referenzieller Integrität ist in folgenden Fällen wahrscheinlich:

- Sichten sind mit Joins definiert.
- Abfragen werden automatisch generiert.

Es gibt zum Beispiel automatisierte Tools in Abfragemanagern, die das Schreiben optimierter Abfragen durch Benutzer verhindern.

Beispiel - Gemeinsame Spaltenberechnungen

Bei Verwendung mehrerer Funktionen in einer Abfrage können zahlreiche Berechnungen entstehen, die zeitintensiv sind. Durch Reduzieren der für die Abfrage erforderlichen Anzahl von Berechnungen kann der Zugriffsplan verbessert werden. Der SQL- und XQuery-Compiler verfügt über eine Abfrage, die mehrere Funktionen verwendet, wie z. B.:

```
SELECT SUM(SALARY+BONUS+COMM) AS OSUM,
AVG(SALARY+BONUS+COMM) AS OAVG,
COUNT(*) AS OCOUNT
FROM EMPLOYEE;
```

wird vom SQL-Compiler wie folgt umgewandelt:

```
SELECT OSUM,
OSUM/OCOUNT
OCOUNT
FROM (SELECT SUM(SALARY+BONUS+COMM) AS OSUM,
COUNT(*) AS OCOUNT
FROM EMPLOYEE) AS SHARED_AGG;
```

Durch dieses Umschreiben benötigt die Abfrage statt 2 Summen und 2 Zählern nur noch 1 Summe und 1 Zähler.

Zugehörige Konzepte:

- „Methoden zum Umschreiben von Abfragen und Beispiele“ auf Seite 236
- „Der SQL- und XQuery-Compilerprozess“ auf Seite 175

Beispiel für das Umschreiben durch den Compiler: Eliminierung von DISTINCT

Für das folgende Abfragebeispiel wird angenommen, dass die Spalte EMPNO als Primärschlüssel der Tabelle EMPLOYEE definiert wurde:

```
SELECT DISTINCT EMPNO, FIRSTNAME, LASTNAME
FROM EMPLOYEE
```

Diese Abfrage würde so umgeschrieben, dass die Klausel DISTINCT entfernt wird:

```
SELECT EMPNO, FIRSTNAME, LASTNAME
FROM EMPLOYEE
```

Da hier der Primärschlüssel ausgewählt wird, weiß der SQL- und XQuery-Compiler, dass jede zurückgegebene Zeile bereits eindeutig ist. In diesem Fall wird das Schlüsselwort DISTINCT nicht benötigt. Wenn die Abfrage nicht umgeschrieben wird, erstellt das Optimierungsprogramm einen Plan, der die nötigen Verarbeitungsschritte, wie zum Beispiel eine Sortierung, enthält, um sicherzustellen, dass die Spalten eindeutig sind.

Beispiel - Allgemeines Verschieben von Vergleichselementen (Pushdown)

Durch das Ändern der Ebene, auf der Vergleichselemente normalerweise angewendet werden, lässt sich eventuell eine Leistungsverbesserung erreichen. Zum Beispiel sei die folgende Sicht gegeben, die eine Liste aller Mitarbeiter der Abteilung „D11“ zusammenstellt:

```
CREATE VIEW D11_EMPLOYEE
(EMPNO, FIRSTNAME, LASTNAME, PHONENO, SALARY, BONUS, COMM)
AS SELECT EMPNO, FIRSTNAME, LASTNAME, PHONENO, SALARY, BONUS, COMM
FROM EMPLOYEE
WHERE WORKDEPT = 'D11'
```

Es wird nun die folgende Abfrage aufgesetzt:

```
SELECT FIRSTNAME, PHONENO
FROM D11_EMPLOYEE
WHERE LASTNAME = 'BROWN'
```

In der Phase des Umschreibens verschiebt der Compiler das Vergleichselement LASTNAME = 'BROWN' nach unten in die Sicht D11_EMPLOYEE. Dadurch kann das Vergleichselement früher und möglicherweise effektiver angewandt werden. Die tatsächliche Abfrage, die in diesem Beispiel zur Ausführung kommen könnte, sieht folgendermaßen aus:

```
SELECT FIRSTNAME, PHONENO
FROM EMPLOYEE
WHERE LASTNAME = 'BROWN'
AND WORKDEPT = 'D11'
```

Das Verschieben von Vergleichselementen ist nicht auf Sichten beschränkt. Beispiele für andere Situationen, in denen Vergleichselemente verschoben werden können, sind Klauseln UNION, GROUP BY und abgeleitete Tabellen (verschachtelte Tabellenausdrücke oder allgemeine Tabellenausdrücke).

Beispiel - Dekorrelierung

In einer Umgebung mit partitionierten Datenbanken kann der SQL- und XQuery-Compiler die folgende Abfrage umschreiben:

Lokalisieren aller Mitarbeiter, die in der Programmierung arbeiten und unterbezahlt sind.

```
SELECT P.PROJNO, E.EMPNO, E.LASTNAME, E.FIRSTNAME,
E.SALARY+E.BONUS+E.COMM AS COMPENSATION
FROM EMPLOYEE E, PROJECT P
WHERE P.EMPNO = E.EMPNO
AND P.PROJNAME LIKE '%PROGRAMMING%'
AND E.SALARY+E.BONUS+E.COMM <
(SELECT AVG(E1.SALARY+E1.BONUS+E1.COMM)
FROM EMPLOYEE E1, PROJECT P1
WHERE P1.PROJNAME LIKE '%PROGRAMMING%'
AND P1.PROJNO = A.PROJNO
AND E1.EMPNO = P1.EMPNO)
```

Da die Abfrage korreliert ist und wahrscheinlich weder PROJECT noch EMPLOYEE über die Spalte PROJNO partitioniert sind, wird möglicherweise jedes Projekt im Broadcastbetrieb an jede Datenbankpartition übermittelt. Außerdem müsste die Unterabfrage viele Male ausgewertet werden.

Der SQL- und XQuery-Compiler kann die Abfrage wie folgt umschreiben:

- Die eindeutige Liste (DISTINCT) der Mitarbeiter (Employees) ermitteln, die an Programmierungsprojekten arbeiten, und diese DIST_PROJS nennen. Diese Liste muss mit DISTINCT erstellt werden, damit sichergestellt wird, dass die Spaltenberechnung nur einmal pro Projekt erfolgt:

```
WITH DIST_PROJS(PROJNO, EMPNO) AS
(SELECT DISTINCT PROJNO, EMPNO
 FROM PROJECT P1
 WHERE P1.PROJNAME LIKE '%PROGRAMMING%')
```

- Die eindeutige Liste der Mitarbeiter, die an Programmierungsprojekten arbeiten mit der Mitarbeitertabelle verknüpfen, um die durchschnittliche Entlohnung je Projekt (AVG_PER_PROJ) zu ermitteln:

```
AVG_PER_PROJ(PROJNO, AVG_COMP) AS
(SELECT P2.PROJNO, AVG(E1.SALARY+E1.BONUS+E1.COMM)
 FROM EMPLOYEE E1, DIST_PROJS P2
 WHERE E1.EMPNO = P2.EMPNO
 GROUP BY P2.PROJNO)
```

- Die neue Abfrage würde wie folgt aussehen:

```
SELECT P.PROJNO, E.EMPNO, E.LASTNAME, E.FIRSTNAME,
E.SALARY+E.BONUS+E.COMM AS COMPENSATION
FROM PROJECT P, EMPLOYEE E, AVG_PER_PROJ A
HERE P.EMPNO = E.EMPNO
AND P.PROJNAME LIKE '%PROGRAMMING%'
AND P.PROJNO = A.PROJNO
AND E.SALARY+E.BONUS+E.COMM < A.AVG_COMP
```

Die umgeschriebene Abfrage berechnet AVG_COMP je Projekt (AVG_PER_PROJ) und kann das Ergebnis an alle Datenbankpartitionen übermitteln, in denen die Tabelle EMPLOYEE enthalten ist.

Zugehörige Konzepte:

- „Methoden zum Umschreiben von Abfragen und Beispiele“ auf Seite 236
- „Der SQL- und XQuery-Compilerprozess“ auf Seite 175

Beispiel für das Umschreiben durch den Compiler: implizierte Vergleichselemente

Die folgende Abfrage erstellt eine Liste der Manager, deren Abteilung an Abteilung „E01“ berichten, und der Projekte, für die die Manager verantwortlich sind:

```
SELECT DEPT.DEPTNAME DEPT.MGRNO, EMP.LASTNAME, PROJ.PROJNAME
FROM DEPARTMENT DEPT,
EMPLOYEE EMP,
PROJECT PROJ
WHERE DEPT.ADMRDEPT = 'E01'
AND DEPT.MGRNO = EMP.EMPNO
AND EMP.EMPNO = PROJ.RESPEMP
```

Beim Umschreiben der Abfrage wird das folgende implizierte Vergleichselement hinzugefügt:

```
DEPT.MGRNO = PROJ.RESPEMP
```

Als Ergebnis dieses Umschreibens kann das Optimierungsprogramm weitere Joins in die Berechnung mit einbeziehen, wenn es versucht, den günstigsten Zugriffsplan für die Abfrage auszuwählen.

Neben der oben gezeigten Ausnutzung der Transitivität der Vergleichselemente werden durch das Umschreiben auch zusätzliche lokale Vergleichselemente aufgrund der durch Gleichheitsvergleichselemente implizierten Transitivität abgeleitet. Zum Beispiel stellt die folgende Abfrage eine Liste der Namen der Abteilungen (deren Abteilungsnummer größer als „E00“ ist) und der Mitarbeiter, die in diesen Abteilungen arbeiten, zusammen.

```
SELECT EMPNO, LASTNAME, FIRSTNAME, DEPTNO, DEPTNAME
FROM EMPLOYEE EMP,
     DEPARTMENT DEPT
WHERE EMP.WORKDEPT = DEPT.DEPTNO
     AND DEPT.DEPTNO > 'E00'
```

Dieser Abfrage wird in der Phase des Umschreibens das folgende implizierte Vergleichselement hinzugefügt:

```
EMP.WORKDEPT > 'E00'
```

Das Ergebnis dieses Umschreibens besteht darin, dass das Optimierungsprogramm die Anzahl der Zeilen, die verknüpft werden müssen, verringern kann.

Beispiel - Transformationen von OR zu IN

Nehmen Sie an, eine Klausel OR verknüpft zwei oder mehr einfache Gleichheitsvergleichselemente für dieselbe Spalte, wie im folgenden Beispiel:

```
SELECT *
FROM EMPLOYEE
WHERE DEPTNO = 'D11'
     OR DEPTNO = 'D21'
     OR DEPTNO = 'E21'
```

Wenn es für die Spalte DEPTNO keinen Index gibt, erlaubt die Umwandlung der Klausel OR in das folgende Vergleichselement IN eine effektivere Verarbeitung der Abfrage:

```
SELECT *
FROM EMPLOYEE
WHERE DEPTNO IN ('D11', 'D21', 'E21')
```

Anmerkung: In einigen Fällen kann der Datenbankmanager ein IN-Vergleichselement in eine Gruppe von Klauseln OR umwandeln, so dass logische OR-Joins für Indizes (Index ORing) durchgeführt werden können.

Zugehörige Konzepte:

- „Methoden zum Umschreiben von Abfragen und Beispiele“ auf Seite 236
- „Der SQL- und XQuery-Compilerprozess“ auf Seite 175

Pushdown-Analyse für föderierte Datenbanken

Für Abfragen in föderierten Datenbanken führt das Optimierungsprogramm eine Pushdown-Analyse durch, um zu ermitteln, ob eine Operation an einer fernen Datenquelle durchgeführt werden kann. Bei dieser Operation kann es sich um eine Funktion, wie zum Beispiel einen relationalen Operator, eine System- oder Benutzerfunktion oder einen SQL-Operator, zum Beispiel GROUP BY, ORDER BY usw., handeln.

Anmerkung: Obwohl der SQL-Compiler von DB2 über zahlreiche Informationen zur SQL-Unterstützung der Datenquelle verfügt, müssen diese Daten möglicherweise mit der Zeit angepasst werden, da Datenquellen aufgerüstet und/oder angepasst werden können. In solchen Fällen müssen Erweiterungen DB2 durch Ändern der lokalen Kataloginformationen bekannt gemacht werden. Verwenden Sie DDL-Anweisungen von DB2 (z. B. CREATE FUNCTION MAPPING und ALTER SERVER), um den Katalog zu aktualisieren.

Wenn Funktionen nicht an die ferne Datenquelle verschoben werden können, können sie sich erheblich auf die Abfrageleistung auswirken. Betrachten Sie die Konsequenzen für den Fall, dass ein selektives Vergleichselement lokal anstatt an der Datenquelle ausgewertet werden muss. Eine solche Auswertung würde DB2 zwingen, die gesamte Tabelle von der fernen Datenquelle abzurufen und anschließend lokal über das Vergleichselement zu filtern. Netzwerkbegrenzungen und eine bedeutende Tabellengröße könnten zu einer Beeinträchtigung der Leistung führen.

Operatoren, die nicht an die Datenquelle verschoben werden, können sich ebenfalls bedeutsam auf die Abfrageleistung auswirken. Wenn zum Beispiel der Operator GROUP BY ferne Daten lokal zusammenfassen soll, muss DB2 möglicherweise ebenfalls die gesamte Tabelle von der fernen Datenquelle abrufen.

Nehmen Sie zum Beispiel an, dass der Kurzname N1 auf die Tabelle EMPLOYEE an einer Datenquelle unter DB2 für OS/390 oder z/OS verweist. Nehmen Sie weiter an, dass die Tabelle 10.000 Zeilen hat und dass eine der Spalten die Familiennamen (Lastname) und eine andere die Gehälter (Salary) enthält. Betrachten Sie die folgende Anweisung:

```
SELECT LASTNAME, COUNT(*) FROM N1
WHERE LASTNAME > 'B' AND SALARY > 50000
GROUP BY LASTNAME;
```

In Abhängigkeit davon, ob die Sortierfolgen unter DB2 und DB2 für OS/390 oder z/OS übereinstimmen, werden verschiedene Möglichkeiten in Betracht gezogen:

- Wenn die Sortierfolgen übereinstimmen, kann das Abfragevergleichselement höchstwahrscheinlich per Pushdown-Aktion nach DB2 für OS/390 oder z/OS verschoben werden. Eine Filterung und Gruppierung der Ergebnisse an der Datenquelle ist in der Regel effizienter, als die gesamte Tabelle in DB2 zu kopieren und die Operationen lokal auszuführen. Für die oben gezeigte Abfrage können die Operationen für das Vergleichselement und für den Operator GROUP BY an der Datenquelle stattfinden.
- Wenn die Sortierfolgen nicht übereinstimmen, kann das gesamte Vergleichselement nicht an der Datenquelle ausgewertet werden. Allerdings kann das Optimierungsprogramm entscheiden, den Teil SALARY > 50000 des Vergleichs-

elements an die Datenquelle zu verschieben. Der Vergleich des Wertebereichs muss immer noch in DB2 ausgeführt werden.

- Wenn die Sortierfolgen übereinstimmen und dem Optimierungsprogramm bekannt ist, dass der lokale DB2-Server sehr schnell ist, kann das Optimierungsprogramm entscheiden, dass die lokale Ausführung der Operation GROUP BY in DB2 das günstigste Verfahren (mit dem geringsten Aufwand) ist. Das Vergleichselement wird an der Datenquelle ausgeführt. Dies wäre ein Beispiel für die Pushdown-Analyse in Kombination mit globaler Optimierung.

Im Allgemeinen besteht das Ziel darin, sicherzustellen, dass das Optimierungsprogramm Funktionen und Operatoren an den Datenquellen auswertet. Zahlreiche Faktoren haben Einfluss auf die Entscheidung, ob eine Funktion oder ein SQL-Operator an der fernen Datenquelle ausgewertet wird. Die zu bewertenden Faktoren werden in folgende drei Gruppen klassifiziert:

- Servermerkmale
- Kurznamenmerkmale
- Abfragemerkmale

Servermerkmale mit Auswirkung auf die Pushdown-Möglichkeiten

Bestimmte datenquellenspezifische Faktoren können sich auf die Pushdown-Möglichkeiten auswirken. Diese Faktoren sind im Allgemeinen deswegen vorhanden, weil DB2 eine sehr facettenreiche SQL-Programmversion unterstützt. Diese SQL-Version bietet u. U. mehr Funktionalität als die SQL-Version, die von einem Server unterstützt wird, auf den durch eine Abfrage zugegriffen wird. DB2 kann diesen Funktionsmangel auf dem Datenserver kompensieren, allerdings kann dies dazu führen, dass die Operation in DB2 ausgeführt werden muss.

SQL-Leistungsspektrum: Jede Datenquelle unterstützt eine Variante der SQL-Version und verschiedene Funktionalitätsebenen. Betrachten Sie zum Beispiel die GROUP BY-Liste. Die meisten Datenquellen unterstützen den Operator GROUP BY. Jedoch begrenzen einige die Anzahl von Elementen in der GROUP BY-Liste oder haben Einschränkungen hinsichtlich der Zulässigkeit bestimmter Ausdrücke in der GROUP BY-Liste. Wenn es eine Einschränkung für die ferne Datenquelle gibt, muss DB2 die Operation GROUP BY möglicherweise lokal ausführen.

SQL-Einschränkungen: Jede Datenquelle kann unterschiedliche SQL-Einschränkungen haben. Zum Beispiel fordern einige Datenquellen, dass Parametermarken Werte an ferne SQL-Anweisungen binden. Daher müssen die Einschränkungen für Parametermarken überprüft werden, um sicherzustellen, dass jede Datenquelle solche Bindeverfahren unterstützen kann. Wenn DB2 keine gute Methode zum Binden eines Werts für eine Funktion finden kann, muss diese Funktion lokal ausgewertet werden.

SQL-Begrenzungen: Zwar ermöglicht DB2 die Verwendung größerer Ganzzahlen (Integer) im Vergleich zu den fernen Datenquellen, aber es können keine Werte, die ferne Grenzwerte überschreiten, in Anweisungen eingebettet werden, die an Datenquellen gesendet werden. Daher muss eine Funktion oder ein Operator, die bzw. der mit einer solchen Konstanten operiert, lokal ausgewertet werden.

Serverspezifische Faktoren: In diese Kategorie fallen verschiedene Faktoren. Ein Beispiel ist, ob NULL-Werte als höchster oder niedrigster Wert sortiert werden oder von der Reihenfolge abhängig sind. Wenn Nullwerte in einer Datenquelle anders als von DB2 sortiert werden, können ORDER BY-Operationen für einen Nullwertausdruck nicht fern ausgewertet werden.

Sortierfolge: Das Abrufen von Daten für lokale Sortierungen und Vergleiche setzt in der Regel die Leistung herab. Daher sollten Sie in Erwägung ziehen, die föderierte Datenbank so zu konfigurieren, dass sie dieselbe Sortierfolge wie die Datenquellen verwendet. Wenn Sie eine föderierte Datenbank zur Verwendung derselben Sortierfolge konfigurieren, die von einer Datenquelle verwendet wird, und die Serveroption *collating_sequence* auf den Wert 'Y' setzen, kann das Optimierungsprogramm in Betracht ziehen, viele Abfrageoperationen an die Datenquelle zu verschieben, wenn sich dadurch die Leistung verbessert.

Die folgenden Operationen können an eine Datenquelle verschoben werden, wenn die Sortierfolgen übereinstimmen:

- Vergleiche von Zeichendaten oder numerischen Daten
- Vergleichselemente für Zeichenbereiche
- Sortierungen

Unerwartete Ergebnisse kommen eventuell dann zustande, wenn die Wertigkeit von Nullzeichen zwischen der föderierten Datenbank und der Datenquelle unterschiedlich ist. Vergleichsanweisungen können unerwartete Ergebnisse liefern, wenn Sie Anweisungen an eine Datenquelle übergeben, auf der die Groß-/Kleinschreibung nicht unterschieden wird. Die Wertigkeiten der Zeichen "I" und "i" sind bei einer Datenquelle ohne Unterscheidung der Groß-/Kleinschreibung identisch. Bei DB2 wird standardmäßig die Groß-/Kleinschreibung beachtet; außerdem werden den Zeichen unterschiedliche Wertigkeiten zugeordnet.

Zur Verbesserung der Leistung lässt der Server der föderierten Datenbank zu, dass Sortierungen und Vergleiche an der Datenquelle stattfinden. Zum Beispiel werden in DB2 für OS/390 oder z/OS Sortierungen, die durch Klauseln ORDER BY definiert werden, mit Hilfe einer Sortierfolge implementiert, die auf einer EBCDIC-Codepage (Extended Binary-Coded Decimal Interchange Code) basiert. Um den Server der föderierten Datenbank zum Abrufen von Daten einer Datenquelle unter DB2 für OS/390 oder z/OS zu verwenden und die Daten an der Datenquelle mit Hilfe von Klauseln des Typs ORDER BY zu sortieren, konfigurieren Sie die föderierte Datenbank so, dass sie eine vordefinierte, auf der EBCDIC-Codepage basierte Sortierfolge verwendet.

Wenn die Sortierfolgen der föderierten Datenbank und der Datenquelle voneinander abweichen, ruft DB2 die Daten für die föderierte Datenbank ab. Da Benutzer die Abfrageergebnisse nach der für den Server der föderierten Datenbank definierten Sortierfolge geordnet erwarten, stellt Server der föderierten Datenbank durch ein lokales Sortieren der Daten sicher, dass diese Erwartung erfüllt wird. Übergeben Sie Ihre Abfrage im Durchgriffsmodus (Pass-through) oder definieren Sie die Abfrage in einer Datenquellsicht, wenn Sie die Daten nach der Sortierfolge der Datenquelle geordnet abrufen müssen.

Serveroptionen: Verschiedene Serveroptionen können die Pushdown-Möglichkeiten beeinflussen. Prüfen Sie insbesondere Ihre Einstellungen für die Serveroptionen *collating_sequence*, *varchar_no_trailing_blanks* und *pushdown*.

Faktoren der DB2-Typenzuordnung und -Funktionszuordnung: Die von DB2 definierten lokalen Standardtypenzuordnungen sind dazu vorgesehen, genügend Pufferspeicherplatz für jeden Datentyp einer Datenquelle bereitzustellen, wodurch Datenverlust vermieden wird. Benutzer können Typenzuordnung für eine bestimmte Datenquelle an die Anforderungen bestimmter Anwendungen anzupassen. Wenn Sie zum Beispiel auf eine Spalte des Datentyps DATE einer Oracle-

Datenquelle zugreifen, die standardmäßig dem DB2-Datentyp `TIMESTAMP` zugeordnet wird, können Sie den lokalen Datentyp in den DB2-Datentyp `DATE` ändern.

In den folgenden drei Fällen kann DB2 Funktionen kompensieren, die von einer Datenquelle nicht unterstützt werden:

- Die Funktion ist an der fernen Datenquelle nicht vorhanden.
- Die Funktion ist vorhanden, jedoch verletzen die Merkmale des Operanden die Einschränkungen der Funktion. Ein Beispiel für diesen Fall ist der relationale Operator `IS NULL`. Die meisten Datenquellen unterstützen ihn, aber einige haben möglicherweise Einschränkungen, wie zum Beispiel, dass nur ein Spaltenname auf der linken Seite des Operators `IS NULL` zulässig ist.
- Die Funktion liefert möglicherweise ein anderes Ergebnis, wenn sie fern ausgewertet wird. Ein Beispiel für diesen Fall ist der Operator `'>'` (größer als). Für Datenquellen mit abweichenden Sortierfolgen kann der Operator 'größer als' andere Ergebnisse liefern als bei einer lokalen Auswertung durch DB2.

Kurznamenmerkmale mit Auswirkung auf die Pushdown-Möglichkeiten

Die folgenden kurznamenspezifischen Faktoren können sich auf die Pushdown-Möglichkeiten auswirken.

Lokaler Datentyp einer Kurznamenspalte: Stellen Sie sicher, dass der lokale Datentyp einer Spalte nicht verhindert, dass ein Vergleichselement an der Datenquelle ausgewertet werden kann. Verwenden Sie die Standarddatentypenzuordnungen, um einen möglichen Überlauf zu vermeiden. Jedoch wird ein Joinvergleichselement zwischen zwei Spalten unterschiedlicher Längen möglicherweise nicht bei der Datenquelle ausgewertet, deren Joinspalte kürzer ist, in Abhängigkeit davon, wie DB2 die längere Spalte bindet. Dieser Umstand kann sich auf die Anzahl der Möglichkeiten auswirken, die das DB2-Optimierungsprogramm in einer Joinsequenz auswerten kann. Zum Beispiel erhalten Spalten einer Oracle-Datenquelle, die mit dem Datentyp `INTEGER` bzw. `INT` erstellt wurden, den Typ `NUMBER(38)`. Eine Kurznamenspalte für diesen Oracle-Datentyp erhält den lokalen Datentyp `FLOAT`, weil die Werte einer Ganzzahl in DB2 den Bereich von 2^{*31} bis $(-2^{*31})-1$ umfassen, was in etwa dem Typ `NUMBER(9)` entspricht. In diesem Fall können Joins zwischen einer DB2-Spalte des Typs `INTEGER` und einer Oracle-Spalte des Typs `INTEGER` nicht in der DB2-Datenquelle stattfinden (kürzere Joinspalte). Wenn jedoch der Wertebereich dieser Oracle-Spalte des Typs `INTEGER` in dem DB2-Datentyp `INTEGER` untergebracht werden kann, ändern Sie den lokalen Datentyp der Spalte mit der Anweisung `ALTER NICKNAME`, sodass der Join an der DB2-Datenquelle stattfinden kann.

Spaltenoptionen: Verwenden Sie die SQL-Anweisung `ALTER NICKNAME`, um Spaltenoptionen für Kurznamen hinzuzufügen oder zu ändern.

Verwenden Sie die Option `varchar_no_trailing_blanks` zur Angabe einer Spalte, die keine folgenden Leerzeichen enthält. Der Pushdown-Analyseschritt des Compilers berücksichtigt diese Information bei der Prüfung aller Operationen für Spalten, die so markiert sind. Aufgrund dieser Angabe kann DB2 ein anderes, aber äquivalentes Format eines Vergleichselements generieren, das in einer fernen, an eine Datenquelle gesendete SQL-Anweisung zu verwenden ist. Ein Benutzer bemerkt vielleicht, dass an der Datenquelle ein anderes Vergleichselement ausgewertet wird, das Nettoergebnis sollte jedoch äquivalent sein.

Verwenden Sie die Option `numeric_string` zur Angabe, ob die Werte in der betreffenden Spalte immer Ziffern ohne folgende Leerzeichen sind.

In der folgenden Tabelle werden diese Optionen beschrieben.

Tabelle 59. Spaltenoptionen und zugehörige Einstellungen

Option	Gültige Einstellungen	Standard-einstellung
numeric_string	'Y' Der Wert 'Y' (Yes) gibt an, dass diese Spalte nur Zeichenfolgen mit numerischen Daten enthält. WICHTIG: Wenn die Spalte nur numerische Zeichenfolgen mit folgenden Leerzeichen enthält, geben Sie 'Y' nicht an. 'N' Der Wert 'N' (No) gibt an, dass diese Spalte nicht auf Zeichenfolgen mit numerischen Daten beschränkt ist. Wenn Sie die Spaltenoption numeric_string auf den Wert 'Y' setzen, teilen Sie dem Optimierungsprogramm mit, dass diese Spalte keine Leerzeichen enthält, die einen störenden Einfluss auf das Sortieren der Daten dieser Spalte haben könnten. Diese Option ist in solchen Fällen nützlich, in denen die Sortierfolge einer Datenquelle von der Sortierfolge in DB2 abweicht. Mit dieser Option markierte Spalten werden nicht von der lokalen Auswertung (der Datenquelle) aufgrund unterschiedlicher Sortierfolgen ausgeschlossen.	'N'
varchar_no_trailing_blanks	Gibt an, ob diese Datenquelle eine VARCHAR-Vergleichssemantik für nicht mit Leerzeichen aufgefüllte Zeichenfolgen verwendet. Für Zeichenfolgen variabler Länge, die keine folgenden Leerzeichen enthalten, liefert die Vergleichssemantik für nicht mit Leerzeichen aufgefüllte Zeichenfolgen einiger DBMSs die gleichen Ergebnisse wie die Vergleichssemantik von DB2. Wenn Sie sicher sind, dass alle VARCHAR-Spalten von Tabellen und Sichten an einer Datenquelle keine folgenden Leerzeichen enthalten, können Sie in Betracht ziehen, diese Serveroption auf den Wert 'Y' für eine Datenquelle zu setzen. Diese Option wird häufig für Datenquellen von Oracle verwendet. Stellen Sie sicher, dass Sie alle Objekte berücksichtigen, die Kurznamen haben können, einschließlich Sichten. 'Y' Diese Datenquelle verfügt über eine Vergleichssemantik für nicht mit Leerzeichen aufgefüllte Zeichenfolgen, die der Vergleichssemantik von DB2 ähnlich ist. 'N' Diese Datenquelle verfügt über keine mit DB2 vergleichbare Vergleichssemantik für nicht mit Leerzeichen aufgefüllte Zeichenfolgen.	'N'

Abfragemerkmale mit Auswirkung auf die Pushdown-Möglichkeiten

Eine Abfrage kann einen SQL-Operator enthalten, der Kurznamen aus verschiedenen Datenquellen mit einbezieht. Die Operation muss in DB2 stattfinden, um die Ergebnisse aus zwei angegebenen Datenquellen zu kombinieren, die nur einen Operator verwenden, wie zum Beispiel einen Gruppenoperator (z. B. UNION). Der Operator kann nicht direkt an der fernen Datenquelle ausgewertet werden.

Zugehörige Konzepte:

- „Richtlinien zur Analyse, wo eine Abfrage für föderierte Datenbanken ausgewertet wird“ auf Seite 249

Richtlinien zur Analyse, wo eine Abfrage für föderierte Datenbanken ausgewertet wird

DB2 stellt zwei Dienstprogramme bereit, damit Sie sehen, wo Abfragen ausgewertet werden:

- Visual Explain. Starten Sie dieses Programm über den Befehl **db2cc**. Dieses Programm dient zum Anzeigen des Zugriffsplandiagramms. Die Ausführungsposition für jeden Operator ist der detaillierten Anzeige für einen Operator zu entnehmen.

Wenn eine Abfrage an eine Datenquelle verschoben wird (Pushdown), sollten Sie einen Operator RETURN sehen. Der Operator RETURN ist ein Standardoperator von DB2. Für eine SELECT-Anweisung, die Daten aus einem Kurznamen auswählt, sehen Sie außerdem einen Operation SHIP. Der Operator SHIP ist für Operationen mit föderierten Datenbanken spezifisch. Er ändert das Servermerkmal des Datenflusses und trennt lokale Operatoren von fernen Operatoren. Die SELECT-Anweisung wird mit Hilfe der von der Datenquelle unterstützten SQL-Version generiert. Sie kann jede für die Datenquelle gültige Abfrage enthalten.

Wenn eine INSERT-, DELETE- oder UPDATE-Abfrage vollständig an die ferne Datenbank verschoben werden kann (Pushdown), wird eventuell keine SHIP-Anweisung im Zugriffsplan ausgewiesen. Alle fern ausgeführten INSERT-, DELETE- oder UPDATE-Anweisungen werden jedoch für den Operator RETURN gezeigt. Wenn eine Abfrage jedoch nicht vollständig verschoben werden kann, zeigt der Operator SHIP, welche Operationen fern durchgeführt wurden.

- SQL-EXPLAIN. Starten Sie dieses Programm mit dem Befehl **db2expln** oder **dynexpln**. Dieses Programm dient zur Erstellung einer Textausgabe des Zugriffsplans.

Verstehen, warum eine Abfrage an einer Datenquelle oder in DB2 ausgewertet wird

Berücksichtigen Sie die folgenden Schlüsselfragen, wenn Sie Methoden zur Ausweitung der Pushdown-Möglichkeiten untersuchen:

- Warum wird dieses Vergleichselement nicht fern ausgewertet?

Diese Frage erhebt sich, wenn ein Vergleichselement sehr selektiv ist und daher zum Filtern von Zeilen herangezogen werden und den Netzwerkverkehr verringern könnte. Die ferne Auswertung von Vergleichselementen wirkt sich auch darauf aus, ob ein Join zwischen zwei Tabellen derselben Datenquelle fern ausgewertet werden kann.

Zu untersuchende Bereiche sind:

- Vergleichselemente mit Unterabfragen. Enthält dieses Vergleichselement eine Unterabfrage, die sich auf eine andere Datenquelle bezieht? Enthält dieses Vergleichselement eine Unterabfrage mit einem SQL-Operator, der von dieser Datenquelle nicht unterstützt wird? Nicht alle Datenquellen unterstützen Gruppenoperatoren in einem Vergleichselement einer Unterabfrage.
- Funktionen in Vergleichselementen. Enthält dieses Vergleichselement eine Funktion, die von dieser fernen Datenquelle nicht ausgewertet werden kann? Relationale Operatoren werden als Funktionen klassifiziert.
- Bindeanforderungen von Vergleichselementen. Erfordert dieses Vergleichselement, wenn es fern ausgewertet werden soll, das Einbinden eines bestimmten Werts? Ist dies der Fall, würde der Wert die SQL-Einschränkungen an dieser Datenquelle verletzen?

- Globale Optimierung. Das Optimierungsprogramm kann entschieden haben, dass der Aufwand bei lokaler Verarbeitung geringer ist.
- Warum wird der Operator GROUP BY nicht fern ausgewertet?
Es gibt verschiedene Bereiche, die überprüft werden können:
 - Wird die Eingabe für den Operator GROUP BY fern ausgewertet? Ist dies zu verneinen, untersuchen Sie die Eingabe.
 - Besitzt die Datenquelle Einschränkungen für diesen Operator? Beispiele hierfür sind:
 - Begrenzte Anzahl von GROUP BY-Elementen
 - Begrenzte Byteanzahl kombinierter GROUP BY-Elemente
 - Spaltenspezifikation nur für die GROUP BY-Liste
 - Unterstützt die Datenquelle diesen SQL-Operator?
 - Globale Optimierung. Das Optimierungsprogramm kann entschieden haben, dass der Aufwand bei lokaler Verarbeitung geringer ist.
 - Enthält die Operator Klausel GROUP BY einen Zeichenausdruck? Ist dies der Fall, überprüfen Sie, ob die ferne Datenquelle die gleichen Einstellungen für die Beachtung der Groß-/Kleinschreibung wie DB2 verwendet.
- Warum wird der Gruppenoperator nicht fern ausgewertet?
Es gibt verschiedene Bereiche, die überprüft werden können:
 - Werden beide Operanden vollständig an derselben Datenquelle ausgewertet? Ist dies nicht der Fall, obwohl es der Fall sein sollte, untersuchen Sie jeden Operanden.
 - Besitzt die Datenquelle Einschränkungen für diesen Gruppenoperator? Sind zum Beispiel große Objekte oder Langfelder gültige Eingaben für diesen speziellen Gruppenoperator?
- Warum wird die Operation ORDER BY nicht fern ausgeführt?
Betrachten Sie Folgendes:
 - Wird die Eingabe für die Operation ORDER BY fern ausgewertet? Ist dies zu verneinen, untersuchen Sie die Eingabe.
 - Enthält die Klausel ORDER BY einen Zeichenausdruck? Wenn ja, besitzt die ferne Datenquelle nicht die gleiche Sortierfolge oder Einstellung zur Beachtung der Groß-/Kleinschreibung wie DB2?
 - Besitzt die Datenquelle Einschränkungen für diesen Operator? Gibt es beispielsweise eine begrenzte Anzahl von ORDER BY-Elementen? Schränkt die Datenquelle die Spaltenangaben für die ORDER BY-Liste ein?

Zugehörige Konzepte:

- „Informationen zu Abfragen auf föderierte Datenbanken“ auf Seite 736
- „Globale Analyse von Abfragen auf föderierte Datenbanken“ auf Seite 250
- „Richtlinien zur Zeichenkonvertierung“ in *Systemverwaltung: Konzept*
- „Generierung von fernem SQL und globale Optimierung in föderierten Datenbanken“ auf Seite 253

Globale Analyse von Abfragen auf föderierte Datenbanken

Die beiden folgenden Dienstprogramme sind zum Anzeigen globaler Zugriffspläne verfügbar:

- Visual Explain. Starten Sie dieses Programm über die Steuerzentrale oder führen Sie den Befehl *db2cc* aus, der die Steuerzentrale startet. Verwenden Sie Visual Explain zum Anzeigen des Zugriffsplandiagramms für eine Abfrage. Die Aus-

führungsposition für jeden Operator ist der detaillierten Anzeige für einen Operator zu entnehmen. Sie können darüber hinaus die ferne SQL-Anweisung, die für die jeweilige Datenquelle generiert wurde, je nach Art der Abfrage dem Operator SHIP oder RETURN entnehmen. Durch eine Untersuchung der Details für jeden Operator können Sie die Anzahl der Zeilen feststellen, die vom DB2-Optimierungsprogramm als Eingabe für den jeweiligen Operator und als Ausgabe aus diesem Operator geschätzt wird. Weiterhin können Sie den für die Ausführung des jeweiligen Operators geschätzten Aufwand, einschließlich des Kommunikationsaufwands, ablesen.

- SQL-EXPLAIN. Starten Sie das Programm mit dem Befehl *db2expln* oder *dynexpln*. SQL-EXPLAIN dient zur Erstellung einer Textausgabe des Zugriffsplans. Obwohl SQL-EXPLAIN keine Informationen über den Aufwand bietet, können Sie der Ausgabe den Zugriffsplan entnehmen, der von dem fernen Optimierungsprogramm für die Datenquellen generiert wurde, die von der fernen EXPLAIN-Funktion unterstützt werden.

Verstehen der Entscheidungen der DB2-Optimierung

Berücksichtigen Sie bei der Untersuchung von Leistungsverbesserungen die folgenden Fragen und Schlüsselbereiche der Optimierung:

- Warum wird ein Join zwischen zwei Kurznamen derselben Datenquelle nicht fern an der Datenquelle ausgewertet?

Zu untersuchende Bereiche sind:

- Joinoperationen. Werden sie von der Datenquelle unterstützt?
- Joinvergleichselemente. Können die Joinvergleichselemente an der fernen Datenquelle ausgewertet werden? Wenn dies zu verneinen ist, untersuchen Sie das Joinvergleichselement.
- Anzahl der Zeilen im Joinergebnis (mit Visual Explain). Ergibt sich aus dem Join eine wesentlich größere Gruppe von Zeilen als durch die Kombination der beiden Kurznamen? Ist die jeweilige Anzahl sinnvoll? Wenn dies zu verneinen ist, ziehen Sie eine manuelle Aktualisierung der Kurznamenstatistik (SYSSTAT.TABLES) in Betracht.

- Warum wird der Operator GROUP BY nicht fern ausgewertet?

Zu untersuchende Bereiche sind:

- Operatorsyntax. Stellen Sie sicher, dass der Operator an der fernen Datenquelle ausgewertet werden kann.
- Anzahl von Zeilen. Prüfen Sie die geschätzte Anzahl von Zeilen in der Eingabe und Ausgabe des Operators GROUP BY mit Hilfe von Visual Explain. Liegt die eine Anzahl sehr nahe an der anderen? Ist dies zu bejahen, betrachtet das DB2-Optimierungsprogramm es als effizienter, diesen Operator GROUP BY lokal auszuwerten. Ist die jeweilige Anzahl zudem sinnvoll? Wenn dies zu verneinen ist, ziehen Sie eine manuelle Aktualisierung der Kurznamenstatistik (SYSSTAT.TABLES) in Betracht.

- Warum wird die Anweisung nicht vollständig durch die ferne Datenquelle ausgewertet?

Das DB2-Optimierungsprogramm führt eine aufwandsorientierte Optimierung durch. Auch wenn die Pushdown-Analyse ergibt, dass jeder Operator an der fernen Datenquelle ausgewertet werden kann, stützt sich das Optimierungsprogramm bei der Generierung eines global optimalen Plans auf die Aufwandschätzung. Es gibt eine Vielzahl von Faktoren, die in diesen Plan einfließen können. Es ist beispielsweise möglich, dass eine ferne Datenquelle zwar jede einzelne Operation in der ursprünglichen Abfrage ausführen kann, aber die CPU-Geschwindigkeit der Datenquelle wesentlich langsamer ist als die CPU-

Geschwindigkeit des DB2-Systems, sodass es vorteilhafter ist, die Operationen in DB2 lokal auszuführen. Wenn die Ergebnisse nicht zufrieden stellend sind, überprüfen Sie die Serverstatistikdaten in SYSCAT.SERVEROPTIONS.

- Warum zeigt ein vom Optimierungsprogramm generierter Plan, der vollständig an einer fernen Datenquelle ausgewertet wird, eine wesentlich schlechtere Leistung als die Originalabfrage bei direkter Ausführung an der fernen Datenquelle?

Zu untersuchende Bereiche sind:

- Die vom DB2-Optimierungsprogramm generierte ferne SQL-Anweisung. Überprüfen Sie, ob sie mit der Originalabfrage identisch ist. Suchen Sie nach Änderungen in der Reihenfolge der Vergleichselemente. Ein gutes Abfrageoptimierungsprogramm sollte nicht von der Reihenfolge der Vergleichselemente einer Abfrage abhängig sein. Leider sind nicht alle DBMS-Optimierungsprogramme identisch, so dass das Optimierungsprogramm der fernen Datenquelle vielleicht aufgrund der Reihenfolge der Vergleichselemente in der Eingabe einen anderen Plan generiert. Wenn dies der Fall ist, liegt das Problem bei dem fernen Optimierungsprogramm. Sie können in diesem Fall entweder eine Änderung der Reihenfolge der Vergleichselemente in der Eingabe für DB2 in Betracht ziehen oder die Serviceorganisation der fernen Datenquelle um Unterstützung bitten.

Prüfen Sie darüber hinaus auf Ersetzungen von Vergleichselementen. Ein gutes Abfrageoptimierungsprogramm sollte nicht von der Ersetzung äquivalenter Vergleichselemente abhängig sein. Leider sind nicht alle DBMS-Optimierungsprogramme identisch, so dass das Optimierungsprogramm der fernen Datenquelle vielleicht aufgrund des Vergleichselements in der Eingabe einen anderen Plan generiert. Zum Beispiel können einige Optimierungsprogramme keine Anweisungen für Vergleichselemente unter Verwendung der Transitivität generieren.

- Die Anzahl der zurückgegebenen Zeilen. Diese Anzahl kann mit Hilfe von Visual Explain festgestellt werden. Wenn die Abfrage eine große Anzahl von Zeilen zurückgibt, kann der Netzwerkverkehr zum potenziellen Engpass werden.
- Zusätzliche Funktionen. Enthält die ferne SQL-Anweisung im Vergleich zur Originalabfrage zusätzliche Funktionen? Es können einige zusätzliche Funktionen zur Umsetzung von Datentypen generiert werden. Stellen Sie sicher, dass diese erforderlich sind.

Zugehörige Konzepte:

- „Beispiel 5: Zugriffsplan für eine föderierte Datenbank“ auf Seite 747
- „Pushdown-Analyse für föderierte Datenbanken“ auf Seite 244
- „Informationen zu Abfragen auf föderierte Datenbanken“ auf Seite 736
- „Richtlinien zur Analyse, wo eine Abfrage für föderierte Datenbanken ausgewertet wird“ auf Seite 249
- „Generierung von fernem SQL und globale Optimierung in föderierten Datenbanken“ auf Seite 253
- „Serveroptionen mit Einfluss auf föderierte Datenbanken“ auf Seite 256

Generierung von fernem SQL und globale Optimierung in föderierten Datenbanken

Für eine Abfrage auf eine föderierte Datenbank, die relationale Kurznamen verwendet, kann die Zugriffsstrategie vorsehen, die Originalabfrage in eine Gruppe ferner Abfrageeinheiten zu zerlegen und anschließend die Ergebnisse zu kombinieren. Diese Generierung von fernem SQL hilft bei der Herstellung einer global optimalen Zugriffsstrategie für eine Abfrage.

Das Optimierungsprogramm verwendet die Ausgabe der Pushdown-Analyse zur Entscheidung, ob die jeweilige Operation lokal in DB2 oder fern in einer Datenquelle ausgewertet wird. Es stützt die Entscheidung auf die Ausgabe des Aufwandsmodells, das nicht nur den Aufwand für die Auswertung der Operation, sondern auch den Aufwand für die Übertragung der Daten oder Nachrichten zwischen DB2 und den Datenquellen berücksichtigt.

Obwohl das Ziel die Erstellung einer optimierten Abfrage ist, wird die Ausgabe aus der globalen Optimierung und somit auch die Leistung der Abfrage von den folgenden Hauptfaktoren beeinflusst.

- Servermerkmale
- Kurznamenmerkmale

Servermerkmale und Optionen mit Auswirkung auf die globale Optimierung

Die folgenden Serverfaktoren einer Datenquelle können sich auf die globale Optimierung auswirken:

- Relatives Verhältnis der CPU-Geschwindigkeiten
Geben Sie mit Hilfe der Serveroption *cpu_ratio* an, wie schnell oder langsam die CPU-Geschwindigkeit der Datenquelle im Verhältnis zur CPU von DB2 ist. Ein niedriger Faktor bedeutet, dass die CPU des Computers der Datenquelle schneller als die CPU des DB2-Computers ist. Wenn der Faktor niedrig ist, zieht das DB2-Optimierungsprogramm mit größerer Wahrscheinlichkeit in Betracht, CPU-intensive Operationen in die Datenquelle zu verschieben (Pushdown).
- Relatives Verhältnis der E/A-Geschwindigkeiten
Geben Sie mit Hilfe der Serveroption *io_ratio* an, um wie viel schneller oder langsamer die E/A-Geschwindigkeit des Datenquellensystems im Verhältnis zum DB2-System ist. Ein niedriger Faktor bedeutet, dass die E/A-Geschwindigkeit der Workstation der Datenquelle schneller als die E/A-Geschwindigkeit der DB2-Workstation ist. Wenn der Faktor niedrig ist, zieht das DB2-Optimierungsprogramm in Betracht, E/A-intensive Operationen in die Datenquelle zu verschieben (Pushdown).
- Übertragungsgeschwindigkeit zwischen DB2 und der Datenquelle
Geben Sie mit Hilfe der Serveroption *comm_rate* die Netzwerkkapazität an. Langsame Geschwindigkeiten, die eine langsame Netzkommunikation zwischen DB2 und der Datenquelle bedeuten, veranlassen das DB2-Optimierungsprogramm, die Anzahl der Nachrichten zu reduzieren, die an die und von dieser Datenquelle gesendet werden. Wenn die Geschwindigkeit auf den Wert 0 gesetzt wird, erstellt das Optimierungsprogramm einen Zugriffsplan, der nur minimalen Netzwerkverkehr erfordert.
- Sortierfolge der Datenquelle
Geben Sie mit Hilfe der Serveroption *collating_sequence* an, ob die Sortierfolge einer Datenquelle mit der lokalen DB2-Datenbanksortierfolge übereinstimmt.

Wenn die Option nicht auf den Wert 'Y' gesetzt ist, betrachtet das Optimierungsprogramm die von dieser Datenquelle abgerufenen Daten als unsortiert.

- **Ferne Planhinweise**

Geben Sie mit Hilfe der Serveroption *plan_hints* an, dass Planhinweise generiert werden oder an einer Datenquelle verwendet werden sollen. Standardmäßig sendet DB2 keine Planhinweise an die Datenquelle.

Planhinweise sind Anweisungsfragmente, die zusätzliche Informationen für Optimierungsprogramme von Datenquellen bereitstellen. Für bestimmte Abfragen können diese Informationen die Leistung verbessern. Die Planhinweise können das Optimierungsprogramm der Datenquelle bei der Entscheidung unterstützen, ob ein Index, und wenn ja, welcher, zu verwenden ist oder nach welcher Reihenfolge beim Join von Tabellen vorzugehen ist.

Wenn Planhinweise aktiviert sind, enthält die Abfrage, die an die Datenquelle gesendet wird, zusätzliche Informationen. Zum Beispiel könnte eine Anweisung, die an ein Oracle-Optimierungsprogramm mit Planhinweisen gesendet wird, folgendermaßen aussehen:

```
SELECT /*+ INDEX (tabelle1, t1index)*/  
      coll  
FROM tabelle1
```

Der Planhinweis ist hier die Zeichenfolge `/*+ INDEX (tabelle1, t1index)*/`.

- **Informationen in der Wissensbasis des DB2-Optimierungsprogramms**

DB2 verfügt über eine Wissensbasis für das Optimierungsprogramm, die Daten über native Datenquellen enthält. Das DB2-Optimierungsprogramm generiert keine fernen Zugriffspläne, die nicht von bestimmten Datenbankverwaltungssystemen (DBMSs) generiert werden können. D. h., DB2 vermeidet die Generierung von Plänen, die von Optimierungsprogrammen an fernen Datenquellen nicht verstanden bzw. akzeptiert werden.

Kurznamenmerkmale mit Auswirkung auf die globale Optimierung

Die folgenden kurznamenspezifischen Faktoren können sich auf die globale Optimierung auswirken.

Indexinformationen: DB2 kann Informationen über Indizes an Datenquellen zur Optimierung von Abfragen heranziehen. Daher ist es wichtig, dass die für DB2 verfügbaren Indexinformationen aktuell sind. Die Indexinformationen für Kurznamen werden zu Anfang bei der Erstellung des Kurznamens erfasst. Indexinformationen werden für Sichtkurznamen nicht gesammelt.

Erstellen von Indexspezifikationen für Kurznamen: Sie können eine Indexspezifikation für einen Kurznamen erstellen. Indexspezifikationen erstellen eine Indexdefinition (keinen tatsächlichen Index) im Katalog, die vom DB2-Optimierungsprogramm verwendet werden kann. Indexspezifikationen werden mit der Anweisung `CREATE INDEX SPECIFICATION ONLY` erstellt. Die Syntax für die Erstellung einer Indexspezifikation für einen Kurznamen ist der Syntax zur Erstellung eines Index für eine lokale Tabelle ähnlich.

Ziehen Sie die Erstellung von Indexspezifikationen in folgenden Fällen in Betracht:

- DB2 kann während der Kurznamenerstellung keine Indexinformationen von einer Datenquelle abrufen.
- Sie wünschen einen Index für einen Sichtkurznamen.

- Sie möchten das DB2-Optimierungsprogramm zur Verwendung eines bestimmten Kurznamens als innere Tabelle eines Joins mit Verschachtelungsschleife veranlassen. Der Benutzer kann einen Index für die Joinspalte erstellen, falls keiner vorhanden ist.

Überlegen Sie vor dem Ausführen von Anweisungen `CREATE INDEX` für einen Kurznamen einer Sicht, ob Sie den Index benötigen. Wenn die Sicht eine einfache `SELECT`-Abfrage auf eine Tabelle mit einem Index ist, kann die Erstellung lokaler Indizes für den Kurznamen, die mit den Indizes für die Tabelle an der Datenquelle übereinstimmen, die Abfrageleistung erheblich erhöhen. Wenn Indizes jedoch lokal für Sichten erstellt werden, die keine einfachen `SELECT`-Anweisungen enthalten, zum Beispiel eine Sicht, die durch den Join zweier Tabellen erstellt wird, kann die Abfrageleistung sinken. Wenn Sie zum Beispiel einen Index für eine Sicht erstellen, die auf dem Join zweier Tabellen basiert, kann das Optimierungsprogramm diese Sicht möglicherweise als inneres Element in einem Join mit Verschachtelungsschleife (Nested Loop Join) wählen. Die Abfrage wird eine geringe Leistung erzielen, weil in diesem Fall der Join mehrere Male ausgewertet wird. Eine Alternative wäre, Kurznamen für jede einzelne Tabelle, auf die in der Sicht der Datenquelle verwiesen wird, zu erstellen und eine lokale Sicht unter DB2 zu definieren, die auf beide Kurznamen verweist.

Katalogstatistiken: Katalogstatistiken beschreiben die allgemeine Größe von Kurznamen und den Wertebereich in den zugehörigen Spalten. Das Optimierungsprogramm verwendet diese Statistikdaten, wenn es den Pfad mit dem geringsten Aufwand zur Verarbeitung von Abfragen berechnet, die Kurznamen enthalten. Die statistischen Daten über Kurznamen werden in den gleichen Katalogsichten wie Tabellenstatistiken gespeichert.

Obwohl DB2 die in einer Datenquelle gespeicherten statistischen Daten abrufen kann, kann es Aktualisierungen an vorhandenen statistischen Daten in Datenquellen nicht automatisch erkennen. Überdies kann DB2 Änderungen in Objektdefinitionen oder strukturelle Änderungen, wie das Hinzufügen einer Spalte, an Objekten von Datenquellen nicht verarbeiten. Wenn die Statistik- bzw. Strukturdaten für ein Objekt geändert wurden, haben Sie zwei Möglichkeiten:

- Führen Sie an der Datenquelle das Programm aus, das dem Dienstprogramm `RUNSTATS` entspricht. Löschen Sie anschließend den aktuellen Kurznamen und erstellen Sie ihn neu. Verwenden Sie diese Methode, wenn sich Strukturinformationen geändert haben.
- Aktualisieren Sie die Statistiken in der Sicht `SYSSTAT.TABLES` manuell. Diese Methode erfordert zwar weniger Schritte, funktioniert jedoch nicht, wenn sich Strukturinformationen geändert haben.

Zugehörige Konzepte:

- „Globale Analyse von Abfragen auf föderierte Datenbanken“ auf Seite 250
- „Richtlinien zur Analyse, wo eine Abfrage für föderierte Datenbanken ausgewertet wird“ auf Seite 249
- „Serveroptionen mit Einfluss auf föderierte Datenbanken“ auf Seite 256

Serveroptionen mit Einfluss auf föderierte Datenbanken

Ein föderiertes System besteht aus einem DB2-Datenbankverwaltungssystem (DBMS), d. h. der föderierten Datenbank, und mindestens einer Datenquelle. Sie geben die Datenquellen für die föderierte Datenbank an, wenn Sie Anweisungen CREATE SERVER absetzen. Beim Absetzen dieser Anweisungen können Sie Serveroptionen angeben, die bestimmte Aspekte des Betriebs des Systems mit der föderierten Datenbank bezüglich DB2 und der angegebenen Datenquelle eingrenzen und steuern. Wenn Sie später Serveroptionen ändern wollen, verwenden Sie Anweisungen ALTER SERVER.

Anmerkung: Sie müssen die Installationsoption für den verteilten Join installieren und den Parameter *federated* des Datenbankmanagers auf den Wert YES setzen, bevor Sie Server erstellen und Serveroptionen angeben können.

Die Werte der Serveroptionen, die Sie angeben, haben Einfluss auf die Pushdown-Analyse von Abfragen, die globale Optimierung und andere Aspekte von Operationen mit föderierten Datenbanken. Zum Beispiel können Sie in der Anweisung CREATE SERVER Leistungsstatistikdaten als Serveroptionswerte angeben, wie zum Beispiel die Option *cpu_ratio*, mit der die relativen Geschwindigkeiten der CPUs an der Datenquelle und auf dem Server der föderierten Datenbank angegeben werden. Sie könnten auch die Option *io_ratio* auf einen Wert setzen, der die relativen Übertragungsgeschwindigkeiten der E/A-Einheiten der Quelle und des Servers der föderierten Datenbank angibt. Bei der Ausführung der Anweisung CREATE SERVER werden diese Daten der Katalogsicht SYSCAT.SERVEROPTIONS hinzugefügt, und das Optimierungsprogramm bezieht sie in die Entwicklung eines Zugriffsplans für die Datenquelle mit ein. Falls sich ein Statistikwert ändert (wie es beispielsweise bei einer Aufrüstung der CPU der Datenquelle möglich ist), können Sie die Katalogsicht SYSCAT.SERVEROPTIONS mit Hilfe der Anweisung ALTER SERVER mit dieser Änderung aktualisieren. Das Optimierungsprogramm verwendet dann die neuen Informationen beim nächsten Auswählen eines Zugriffsplans für die Datenquelle.

Zugehörige Referenzen:

- „ALTER SERVER statement“ in *SQL Reference, Volume 2*
- „CREATE SEQUENCE statement“ in *SQL Reference, Volume 2*
- „federated - Unterstützung für Systeme föderierter Datenbanken “ auf Seite 569
- „SYSCAT.SERVEROPTIONS catalog view“ in *SQL Reference, Volume 1*

Kapitel 10. Analysieren von Zugriffsplänen mit der EXPLAIN-Funktion

EXPLAIN-Einrichtung

Der SQL- oder XQuery-Compiler kann Informationen über den Zugriffsplan und die Umgebung statischer oder dynamischer SQL- oder XQuery-Anweisungen erfassen. Die erfassten Informationen helfen Ihnen zu verstehen, wie einzelne SQL- oder XQuery-Anweisungen ausgeführt werden, sodass Sie die Anweisungen und die Konfiguration des Datenbankmanagers zur Leistungsverbesserung optimieren können.

Sie können EXPLAIN-Daten zu folgenden Zwecken erfassen und verwenden:

- Um zu verstehen, wie der Datenbankmanager auf Tabellen und Indizes zur Erfüllung Ihrer Abfrage zugreift.
- Um Ihre Maßnahmen zur Leistungsverbesserung zu beurteilen.

Wenn Sie einen Aspekt des Datenbankmanagers, der SQL- oder XQuery-Anweisungen oder der Datenbank ändern, sollten Sie die EXPLAIN-Daten untersuchen, um herauszufinden, wie sich Ihre Maßnahme auf die Leistung ausgewirkt hat.

Zu den erfassten Informationen gehören folgende:

- Informationen über die Reihenfolge der Operationen zur Verarbeitung der Abfrage
- Informationen über den Aufwand
- Vergleichselemente und Selektivitätsschätzwerte für jedes Vergleichselement
- Statistikdaten für alle Objekte, auf die in der SQL- oder XQuery-Anweisung zum Zeitpunkt der EXPLAIN-Datenerfassung verwiesen wird
- Werte für die Hostvariablen, Parametermarken oder Sonderregister, die zur Reoptimierung der SQL- oder XQuery-Anweisung verwendet werden

Bevor Sie EXPLAIN-Informationen erfassen können, erstellen Sie die relationalen Tabellen, in denen das Optimierungsprogramm die EXPLAIN-Informationen speichert, und definieren die Sonderregister, die die Art der zu erfassenden EXPLAIN-Informationen festlegen.

Zum Anzeigen von EXPLAIN-Informationen können Sie entweder ein Befehlszeilentool oder Visual Explain verwenden. Mit dem verwendeten Tool wird festgelegt, wie Sonderregister definiert werden, die festlegen, welche EXPLAIN-Daten erfasst werden. Wenn Sie zum Beispiel davon ausgehen, dass Sie nur Visual Explain verwenden, brauchen Sie nur Momentaufnahmeninformationen zu erfassen. Wenn Sie beabsichtigen, eine detaillierte Analyse mit einem der Befehlszeilendienstprogramme oder mit angepassten SQL- oder XQuery-Anweisungen für die EXPLAIN-Tabellen durchzuführen, sollten Sie alle EXPLAIN-Informationen erfassen.

Zugehörige Konzepte:

- „EXPLAIN-Tools“ auf Seite 258
- „Richtlinien zur Analyse von EXPLAIN-Informationen“ auf Seite 271
- „Richtlinien zur Erfassung von EXPLAIN-Informationen“ auf Seite 269

- „Richtlinien zur Verwendung von EXPLAIN-Informationen“ auf Seite 260
- „SQL- und XQuery-EXPLAIN-Tools“ auf Seite 711
- „Die EXPLAIN-Tabellen und die Organisation von EXPLAIN-Informationen“ auf Seite 261

dynexpln

Das Tool `dynexpln` ist aus Gründen der Abwärtskompatibilität weiterhin verfügbar. Sie können jedoch die **dynamischen-optionen** von `db2expln` verwenden, um alle Funktionen von `dynexpln` auszuführen.

Wenn Sie die dynamischen Optionen von `db2expln` verwenden, wird die Anweisung als wahre dynamische SQL- und XQuery-Anweisung vorbereitet (PREPARE), und der generierte Plan wird aus dem Abfragecache heraus mit EXPLAIN bearbeitet. Diese EXPLAIN-Ausgabemethode stellt genauere Zugriffspläne als das Tool `dynexpln` bereit, das die Anweisung als statische SQL- und XQuery-Anweisung vorbereitet. Sie ermöglicht darüber hinaus die Verwendung von Funktionen, die nur in dynamischen SQL- und XQuery-Anweisungen verfügbar sind, wie zum Beispiel Parametermarken.

Zugehörige Konzepte:

- „Beispiele für die Ausgabe von `db2expln` und `dynexpln`“ auf Seite 739
- „SQL- und XQuery-EXPLAIN-Tools“ auf Seite 711

EXPLAIN-Tools

DB2 stellt eine umfassende EXPLAIN-Einrichtung bereit, die ausführliche Informationen über den Zugriffsplan bereitstellt, den das Optimierungsprogramm für eine SQL- oder XQuery-Anweisung auswählt. Der Zugriff auf die Tabellen, in denen EXPLAIN-Daten, d. h., Informationen zu statischen und dynamischen SQL- oder XQuery-Anweisungen, gespeichert werden, ist auf allen unterstützten Plattformen möglich. Verschiedene Tools oder Methoden bieten flexible Möglichkeiten, die EXPLAIN-Informationen zu erfassen, anzuzeigen und zu analysieren.

Detaillierte Informationen des Optimierungsprogramms, die eine eingehende Analyse eines Zugriffsplans ermöglichen, werden getrennt vom eigentlichen Zugriffsplan in EXPLAIN-Tabellen gespeichert. Es stehen mehrere Methoden zum Abrufen von Informationen aus den EXPLAIN-Tabellen zur Verfügung:

- Visual Explain zum Anzeigen von EXPLAIN-Momentaufnahmen
Rufen Sie Visual Explain über die Steuerzentrale auf, um eine grafische Darstellung eines Abfragezugriffsplans anzuzeigen. Sie können sowohl statische als auch dynamische SQL- oder XQuery-Anweisungen analysieren.
Visual Explain ermöglicht Ihnen, auf einer anderen Plattform erfasste oder erstellte Momentaufnahmen anzuzeigen. Zum Beispiel kann ein Windows-Client Momentaufnahmen grafisch darstellen, die auf einem Server unter DB2 für HP-UX generiert wurden.
- Das Tool `db2exfmt` zum Anzeigen von EXPLAIN-Informationen in einer formatierten Ausgabe
- Die Tools `db2expln` und `dynexpln`
Zugriffsplaninformationen, die für ein oder mehrere Pakete mit statischen SQL- oder XQuery-Anweisungen verfügbar sind, können mit dem Tool `db2expln` über

die Befehlszeile angezeigt werden. Das Tool *db2expln* zeigt die tatsächliche Implementierung des gewählten Zugriffsplans. Informationen des Optimierungsprogramms werden nicht angezeigt.

Das Tool *dynexpln*, das intern auf *db2expln* zurückgreift, stellt ein zeiteffizientes Verfahren zur Bearbeitung dynamischer SQL- oder XQuery-Anweisungen mit EXPLAIN dar, die keine Parametermarken enthalten. Die Verwendung von *db2expln* innerhalb des Tools *dynexpln* wird durch die Umsetzung der SQL- oder XQuery-Eingabeanweisung in eine statische Anweisung innerhalb eines Pseudopakets ermöglicht. Wenn dieses Verfahren angewandt wird, sind die Informationen nicht immer ganz exakt. Wenn es auf Genauigkeit ankommt, verwenden Sie die EXPLAIN-Einrichtung.

Das Tool *db2expln* liefert einen relativ kompakten und englisch aufbereiteten Überblick über die Operationen, die bei der Ausführung stattfinden, indem der tatsächlich generierte Zugriffsplan analysiert wird.

- Schreiben eigener Abfragen für die EXPLAIN-Tabellen

Durch Schreiben eigener Abfragen können Sie die Ausgabe leicht bearbeiten und verschiedene Abfragen oder auch die gleiche Abfrage über einen Zeitraum hinweg vergleichen.

Anmerkung: Die EXPLAIN-Tools und andere Tools, die über die Befehlszeile ausgeführt werden, wie *db2batch*, *dynexpln* und *db2_all*, befinden sich im Unterverzeichnis *misc* des Verzeichnisses *sqllib*. Wenn die Tools aus diesem Pfad an eine andere Position versetzt werden, funktionieren die Befehlszeilenmethoden möglicherweise nicht.

Die folgende Tabelle enthält eine Übersicht über die verschiedenen, in der EXPLAIN-Einrichtung von DB2 verfügbaren Tools und ihre Merkmale. Verwenden Sie die Tabelle zur Auswahl des Tools, das für Ihre Anforderungen und Ihre Umgebung am besten geeignet ist.

Tabelle 60. Tools der EXPLAIN-Einrichtung

Merkmale	Visual Explain	EXPLAIN-Tabellen	db2exfmt	db2expln	dynexpln
Grafische Benutzerschnittstelle	Ja				
Textausgabe			Ja	Ja	Ja
Schnelle Grobanalyse für statisches SQL und XQuery				Ja	
Unterstützung für statisches SQL und XQuery	Ja	Ja	Ja	Ja	
Unterstützung für dynamisches SQL und XQuery	Ja	Ja	Ja	Ja	Ja*
Unterstützung für CLI-Anwendungen	Ja	Ja	Ja		
Verfügbar für DRDA-Anwendungsrequester		Ja			
Detaillierte Informationen des Optimierungsprogramms	Ja	Ja	Ja		
Geeignet zur Analyse mehrerer Anweisungen		Ja	Ja	Ja	Ja
Zugriff auf die Informationen aus einer Anwendung heraus		Ja			

Tabelle 60. Tools der EXPLAIN-Einrichtung (Forts.)

Merkmale	Visual Explain	EXPLAIN-Tabellen	db2exfmt	db2expln	dynexpln
Anmerkung:					
* Verwendet indirekt db2expln; es gelten einige Einschränkungen.					

Zugehörige Konzepte:

- „dynexpln“ auf Seite 258
- „Beschreibung der Ausgabe von db2expln und dynexpln“ auf Seite 719
- „Beispiele für die Ausgabe von db2expln und dynexpln“ auf Seite 739

Zugehörige Referenzen:

- Anhang F, „db2exfmt - EXPLAIN-Tabellen formatieren“, auf Seite 755
- „db2expln - SQL- und XQuery-EXPLAIN“ auf Seite 712

Richtlinien zur Verwendung von EXPLAIN-Informationen

EXPLAIN-Informationen dienen in erster Linie den beiden folgenden Zwecken:

- Analysieren der Gründe, aus denen sich die Anwendungsleistung geändert hat
- Beurteilen von Maßnahmen zur Leistungsoptimierung

Analysieren von Leistungsänderungen

Um sich ein besseres Verständnis für die Ursachen für Änderungen in der Abfrageleistung zu verschaffen, benötigen Sie EXPLAIN-Informationen vor und nach den Änderungen, die Sie durch die folgenden Schritte erhalten können:

- Erfassen Sie EXPLAIN-Informationen für die Abfrage, bevor Sie Änderungen vornehmen, und sichern Sie die resultierenden EXPLAIN-Tabellen. Alternativ könnten Sie auch die Ausgabe aus dem EXPLAIN-Tool db2exfmt sichern.
- Sichern oder drucken Sie die aktuellen Katalogstatistiken, wenn Sie zum Anzeigen der entsprechenden Informationen nicht auf Visual Explain zugreifen wollen oder können. Zu diesem Zweck können Sie auch das Produktivitätstool db2look verwenden.
- Sichern oder drucken Sie die Anweisungen der Datendefinitionssprache (DDL), einschließlich der Anweisungen für CREATE TABLE, CREATE VIEW, CREATE INDEX, CREATE TABLESPACE.

Die Informationen, die Sie auf diese Weise erfassen, stellen einen Referenzpunkt für zukünftige Analysen dar. Bei dynamischen SQL- oder XQuery-Anweisungen können Sie diese Informationen bei der ersten Ausführung Ihrer Anwendung erfassen. Bei statischen SQL- oder XQuery-Anweisungen können Sie diese Informationen auch beim Binden erfassen. Zur Analyse einer Leistungsänderung vergleichen Sie die Informationen, die Sie erfasst haben, mit Informationen, die Sie über die Abfrage und die Umgebung erfassen, wenn Sie Ihre Analyse beginnen.

Ein einfaches Beispiel wäre eine Analyse, die ergäbe, dass ein Index nicht mehr als Bestandteil des Zugriffspfads verwendet wird. Mit Hilfe der in Visual Explain angezeigten Informationen der Katalogstatistiken könnten Sie feststellen, dass die Anzahl von Indexstufen (Spalte NLEVELS) nun wesentlich höher ist als zu dem Zeitpunkt, als die Abfrage zum ersten Mal an die Datenbank gebunden wurde. Sie könnten in diesem Fall eine der folgenden Maßnahmen durchführen:

- Reorganisieren des Index
- Erfassen neuer Statistikdaten für die Tabelle und Indizes
- Sammeln von EXPLAIN-Informationen beim Rebind der Abfrage

Nach der Durchführung einer dieser Maßnahmen untersuchen Sie den Zugriffsplan erneut. Wenn der Index wieder verwendet wird, ist die Leistung der Abfrage vielleicht kein Problem mehr. Wenn der Index weiterhin nicht verwendet wird oder die Leistung problematisch bleibt, führen Sie eine zweite Maßnahme durch und untersuchen die Ergebnisse. Wiederholen Sie diese Schritte, bis das Problem gelöst ist.

Beurteilen von Maßnahmen zur Leistungsoptimierung

Sie haben die Möglichkeit, die Abfrageleistung durch eine Reihe von Maßnahmen zu verbessern, zu denen das Anpassen von Konfigurationsparametern, das Hinzufügen von Containern, das Erfassen aktueller Katalogstatistiken u. a. gehören.

Nach einer Änderung in einem dieser Bereiche können Sie mit der EXPLAIN-Einrichtung die Auswirkungen ermitteln, die die Änderung auf den ausgewählten Zugriffsplan hat. Wenn sie zum Beispiel einen Index oder eine MQT (Materialized Query Table, gespeicherte Abfragetabelle) gemäß den Richtlinien für Indizes hinzufügen, können Sie mit Hilfe der EXPLAIN-Daten feststellen, ob der Index oder die MQT tatsächlich in der erwarteten Weise genutzt wird.

Obwohl die EXPLAIN-Ausgabe Informationen liefert, die Ihnen ermöglichen, den ausgewählten Zugriffsplan und seinen relativen Aufwand zu ermitteln, besteht die einzige Möglichkeit, die Verbesserung der Leistung für eine Abfrage genau zu messen, in der Anwendung von Vergleichstesttechniken (Benchmark Tests).

Zugehörige Konzepte:

- „MQTs (MQT - Materialized Query Table, gespeicherte Abfragetabelle)“ auf Seite 11
- „Richtlinien zur Erfassung von EXPLAIN-Informationen“ auf Seite 269
- „EXPLAIN-Einrichtung“ auf Seite 257
- „Designadvisor“ auf Seite 51
- „Die EXPLAIN-Tabellen und die Organisation von EXPLAIN-Informationen“ auf Seite 261

Zugehörige Referenzen:

- Anhang F, „db2exfmt - EXPLAIN-Tabellen formatieren“, auf Seite 755

Die EXPLAIN-Tabellen und die Organisation von EXPLAIN-Informationen

Sämtliche EXPLAIN-Informationen sind nach dem Konzept einer EXPLAIN-Instanz organisiert. Eine EXPLAIN-Instanz stellt einen Aufruf der EXPLAIN-Einrichtung für eine oder mehrere SQL- oder XQuery-Anweisungen dar. Die EXPLAIN-Informationen, die in einer EXPLAIN-Instanz erfasst werden, umfassen die Umgebung der Kompilierung und den zur Ausführung der SQL- oder XQuery-Anweisung, die kompiliert wird, ausgewählten Zugriffsplan. Zum Beispiel kann eine EXPLAIN-Instanz aus jeder der folgenden Informationsgruppen bestehen:

- Alle in Frage kommenden SQL- oder XQuery-Anweisungen in einem Paket für statische Abfrageanweisungen. Bei SQL-Anweisungen (einschließlich derer, die

XML-Daten abfragen) können Sie EXPLAIN-Informationen für Anweisungen mit SELECT, SELECT INTO, UPDATE, INSERT, VALUES, VALUES INTO und DELETE erfassen. Bei XQuery-Anweisungen können Sie EXPLAIN-Informationen für Anweisungen mit XQUERY db2-fn:xmlcolumn und XQUERY db2-fn:sqlquery abrufen.

- Eine bestimmte SQL-Anweisung für SQL-Anweisungen zum inkrementellen Binden (Incremental Bind).
- Eine bestimmte SQL-Anweisung für dynamische SQL-Anweisungen.
- Jede SQL-Anweisung EXPLAIN (dynamisch oder statisch).

Die Informationen der EXPLAIN-Tabellen spiegeln die Beziehungen zwischen Operatoren und Datenobjekten im Zugriffsplan wieder. Die folgende Abbildung zeigt die Beziehungen zwischen diesen Tabellen.

EXPLAIN-Informationen werden in den folgenden Tabellen gespeichert:

Tabelle 61. Relationale Tabellen zur Speicherung von EXPLAIN-Daten

Tabellenname	Beschreibung
EXPLAIN_ARGUMENT	Enthält die spezifischen Merkmale für jeden einzelnen Operator (sofern vorhanden).
EXPLAIN_INSTANCE	Die Hauptsteuertabelle für alle EXPLAIN-Informationen. Jede Datenzeile in den EXPLAIN-Tabellen ist explizit mit einer eindeutigen Zeile in dieser Tabelle verbunden. In dieser Tabelle werden grundlegende Informationen über die Quelle der SQL- oder XQuery-Anweisungen, die mit EXPLAIN bearbeitet werden, und Umgebungsinformationen aufbewahrt.
EXPLAIN_OBJECT	Identifiziert die Datenobjekte, die für den Zugriffsplan erforderlich sind, der zur Erfüllung der SQL- oder XQuery-Anweisung generiert wurde.
EXPLAIN_OPERATOR	Enthält alle Operatoren, die zur Erfüllung der SQL- oder XQuery-Anweisung durch den Abfragecompiler erforderlich sind.
EXPLAIN_PREDICATE	Enthält Informationen über die Vergleichselemente, die von einem bestimmten Operator angewandt werden.
EXPLAIN_STATEMENT	Enthält den Text der SQL- oder XQuery-Anweisung, wie er für die verschiedenen Stufen der EXPLAIN-Informationen vorhanden ist. Die ursprüngliche, vom Benutzer eingegebene SQL- oder XQuery-Anweisung wird in dieser Tabelle mit der vom Optimierungsprogramm zum Auswählen eines Zugriffsplans verwendeten Version gespeichert. Wenn eine EXPLAIN-Momentaufnahme angefordert wird, werden zusätzliche EXPLAIN-Informationen aufgezeichnet, um den Zugriffsplan zu beschreiben, der vom Abfrageoptimierungsprogramm ausgewählt wurde. Diese Informationen werden in der Spalte SHAPSHOT der Tabelle EXPLAIN_STATEMENT in dem für Visual Explain erforderlichen Format gespeichert. Dieses Format kann von anderen Anwendungen nicht verwendet werden.
EXPLAIN_STREAM	Stellt die Eingabe- und Ausgabedatenströme zwischen einzelnen Operatoren und Datenobjekten dar. Die Datenobjekte selbst werden in der Tabelle EXPLAIN_OBJECT dargestellt. Die an einem Datenstrom beteiligten Operatoren werden in der Tabelle EXPLAIN_OPERATOR dargestellt.
EXPLAIN_DIAGNOSTIC	Enthält einen Eintrag für jede einzelne Diagnosenachricht, die für eine bestimmte Instanz einer EXPLAIN-Anweisung in der Tabelle EXPLAIN_STATEMENT erstellt wird.

Tabelle 61. Relationale Tabellen zur Speicherung von EXPLAIN-Daten (Forts.)

Tabellenname	Beschreibung
EXPLAIN_DIAGNOSTIC_DATA	Enthält Nachrichtentokens für bestimmte Diagnosenachrichten, die in der Tabelle EXPLAIN_DIAGNOSTIC aufgezeichnet werden. Die Nachrichtentokens enthalten weitere Informationen, die sich auf die Ausführung der durch die Nachricht generierten SQL-Anweisung beziehen.
ADVISE_WORKLOAD	Ermöglicht Benutzern, eine Auslastung für die Datenbank zu beschreiben. Jede Zeile in der Tabelle stellt eine SQL- oder XQuery-Anweisung in der Auslastung dar und wird durch eine zugeordnete Häufigkeit beschrieben. Das Tool db2advise verwendet diese Tabelle, um Auslastungsdaten zu erfassen und zu speichern.
ADVISE_INSTANCE	Enthält Informationen zur Ausführung von db2advise. Diese Informationen beinhalten auch die Startzeit. Für jede Ausführung von db2advise wird eine Zeile eingefügt.
ADVISE_INDEX	Speichert Informationen über empfohlene Indizes. Die Tabelle kann durch den Abfragecompiler, das Dienstprogramm db2advise oder einen Benutzer mit Daten gefüllt werden. Diese Tabelle wird zu zwei Zwecken verwendet: • Zum Abrufen empfohlener Indizes • Zum Bewerten von Indizes auf der Grundlage von Eingaben zu vorgeschlagenen Indizes
ADVISE_MQT	Enthält die CREATE-DDL-Anweisungen, die Abfrage, die die einzelnen MQTs (MQT - Materialized Query Table, gespeicherte Abfragetabelle) definiert, die Statistikdaten für die MQTs wie beispielsweise COLSTATS (pro Spalte) im XML-Format, NUMROWS usw. sowie die Stichprobenabfrage zum Abrufen von Stichprobenstatistiken für die MQT.
ADVISE_TABLE	Speichert die DDL zur Tabellenerstellung mit den endgültigen Empfehlungen des Designadvisors für empfohlene MQTs, MDC-Tabellen und Datenbankpartitionen in Abhängigkeit von den angegebenen Optionen und den generierten Empfehlungen.
ADVISE_PARTITION	Speichert von db2advise generierte und bewertete virtuelle Datenbankpartitionen.

Anmerkung: Nicht alle der oben genannten Tabellen werden standardmäßig erstellt. Zur Erstellung der Tabellen führen Sie das Script EXPLAIN.DDL aus, das sich im Unterverzeichnis *misc* des Unterverzeichnisses *sqllib* befindet.

EXPLAIN-Tabellen können für mehrere Benutzer gemeinsame Daten enthalten. Jedoch können die EXPLAIN-Tabellen auch für nur einen Benutzer definiert werden. Anschließend können Aliasnamen für jeden weiteren Benutzer definiert werden, der den gleichen Namen verwendet, um auf die definierten Tabellen zu verweisen. Die EXPLAIN-Tabellen können auch unter dem Schema SYSTOOLS definiert werden. Die EXPLAIN-Funktion verwendet standardmäßig das Schema SYSTOOLS, wenn keine anderen EXPLAIN-Tabellen oder -Aliasnamen unter der Sitzungs-ID des Benutzers für dynamische SQL- oder XQuery-Anweisungen oder unter der Berechtigungs-ID der Anweisung für statische SQL- oder XQuery-Anweisungen gefunden werden. Jeder Benutzer, der auf die gemeinsamen EXPLAIN-Tabellen zugreift, muss das Zugriffsrecht INSERT zum Einfügen für diese Tabellen aufweisen. Der Lesezugriff für die allgemeinen EXPLAIN-Tabellen muss ebenfalls eingeschränkt werden, speziell für Benutzer, die die EXPLAIN-Informationen analysieren.

Zugehörige Konzepte:

- „EXPLAIN-Informationen für Datenobjekte“ auf Seite 264
- „EXPLAIN-Informationen für Datenoperatoren“ auf Seite 268
- „EXPLAIN-Informationen für Instanzen“ auf Seite 264
- „EXPLAIN-Einrichtung“ auf Seite 257
- „SQL- und XQuery-EXPLAIN-Tools“ auf Seite 711

EXPLAIN-Informationen für Datenobjekte

Ein einziger Zugriffsplan kann zur Erfüllung der SQL- oder XQuery-Anweisung ein oder mehrere Datenobjekte verwenden.

Objektstatistiken: Die EXPLAIN-Einrichtung zeichnet Informationen über das Objekt auf, wie z. B. folgende Angaben:

- Die Erstellungszeit
- Den Zeitpunkt, an dem zum letzten Mal Statistiken für das Objekt gesammelt wurden
- Eine Angabe, ob die Daten im Objekt sortiert sind (nur Tabellen- oder Indexobjekte)
- Die Anzahl von Spalten im Objekt (nur Tabellen- oder Indexobjekte)
- Die geschätzte Anzahl von Zeilen im Objekt (nur Tabellen- oder Indexobjekte)
- Die Anzahl von Seiten, die das Objekt im Pufferpool einnimmt
- Den geschätzten Gesamtaufwand in Millisekunden für eine einzelne wahlfreie Ein-/Ausgabeoperation für den angegebenen Tabellenbereich, in dem dieses Objekt gespeichert ist
- Die geschätzte Übertragungsrate in Millisekunden zum Lesen einer 4-KB-Seite aus dem angegebenen Tabellenbereich
- Die Größen für PREFETCHSIZE und EXTENTSIZE (in 4-KB-Seiten)
- Den Grad der Datenclusterbildung in Bezug auf den Index
- Die Anzahl von Blattseiten (Leaf Pages), die vom Index für dieses Objekt verwendet werden, und die Anzahl der Indexstufen in der Indexbaumstruktur
- Die Anzahl unterschiedlicher vollständiger Schlüsselwerte im Index für dieses Objekt
- Die Gesamtzahl der Überlaufsätze in der Tabelle

Zugehörige Konzepte:

- „EXPLAIN-Informationen für Datenoperatoren“ auf Seite 268
- „EXPLAIN-Informationen für Instanzen“ auf Seite 264
- „Richtlinien zur Analyse von EXPLAIN-Informationen“ auf Seite 271
- „Die EXPLAIN-Tabellen und die Organisation von EXPLAIN-Informationen“ auf Seite 261

EXPLAIN-Informationen für Instanzen

Informationen über EXPLAIN-Instanzen werden in der Tabelle EXPLAIN_INSTANCE gespeichert. Zusätzliche bestimmte Informationen zu jeder Abfrageanweisung in einer Instanz werden in der Tabelle EXPLAIN_STATEMENT gespeichert.

Kennzeichnung von EXPLAIN-Instanzen: Anhand der in den folgenden Punkten bereitgestellten Informationen können Sie jede EXPLAIN-Instanz eindeutig identifizieren und die Informationen für Abfrageanweisungen einem bestimmten Aufruf der Einrichtung zuordnen:

- Der Benutzer, der die EXPLAIN-Informationen anforderte
- Der Zeitpunkt, zu dem die EXPLAIN-Anforderung begann
- Der Name des Pakets, das die mit EXPLAIN bearbeitete Abfrageanweisung enthält
- Das SQL-Schema des Pakets, das die mit EXPLAIN bearbeitete Abfrageanweisung enthält
- Die Version des Pakets, das die Anweisung enthält
- Eine Angabe, ob Momentaufnahmendaten (Snapshot) erfasst wurden

Einstellungen der Umgebung: Es werden Informationen zur Umgebung des Datenbankmanagers erfasst, in der der Abfragecompiler die Abfragen optimiert hat. Zu den Umgebungsinformationen gehören die folgenden:

- Die Versions- und Releasenummer für die DB2-Stufe
- Der Grad der Parallelität, für den die Abfrage kompiliert wurde
Das Sonderregister CURRENT DEGREE, die Bindeoption DEGREE, die API SET RUNTIME DEGREE und der Konfigurationsparameter *dft_degree* bestimmen den Grad der Parallelität, für den eine bestimmte Abfrage kompiliert wird.
- Ob die Abfrageanweisung dynamisch oder statisch ist
- Die zum Kompilieren der Abfrage verwendete Optimierungsklasse
- Der Typ der Zeilenblockung für Cursor, der beim Kompilieren der Abfrage angegeben wurde
- Die Isolationsstufe, in der die Abfrage ausgeführt wird
- Die Werte verschiedener Konfigurationsparameter zum Zeitpunkt der Kompilierung der Abfrage. Die folgenden Parameter werden bei der Erfassung einer EXPLAIN-Momentaufnahme (Snapshot) aufgezeichnet:
 - Zwischenspeicher für Sortierlisten (*sortheap*)
 - Durchschnittliche Anzahl aktiver Anwendungen (*avg_appls*)
 - Datenbankzwischenspeicher (*dbheap*)
 - Maximaler Speicher für Sperrenliste (*locklist*)
 - Maximale Anzahl Sperren pro Anwendung (*maxlocks*)
 - CPU-Geschwindigkeit (*cpuspeed*)
 - Kommunikationsbandbreite (*comm_bandwidth*)

Kennzeichnung von Anweisungen: Für jede EXPLAIN-Instanz können mehrere Abfrageanweisungen mit EXPLAIN bearbeitet werden. Zusätzlich zu den Informationen, die die EXPLAIN-Instanz eindeutig kennzeichnen, kann jede einzelne Abfrageanweisung anhand der folgenden Informationen identifiziert werden:

- Der Typ der Anweisung: SELECT, DELETE, INSERT, UPDATE, positioniertes DELETE, positioniertes UPDATE
- Die Anweisungs- und Abschnittsnummer des Pakets, das die Abfrageanweisung absetzt, die in der Katalogsicht SYSCAT.STATEMENTS aufgezeichnet wurde

Die Felder QUERYTAG und QUERYNO in der Tabelle EXPLAIN_STATEMENT enthalten Kennungen, die im Rahmen des EXPLAIN-Prozesses mit Werten gefüllt werden. Für dynamische EXPLAIN-Abfrageanweisungen, die in einer CLP- oder CLI-Sitzung übergeben wurden, wird QUERYTAG auf „CLP“ bzw. „CLI“ gesetzt,

wenn EXPLAIN MODE oder EXPLAIN SNAPSHOT aktiv ist. In diesem Fall wird für QUERYNO standardmäßig eine Nummer angegeben, die mindestens um den Wert 1 für jede Anweisung erhöht wird. Für alle anderen dynamischen EXPLAIN-Abfrageanweisungen, die nicht über CLP bzw. CLI angefordert werden oder die EXPLAIN-Abfrageanweisung nicht verwenden, wird das Feld QUERYTAG mit Leerzeichen gefüllt, und QUERYNO wird immer auf „1“ gesetzt.

Schätzung des Aufwands: Für jede mit EXPLAIN bearbeitete Anweisung zeichnet das Optimierungsprogramm einen Schätzwert für den relativen Aufwand zur Ausführung des ausgewählten Zugriffsplans auf. Dieser Aufwand wird in einer künstlich geschaffenen relativen Maßeinheit mit der Bezeichnung *timeron* angegeben. Schätzwerte für die benötigten Ausführungszeiten werden aus folgenden Gründen nicht bereitgestellt:

- Das Abfrageoptimierungsprogramm schätzt nicht die benötigte Zeit, sondern nur den Ressourcenbedarf ab.
- Das Optimierungsprogramm modelliert nicht alle Faktoren nach, die die benötigte Zeit beeinflussen können. Es ignoriert Faktoren, die keine Auswirkung auf die Effizienz des Zugriffsplans haben. Die benötigte Zeit wird von einer Reihe Laufzeitfaktoren beeinflusst, zu denen die folgenden gehören: die Systemauslastung, die Ressourcenverfügbarkeit, der Umfang der Parallelverarbeitung und der Ein-/Ausgabeoperationen, der Aufwand für die Rückgabe von Zeilen an den Benutzer sowie die Übertragungszeit zwischen Client und Server.

Anweisungstext: Für jede mit EXPLAIN bearbeitete Anweisung werden zwei Versionen des Texts der Abfrageanweisung aufgezeichnet. Eine Version ist der Code, den der Abfragecompiler von der Anwendung empfängt. Die andere Version ist die rückübersetzte Version aus der compilerinternen Darstellung der Abfrage. Obwohl diese Übersetzung anderen Abfrageanweisungen sehr ähnlich ist, entspricht sie nicht unbedingt der richtigen Syntax der Abfragesprache und spiegelt nicht in jedem Fall den tatsächlichen Inhalt der internen Darstellung als Ganzes wider. Diese Übersetzung wird nur zur Verfügung gestellt, um Ihnen einen Einblick in den Kontext zu geben, in dem das SQL- oder XQuery-Optimierungsprogramm den Zugriffsplan ausgewählt hat. Um zu verstehen, wie der SQL- und XQuery-Compiler Ihre Abfrage zur Optimierung umgeschrieben hat, vergleichen Sie den vom Benutzer geschriebenen Anweisungstext mit der internen Darstellung der Abfrageanweisung. Die umgeschriebene Anweisungen zeigt Ihnen außerdem noch weitere Elemente in der Umgebung, die sich auf Ihre Anweisung auswirken, wie zum Beispiel Trigger und Integritätsbedingungen. Einige Schlüsselwörter, die in diesem „optimierten“ Text verwendet werden, sind:

\$C_n	Der Name einer abgeleiteten Spalte, wobei <i>n</i> ein ganzzahliger Wert ist.
\$CONSTRAINT\$	Diese Kennung weist auf den Namen einer Integritätsbedingung hin, die der ursprünglichen Abfrageanweisung bei der Kompilierung hinzugefügt wurde. Es tritt in Kombination mit dem Präfix \$WITH_CONTEXT\$ auf.
\$DERIVED.T_n	Der Name einer abgeleiteten Tabelle, wobei <i>n</i> ein ganzzahliger Wert ist.
\$INTERNAL_FUNC\$	Diese Kennung weist auf das Vorhandensein einer Funktion hin, die vom SQL- und XQuery-Compiler für die mit EXPLAIN bearbeitete Abfrage verwendet wurde, jedoch nicht zur allgemeinen Verwendung verfügbar ist.

\$INTERNAL_PRED\$	Diese Kennung weist auf das Vorhandensein eines Vergleichselements hin, das vom SQL- und XQuery-Compiler bei der Kompilierung der mit EXPLAIN bearbeiteten Abfrage hinzugefügt wurde, jedoch nicht zur allgemeinen Verwendung verfügbar ist. Vom Compiler wird ein internes Vergleichselement verwendet, um den infolge von Triggern und Integritätsbedingungen der ursprünglichen Abfrageanweisung hinzugefügten Zusatzkontext zu erfüllen.
\$INTERNAL_XPATH\$	Zeigt eine interne Tabellenfunktion an, die als Parameter ein einziges eingegebenes, mit Anmerkungen versehenes XPath-Muster enthält und eine Tabelle mit mindestens einer Spalte zurückgibt, die mit dem Muster abgeglichen wird.
\$RID\$	Dieses Kennzeichen dient zur Identifizierung der Spalte der Satz-ID (RID) für eine bestimmte Zeile.
\$TRIGGER\$	Diese Kennung weist auf den Namen eines Triggers hin, der der ursprünglichen Abfrageanweisung bei der Kompilierung hinzugefügt wurde. Es tritt in Kombination mit dem Präfix \$WITH_CONTEXT\$ auf.
\$WITH_CONTEXT\$(...)	Dieses Präfix tritt am Anfang des Texts auf, wenn zusätzliche Trigger oder Integritätsbedingungen in die ursprüngliche Abfrageanweisung eingefügt wurden. Diesem Präfix folgt eine Liste der Namen aller Trigger oder Integritätsbedingungen, die sich auf die Kompilierung und Auflösung der Abfrageanweisung auswirken.

Zugehörige Konzepte:

- „EXPLAIN-Informationen für Datenobjekte“ auf Seite 264
- „EXPLAIN-Informationen für Datenoperatoren“ auf Seite 268
- „Richtlinien zur Analyse von EXPLAIN-Informationen“ auf Seite 271
- „Die EXPLAIN-Tabellen und die Organisation von EXPLAIN-Informationen“ auf Seite 261

Zugehörige Referenzen:

- „avg_appls - Durchschnittliche Anzahl aktiver Anwendungen “ auf Seite 484
- „comm_bandwidth - Kommunikationsbandbreite “ auf Seite 566
- „cpuspeed - CPU-Geschwindigkeit “ auf Seite 567
- „dbheap - Zwischenspeicher für Datenbank “ auf Seite 437
- „dft_degree - Standardgrad der Parallelität “ auf Seite 541
- „maxlocks - Maximale Anzahl Sperren pro Anwendung “ auf Seite 474
- „locklist - Maximaler Speicher für Sperrenliste “ auf Seite 438
- „sortheap - Zwischenspeichergröße für Sortierlisten “ auf Seite 458

EXPLAIN-Informationen für Datenoperatoren

Ein einziger Zugriffsplan kann mehrere Operationen mit den Daten ausführen, um die SQL- oder XQuery-Anweisung zu erfüllen und Ihnen die Ergebnisse bereitzustellen. Der Abfragecompiler ermittelt, welche Operationen erforderlich sind, wie zum Beispiel eine Tabellensuche, eine Indexsuche, ein Join mit Verschachtelungsschleife oder ein GROUP BY-Operator.

Neben den Operatoren, die in einem Zugriffsplan verwendet werden, und den Informationen zu jedem einzelnen Operator zeigen EXPLAIN-Informationen auch die kumulativen Auswirkungen des Zugriffsplans.

Informationen über den geschätzten Aufwand: Für die Operatoren können die folgenden geschätzten kumulativen Aufwände angezeigt werden. Diese Aufwände beziehen sich auf den ausgewählten Zugriffsplan bis zu dem Operator (einschließlich), für den die Informationen erfasst wurden.

- Der Gesamtaufwand (in Timerons)
- Die Anzahl der Seiten-E/As
- Die Anzahl von CPU-Instruktionen
- Der Aufwand (in Timerons) für das Abrufen der ersten Zeile einschließlich eines evtl. zusätzlich erforderlichen einleitenden Systemaufwands
- Der Übertragungsaufwand (in Rahmen (Frames))

Timerons ist die Bezeichnung für eine künstlich geschaffene relative Maßeinheit. Timerons werden vom Optimierungsprogramm auf der Basis interner Werte bestimmt, zum Beispiel Statistikdaten, die sich aufgrund der Datenbanknutzung ändern. Daher gibt es keine Garantie, dass das Timerons-Maß für eine SQL- oder XQuery-Anweisung bei jeder Ermittlung der geschätzten Kosten in Timerons gleich ist.

Operatoreigenschaften: Die folgenden Informationen werden von der EXPLAIN-Einrichtung aufgezeichnet, um die Eigenschaften jedes Operators zu beschreiben:

- Die Menge der Tabellen, auf die zugegriffen wurde
- Die Menge der Spalten, auf die zugegriffen wurde
- Die Spalten, nach denen die Daten sortiert werden, wenn das Optimierungsprogramm ermittelt hat, dass diese Sortierung von nachfolgenden Operatoren verwendet werden kann
- Die Menge der Vergleichselemente, die angewandt wurden
- Die geschätzte Anzahl von Zeilen, die übergeben werden sollen (Kardinalität)

Zugehörige Konzepte:

- „EXPLAIN-Informationen für Datenobjekte“ auf Seite 264
- „EXPLAIN-Informationen für Instanzen“ auf Seite 264
- „Richtlinien zur Analyse von EXPLAIN-Informationen“ auf Seite 271
- „Die EXPLAIN-Tabellen und die Organisation von EXPLAIN-Informationen“ auf Seite 261

Richtlinien zur Erfassung von EXPLAIN-Informationen

EXPLAIN-Daten werden bei der Kompilierung einer SQL- oder XQuery-Anweisung erfasst, wenn Sie dies anfordern. Bei der Anforderung von EXPLAIN-Daten sollten Sie berücksichtigen, wie die erfassten Informationen später verwendet werden sollen.

Anmerkungen:

1. Wenn SQL- oder XQuery-Anweisungen zum inkrementellen Binden während der Ausführung kompiliert werden, werden Daten während der Bearbeitungszeit, nicht während der Bindezeit, in die EXPLAIN-Tabellen eingetragen. Für solche Anweisungen sind das Qualifikationsmerkmal und die Berechtigungs-ID, die in die EXPLAIN-Tabelle eingetragen werden, die des Paketeigners und nicht die des Benutzers, der das Paket ausführt.
2. Die EXPLAIN-Informationen werden nur erfasst, wenn die SQL- oder XQuery-Anweisung kompiliert wird. Nach der Erstkompilierung werden dynamische Abfrageanweisungen erneut kompiliert, wenn eine Änderung an der Umgebung dies erfordert oder wenn die EXPLAIN-Einrichtung aktiv ist. Wenn Sie dieselbe Anweisung PREPARE mehrmals für die gleiche Abfrageanweisung ausführen, erfolgt das Kompilieren der Anweisung und das Erfassen der EXPLAIN-Daten jedes Mal, wenn diese Anweisung vorbereitet oder ausgeführt wird.
3. Wenn ein Paket mit der Bindeoption REOPT ONCE/ALWAYS gebunden wird, werden SQL- oder XQuery-Anweisungen, die Hostvariablen, Parametermarken oder Sonderregister enthalten, kompiliert; der Zugriffspfad wird mit den realen Werten dieser Variablen, wenn sie bekannt sind, und mit Standardschätzwerten, wenn die Werte nicht bekannt sind, erstellt.
4. Wenn die Klausel FOR REOPT ONCE verwendet wird, wird ein Versuch unternommen, die angegebene SQL- oder XQuery-Anweisung mit der gleichen Anweisung im Paketcache abzugleichen. Die Werte dieser bereits reoptimierten Abfrageanweisung im Cache werden zur Reoptimierung der angegebenen Abfrageanweisung verwendet. Die EXPLAIN-Tabellen enthalten in diesem Fall den neu generierten, reoptimierten Zugriffsplan und die Werte, die für diese Reoptimierung verwendet wurden, sofern der Benutzer über die erforderlichen Zugriffsrechte verfügt.
5. In einem System mit mehreren Partitionen muss die Anweisung in der gleichen Datenbankpartition mit EXPLAIN bearbeitet werden, in der sie auch ursprünglich kompiliert und mit REOPT ONCE reoptimiert wurde. Ansonsten wird ein Fehler zurückgegeben.

Erfassen von Informationen in den EXPLAIN-Tabellen

- **Statische SQL- und XQuery-Anweisungen oder SQL- und XQuery-Anweisungen zum inkrementellen Binden:**

Geben Sie für die Befehle BIND oder PREP die Optionen EXPLAIN ALL oder EXPLAIN YES an, oder fügen Sie eine statische EXPLAIN-Anweisung in das Quellenprogramm ein.

- **Dynamische SQL- und XQuery-Anweisungen:**

Informationen für EXPLAIN-Tabellen werden in folgenden Fällen erfasst:

- Das Sonderregister CURRENT EXPLAIN MODE enthält einen der folgenden Werte:
 - YES: Der SQL- und XQuery-Compiler erfasst EXPLAIN-Daten und führt die Abfrageanweisung aus.

- EXPLAIN: Der SQL- und XQuery-Compiler erfasst EXPLAIN-Daten, führt die Abfrageanweisung jedoch nicht aus.
- RECOMMEND INDEXES: Der SQL- und XQuery-Compiler erfasst EXPLAIN-Daten und die Daten zu den empfohlenen Indizes werden in der Tabelle ADVISE_INDEX gespeichert; die Abfrageanweisung wird allerdings nicht ausgeführt.
- EVALUATE INDEXES: Der SQL- und XQuery-Compiler verwendet die vom Benutzer in die Tabelle ADVISE_INDEX eingefügten Indizes zur Bewertung. Im Modus EVALUATE INDEXES werden alle dynamischen Anweisungen mit EXPLAIN so bearbeitet, als wären diese virtuellen Indizes verfügbar. Der Abfragecompiler wählt anschließend die virtuellen Indizes aus, wenn sie die Leistung der Anweisungen verbessern. Ansonsten werden die Indizes ignoriert. Durch eine Analyse der EXPLAIN-Ergebnisse können Sie feststellen, ob die vorgeschlagenen Indizes nützlich wären.
- REOPT: Der Abfragecompiler erfasst während der Ausführung EXPLAIN-Daten für statische oder dynamische SQL- oder XQuery-Anweisungen bei der Reoptimierung der Anweisung, wenn für die Hostvariablen, Sonderregister oder Parametermarken tatsächliche Werte zur Verfügung stehen.
- Die Option EXPLAIN ALL wurde im Befehl BIND oder PREP angegeben. Der Abfragecompiler erfasst während der Ausführung EXPLAIN-Daten für dynamisches SQL und XQuery, selbst wenn das Sonderregister CURRENT EXPLAIN MODE auf NO gesetzt ist. Die SQL- oder XQuery-Anweisung wird ebenfalls ausgeführt und gibt die Abfrageergebnisse zurück.

Erfassen von EXPLAIN-Momentaufnahmen

Wenn eine EXPLAIN-Momentaufnahme (Snapshot) angefordert wird, werden EXPLAIN-Informationen in der Spalte SNAPSHOT der Tabelle EXPLAIN_STATEMENT in dem für Visual Explain erforderlichen Format gespeichert. Dieses Format kann von anderen Anwendungen nicht verwendet werden. Weitere Informationen zum Inhalt der Informationen der EXPLAIN-Momentaufnahmen erhalten Sie direkt mit Hilfe von Visual Explain. Zu diesen Informationen gehören Informationen zu Datenobjekten und Datenoperatoren.

EXPLAIN-Momentaufnahmedaten werden erfasst, wenn eine SQL- oder XQuery-Anweisung kompiliert wird und EXPLAIN-Daten wie folgt angefordert wurden:

- **Statische SQL-Anweisungen oder SQL- und XQuery-Anweisungen zum inkrementellen Binden:**

Eine EXPLAIN-Momentaufnahme wird erfasst, wenn eine der beiden Klauseln EXPLSNAP ALL oder EXPLSNAP YES für die Befehle BIND oder PREP angegeben wird oder wenn das Quellenprogramm eine statische Anweisung EXPLAIN mit der Klausel FOR SNAPSHOT oder WITH SNAPSHOT enthält.

- **Dynamische SQL- und XQuery-Anweisungen:**

Eine EXPLAIN-Momentaufnahme wird in folgenden Fällen erfasst:

- Sie führen eine Anweisung EXPLAIN mit der Klausel FOR SNAPSHOT oder WITH SNAPSHOT aus. Mit der Klausel FOR SNAPSHOT werden nur Momentaufnahmedaten erfasst. Mit der Klausel WITH SNAPSHOT werden neben den Momentaufnahmedaten auch sämtliche anderen EXPLAIN-Informationen erfasst.
- Das Sonderregister CURRENT EXPLAIN SNAPSHOT enthält einen der folgenden Werte:
 - YES: Der Abfragecompiler erfasst EXPLAIN-Momentaufnahmedaten und führt die SQL- oder XQuery-Anweisung aus.

- EXPLAIN: Der Abfragecompiler erfasst EXPLAIN-Momentaufnahmedaten, führt die SQL- oder XQuery-Anweisung jedoch nicht aus.
- Sie geben die Option EXPLSNAP ALL im Befehl BIND oder PREP an. Der Abfragecompiler erfasst EXPLAIN-Momentaufnahmedaten bei der Ausführung, selbst wenn das Sonderregister CURRENT EXPLAIN SNAPSHOT auf NO gesetzt ist. Außerdem wird die SQL- oder XQuery-Anweisung ausgeführt.

Zugehörige Konzepte:

- „Designadvisor“ auf Seite 51
- „Richtlinien zur Analyse von EXPLAIN-Informationen“ auf Seite 271
- „Richtlinien zur Verwendung von EXPLAIN-Informationen“ auf Seite 260
- „EXPLAIN-Einrichtung“ auf Seite 257
- „SQL- und XQuery-EXPLAIN-Tools“ auf Seite 711
- „Die EXPLAIN-Tabellen und die Organisation von EXPLAIN-Informationen“ auf Seite 261

Richtlinien zur Analyse von EXPLAIN-Informationen

Der primäre Zweck der EXPLAIN-Informationen ist die Analyse des Zugriffspaths für Abfrageanweisungen. Die Analyse der EXPLAIN-Daten kann Ihnen auf verschiedene Weise helfen, Ihre Abfragen und Ihre Umgebung zu optimieren. Betrachten Sie die folgende Art einer Analyse:

• Verwendung von Indizes

Die geeigneten Indizes können die Leistung erheblich fördern. Anhand der EXPLAIN-Ausgabe können Sie ermitteln, ob die von Ihnen für eine bestimmte Gruppe von Abfragen erstellten Indizes tatsächlich verwendet werden. In der EXPLAIN-Ausgabe sollten Sie folgende Bereiche auf die Verwendung von Indizes hin überprüfen:

- Joinvergleichselemente
- Lokale Vergleichselemente
- Die Klausel GROUP BY
- Die Klausel ORDER BY
- Die Klausel WHERE XMLEXISTS
- Die SELECT-Liste

Sie können die EXPLAIN-Einrichtung auch verwenden, um zu ermitteln, ob ein anderer Index anstelle des vorhandenen Index oder kein Index verwendet werden könnte. Führen Sie nach der Erstellung eines neuen Index den Befehl RUNSTATS aus, um Statistikdaten für diesen Index zu erfassen, und kompilieren Sie die Abfrage erneut. Mit der Zeit werden Sie möglicherweise durch die EXPLAIN-Daten feststellen, dass anstelle einer Indexsuche eine Tabellensuche verwendet wird. Dies kann sich aus einer Änderung in der Clusterbildung der Tabellendaten ergeben. Wenn der zuvor verwendete Index nun ein niedriges Clusterverhältnis aufweist, sollten Sie in Erwägung ziehen, die Tabelle zu reorganisieren, um ihre Daten in Clustern entsprechend dem Index anzuordnen, den Befehl RUNSTATS auszuführen, um Statistikdaten für Index und Tabelle zu erfassen, und anschließend die Abfrage erneut zu kompilieren. Prüfen Sie die EXPLAIN-Ausgabe erneut, um herauszufinden, ob das Reorganisieren der Tabelle zu einer Verbesserung des Zugriffsplans geführt hat.

- **Zugriffstyp**

Analysieren Sie die EXPLAIN-Ausgabe und suchen Sie nach Arten des Zugriffs auf die Daten, die für die Art von Anwendung, die Sie ausführen, normalerweise nicht optimal sind. Zum Beispiel:

- **OLTP-Abfragen (Online Transaction Processing)**

In OLTP-Anwendungen werden häufig Indexsuchen mit Vergleichselementen durchgeführt, die eine Bereichsbegrenzung vornehmen, weil diese Anwendungen in der Regel nur wenige Zeilen zurückgeben, die den mit einem Gleichheitsvergleichselement für eine Spalte angegebenen Bedingungen entsprechen. Wenn Ihre OLTP-Abfragen eine Tabellensuche verwenden, können Sie die EXPLAIN-Daten analysieren, um herauszufinden, warum keine Indexsuche verwendet wurde.

- **Reine Suchabfragen**

Die Suchbedingungen für eine Abfrage, bei der Daten nur abgerufen werden, können sehr vage sein, was jedoch bewirkt, dass eine große Menge von Zeilen den Bedingungen entspricht. Wenn sich Benutzer normalerweise nur einige Seiten der Ausgabedaten anzeigen lassen, können Sie dafür sorgen, dass nicht die gesamte Antwortmenge errechnet werden muss, bevor einige Ergebnisse zurückgegeben werden. In diesem Fall unterscheiden sich die Ziele des Benutzers vom grundlegenden Verarbeitungsprinzip des Optimierungsprogramms, das versucht, den Ressourcenbedarf für die gesamte Abfrage und nicht nur für die ersten wenigen Anzeigen mit Daten zu minimieren.

Wenn zum Beispiel die EXPLAIN-Ausgabe zeigt, dass Operatoren sowohl für Mischjoins als auch für Sortierungen im Zugriffsplan verwendet wurden, wird die gesamte Antwortmenge in einer temporären Tabelle gespeichert, bevor Zeilen an die Anwendung zurückgegeben werden. In diesem Fall können Sie versuchen, den Zugriffsplan durch die Verwendung der Klausel OPTIMIZE FOR in der SELECT-Anweisung zu ändern. Wenn Sie diese Option angeben, kann das Optimierungsprogramm versuchen, einen Zugriffsplan auszuwählen, der nicht die gesamte Antwortmenge in einer temporären Tabelle erstellt, bevor die ersten Zeilen an die Anwendung zurückgegeben werden.

- **Joinmethoden**

Wenn bei einer Abfrage zwei Tabellen verknüpft werden, sollten Sie die Art der verwendeten Joinverarbeitung überprüfen. Joins mit mehreren Zeilen, wie sie zum Beispiel bei Abfragen von Entscheidungshilfedaten auftreten, werden in der Regel schneller als Hash-Joins oder Mischjoins ausgeführt. Joins, an denen nur einige wenige Zeilen beteiligt sind, wie zum Beispiel für OLTP-Abfragen, sind zumeist als Joins mit Verschachtelungsschleife effizienter. In beiden Fällen kann es jedoch auch Umstände geben, wie beispielsweise die Verwendung lokaler Vergleichselemente oder Indizes, die die normale Verarbeitung dieses Joins möglicherweise ändern.

Zugehörige Konzepte:

- „Beschreibung der Ausgabe von db2expln und dynexpln“ auf Seite 719
- „EXPLAIN-Einrichtung“ auf Seite 257
- „SQL- und XQuery-EXPLAIN-Tools“ auf Seite 711

Hinweise zur Verwendung von dynexpln

Zur Bearbeitung dynamischer Anweisungen erstellt dynexpln eine statische Anwendung für die Anweisungen und ruft anschließend db2expln auf. Zur Erstellung der statischen Anweisungen generiert dynexpln ein triviales C-Programm mit den Anweisungen und ruft anschließend den DB2-Precompiler zur Erstellung des Pakets auf. (Das generierte C-Programm ist nicht vollständig und kann nicht kompiliert werden. Es enthält nur so viele Informationen, dass der Precompiler das Paket erstellen kann.)

Es folgen einige allgemeine Nachrichten, die dynexpln ausgeben kann:

- Alle Fehlernachrichten aus db2expln

Da dynexpln das Programm db2expln aufruft, können die Mehrzahl der Fehlernachrichten von db2expln in der Ausgabe auftreten.

- Error connecting to the database.

Diese Nachricht wird in der Ausgabe angezeigt, wenn ein Fehler bei der Herstellung der Verbindung zur Datenbank aufgetreten ist. In einer weiteren CLI-Fehlernachricht wird die Ursache für das Fehlschlagen der Verbindung angezeigt. Beheben Sie die Fehlerursache, und führen Sie dynexpln erneut aus.

- The file "<dateiname>" must be removed before dynexpln will run.

Diese Nachricht wird in der Ausgabe angezeigt, wenn die angegebene Datei zur Zeit der Ausführung von dynexpln existiert. Entfernen Sie die Datei, oder ändern Sie den Wert für die Umgebungsvariable DYNEXPLN_PACKAGE, um den Namen der Datei zu ändern, die erstellt wird, und führen Sie dynexpln erneut aus.

- The package "<ersteller>.<paket>" must be dropped before dynexpln will run.

Diese Nachricht wird in der Ausgabe angezeigt, wenn das angegebene Paket zur Zeit der Ausführung von dynexpln existiert. Löschen Sie das Paket, oder ändern Sie den Wert für die Umgebungsvariable DYNEXPLN_PACKAGE, um den Namen des Pakets zu ändern, das erstellt wird, und führen Sie dynexpln erneut aus.

- Error writing file "<dateiname>".

Diese Nachricht wird in der Ausgabe angezeigt, wenn in die Datei nicht geschrieben werden kann. Stellen Sie sicher, dass dynexpln in die Dateien des aktuellen Verzeichnisses schreiben kann, und führen Sie dynexpln erneut aus.

- Error reading input file "<dateiname>".

Diese Nachricht wird in der Ausgabe angezeigt, wenn die mit der Option **-f** angegebene Datei nicht gelesen werden kann. Stellen Sie sicher, dass die Datei vorhanden ist und dass dynexpln sie lesen kann. Führen Sie anschließend dynexpln erneut aus.

Umgebungsvariablen: Es gibt zwei verschiedene Umgebungsvariablen, die in Verbindung mit dynexpln verwendet werden können:

- **DYNEXPLN_OPTIONS** definiert die Optionen für den SQL- und XQuery-Precompiler, die Sie bei der Erstellung des Pakets für Ihre Anweisungen verwenden. Verwenden Sie dieselbe Syntaxvariable wie bei der Eingabe eines Befehls PREP über CLP.

Beispiel: DYNEXPLN_OPTIONS="OPTLEVEL 5 BLOCKING ALL"

- **DYNEXPLN_PACKAGE** definiert den Namen des Pakets, das in der Datenbank erstellt wird. Die mit EXPLAIN zu bearbeitenden Anweisungen werden in dieses Paket geschrieben. Falls diese Variable nicht definiert ist, erhält das Paket den Standardnamen **DYNEXPLN**. (Es werden nur die ersten acht Zeichen des Namens in dieser Umgebungsvariablen verwendet.)

Der Name wird auch zur Generierung von Namen für von dynexpln benötigte temporäre Dateien verwendet.

Zugehörige Konzepte:

- „dynexpln“ auf Seite 258
- „Beschreibung der Ausgabe von db2expln und dynexpln“ auf Seite 719
- „Beispiele für die Ausgabe von db2expln und dynexpln“ auf Seite 739

Kapitel 11. Konfigurieren von DB2-Instanzen und -Datenbanken

Konfigurationsparameter

Bei der Erstellung einer DB2-Datenbankinstanz oder -Datenbank wird eine entsprechende Konfigurationsdatei mit Standardparameterwerten erstellt. Diese Parameterwerte können zur Verbesserung der Leistung und anderer Merkmale der Instanz oder der Datenbank geändert werden.

Konfigurationsdateien enthalten Parameter, die Werte definieren, wie zum Beispiel die den DB2-Datenbankprodukten und einzelnen Datenbanken zugeordneten Ressourcen und die Diagnoseebene. Es gibt zwei Arten von Konfigurationsdateien:

- Die Konfigurationsdatei des Datenbankmanagers für jede DB2-Instanz
- Die Datenbankkonfigurationsdatei für jede einzelne Datenbank

Eine *Konfigurationsdatei des Datenbankmanagers* wird erstellt, wenn eine DB2-Instanz erstellt wird. Die in ihr enthaltenen Parameter betreffen Systemressourcen auf Instanzebene, die von keiner Datenbank abhängig sind, die zu dieser Instanz gehört. Die Werte vieler dieser Parameter können von den Standardwerten des Systems zur Leistungsoptimierung oder Kapazitätserhöhung abhängig von der Konfiguration des Systems geändert werden.

Darüber hinaus gibt es auch für jede Clientinstallation eine Konfigurationsdatei des Datenbankmanagers. Diese Datei enthält Informationen über den Client Enabler für eine bestimmte Workstation. Eine Untergruppe der für einen Server verfügbaren Parameter gilt für den Client.

Konfigurationsparameter des Datenbankmanagers werden in einer Datei mit dem Namen `db2system` gespeichert. Diese Datei wird erstellt, wenn die Instanz des Datenbankmanagers erstellt wird. In UNIX-basierten Umgebungen befindet sich diese Datei im Unterverzeichnis `sql1ib` für die Instanz des Datenbankmanagers. Unter Windows ist die Standardposition dieser Datei das Instanzunterverzeichnis des Verzeichnisses `sql1ib`. Wenn die Variable `DB2INSTPROF` definiert ist, befindet sich die Datei im Unterverzeichnis `instance` des Verzeichnisses, das durch die Variable `DB2INSTPROF` angegeben wird.

In einer Umgebung mit partitionierten Datenbanken befindet sich diese Datei in einem gemeinsamen Dateisystem, so dass alle Datenbankpartitionsserver Zugriff auf dieselbe Datei haben. Die Konfiguration des Datenbankmanagers ist auf allen Datenbankpartitionsservern identisch.

Die meisten Parameter haben entweder Einfluss darauf, wie viel Systemressourcen einer einzelnen Datenbankmanagerinstanz zugeordnet werden, oder sie konfigurieren die Einrichtung des Datenbankmanagers und der verschiedenen Kommunikationssysteme nach umgebungsspezifischen Aspekten. Darüber hinaus gibt es noch weitere Parameter, die nur der Information dienen und deren Werte nicht geändert werden können. Alle diese Parameter sind allgemein gültig und unabhängig von einzelnen Datenbanken, die unter dieser Datenbankmanagerinstanz gespeichert sind.

Eine *Konfigurationsdatei der Datenbank* wird erstellt, wenn eine Datenbank erstellt wird. Sie befindet sich dort, wo sich die Datenbank befindet. Pro Datenbank gibt es eine Konfigurationsdatei. Die in ihr enthaltenen Parameter geben neben anderen Merkmalen die Menge der Ressourcen an, die der zugehörigen Datenbank zugeordnet sind. Die Werte vieler dieser Parameter können zur Leistungsoptimierung und Kapazitätserhöhung geändert werden. Abhängig von der Art der Aktivitäten in einer bestimmten Datenbank können unterschiedliche Änderungen erforderlich sein.

Parameter für eine einzelne Datenbank werden in einer Konfigurationsdatei namens `SQLDBCON` gespeichert. Diese Datei wird zusammen mit anderen Steuerdateien für die Datenbank im Verzeichnis `SQLnnnnn` gespeichert, wobei `nnnnn` eine Nummer ist, die bei der Erstellung der Datenbank zugeordnet wurde. Jede Datenbank hat eine eigene Konfigurationsdatei, und die meisten Parameter in der Datei geben an, wie viele Ressourcen der betreffenden Datenbank zugeordnet werden. Die Datei enthält außerdem beschreibende Informationen sowie Markierungen, die den Status der Datenbank angeben.

In einer Umgebung mit partitionierten Datenbanken ist für jede Datenbankpartition eine separate Datei `SQLDBCON` vorhanden. Die Werte in der Datei `SQLDBCON` können in den einzelnen Datenbankpartitionen gleich sein oder sich unterscheiden; es ist allerdings empfehlenswert, in allen Datenbankpartitionen die gleichen Werte für die Datenbankkonfigurationsparameter zu verwenden.

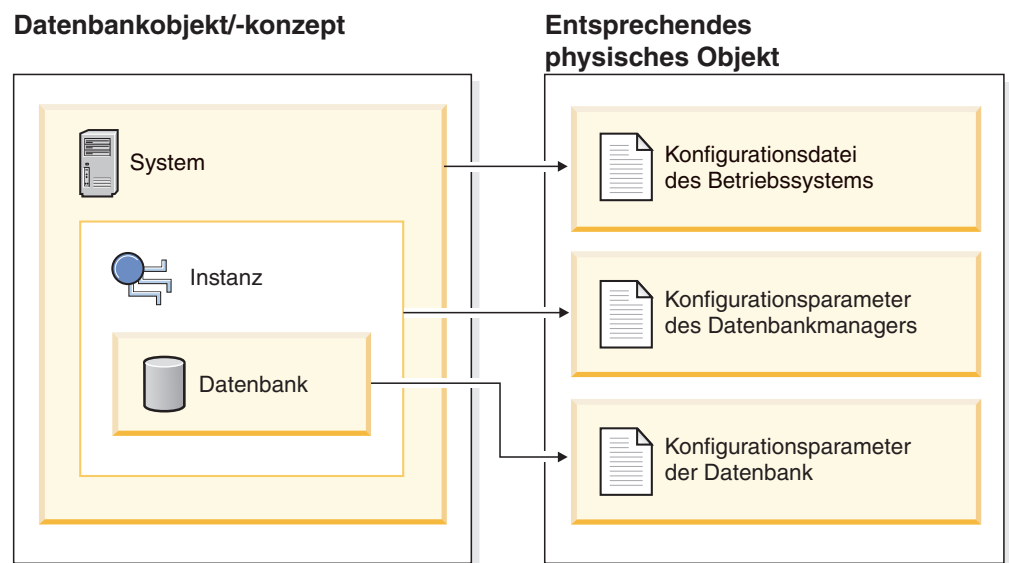


Abbildung 26. Beziehung zwischen Datenbankobjekten und Konfigurationsdateien

Zugehörige Konzepte:

- „Konfigurationsparameter mit Einfluss auf die Abfrageoptimierung“ auf Seite 186

Zugehörige Tasks:

- „Konfigurieren von DB2 mit Konfigurationsparametern“ auf Seite 277

Konfigurieren von DB2 mit Konfigurationsparametern

Der vom Datenbankmanager auf der Basis der Standardwerte für die Parameter zugeordnete Plattenspeicherplatz und der Hauptspeicher können für Ihre Anforderungen in einigen Fällen ausreichend sein. In einigen Situationen wird jedoch die maximale Leistung bei Verwendung dieser Standardwerte vielleicht nicht erreicht.

Da die Standardwerte für Systeme mit relativ geringen Hauptspeicherressourcen, die als Datenbankserver eingesetzt werden, ausgelegt sind, müssen Sie diese Werte möglicherweise ändern, wenn Ihre Umgebung folgende Merkmale aufweist:

- Umfangreiche Datenbanken
- Große Anzahl von Verbindungen
- Hohe Leistungsanforderungen für eine bestimmte Anwendung
- Spezifische Abfrage-/Transaktionsladevorgänge bzw. -arten

Jede Umgebung, die Transaktionen verarbeitet, hat einen oder mehrere spezifische, nur für sie geltende Aspekte. Diese Unterschiede können sich tief greifend auf die Leistung des Datenbankmanagers auswirken, wenn die Standardkonfiguration verwendet wird. Aus diesem Grund wird ausdrücklich empfohlen, die Konfiguration für die Umgebung zu optimieren.

Ein guter Ausgangspunkt für die Optimierung Ihrer Konfiguration ist die Verwendung des Konfigurationsadvisors bzw. des Befehls `AUTOCONFIGURE`, mit dessen Hilfe Parameterwerte auf der Basis Ihrer Antworten auf die Fragen zu den Auslastungsmerkmalen generiert werden.

Einige Konfigurationsparameter können auf den Wert *automatic* gesetzt werden. In diesem Fall werden diese Parameter von DB2 automatisch angepasst, sodass sie die aktuellen Ressourcenanforderungen widerspiegeln.

Die Konfigurationsparameter des Datenbankmanagers sind in einer Datei mit dem Namen `db2system` gespeichert. Die Konfigurationsparameter einer Datenbank sind in einer Datei mit dem Namen `SQLDBCON` gespeichert. Diese Dateien können nicht direkt editiert werden, sondern lediglich mit Hilfe einer bereitgestellten Anwendungsprogrammierschnittstelle (API) oder mit einem Tool, das diese API aufruft, geändert oder angezeigt werden.

Achtung: Wenn Sie die Datei `db2system` oder `SQLDBCON` mit anderen als den von DB2 vorgesehenen Methoden editieren, können Sie die Datenbank unbrauchbar machen. **Es wird ausdrücklich davon abgeraten**, diese Dateien mit anderen als den von DB2 unterstützten und dokumentierten Methoden zu ändern.

Sie können eine der folgenden Methoden zum Anzeigen, Aktualisieren und Zurücksetzen der Konfigurationsparameter verwenden:

- Über die Steuerzentrale. Das Notizbuch **Instanz konfigurieren** kann zur Einstellung der Konfigurationsparameter des Datenbankmanagers entweder auf einem Client oder auf einem Server verwendet werden. Das Notizbuch **Datenbank konfigurieren** kann zum Ändern der Werte von Datenbankkonfigurationsparametern verwendet werden. Die DB2-Steuerzentrale stellt auch den Konfigurationsadvisor zum Ändern der Werte der Konfigurationsparameter bereit. Dieser Advisor generiert Werte für Parameter auf der Grundlage Ihrer Antworten zu einer Reihe von Fragen, die zum Beispiel die Auslastung und die Art von Transaktionen betreffen, die mit der Datenbank ausgeführt werden.

In einer Umgebung mit partitionierten Datenbanken ist die Datei SQLDBCON für jede Datenbankpartition vorhanden. Das Notizbuch zum Konfigurieren einer Datenbank ändert den Wert in allen Datenbankpartitionen, wenn Sie es vom Datenbankobjekt in der Baumstruktursicht der Steuerzentrale aus starten. Wenn Sie das Notizbuch von einem Datenbankpartitionsobjekt aus starten, ändert es nur die Werte für die betreffende Datenbankpartition. (Es wird jedoch empfohlen, dass die Werte der Konfigurationsparameter in allen Datenbankpartitionen übereinstimmen.)

Anmerkung: Der Konfigurationsadvisor ist in Umgebungen mit partitionierten Datenbanken nicht verfügbar.

- Über den Befehlszeilenprozessor. Befehle zum Ändern der Einstellungen können schnell und bequem eingegeben werden:

Für Konfigurationsparameter des Datenbankmanagers:

- GET DATABASE MANAGER CONFIGURATION (oder GET DBM CFG)
- UPDATE DATABASE MANAGER CONFIGURATION (oder UPDATE DBM CFG)
- RESET DATABASE MANAGER CONFIGURATION (oder RESET DBM CFG) zum Zurücksetzen *aller* Parameter des Datenbankmanagers auf ihre Standardwerte
- AUTOCONFIGURE

Für Konfigurationsparameter der Datenbank:

- GET DATABASE CONFIGURATION (oder GET DB CFG)
- UPDATE DATABASE CONFIGURATION (oder UPDATE DB CFG)
- RESET DATABASE CONFIGURATION (oder RESET DB CFG) zum Zurücksetzen *aller* Datenbankparameter auf ihre Standardwerte
- AUTOCONFIGURE

- Über APIs (Anwendungsprogrammierschnittstellen). APIs können aus einer Anwendung oder einem Programm in einer Hostprogrammiersprache aufgerufen werden. Rufen Sie die folgenden DB2-APIs auf, um Konfigurationsparameter anzuzeigen oder zu aktualisieren:
 - db2AutoConfig - Zugreifen auf den Konfigurationsadvisor
 - db2CfgGet - Abrufen der Parameter für den Datenbankmanager oder die Datenbankkonfiguration
 - db2CfgSet - Definieren der Parameter für den Datenbankmanager oder die Datenbankkonfiguration
- Über den Konfigurationsassistenten (für Konfigurationsparameter des Datenbankmanagers). Mit Hilfe des Konfigurationsassistenten können Sie nur die Konfigurationsparameter des Datenbankmanagers auf einem Client einstellen.

Für einige Konfigurationsparameter des Datenbankmanagers muss der Datenbankmanager gestoppt (db2stop) und erneut gestartet (db2start) werden, um die neuen Parameterwerte in Kraft zu setzen.

Für Datenbankparameter werden Änderungen erst wirksam, wenn die Datenbank erneut aktiviert wird. Dazu müssen zunächst alle Anwendungen ihre Verbindung zur Datenbank trennen. (Wenn die Datenbank aktiviert war, muss sie inaktiviert und anschließend erneut aktiviert werden.) Bei der Herstellung der ersten neuen Verbindung zur Datenbank werden die Änderungen wirksam.

Andere Parameter können online geändert werden. Diese werden als *online konfigurierbare Konfigurationsparameter* bezeichnet.

Wenn Sie die Einstellung eines online konfigurierbaren Konfigurationsparameters des Datenbankmanagers ändern, während Sie mit einer Instanz verbunden sind, wendet der Befehl UPDATE DBM CFG die Änderung standardmäßig unverzüglich an. Wenn die Änderung nicht sofort angewendet werden soll, verwenden Sie die Option DEFERRED im Befehl UPDATE DBM CFG.

Geben Sie folgende Befehle ein, um einen Konfigurationsparameter des Datenbankmanagers online zu ändern:

```
db2 attach to <instanzname>
db2 update dbm cfg using <parametername> <wert>
db2 detach
```

Bei Clients werden Änderungen an den Konfigurationsparametern des Datenbankmanagers wirksam, wenn der Client das nächste Mal die Verbindung zu einem Server herstellt.

Wenn Sie einen online konfigurierbaren Datenbankkonfigurationsparameter ändern, während Sie mit einer Datenbank verbunden sind, wird die Änderung standardmäßig, soweit möglich, online angewendet. Sie sollten jedoch beachten, dass einige Parameteränderungen aufgrund des Systemaufwands für die Speichervuordnung eine relativ lange Zeit benötigen, um in Kraft zu treten. Zur Änderung von Konfigurationsparametern online über den Befehlszeilenprozessor ist eine Verbindung zur Datenbank erforderlich. Geben Sie folgende Befehle ein, um einen Datenbankkonfigurationsparameter online zu ändern:

```
db2 connect to <dbname>
db2 update db cfg using <parametername> <parameterwert>
db2 connect reset
```

Jedem online konfigurierbaren Konfigurationsparameter ist eine *Weitergabeklasse* zugeordnet. Die Weitergabeklasse gibt an, wann Sie damit rechnen können, dass eine Änderung an dem Konfigurationsparameter in Kraft tritt. Es gibt drei Weitergabeklassen:

- **Sofort:** Gibt Parameterwerte an, die sich sofort beim Aufruf des Befehls oder der API ändern. Zum Beispiel hat der Parameter *diaglevel* die Weitergabeklasse „Sofort“.
- **Anweisungsgrenzwert:** Gibt Parameter an, die sich an Anweisungsgrenzen und anweisungsähnlichen Grenzen ändern. Wenn Sie zum Beispiel den Wert des Parameters *sortheap* ändern, beginnen alle neuen Anforderungen mit der Verwendung des neuen Werts.
- **Transaktionsgrenzwert:** Gibt Parameter an, die sich an Transaktionsgrenzen ändern. Zum Beispiel wird ein neuer Wert für den Parameter *dl_expint* nach einer Anweisung COMMIT aktualisiert.

Das Ändern einiger Konfigurationsparameter der Datenbank kann den Zugriffsplan beeinflussen, der vom SQL- und XQuery-Optimierungsprogramm ausgewählt wird. Nach der Änderung eines dieser Parameter sollten Sie in Betracht ziehen, einen Rebind für die Anwendungen durchzuführen, um sicherzustellen, dass der beste Zugriffsplan für die SQL- und XQuery-Anweisungen verwendet wird. Alle Parameter, die online modifiziert werden (z. B. durch den Befehl UPDATE DATABASE CONFIGURATION IMMEDIATE) veranlassen das SQL- und XQuery-Optimierungsprogramm, neue Zugriffspläne für neue Abfrageanweisungen auszu-

wählen. Allerdings wird der Abfrageanweisungscache nicht von vorhandenen Einträgen bereinigt. Zur Bereinigung des Abfragecaches verwenden Sie die Anweisung `FLUSH PACKAGE CACHE`.

Obwohl die neuen Parameterwerte möglicherweise nicht sofort in Kraft treten, werden beim Anzeigen der Parametereinstellungen (mit Hilfe der Befehle `GET DATABASE MANAGER CONFIGURATION` und `GET DATABASE CONFIGURATION`) stets die zuletzt aktualisierten Werte angezeigt. Wenn Sie die Klausel `SHOW DETAIL` in diesen Befehlen zum Anzeigen der Parametereinstellungen angeben, werden sowohl die zuletzt aktualisierten Werte als auch die im Hauptspeicher befindlichen Werte angezeigt.

Anmerkung: Eine Reihe von Konfigurationsparametern (z. B. `userexit`) verfügen laut Beschreibung in der Hilfe und anderer DB2-Dokumentationen über die zulässigen Werte „Yes“ oder „No“ bzw. „On“ oder „Off“. Zur Vermeidung von Unklarheiten sei hier angemerkt, dass „Yes“ als äquivalent zu „On“ und „No“ als äquivalent zu „Off“ anzusehen sind.

Zugehörige Konzepte:

- „Konfigurationsparameter“ auf Seite 275

Zugehörige Referenzen:

- „Konfigurationsparameter - Übersicht“ auf Seite 305
- „`FLUSH PACKAGE CACHE` statement“ in *SQL Reference, Volume 2*
- „`AUTOCONFIGURE` command using the `ADMIN_CMD` procedure“ in *Administrative SQL Routines and Views*
- „`RESET DATABASE CONFIGURATION` command using the `ADMIN_CMD` procedure“ in *Administrative SQL Routines and Views*
- „`RESET DATABASE MANAGER CONFIGURATION` command using the `ADMIN_CMD` procedure“ in *Administrative SQL Routines and Views*
- „`UPDATE DATABASE CONFIGURATION` command using the `ADMIN_CMD` procedure“ in *Administrative SQL Routines and Views*
- „`UPDATE DATABASE MANAGER CONFIGURATION` command using the `ADMIN_CMD` procedure“ in *Administrative SQL Routines and Views*
- „db2AutoConfig API - Access the Configuration Advisor“ in *Administrative API Reference*
- „db2CfgGet API - Get the database manager or database configuration parameters“ in *Administrative API Reference*
- „db2CfgSet API - Set the database manager or database configuration parameters“ in *Administrative API Reference*
- „`AUTOCONFIGURE` command“ in *Command Reference*
- „`GET DATABASE CONFIGURATION` command“ in *Command Reference*
- „`GET DATABASE MANAGER CONFIGURATION` command“ in *Command Reference*
- „`RESET DATABASE CONFIGURATION` command“ in *Command Reference*
- „`RESET DATABASE MANAGER CONFIGURATION` command“ in *Command Reference*
- „`UPDATE DATABASE CONFIGURATION` command“ in *Command Reference*
- „`UPDATE DATABASE MANAGER CONFIGURATION` command“ in *Command Reference*

Dynamische Konfiguration von Parametern

Unter DB2 können Sie sich eine dynamische Konfiguration zunutze machen. Sie können bestimmte Konfigurationsparameter in einer DB2-Datenbank oder -Instanz ändern, während die Datenbank aktiv ist, Verbindungen empfängt oder Transaktionen verarbeitet. Das folgende Beispiel erläutert, wie Sie sich die Funktionen zur dynamischen Konfiguration in DB2 zunutze machen können.

In diesem Szenario ist ein Datenbankserver mit einem Speicher von 4 GB konfiguriert, von denen 3,5 GB dem Datenbankmanager zur Verfügung stehen. Es gibt acht Prozessoren. Der Datenbankserver ist für eine einzige Datenbank mit dem Namen DB2DYN dediziert, die in der Instanz DB2INST ausgeführt wird. Die Datenbankauslastung variiert über den Tag und die Woche wie folgt:

- Die Auslastung des Tagesgeschäfts (05:00-20:00) beinhaltet zahlreiche Verbindungen und gleichzeitige Transaktionen.
- Die Auslastung nach Tagesschäftsschluss (20:00-24:00) beinhaltet die Erstellung von Ergebnisberichten und die Ausführung von Entscheidungshilfeabfragen mit einigen wenigen Verbindungen und Transaktionen.
- Die tägliche Systempflegeauslastung (24:00-05:00) beinhaltet Onlineladeoperationen, inkrementelle Backups, Indexerstellungen usw.
- Die wöchentliche Systempflegeauslastung beinhaltet umfangreiche Reorganisations von Tabellen, RUNSTATS-Operationen und die Erstellung größerer Indizes.

In Anbetracht dieser Auslastungsmerkmale könnten Sie das System wie in der folgenden Tabelle gezeigt konfigurieren.

	Tagsüber (05:00 - 20:00)	Tagesabschluss (20:00 - 24:00)	Tägliche Systempflege (24:00 - 05:00)	Wöchentliche Systempflege (sonntags)
Datenbankaktivität	Hohe Auslastung durch Transaktionen	Entscheidungshilfeabfragen	Laden, Backup, Indexerstellung	Ausführen von REORG und RUNSTATS in Pufferpool 1
Pufferpool 1 (MB)	1000	500	500	2000
Pufferpool 2 (MB)	1000	500	500	200
Sortierspeicher (MB)	0,1	20	200	200
Katalogcache (MB)	200	200	50	50
Paketcache (MB)	800	200	200	200
Zwischenspeicher für Dienstprogramme (MB)	0	0	1000	0
Diagnosestufe	1	3	4	4

Sie könnten beginnen, indem Sie den Datenbankkonfigurationsparameter *database_memory* auf den Wert 3,5 GB setzen, so dass diese angegebene Größe des verfügbaren Speicherbereichs für die Datenbank reserviert wird. Dies ist eine einmalige Operation, die nach ihrer Ausführung der Datenbank 3,5 GB (bzw. 917.504 4-KB-Seiten) reservierten Speichers für die Pufferpoolerstellung oder für die Konfigurationsanpassung zur Verfügung stellt.

```
db2start
db2 update db cfg for db2dyn using database_memory 917504
db2stop
```

Anschließend könnten Sie die folgenden Scripts verwenden, um den Übergang der Datenbank von einer Konfiguration zu einer anderen zu vollziehen. (Diese Scripts können zur Ausführung an geeigneten Zeitpunkten terminiert werden.)

MorningConfiguration.sh

```
# Dieses Script dient zur Vorbereitung
# des Datenbankservers für die
# morgendliche Konfiguration
# für eine Auslastung bestehend aus
# einer großen Anzahl OLTP-Verbindungen
# und gleichzeitiger Transaktionen.

db2 connect to db2dyn

db2 update db cfg using sortheap 25
db2 update db cfg using util_heap_sz 32
db2 alter bufferpool bufferpool1 size 262144
db2 commit
db2 update db cfg using catcachesz 51200
db2 update db cfg using pkcachesz 204800
db2 alter bufferpool bufferpool2 size 262144
db2 commit
db2 flush package cache dynamic
db2 get db cfg show detail
db2 connect reset

db2 attach to instance db2inst
db2 update dbm cfg using diaglevel 1
db2 get dbm cfg show detail
db2 detach
```

EveningConfiguration.sh

```
# Dieses Script dient zur Vorbereitung
# des Datenbankservers für die
# abendliche Konfiguration
# für eine Auslastung bestehend aus
# Entscheidungshilfeabfragen.

db2 connect to db2dyn

db2 alter bufferpool bufferpool1 size 131072
db2 commit
db2 alter bufferpool bufferpool2 size 131072
db2 commit
db2 update db cfg using pkcachesz 51200
db2 update db cfg using catcachesz 51200
db2 update db cfg using util_heap_sz 32
db2 update db cfg using sortheap 5120
db2 flush package cache dynamic
db2 get db cfg show detail
db2 connect reset

db2 attach to instance db2inst
db2 update dbm cfg using diaglevel 3
db2 get dbm cfg show detail
db2 detach
```

NightTimeConfiguration.sh

```
# Dieses Script dient zur Vorbereitung
# des Datenbankservers für die
# tägliche Systempflegekonfiguration
# für eine Auslastung bestehend aus
# Indexerstellungs- sowie Lade- und
# Backup-Operationen.

db2 connect to db2dyn
```

```

db2 alter bufferpool bufferpool1 size 131072
db2 commit
db2 alter bufferpool bufferpool2 size 131072
db2 commit
db2 update db cfg using catcachesz 51200
db2 update db cfg using pkcachesz 51200
db2 update db cfg using sortheap 51200
db2 update db cfg using util_heap_sz 262144
db2 flush package cache dynamic
db2 get db cfg show detail
db2 connect reset

db2 attach to instance db2inst
db2 update dbm cfg using diaglevel 4
db2 get dbm cfg show detail
db2 detach

```

MaintenanceConfiguration.sh

```

# Dieses Script dient zur Vorbereitung
# des Datenbankservers für die
# wöchentliche Systempflegekonfiguration
# für eine Auslastung bestehend aus
# REORG- und RUNSTATS-Operationen, die in
# Pufferpool 1 ausgeführt werden.

db2 connect to db2dyn

db2 update db cfg using util_heap_sz 32
db2 alter bufferpool bufferpool2 size 51200
db2 commit
db2 update db cfg using sortheap 5120
db2 update db cfg using catcachesz 51200
db2 update db cfg using pkcachesz 51200
db2 alter bufferpool bufferpool1 size 524288
db2 commit
db2 flush package cache dynamic
db2 get db cfg show detail
db2 connect reset

db2 attach to instance db2inst
db2 update dbm cfg using diaglevel 4
db2 get dbm cfg show detail
db2 detach

```

Beachten Sie, dass die Reihenfolge der Operationen von einem Script zum nächsten unterschiedlich ist. Operationen, die den Speicherbedarf senken, sollten vor Operationen ausgeführt werden, die den Speicherbedarf heraufsetzen. Ansonsten könnte das Heraufsetzen fehlschlagen.

Auf alle Änderungen an der Größe von Pufferpools folgt ein Commit, weil Pufferpooländerungen SQL-Operationen und somit Teil der Transaktion sind.

Der Paketcache wird nach jeder Rekonfiguration gelöscht (FLUSH), um dem Optimierungsprogramm zu signalisieren, dass die Konfiguration eine wesentliche Änderung erfahren hat und dass jetzt alle vorhandenen dynamischen SQL- und XQuery-Zugriffspläne ungültig sind.

Dynamische Operationen zur Rekonfiguration des Datenbankmanagers werden ausgeführt, während eine Anwendung mit der Instanz verbunden ist. Dynamische Datenbankoperationen werden ausgeführt, während eine Anwendung mit der Datenbank verbunden ist.

Mit der Klausel `SHOW DETAIL` im Befehl `GET DATABASE MANAGER CONFIGURATION` oder `GET DATABASE CONFIGURATION` kann überprüft werden, ob eine dynamische Rekonfigurationsoperation in Kraft getreten ist.

Weitere Datenbankkonfigurationsparameter, die im Rahmen dieses Szenarios ebenfalls hätten dynamisch geändert werden können, sind zum Beispiel:

- Der Parameter *locklist*, dessen Wert dynamisch aktualisiert werden kann, wenn in der Datenbank häufige Sperreneskaltungen auftreten.
- Der Parameter *dft_queryopt*, der nur für neue Verbindungen gültig ist und zur Erhöhung des Optimierungsgrades vor dem Start der Entscheidungshilfeauslastung verwendet werden kann.
- Der Parameter *dft_degree*, der ebenfalls nur für neue Verbindungen gilt und dazu verwendet werden kann, Entscheidungshilfeabfragen mit zusätzlicher abfrageinterner Parallelität zu unterstützen.

Eine weitere Option zur Anpassung an geänderte Auslastungsmerkmale besteht darin, die automatische Speicheroptimierungsfunktion zu aktivieren, die Speicherkapazitäten unter verschiedenen Speicherbereichen in Reaktion auf Änderungen in den Auslastungsanforderungen verteilt.

Zugehörige Konzepte:

- „Speicher mit automatischer Leistungsoptimierung“ auf Seite 295

Generieren von Empfehlungen für die Datenbankkonfiguration

Der Konfigurationsadvisor dient zur Generierung von Empfehlungen für die Anfangswerte der Pufferpoolgröße, der Datenbankkonfigurationsparameter und der Konfigurationsparameter des Datenbankmanagers. Die empfohlenen Werte können angezeigt oder mit der Option `APPLY` angewendet werden. Die Empfehlungen basieren auf Ihrer Eingabe sowie auf Systeminformationen, die von der Advisorfunktion erfasst werden.

Die vom Konfigurationsadvisor vorgeschlagenen Werte sind nur für eine Datenbank pro Instanz relevant. Wenn Sie diese Advisorfunktion für mehrere Datenbanken verwenden möchten, sollte jede Datenbank zu einer separaten Instanz gehören.

In Version 9.1 wird der Konfigurationsadvisor automatisch aufgerufen, wenn Sie eine Datenbank erstellen. Zur Inaktivierung dieser Funktion für eine bestimmte Datenbank oder zur expliziten Aktivierung dieser Funktion verwenden Sie den Befehl **db2set**. Beispiele:

```
db2set DB2_ENABLE_AUTOCONFIG_DEFAULT=NO
db2set DB2_ENABLE_AUTOCONFIG_DEFAULT=YES
```

Voraussetzungen:

Sie können die Option `AUTOCONFIGURE` für eine vorhandene Datenbank oder als Option im Befehl **CREATE DATABASE** verwenden.

Zur Konfiguration Ihrer Datenbank müssen Sie über die Berechtigung SYSADM, SYSCTRL oder SYSMAINT verfügen.

Vorgehensweise:

Der Konfigurationsadvisor kann durch den Befehl **AUTOCONFIGURE** über den Befehlszeilenprozessor (CLP), durch die Konfigurationsadvisor-GUI der Steuerzentrale oder durch Aufrufen der API **db2AutoConfig** ausgeführt werden.

Gehen Sie wie folgt vor, um den Konfigurationsadvisor über die Steuerzentrale zu öffnen:

1. Erweitern Sie die Objektbaumstruktur, bis Sie das Datenbankobjekt finden, für das DB2 die Konfigurationsempfehlungen generieren soll.
2. Klicken Sie die Datenbank mit der rechten Maustaste an, und wählen Sie **Konfigurationsadvisor** aus. Der Konfigurationsadvisor wird geöffnet.

Detaillierte Informationen finden Sie in der Onlinehilfefunktion der Steuerzentrale.

Geben Sie in der Befehlszeile Folgendes ein, um Konfigurationsempfehlungen anzufordern:

```
AUTOCONFIGURE
  USING <eingabeschlüsselwort> <parameterwert>
  APPLY <wert>
```

Das folgende Beispiel zeigt einen Befehl **AUTOCONFIGURE**, mit dem Konfigurationsempfehlungen auf der Basis von Eingaben über die Nutzung der Datenbank angefordert werden, ohne die Empfehlungen jedoch anzuwenden:

```
DB2 AUTOCONFIGURE USING
  MEM_PERCENT 60
  WORKLOAD_TYPE MIXED
  NUM_STMTS 500
  ADMIN_PRIORITY BOTH
  IS_POPULATED YES
  NUM_LOCAL_APPS 0
  NUM_REMOTE_APPS 20
  ISOLATION RR
  BP_RESIZEABLE YES
  APPLY NONE
```

Zugehörige Konzepte:

- „Standardmäßig aktivierte automatische Funktionen“ in *Systemverwaltung: Konzept*

Zugehörige Tasks:

- „Erstellen einer Datenbank“ in *Systemverwaltung: Implementierung*

Zugehörige Referenzen:

- „Beispielausgabe des Konfigurationsadvisors“ in *Systemverwaltung: Implementierung*
- „AUTOCONFIGURE command“ in *Command Reference*
- „db2AutoConfig API - Access the Configuration Advisor“ in *Administrative API Reference*

Konfigurieren der DB2-Hauptspeicherzuordnung

Hauptspeicherzuordnung in DB2

Die Hauptspeicherzuordnung bzw. die Aufhebung der Hauptspeicherzuordnung tritt zu unterschiedlichen Zeiten in DB2 auf. Der Hauptspeicher kann einem bestimmten Hauptspeicherbereich zugeordnet werden, wenn ein angegebenes Ereignis auftritt, wie z. B. die Herstellung einer Verbindung zu einer Anwendung, oder er kann auf der Basis einer Änderung einer Konfigurationsparametereinstellung neu zugeordnet werden.

In der unten stehenden Abbildung werden die unterschiedlichen Bereiche des Hauptspeichers angezeigt, die der Datenbankmanager für unterschiedliche Zwecke zuordnet; außerdem werden die Konfigurationsparameter angezeigt, mit denen Sie die Größe dieses Speichers steuern können. Beachten Sie, dass in einer Enterprise Server Edition-Umgebung, die mehrere logische Datenbankpartitionen umfasst, jede Datenbankpartition über ihre eigene Gruppe von gemeinsamem Speicher des Datenbankmanagers verfügt.

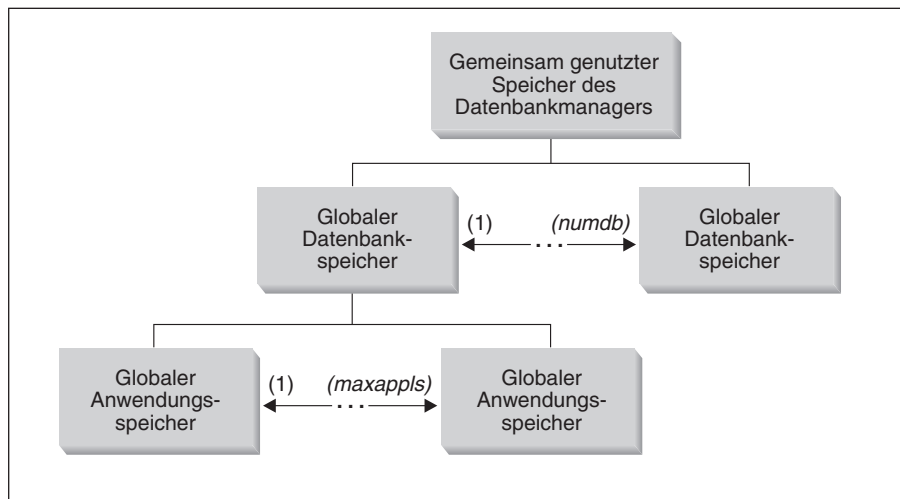


Abbildung 27. Vom Datenbankmanager verwendete Speichertypen

Der Speicher wird für jede Instanz des Datenbankmanagers zugeordnet, wenn die folgenden Ereignisse eintreten:

- **Wenn der Datenbankmanager gestartet wird (db2start):** Der globale gemeinsam genutzte Speicher (gemeinsam genutzter Speicher der Instanz) des Datenbankmanagers wird zugeordnet und bleibt so lange zugeordnet, bis der Datenbankmanager gestoppt wird (db2stop). Dieser Bereich enthält Informationen, die der Datenbankmanager zur Verwaltung von Aktivitäten für alle Datenbankverbindungen verwendet. Der gemeinsame Instanzspeicher kann durch den Konfigurationsparameter *instance_memory* gesteuert werden. Standardmäßig ist dieser Parameter auf *automatic* gesetzt, sodass DB2 die Größe des für die Instanz zugeordneten Speichers berechnet.
- **Wenn eine Datenbank aktiviert oder zum ersten Mal eine Verbindung zu ihr hergestellt wird:** Der globale Datenbankspeicher wird zugeordnet. Der globale Datenbankspeicher wird für alle Anwendungen verwendet, die eine Verbindung zur Datenbank herstellen. Die Größe des globalen Datenbankspeichers wird durch den Konfigurationsparameter *database_memory* festgelegt. Standardmäßig ist dieser Parameter auf *automatic* gesetzt, sodass DB2 die Größe des für die

Datenbank zugeordneten Speichers berechnet. Sie können *database_memory* definieren, damit zu Anfang mehr Speicher als benötigt zugeordnet wird, sodass der zusätzliche Speicher später dynamisch verteilt werden kann.

Obwohl die Gesamtgröße des globalen Datenbankspeichers nicht geändert werden kann, während die Datenbank aktiv ist, können Bereiche angepasst werden, die im globalen Datenbankspeicher enthalten sind. Die folgenden Speicherbereiche können dynamisch angepasst werden, um beispielsweise Speicher zu verkleinern, der einem Bereich zugeordnet ist, und Speicher zu vergrößern, der einem anderen Bereich zugeordnet ist.

- Pufferpools (unter Verwendung der DDL-Anweisung ALTER BUFFERPOOL)
- Datenbankzwischenpeicher (einschließlich Protokollpuffer)
- Zwischenpeicher für Dienstprogramme
- Paketcache
- Katalogcache
- Sperrenliste (Dieser Speicherbereich kann nur dynamisch vergrößert und nicht verkleinert werden.)

In einer Umgebung, in der der Konfigurationsparameter für partitionsinterne Parallelität (*intra_parallel*) des Datenbankmanagers aktiviert ist, oder einer Umgebung, in der der Verbindungskonzentrator aktiviert ist, wird auch der gemeinsame Sortierspeicher als Teil des globalen Datenbankspeichers zugeordnet.

- **Wenn eine Anwendung die Verbindung zu einer Datenbank herstellt:** In einer partitionierten Datenbankumgebung, in einer nicht partitionierten Datenbank bei aktiviertem Konfigurationsparameter für partitionsinterne Parallelität (*intra_parallel*) des Datenbankmanagers oder in einer Umgebung mit aktiviertem Verbindungskonzentrator können mehrere Anwendungen *Anwendungsgruppen* zugeordnet werden, um Speicher gemeinsam zu nutzen. Jede Anwendungsgruppe verfügt über eine eigene Zuordnung an gemeinsamem Speicher. In dem gemeinsam genutzten Speicher der Anwendungsgruppe verfügt jede Anwendung über einen eigenen Zwischenpeicher zur Anwendungssteuerung, nutzt jedoch den gemeinsamen Zwischenpeicher der Anwendungsgruppe, wodurch die Effizienz der Cache- und Speicherverwendung verbessert wird.

Die folgenden drei Datenbankkonfigurationsparameter bestimmen die Größe des Speichers von Anwendungsgruppen:

- Der Parameter *appgroup_mem_sz*, der die Größe des gemeinsamen Speichers für die Anwendungsgruppe angibt
- Der Parameter *groupheap_ratio*, der den Prozentsatz des gemeinsamen Anwendungsgruppenspeichers angibt, der für den gemeinsamen Zwischenpeicher bereitgestellt wird
- Der Parameter *app_ctl_heap_sz*, der die Größe des Steuerungszwischenspeichers für jede Anwendung in der Gruppe angibt

Der Konfigurationsparameter *max_connections* des Datenbankmanagers legt eine obere Begrenzung für die Anzahl der Anwendungen fest, die eine Verbindung zu einer Datenbank herstellen können. Da jede Anwendung, die die Verbindung zu einer Datenbank herstellt, die Zuordnung von Speicher erforderlich macht, entsteht durch das Zulassen einer höheren Anzahl gleichzeitiger Anwendungen potenziell ein höherer Speicherbedarf.

Bis zu einem gewissen Grad wird die maximale Anzahl von Anwendungen auch durch den Konfigurationsparameter *maxagents* des Datenbankmanagers oder, bei Umgebungen mit partitionierten Datenbanken, durch den Parameter *max_coordagents* bestimmt. Der Parameter *maxagents* legt eine Obergrenze für die Gesamtzahl an Datenbankmanageragenten in einer Datenbankpartition fest. Zu diesen Datenbankmanageragenten gehören aktive Koordinatoragenten, Subagenten, inaktive Agenten sowie nicht zugeordnete Agenten im Bereitschaftsmodus.

- **Bei der Erstellung eines Agenten:** Der private Agentenspeicher wird für einen Agenten zugeordnet, wenn der Agent infolge einer Verbindungsanforderung oder einer neuen SQL-Anforderung in einer parallelen Umgebung zugeordnet wird. Der private Agentenspeicher wird für den Agenten zugeordnet und enthält Speicher, der nur von diesem bestimmten Agenten verwendet wird, wie zum Beispiel der Sortierspeicher und der Anwendungszwischenspeicher. Wenn bereits eine Datenbank von einer Anwendung verwendet wird, werden für Anwendungen, die nachfolgend eine Verbindung herstellen, nur der private Agentenspeicher und der globale gemeinsame Anwendungsspeicher zugeordnet.

Die Abbildung gibt außerdem die folgenden Konfigurationsparametereinstellungen an, die die Größe des Speichers begrenzen, der für die einzelnen Speicherbereichstypen zugeordnet wird. Beachten Sie, dass dieser Speicher in einer partitionierten Datenbankumgebung in jeder Datenbankpartition zugeordnet wird.

- *instance_memory*
Dieser Parameter gibt an, wie viel Speicher für die Instanzverwaltung zugeordnet werden soll.
- *numdb*
Dieser Parameter gibt die maximale Anzahl gleichzeitig aktiver Datenbanken an, die von verschiedenen Anwendungen verwendet werden können. Da jede Datenbank über einen eigenen globalen Speicherbereich verfügt, wächst die Menge an Speicher, die möglicherweise zugeordnet wird, wenn Sie den Wert dieses Parameters erhöhen.
- *maxappls*
Dieser Parameter definiert die maximale Anzahl von Anwendungen, die gleichzeitig mit einer einzigen Datenbank verbunden sein können. Er wirkt sich auf die Menge an Speicher aus, die möglicherweise für privaten Agentenspeicher und globalen Anwendungsspeicher für diese Datenbank zugeordnet wird. Beachten Sie, dass dieser Parameter für jede Datenbank unterschiedlich eingestellt werden kann.
- *maxagents* und *max_coordagents* bei Parallelverarbeitung
Diese Parameter begrenzen die Anzahl von Datenbankmanageragenten, die gleichzeitig in allen aktiven Datenbanken in einer Instanz vorhanden sein können. Zusammen mit dem Parameter *maxappls* begrenzen diese Parameter die Größe des Speichers, der für den privaten Agentenspeicher und den globalen Anwendungsspeicher zugeordnet wird.

Mit dem Speichertracker, der über den Befehl `db2mtrk` aufgerufen wird, können Sie die momentane Speicherzuordnung in der Instanz anzeigen, einschließlich der folgenden Informationstypen für die einzelnen Speicherpools:

- Aktuelle Größe
- Maximalgröße (fester Grenzwert)
- Große Größe (obere Grenze)

Zugehörige Konzepte:

- „Gemeinsam genutzter Speicher des Datenbankmanagers“ auf Seite 289
- „Der FCM-Pufferpool und Speicheranforderungen“ auf Seite 291
- „Optimieren der Parameter für die Hauptspeicherzuordnung“ auf Seite 293

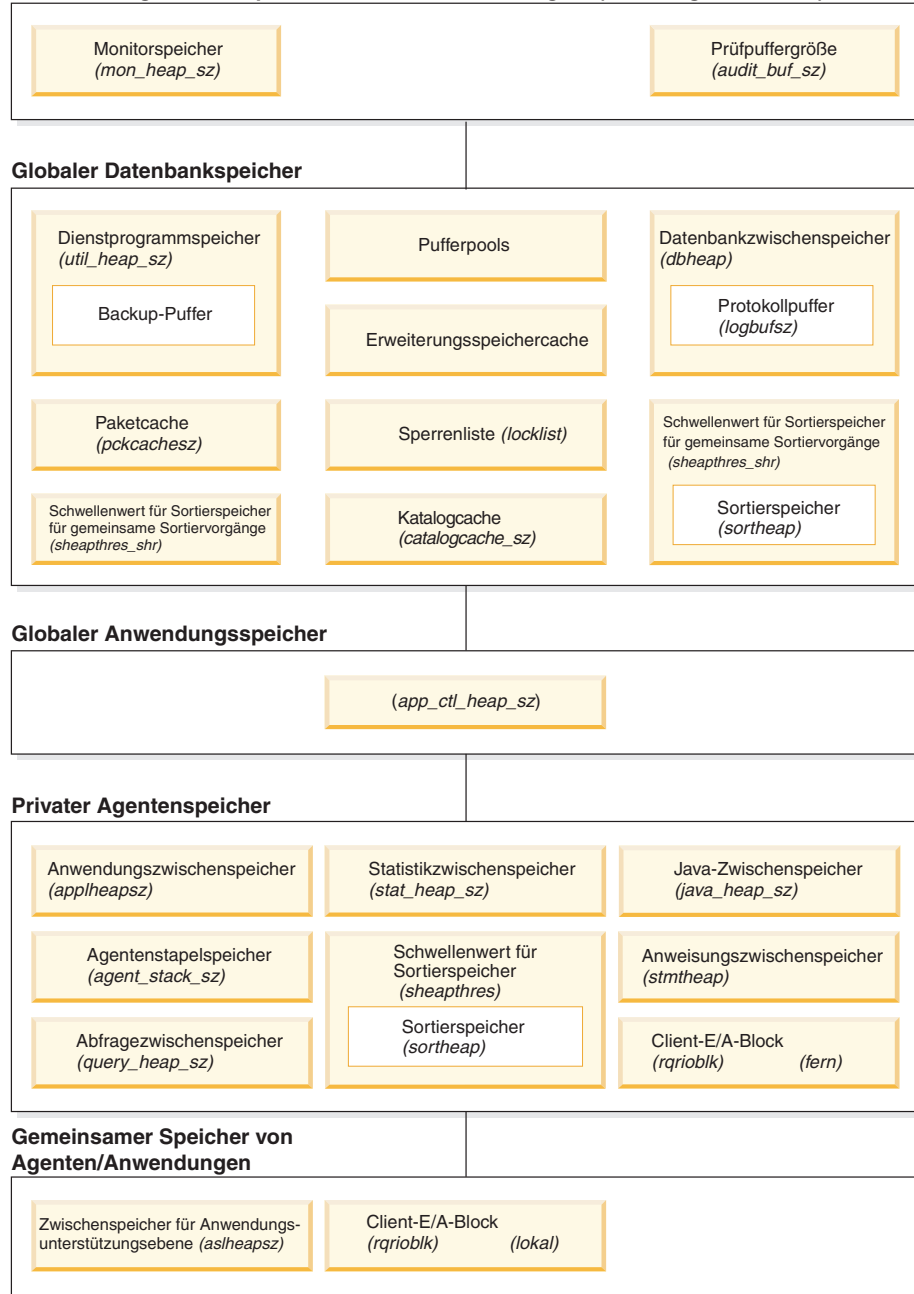
Zugehörige Referenzen:

- „db2mtrk - Memory tracker command“ in *Command Reference*

Gemeinsam genutzter Speicher des Datenbankmanagers

Der gemeinsam genutzte Speicher des Datenbankmanagers wird in vielen verschiedenen Speicherbereichen verwaltet. In der folgenden Abbildung sehen Sie, wie gemeinsam genutzter Datenbankmanagerspeicher zugeordnet wird. Mit den angezeigten Konfigurationsparametern können Sie die Größe dieses Speichers steuern, indem Sie die Anzahl von Speichersegmenten (d. h. logischen Speicherbereichen) und ihre Größe begrenzen.

Gemeinsam genutzter Speicher des Datenbankmanagers (FCM eingeschlossen)



Hinweis: Die Rahmengröße gibt keine relative Speichergröße an.

Abbildung 28. Verwendung von Speicher durch den Datenbankmanager

Prüfpuffer

Dieser Speicherbereich wird verwendet, um die Größe des Puffers zu berechnen, der in Datenbankprüfungsaktivitäten verwendet wird. Die Größe dieses Puffers wird durch den Konfigurationsparameter *audit_buf_sz* festgelegt.

Zwischenspeicher für Monitor

Dieser Speicherbereich wird für Datenbanksystemmonitordaten verwendet. Die Größe dieses Bereichs wird durch den Konfigurationsparameter *mon_heap_sz* festgelegt.

FCM-Pufferpool (FCM = Fast Communication Manager)

Bei partitionierten Datenbanksystemen benötigt FCM (Fast Communications Manager) einen beträchtlichen Speicherbereich, insbesondere wenn der Wert des Parameters *fc_num_buffers* hoch eingestellt ist. Der benötigte FCM-Speicher wird entweder aus dem FCM-Pufferpool oder aus dem gemeinsam genutzten Speicher des Datenbankmanagers und dem FCM-Pufferpool zugeordnet; dies ist davon abhängig, ob ein partitioniertes Datenbanksystem mehrere logische Knoten verwendet oder nicht. Wenn es mehrere logische Knoten gibt, wird ein separater gemeinsam genutzter Speicherbereich des FCM-Puffers zur gemeinsamen Nutzung allen logischen Knoten zugeordnet.

Globaler Datenbankspeicher

Der globale Datenbankspeicher wird durch folgende Konfigurationsparameter beeinflusst:

- Der Parameter *database_memory* stellt einen unteren Grenzwert für die Größe des globalen Datenbankspeichers bereit.
- Die folgenden Parameter oder Faktoren geben die maximale Größe von Speichersegmenten an:
 - Die Größe der Pufferpools.
 - Maximaler Speicher für Sperrenliste (*locklist*)
 - Datenbankzwischenspeicher (*dbheap*)
 - Zwischenspeichergröße für Dienstprogramme (*util_heap_sz*)
 - Größe des Paketcache (*pckcachesz*)
 - Schwellenwert für Sortierspeicher für gemeinsame Sortiervorgänge (*sheapthres_shr*)
 - Katalogcache (*catalogcache_sz*)

Globaler Anwendungsspeicher

Der globale Anwendungsspeicher wird durch den Konfigurationsparameter für die Größe des Zwischenspeichers zur Anwendungssteuerung (*app_ctl_heap_sz*) beeinflusst.

Für parallele Systeme ist außerdem Speicherbereich für den Zwischenspeicher für Anwendungssteuerung erforderlich, der von den Agenten gemeinsam genutzt wird, die für dieselbe Anwendung in einer Datenbankpartition aktiv sind. Der Zwischenspeicher wird zugeordnet, wenn der erste Agent, der eine Anforderung von der Anwendung empfängt, eine Verbindung anfordert. Der Agent kann entweder ein koordinierender Agent oder ein Subagent sein.

Privater Agentenspeicher

- Die Anzahl von Speichersegmenten wird durch den niedrigeren der folgenden Werte begrenzt:
 - Die Summe des Konfigurationsparameters *maxappls* für alle aktiven Datenbanken, die die maximal zulässige Anzahl aktiver Anwendungen angibt
 - Der Wert des Konfigurationsparameters *maxagents*, der die maximal zulässige Anzahl von Agenten angibt
- Die maximale Größe von Speichersegmenten wird von den Werten der folgenden Parameter bestimmt:
 - Zwischenspeichergröße für Anwendungen (*applheapsz*)
 - Zwischenspeicher für Sortierlisten (*sortheap*)
 - Größe des Anweisungszwischenspeichers (*stmtheap*)
 - Größe des Statistikzwischenspeichers (*stat_heap_sz*)
 - Größe des Abfragezwischenspeichers (*query_heap_sz*)
 - Größe des Agentenstacks (*agent_stack_sz*)

Gemeinsamer Agenten-/Anwendungsspeicher

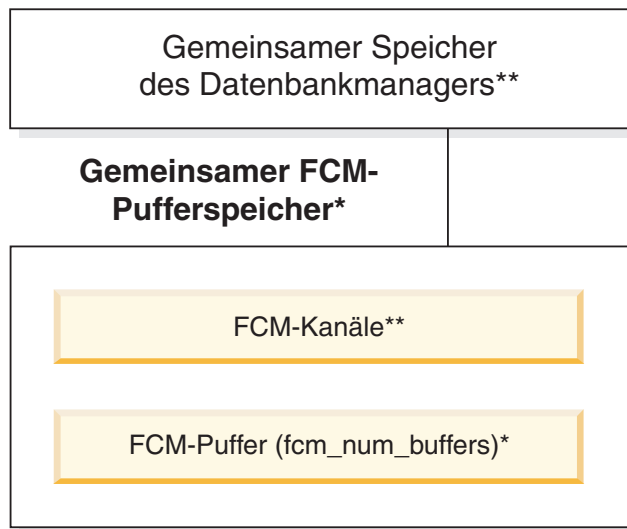
- Die Gesamtanzahl der Segmente für gemeinsamen Agenten-/Anwendungsspeicher für lokale Clients wird durch den niedrigeren der folgenden Datenbankkonfigurationsparameter begrenzt:
 - Die Summe von *maxappls* für alle aktiven Datenbanken
 - Der Wert des Parameters *maxagents* bzw. des Parameters *max_coordinating_agents* für Parallelsysteme.
- Der gemeinsame Agenten-/Anwendungsspeicher wird außerdem durch folgende Datenbankkonfigurationsparameter beeinflusst:
 - Zwischenspeichergröße für Anwendungsunterstützungsebene (*aslheapsz*)
 - E/A-Blockgröße für Clients (*rqrioblk*)

Zugehörige Konzepte:

- „Pufferpoolverwaltung“ auf Seite 131
- „Hauptspeicherzuordnung in DB2“ auf Seite 286
- „Optimieren der Parameter für die Hauptspeicherzuordnung“ auf Seite 293

Der FCM-Pufferpool und Speicheranforderungen

In einem partitionierten Datenbanksystem werden der gemeinsame Speicher des Datenbankmanagers und der FCM-Pufferpool wie folgt genutzt.



Legende

- * ein gemeinsamer für alle logischen Knoten
- ** einer für jeden logischen Knoten

Abbildung 29. FCM-Pufferpool bei Verwendung mehrerer logischer Knoten

Die Anzahl der FCM-Puffer für jede Datenbankpartition wird durch den Konfigurationsparameter *fcm_num_buffers* gesteuert. Standardmäßig ist dieser Parameter auf den Wert AUTOMATIC gesetzt. Zur manuellen Optimierung dieses Parameters verwenden Sie die Daten aus den Systemmonitorelementen 'buff_free' (Derzeit freie FCM-Puffer) und 'buff_free_bottom' (Mindestanzahl freier FCM-Puffer).

Die Anzahl der FCM-Kanäle für jede Datenbankpartition wird durch den Konfigurationsparameter *fcm_num_channels* gesteuert. Standardmäßig ist dieser Parameter auf den Wert AUTOMATIC gesetzt. Zur manuellen Optimierung dieses Parameters verwenden Sie die Daten aus den Systemmonitorelementen 'ch_free' (Derzeit freie Kanäle) und 'ch_free_bottom' (Mindestanzahl freier Kanäle).

Zugehörige Konzepte:

- „Gemeinsam genutzter Speicher des Datenbankmanagers“ auf Seite 289

Zugehörige Referenzen:

- „buff_free - FCM Buffers Currently Free monitor element“ in *Systemmonitor Handbuch und Referenz*
- „buff_free_bottom - Minimum FCM Buffers Free monitor element“ in *Systemmonitor Handbuch und Referenz*
- „ch_free - Channels Currently Free monitor element“ in *Systemmonitor Handbuch und Referenz*
- „ch_free_bottom - Minimum Channels Free monitor element“ in *Systemmonitor Handbuch und Referenz*
- „fcm_num_buffers - Anzahl FCM-Puffer “ auf Seite 554
- „fcm_num_channels - Anzahl FCM-Kanäle “ auf Seite 555

Optimieren der Parameter für die Hauptspeicherzuordnung

Die erste Regel zur Einstellung von Parametern zur Speicherzuordnung ist, sie nie auf die höchsten Werte setzen, sofern ein solcher Wert nicht sorgfältig auf seine Eignung geprüft wurde. Diese Regel gilt selbst für Systeme, die über maximale Speicherkapazitäten verfügen. Viele Parameter, die sich auf den Speicher auswirken, können zulassen, dass der Datenbankmanager leicht und schnell den gesamten auf einem Computer verfügbaren Speicher belegt. Darüber hinaus kann die Verwaltung einer großen Speichermenge wesentlich mehr Zusatzaufwand auf Seiten des Datenbankmanagers verursachen, so dass der Systemaufwand steigt.

Einige UNIX-Betriebssysteme ordnen Auslagerungsspeicher zu, wenn ein Prozess Speicher zuordnet, und nicht, wenn der Prozess in den Auslagerungsspeicher ausgelagert wird. Stellen Sie für solche Systeme sicher, dass Sie Paging-Bereich in der Größe des gesamten gemeinsamen Speicherbereichs bereitstellen.

Bei der Mehrzahl der Konfigurationsparameter wird der entsprechende Speicher erst festgeschrieben, wenn er erforderlich ist; die Parametereinstellungen legen die maximale Größe eines bestimmten Zwischenspeicherbereichs fest. In den folgenden Fällen wird jedoch die volle Größe des Speichers zugeordnet, die durch den Parameter angegeben wird:

- Maximaler Speicher für Sperrenliste (*locklist*)
- Zwischenspeichergröße für Anwendungsunterstützungsebene (*aslheapsz*)
- Anzahl der FCM-Puffer (*fcm_num_buffers*)
- Anzahl der FCM-Kanäle (*fcm_num_channels*)

Parameter mit Auswirkung auf Speichernutzung durch Anwendungsgruppen

Die Parameter, die sich auf die Nutzung von Speicher durch Anwendungsgruppen auswirken, gelten nur für partitionierte Datenbanken, für Datenbanken mit aktivierter partitionsinterner Parallelverarbeitung sowie für Datenbanken, für die der Verbindungskonzentrator aktiviert ist. Die folgenden Parameter legen fest, wie Anwendungen in Anwendungsgruppen ihren gemeinsamen Speicher verwenden:

- Der Parameter *appgroup_mem_sz* gibt die Größe des gemeinsamen Speichers für die Anwendungsgruppe an.

Eine zu hohe Einstellung des Parameters *appgroup_mem_sz* hat einen negativen Effekt. Da alle Anwendungen in der Anwendungsgruppe die Caches im Zwischenspeicher der Anwendungsgruppe gemeinsam verwenden, erhöhen zu viele Anwendungen die Konkurrenzsituation bei der Cachenutzung. Wenn jede Anwendungsgruppe andererseits zu wenig Anwendungen enthält, ist die Wirkung des Cache ebenfalls begrenzt.

- Der Parameter *groupheap_ratio* gibt den zulässigen Prozentsatz an Speicher für den gemeinsamen Speicher an.

Eine zu niedrige Einstellung des Parameters *groupheap_ratio* beschränkt die Größe von Caches. Eine zu hohe Einstellung von *groupheap_ratio* bewirkt, dass der Steuerungszwischenspeicher für Anwendungen zu klein wird und verursacht eventuell den SQL-Fehler SQL0973, der Sie darauf hinweist, dass zur Laufzeit zu wenig Speicher für den Zwischenspeicher zur Anwendungssteuerung verfügbar ist.

- Der Parameter *app_ctl_heap_sz* gibt die Größe des Steuerungszwischenspeichers für jede Anwendung in der Gruppe an.

Übernehmen Sie die Standardwerte für diese Parameter, wenn Sie Ihren Datenbankserver konfigurieren. Passen Sie die Einstellungen nur an, wenn die Leistung beeinträchtigt ist. Definieren Sie zum Beispiel den Parameter *appgroup_mem_sz*, um die Anzahl von Anwendungen in jeder Anwendungsgruppe zu steuern. Als Faustregel ist zu beachten, dass 10 zu wenige und 100 zu viele sind. Der Standardwert ist wahrscheinlich angemessen. Führen Sie anschließend eine durchschnittliche Auslastung aus und verwenden Sie die Diagnosezentrale in der Steuerzentrale oder den Systemmonitor, um Informationen zu den Trefferquoten für den Katalogcache, den Paketcache und den gemeinsamen Arbeitsbereich zu erfassen.

- Wenn zahlreiche Fehler *sql0973* auftreten, ist die Einstellung für *groupheap_ratio* zu hoch.
- Wenn die Überwachungsdaten zeigen, dass die durchschnittliche und die maximale Nutzung des Zwischenspeichers für die Anwendungssteuerung weit unter dem Wert $\text{app\_ctl\_heap\_sz} * (100 - \text{groupheap\_ratio}) / 100$ liegt, verringern Sie den Wert des Konfigurationsparameters *app_ctl_heap_sz*.
- Wenn die Werte für die Cachenutzung darauf hinweisen, dass die Caches ihre Grenze erreichen, erhöhen Sie den Wert des Konfigurationsparameters *group_heap_ratio*, oder verringern Sie die Anzahl von Anwendungen in der Anwendungsgruppe.

Hinweise:

- Vergleichstests liefern die besten Informationen zur Einstellung geeigneter Werte für Speicherparameter. Bei der Durchführung von Vergleichstests werden repräsentative SQL-Anweisungen und Extremfall-SQL-Anweisungen für den Server ausgeführt und die Werte der Parameter modifiziert, bis der Punkt gefunden wird, an dem die Leistung wieder sinkt. Wenn die Leistung gegen einen Parameterwert in einem Diagramm aufgezeichnet würde, zeigt die Stelle, an dem die Kurve nicht weiter steigt bzw. wieder zu sinken beginnt, den Punkt an, an dem eine weitere Speicherzuordnung keinen weiteren Vorteil für die Anwendung bringt und daher nur zu einer Verschwendung von Speicher führen würde.
- Die oberen Grenzen der Speicherzuordnung einiger Parameter können über den Speicherkapazitäten der vorhandenen Hardware und des Betriebssystems liegen. Diese Grenzen tragen steigenden Anforderungen in der Zukunft Rechnung.
- Gültige Wertebereiche der Parameter finden Sie in den detaillierten Informationen zu den einzelnen Parametern.

Zugehörige Konzepte:

- „Gemeinsam genutzter Speicher des Datenbankmanagers“ auf Seite 289
- „Hauptspeicherzuordnung in DB2“ auf Seite 286

Zugehörige Referenzen:

- „app_ctl_heap_sz - Zwischenspeichergröße für Anwendungssteuerung “ auf Seite 449
- „appgroup_mem_sz - Maximale Speichergröße für Anwendungsgruppe “ auf Seite 450
- „aslheapsz - Zwischenspeichergröße für Anwendungsunterstützungsebene “ auf Seite 461
- „catalogcache_sz - Katalogcachegröße “ auf Seite 432
- „dbheap - Zwischenspeicher für Datenbank “ auf Seite 437

- „fcm_num_buffers - Anzahl FCM-Puffer “ auf Seite 554
- „groupheap_ratio - Für Anwendungsgruppenzwischenspeicher vorgesehener Speicher in Prozent “ auf Seite 451
- „locklist - Maximaler Speicher für Sperrenliste “ auf Seite 438
- „pckcachesz - Größe des Paketcache “ auf Seite 442
- „sheapthres - Schwellenwert für Sortierspeicher “ auf Seite 456
- „util_heap_sz - Zwischenspeichergröße für Dienstprogramme “ auf Seite 448

Automatisches Konfigurieren mit dem Speicher mit automatischer Leistungsoptimierung

Speicher mit automatischer Leistungsoptimierung

Mit IBM DB2 Version 9 wird eine neue Speicheroptimierungsfunktion eingeführt, welche die Speicherkonfiguration vereinfacht, indem sie automatisch Werte für verschiedene Speicherkonfigurationsparameter einstellt. Wenn sie aktiviert ist, verteilt die Speicheroptimierungsfunktion verfügbare Speicherressourcen dynamisch unter mehreren Speicherkonsumenten, zu denen der Sortierzwischenspeicher, der Paketcache, die Sperrenliste und Pufferpools zählen.

Die Speicheroptimierungsfunktion reagiert auf signifikante Änderungen an den Auslastungsmerkmalen und passt die Werte von Speicherkonfigurationsparametern sowie die Größen der Pufferpools an, um die Leistung zu optimieren.

Die Optimierungsfunktion arbeitet innerhalb der Speicherbegrenzungen, die durch den Konfigurationsparameter *database_memory* definiert sind. Der Wert des Parameters *database_memory* kann unter Windows und AIX selbst wiederum automatisch optimiert werden. Wenn die automatische Optimierung für den Parameter *database_memory* aktiviert ist (d. h. wenn der Wert AUTOMATIC eingestellt ist), bestimmt die Optimierungsfunktion den Gesamtspeicherbedarf für die Datenbank und erhöht bzw. verringert die Menge an Speicher, die für gemeinsam genutzten Datenbankspeicher zugeordnet ist, abhängig von den aktuellen Anforderungen der Datenbank. Wenn zum Beispiel der aktuelle Bedarf der Datenbank hoch ist und ausreichend freier Speicher auf dem System zur Verfügung steht, wird mehr Speicher für den gemeinsam genutzten Datenbankspeicher in Anspruch genommen. Wenn der Bedarf an Datenbankspeicher sinkt oder die Größe des freien Speichers auf dem System auf einen zu niedrigen Wert zurückgeht, wird ein Teil des gemeinsam genutzten Datenbankspeichers freigegeben.

Wenn der Parameter *database_memory* nicht für die automatische Optimierung aktiviert ist (d. h. wenn er nicht auf AUTOMATIC gesetzt ist), verwendet die gesamte Datenbank die angegebene Größe an Speicher und verteilt diese nach Bedarf an die Konsumenten des Datenbankspeichers. Wenn der Parameter *database_memory* für die automatische Optimierung nicht aktiviert ist, kann die Größe des Speichers, der von der Datenbank verwendet wird, auf zwei Arten angegeben werden: Durch Setzen des Parameters *database_memory* auf einen numerischen Wert oder auf den Wert COMPUTED. Im zweiten Fall wird der Gesamtspeicher auf der Basis der Summe der Anfangswerte der Datenbankzwischenspeicher beim Starten der Datenbank berechnet.

Die folgenden Speicherkonsumenten können für die automatische Optimierung aktiviert werden:

- Pufferpools (Steuerung durch die Anweisungen ALTER BUFFERPOOL und CREATE BUFFERPOOL)
- Paketcache (Steuerung durch den Konfigurationsparameter *pckcachesz*)
- Sperrenspeicher (Steuerung durch die Konfigurationsparameter *locklist* und *maxlocks*)
- Sortierspeicher (Steuerung durch die Konfigurationsparameter *sheapthres_shr* und *sortheap*)
- Gemeinsam genutzter Datenbankspeicher (Steuerung durch den Konfigurationsparameter *database_memory*)

Zugehörige Konzepte:

- „Verwenden von Speicher mit automatischer Leistungsoptimierung in Umgebungen mit partitionierten Datenbanken“ auf Seite 302

Zugehörige Tasks:

- „Configuring health indicators using a client application“ in *Systemmonitor Handbuch und Referenz*
- „Configuring health indicators using the Health Center“ in *Systemmonitor Handbuch und Referenz*
- „Ermitteln der Speicherkonsumenten mit aktivierter automatischer Leistungsoptimierung“ auf Seite 299
- „Aktivieren des Speichers mit automatischer Leistungsoptimierung“ auf Seite 296

Aktivieren des Speichers mit automatischer Leistungsoptimierung

Speicher mit automatischer Leistungsoptimierung vereinfacht die Speicherkonfiguration, indem automatisch Werte für Speicherkonfigurationsparameter eingestellt und die Größe von Pufferpools gesteuert werden. Wenn sie aktiviert ist, verteilt die Speicheroptimierungsfunktion verfügbare Speicherressourcen unter mehreren Speicherkonsumenten, zu denen Bereiche für Sortierspeicher, Paketcache, Sperrenliste und Pufferpools zählen.

Vorgehensweise:

1. Die automatische Speicheroptimierungsfunktion für die Datenbank wird dadurch aktiviert, dass der Parameter *self_tuning_mem* auf den Wert ON gesetzt wird. Sie können den Parameter *self_tuning_mem* mit Hilfe des Befehls UPDATE DATABASE CONFIGURATION, mit der API SQLFUPD oder über das Fenster **Datenbankkonfigurationsparameter ändern** der Steuerzentrale auf ON setzen.
2. Zur Aktivierung der automatischen Optimierungsfunktion für Speicherbereiche, die durch Speicherkonfigurationsparameter gesteuert werden, setzen Sie die relevanten Konfigurationsparameter mit Hilfe des Befehls UPDATE DATABASE CONFIGURATION, mit der API SQLFUPD oder über das Fenster **Datenbankkonfigurationsparameter ändern** der Steuerzentrale auf den Wert AUTOMATIC.
3. Zur Aktivierung der automatischen Optimierung für Pufferpools setzen Sie die Größe (SIZE) des Pufferpools auf den Wert AUTOMATIC. Dies können Sie mit Hilfe der Anweisung ALTER BUFFER POOL für vorhandene Pufferpools und der Anweisung CREATE BUFFER POOL für neue Pufferpools erreichen. Wenn

die Größe eines Pufferpools in der DPF-Umgebung auf AUTOMATIC gesetzt wird, darf dieser Pufferpool keine Einträge in SYSIBM.SYSBUFFERPOOLNODES haben.

Anmerkungen:

1. Da die automatische Optimierungsfunktion Speicherressourcen unter verschiedenen Speicherbereichen umverteilt, müssen mindestens zwei Speicherkonsumenten, zum Beispiel der Sperrenspeicherbereich und der gemeinsam genutzte Datenbankspeicher, für die automatische Optimierung aktiviert sein, damit eine automatische Optimierung erfolgen kann. Die einzige Ausnahme in dieser Hinsicht ist der Speicher, der durch den Konfigurationsparameter *sortheap* gesteuert wird. Wenn *sortheap* allein auf AUTOMATIC gesetzt ist, ist die automatische Optimierung von *sortheap* aktiviert.
2. Zur Aktivierung des Konfigurationsparameters *locklist* für die automatische Optimierung muss auch der Parameter *maxlocks* für die automatische Optimierung aktiviert werden. Daher wird *maxlocks* auf AUTOMATIC gesetzt, wenn *locklist* auf AUTOMATIC gesetzt wird. Zur Aktivierung des Konfigurationsparameters *sheapthres_shr* für die automatische Optimierung muss auch der Parameter *sortheap* für die automatische Optimierung aktiviert werden. Daher wird *sortheap* auf AUTOMATIC gesetzt, wenn *sheapthres_shr* auf AUTOMATIC gesetzt wird.
3. Die automatische Optimierung von *sheapthres_shr* oder *sortheap* ist nur zulässig, wenn der Datenbankkonfigurationsparameter *sheapthres* auf den Wert 0 gesetzt ist.
4. Speicher mit automatischer Leistungsoptimierung wird nur auf dem primären HADR-Server verwendet. Wenn die Funktion für die automatische Speicheroptimierung auf einem HADR-System aktiviert wird, wird sie nie auf dem sekundären Server verwendet, und auch nur dann auf dem primären Server, wenn die Konfiguration ordnungsgemäß eingestellt ist. Wenn ein Befehl ausgeführt wird, der die Rollen der HADR-Datenbanken vertauscht, wird die Nutzung der Funktion für die automatische Speicheroptimierung ebenfalls übertragen, so dass sie auf dem primären Server erfolgt.

Zugehörige Konzepte:

- „Speicher mit automatischer Leistungsoptimierung“ auf Seite 295

Zugehörige Tasks:

- „Ermitteln der Speicherkonsumenten mit aktivierter automatischer Leistungsoptimierung“ auf Seite 299
- „Inaktivieren des Speichers mit automatischer Leistungsoptimierung“ auf Seite 297

Zugehörige Referenzen:

- „self_tuning_mem - Speicher mit automatischer Leistungsoptimierung “ auf Seite 444
- „sqlfupd data structure“ in *Administrative API Reference*

Inaktivieren des Speichers mit automatischer Leistungsoptimierung

Die automatische Leistungsoptimierung kann für die gesamte Datenbank inaktiviert werden, indem der Parameter *self_tuning_mem* auf den Wert OFF gesetzt wird. Wenn der Parameter *self_tuning_mem* auf den Wert OFF gesetzt wird, behal-

ten die Parameter für die Speicherkonfiguration und die Pufferpools, die den Wert AUTOMATIC haben, den Wert AUTOMATIC, und die Speicherbereiche behalten ihre aktuelle Größe bei.

Sie können den Parameter *self_tuning_mem* mit Hilfe des Befehls UPDATE DATABASE CONFIGURATION, mit der API SQLFUPD oder über das Fenster **Datenbankkonfigurationsparameter ändern** der Steuerzentrale auf OFF setzen.

Die automatische Leistungsoptimierung kann für die gesamte Datenbank effektiv außerdem dadurch inaktiviert werden, dass nur ein einziger Speicherkonsument für die automatische Leistungsoptimierung aktiviert wird. Dies liegt daran, dass kein Speicher umverteilt werden kann, wenn nur ein Hauptspeicherbereich aktiviert ist.

Zur Inaktivierung der automatisch Optimierung des Konfigurationsparameters *sorthheap* könnten Sie zum Beispiel den folgenden Befehl eingeben:

```
UPDATE DATABASE CONFIGURATION USING SORTHEAP MANUAL
```

Zur Inaktivierung der automatischen Optimierung des Konfigurationsparameters *sorthheap* und gleichzeitigen Änderung des aktuellen Werts von *sorthheap* in den Wert 2000 geben Sie den folgenden Befehl ein:

```
UPDATE DATABASE CONFIGURATION USING SORTHEAP 2000
```

In einigen Fällen kann ein Speicherkonfigurationsparameter für die automatische Optimierungsfunktion nur dann aktiviert werden, wenn ein anderer Speicherkonfigurationsparameter ebenfalls aktiviert wird. Zum Beispiel kann die automatische Optimierung für den Konfigurationsparameter *maxlocks* nur aktiviert werden, wenn der Konfigurationsparameter *locklist* ebenfalls für diese Funktion aktiviert wird. Desgleichen kann die automatische Optimierungsfunktion für den Konfigurationsparameter *sheapthres_shr* nur aktiviert werden, wenn sie auch für den Konfigurationsparameter *sorthheap* aktiviert wird. Dies bedeutet, dass eine Inaktivierung der automatischen Optimierung des Parameters *locklist* oder *sorthheap* auch die automatische Optimierung für den Parameter *maxlocks* bzw. *sheapthres_shr* inaktiviert.

Die automatische Optimierung kann für einen Pufferpool inaktiviert werden, indem der Pufferpool auf MANUAL oder auf eine bestimmte Größe gesetzt wird. Die beiden folgenden Anweisungen würden zum Beispiel die automatische Optimierung für den Pufferpool *bufferpool1* inaktivieren:

```
ALTER BUFFERPOOL bufferpool1 SIZE MANUAL  
ALTER BUFFERPOOL bufferpool1 SIZE 1000
```

Zugehörige Konzepte:

- „Speicher mit automatischer Leistungsoptimierung“ auf Seite 295
- „Verwenden von Speicher mit automatischer Leistungsoptimierung in Umgebungen mit partitionierten Datenbanken“ auf Seite 302

Zugehörige Tasks:

- „Aktivieren des Speichers mit automatischer Leistungsoptimierung“ auf Seite 296

Zugehörige Referenzen:

- „self_tuning_mem - Speicher mit automatischer Leistungsoptimierung “ auf Seite 444

Ermitteln der Speicherkonsumenten mit aktivierter automatischer Leistungsoptimierung

Zum Anzeigen der Einstellungen für die automatische Optimierungsfunktion für Speicherkonsumenten, die durch Konfigurationsparameter gesteuert werden, können Sie eine der folgenden Methoden verwenden.

- Zum Anzeigen der Einstellungen der automatischen Optimierungsfunktion für Konfigurationsparameter über die Befehlszeile verwenden Sie den Befehl `GET DATABASE CONFIGURATION` mit dem Parameter `SHOW DETAIL`.

Die Speicherkonsumenten, die für die automatische Optimierung aktiviert werden können, werden in der Ausgabe wie folgt zusammengruppiert:

Beschreibung	Parameter	Aktueller Wert	Verzögerter Wert
Speicher mit automatischer Leistungsoptimierung	(SELF_TUNING_MEM) = ON (Aktiv)	ON	ON
Größe des gemeinsamen Datenbankspeichers (4 KB)	(DATABASE_MEMORY) = AUTOMATIC(37200)	AUTOMATIC(37200)	AUTOMATIC(37200)
Max. Speicher für Sperrenliste (4 KB)	(LOCKLIST) = AUTOMATIC(7456)	AUTOMATIC(7456)	AUTOMATIC(7456)
Anzahl der Sperrenlisten pro Anwend. (in %)	(MAXLOCKS) = AUTOMATIC(98)	AUTOMATIC(98)	AUTOMATIC(98)
Größe des Paketcache (4 KB)	(PCKCACHESZ) = AUTOMATIC(5600)	AUTOMATIC(5600)	AUTOMATIC(5600)
Sortierspeicherschwelle für gemeinsame Sortierungen (4 KB)	(SHEAPTHRES_SHR) = AUTOMATIC(5000)	AUTOMATIC(5000)	AUTOMATIC(5000)
Zwischenspeicher für Sortierlisten (4 KB)	(SORTHEAP) = AUTOMATIC(256)	AUTOMATIC(256)	AUTOMATIC(256)

- Sie können auch die API `db2CfgGet` verwenden, um zu ermitteln, ob die Optimierung aktiviert ist oder nicht. Die folgenden Werte werden zurückgegeben:

SQLF_OFF	0
SQLF_ON_ACTIVE	2
SQLF_ON_INACTIVE	3

Der Wert `SQLF_ON_ACTIVE` beschreibt eine Situation, in der die automatische Optimierung aktiviert und aktiv ist, während der Wert `SQLF_ON_INACTIVE` anzeigt, dass die automatische Optimierung zwar aktiviert, jedoch zurzeit nicht aktiv ist.

- Sie können die Konfigurationseinstellungen auch im Fenster **Datenbank-konfiguration** der Steuerzentrale prüfen.

Zum Anzeigen der Einstellungen für die automatische Optimierungsfunktion für Pufferpools können Sie eine der folgenden Methoden verwenden.

- Zum Abrufen einer Liste von Pufferpools, für die die automatische Optimierung aktiviert ist, geben Sie den folgenden Befehl in die Befehlszeile ein:

```
db2 "select BPNAME, NPAGES from sysibm.sysbufferpools"
```

Wenn die automatische Optimierung für einen Pufferpool aktiviert ist, ist das Feld `NPAGES` in der Tabelle `SYSIBM.SYSBUFFERPOOLS` für den betreffenden Pufferpool auf den Wert `-2` gesetzt. Wenn die automatische Optimierung inaktiviert ist, enthält das Feld `NPAGES` die aktuelle Größe des Pufferpools.

- Zur Bestimmung der aktuellen Größe von Pufferpools, die für die automatische Optimierung aktiviert wurden, verwenden Sie den Snapshot Monitor wie folgt und prüfen die Daten des Monitorelements `'bp_cur_buffsz'` (Aktuelle Größe des Pufferpools):

```
db2 get snapshot for bufferpools on db_name
```

- Zum Anzeigen der Einstellungen für die automatische Optimierung Ihrer Pufferpools über die Steuerzentrale klicken Sie einen Pufferpool mit der rechten Maustaste an und prüfen die Attribute der Pufferpools im Teilfenster mit den Objektdetails.

Es ist wichtig zu beachten, dass die Reaktionsfähigkeit der Speicheroptimierungsfunktion durch die Zeit eingeschränkt wird, die zur Änderung der Größe eines Speicherkonsumenten erforderlich ist. Zum Beispiel kann die Verringerung der

Größe eines Pufferpools ein längerer Prozess sein, weshalb die Vorteile der Verkleinerung des Pufferpoolspeichers zu Gunsten einer Erweiterung des Sortierspeichers vielleicht nicht sofort realisiert werden.

Zugehörige Konzepte:

- „Speicher mit automatischer Leistungsoptimierung“ auf Seite 295

Zugehörige Tasks:

- „Aktivieren des Speichers mit automatischer Leistungsoptimierung“ auf Seite 296
- „Inaktivieren des Speichers mit automatischer Leistungsoptimierung“ auf Seite 297

Zugehörige Referenzen:

- „db2CfgGet API - Get the database manager or database configuration parameters“ in *Administrative API Reference*
- „db2GetAlertCfg API - Get the alert configuration settings for the health indicators“ in *Administrative API Reference*
- „GET DATABASE CONFIGURATION command“ in *Command Reference*
- „bp_cur_buffsz - Current Size of Buffer Pool monitor element“ in *Systemmonitor Handbuch und Referenz*

Speicher mit automatischer Leistungsoptimierung in Umgebungen mit partitionierten Datenbanken

Wenn die automatische Speicheroptimierungsfunktion in Umgebungen mit partitionierten Datenbanken verwendet wird, bestimmen einige wenige Faktoren, ob die Funktion das System geeignet optimiert.

Wenn der Speicher mit automatischer Leistungsoptimierung in partitionierten Datenbanken aktiviert wird, wird eine Datenbankpartition als Optimierungspartition bestimmt. Alle Entscheidungen bezüglich der Speicheroptimierung werden auf der Basis der Speicher- und Auslastungsmerkmale dieser Datenbankpartition getroffen. Wenn die Optimierungsentscheidungen in der Optimierungspartition getroffen sind, werden die Speicheranpassungen an alle anderen Datenbankpartitionen verteilt, um sicherzustellen, dass alle Datenbankpartitionen ähnliche Konfigurationen behalten.

Das auf einer Optimierungspartition basierende Modell setzt voraus, dass die Funktion nur in Datenbankpartitionen mit ähnlichen Speicheranforderungen verwendet wird. Die folgenden Richtlinien sind bei der Entscheidung zu beachten, ob die automatische Speicheroptimierung für eine partitionierte Datenbank aktiviert werden sollte.

Fälle, in denen die automatische Optimierung in partitionierten Datenbanken empfohlen wird

Wenn alle Datenbankpartitionen ähnliche Speicheranforderungen haben und auf ähnlicher Hardware betrieben werden, kann die automatische Speicheroptimierung ohne Modifikationen aktiviert werden. Solche Typen von Umgebungen haben die folgenden gemeinsamen Merkmale:

- Alle Datenbankpartitionen befinden sich auf identischer Hardware, und mehrere logische Knoten sind gleichmäßig auf mehrere physische Knoten verteilt.
- Sie weisen eine perfekte oder nahezu perfekte Verteilung der Daten auf.

- Die Auslastung, die in den Datenbankpartitionen verarbeitet wird, wird gleichmäßig auf die Datenbankpartitionen verteilt. Dies bedeutet, dass keine einzelne Datenbankpartition erhöhte Speicheranforderungen für einen oder mehrere Zwischenspeicher hat.

In einer solchen Umgebung ist für alle Datenbankpartitionen die gleiche Konfiguration wünschenswert, und die automatische Speicheroptimierung kann das System geeignet konfigurieren.

Fälle, in denen die automatische Optimierung in partitionierten Datenbanken nur mit Vorsicht empfohlen wird

In Fällen, in denen die meisten Datenbankpartitionen in einer Umgebung ähnliche Speicheranforderungen haben und auf ähnlicher Hardware betrieben werden, kann die automatische Speicheroptimierung eingesetzt werden, solange die Anfangskonfiguration mit Vorsicht erfolgt. Ein solches System hat möglicherweise eine Gruppe von Datenbankpartitionen für Daten und eine wesentlich kleinere Gruppe von Koordinatorpartitionen und Katalogpartitionen. In solchen Umgebungen kann es von Vorteil sein, die Koordinatorpartitionen und Katalogpartitionen anders zu konfigurieren als die Datenbankpartitionen mit den Daten.

Auch eine solche Umgebung kann von der automatischen Speicheroptimierungsfunktion profitieren, wenn einige kleinere Konfigurationsschritte ausgeführt werden. Da die Datenbankpartitionen mit den Daten den Hauptteil der Datenbankpartitionen ausmachen, sollte die automatische Optimierung für alle diese Datenbankpartitionen aktiviert und eine dieser Datenbankpartitionen als Optimierungspartition angegeben werden. Außerdem sollte die automatische Speicheroptimierung für die Katalog- und Koordinatorpartitionen inaktiviert werden, da sie möglicherweise eine andere Konfiguration aufweisen. Zur Inaktivierung der automatischen Optimierung in den Katalog- und Koordinatorpartitionen aktualisieren Sie den Datenbankkonfigurationsparameter *self_tuning_mem* in diesen Partitionen mit dem Wert OFF.

Fälle, in denen die automatische Optimierung in partitionierten Datenbanken nicht empfohlen wird

In Umgebungen, in denen sich die Speicheranforderungen jeder Datenbankpartition unterscheiden oder verschiedene Datenbankpartitionen auf völlig anderer Hardware betrieben werden, ist es empfehlenswert, die Funktion der automatischen Speicheroptimierung zu inaktivieren. Sie können dies erreichen, indem Sie in allen Partitionen den Datenbankkonfigurationsparameter *self_tuning_mem* auf den Wert OFF setzen.

Vergleich der Speicheranforderungen verschiedener Datenbankpartitionen

Die beste Methode zur Bestimmung, ob die Speicheranforderungen verschiedener Datenbankpartitionen ausreichend ähnlich sind, ist die Verwendung des Snapshot Monitors. Wenn die folgenden Monitorelemente in allen Partitionen ähnliche Werte liefern (mit Abweichungen von unter 20 %), können die Partitionen als ähnlich betrachtet werden.

Erfassen Sie die folgenden Daten, indem Sie den Befehl `get snapshot for database on <datenbankname>` ausführen.

Gesamter zugeordneter gemeinsamer Sortierspeicher	= 0
Obere Grenze für gemeinsamen Sortierspeicher	= 0
Sortiervorgang nach Schwellenwertüberschreitung (gemeinsam genutzter Speicher)	= 0
Überläufe bei Sortierung	= 0
Suchoperationen im Paket-Cache	= 13
Einfügungen im Paket-Cache	= 1

Überläufe des Paket-Caches	= 0
Obere Grenze für Paket-Cache (Byte)	= 655360
Anzahl Hash-Joins	= 0
Anzahl Hash-Schleifen	= 0
Anzahl Überläufe von Hash-Joins	= 0
Anzahl kleiner Überläufe von Hash-Joins	= 0
Hash-Joins nach Schwellenwertüberschreitung (gemeinsam genutzter Speicher)	= 0
Aktuelle Sperren	= 0
Warten bei Sperren	= 0
Wartezeit der Datenbank bei Sperren (ms)	= 0
Verwendeter Speicher für Sperrenlisten (Byte)	= 4968
Sperreneskalationen	= 0
Exklusive Sperreneskalationen	= 0

Erfassen Sie die folgenden Daten, indem Sie den Befehl `get snapshot for bufferpools on <datenbankname>` ausführen.

Logische Lesevorgänge im Pufferpool	= 0
Physische Lesevorgänge im Pufferpool	= 0
Logische Lesevorgänge im Pufferpoolindex	= 0
Physische Lesevorgänge im Pufferpoolindex	= 0
Gesamtzeit der Lesevorgänge im Pufferpool (ms)	= 0
Gesamtzeit der Schreibvorgänge im Pufferpool (ms)	= 0

Zugehörige Konzepte:

- „Speicher mit automatischer Leistungsoptimierung“ auf Seite 295
- „Verwenden von Speicher mit automatischer Leistungsoptimierung in Umgebungen mit partitionierten Datenbanken“ auf Seite 302

Zugehörige Referenzen:

- „self_tuning_mem - Speicher mit automatischer Leistungsoptimierung “ auf Seite 444
- „GET SNAPSHOT command“ in *Command Reference*

Verwenden von Speicher mit automatischer Leistungs-optimierung in Umgebungen mit partitionierten Datenbanken

Wenn die Funktion für die automatische Optimierung in Umgebungen mit partitionierten Datenbanken aktiviert wird, überwacht eine einzige Datenbankpartition, die als *Optimierungspartition* bezeichnet wird, die Speicherkonfiguration und gibt alle Konfigurationsänderungen an alle anderen Datenbankpartitionen weiter, um eine konsistente Konfiguration über alle Datenbankpartitionen hinweg sicherzustellen.

Die Optimierungspartition wird nach einer Reihe von Merkmalen ausgewählt, wie zum Beispiel der Anzahl von Datenbankpartitionen in der Partitionsgruppe und der Anzahl der definierten Pufferpools.

- Wenn Sie ermitteln möchten, welche Datenbankpartition zurzeit als Optimierungspartition angegeben ist, verwenden Sie die Prozedur `ADMIN_CMD` wie folgt:
`CALL SYSPROC.ADMIN_CMD( 'get stmm tuning dbpartitionnum' )`
- Zum Ändern der Optimierungspartition verwenden Sie die Prozedur `ADMIN_CMD` wie folgt:
`CALL SYSPROC.ADMIN_CMD( 'update stmm tuning dbpartitionnum <db_partitionsnummer>' )`

Wenn Sie diesen Befehl absetzen, wird die Optimierungspartition asynchron oder beim nächsten Starten der Datenbank aktualisiert.

- Wenn die Optimierungspartition von der Speicheroptimierungsfunktion automatisch neu ausgewählt werden soll, geben Sie für <db_partitionsnummer> den Wert -1 ein.

Starten der Speicheroptimierungsfunktion auf DPF-Systemen

In einer DPF-Umgebung wird die Speicheroptimierungsfunktion nur gestartet, wenn die Datenbank explizit mit dem Befehl `ACTIVATE DATABASE` aktiviert wird, weil die automatische Optimierung voraussetzt, dass alle Partitionen aktiv sind, bevor sie den Speicher in einem Mehrpartitionssystem richtig optimieren kann.

Inaktivieren der automatischen Optimierung für eine bestimmte Datenbankpartition

- Zur Inaktivierung der automatischen Optimierung für eine Teilgruppe von Datenbankpartitionen setzen Sie für die Datenbankpartitionen, die nicht optimiert werden sollen, den Konfigurationsparameter `self_tuning_mem` auf den Wert `OFF`.
- Zur Inaktivierung der automatischen Optimierung für eine Teilgruppe von Speicherkonsumenten, die durch Konfigurationsparameter gesteuert werden, in einer bestimmten Datenbankpartition, setzen Sie den Wert des relevanten Konfigurationsparameters oder die Pufferpoolgröße auf `MANUAL` bzw. einen bestimmten Wert in dieser Datenbankpartition.
- Wenn Sie die Optimierung für einen bestimmten Pufferpool in einer Datenbankpartition inaktivieren, führen Sie den Befehl `ALTER BUFFER POOL` mit der Angabe eines Größenwerts und eines Werts für den Parameter `PARTITIONNUM` für die Partition aus, in der die automatische Optimierung inaktiviert werden soll.

Dieser Befehl `ALTER BUFFERPOOL` erstellt einen Ausnahmeeintrag für den angegebenen Pufferpool in der Tabelle `SYSIBM.SYSBUFFERPOOLNODES`. Wenn in dieser Tabelle ein Eintrag für einen Pufferpool vorhanden ist, kann der Pufferpool für die automatische Optimierung nicht reaktiviert werden. Gehen Sie daher wie folgt vor, wenn Sie einen Ausnahmeeintrag entfernen möchten, so dass ein Pufferpool für die automatische Optimierung wieder aktiviert werden kann:

1. Führen Sie die Anweisung `ALTER BUFFERPOOL` aus, indem Sie die Pufferpoolgröße (`SIZE`) auf `MANUAL` setzen.
2. Führen Sie eine weitere Anweisung `ALTER BUFFERPOOL` aus, indem Sie die Pufferpoolgröße in der Partition mit dem Ausnahmeeintrag auf denselben Wert setzen, der in der Tabelle `SYSIBM.SYSBUFFERPOOLS` angegeben ist.
3. Aktivieren Sie die automatische Optimierung, indem Sie eine weitere Anweisung `ALTER BUFFERPOOL` ausführen, in der die Größe (`SIZE`) auf den Wert `AUTOMATIC` gesetzt ist.

Aktivieren der automatischen Optimierung in nicht einheitlichen Umgebungen

Im Idealfall sollten Ihre Daten gleichmäßig auf alle Datenbankpartitionen verteilt und die Auslastung in jeder Partition durch ähnliche Speicheranforderungen gekennzeichnet sein. Wenn die Datenverteilung ungleichmäßig ist, so dass mindestens eine Datenbankpartition erheblich mehr oder weniger Daten als andere Datenbankpartitionen enthält, sollten solche anomalen Datenbankpartitionen nicht für die automatische Optimierung aktiviert werden. Dasselbe gilt, wenn die

Speicheranforderungen in den Datenbankpartitionen unterschiedlich sind. Dies kann geschehen, wenn zum Beispiel ressourcenintensive Sortiervorgänge nur in einer Partition ausgeführt werden oder wenn einige Datenbankpartitionen mit anderer Hardware und mehr verfügbarem Speicher als andere Partitionen ausgestattet sind. Die automatische Optimierung kann dennoch in einigen Datenbankpartitionen in diesem Typ von Umgebung aktiviert werden. Zur Nutzung der Vorteile der automatischen Speicheroptimierung in Umgebungen mit ungleich verteilten Anforderungen, geben Sie eine Gruppe von Datenbankpartitionen an, die ähnliche Daten- und Speicheranforderungen haben, und aktivieren sie für die automatische Optimierung. Die Speicherkonfiguration in den übrigen Partitionen muss in diesem Fall manuell erfolgen.

Zugehörige Konzepte:

- „Speicher mit automatischer Leistungsoptimierung“ auf Seite 295

Zugehörige Tasks:

- „Ermitteln der Speicherkonsumenten mit aktivierter automatischer Leistungsoptimierung“ auf Seite 299

Zugehörige Referenzen:

- „self_tuning_mem - Speicher mit automatischer Leistungsoptimierung “ auf Seite 444

Betriebsmerkmale und Einschränkungen von Speicher mit automatischer Leistungsoptimierung

Ermitteln der Optimierungsanforderungen

Zur Sicherstellung einer sachgerechten und relevanten Vergleichbarkeit von Speicherkonsumenten wurde eine neue einheitliche Metrik entwickelt. Jeder optimierte Speicherkonsument berechnet eine Voraussage über den durch zusätzlichen Speicher erzielbaren Nutzen und meldet diesen Wert an den Prozess der automatischen Speicheroptimierung. Der Speicher mit automatischer Leistungsoptimierung verwendet diese Werte als Basis für die Speicheroptimierung, indem er Speicherressourcen von Konsumenten mit dem geringsten Bedarf abzieht und Speicher den Speicherkonsumenten zuordnet, die am meisten davon profitieren.

Häufigkeit der Speicheroptimierung

Wenn die automatische Leistungsoptimierung aktiviert ist, überprüft sie in regelmäßigen Abständen Veränderungen der Datenbankauslastung. Wenn die Auslastung nicht konstant ist (d. h., wenn die ausgeführten Abfragen keine ähnlichen Speichermerkmale zeigen), ordnet die Speicheroptimierung Speicher weniger häufig (in Abständen von bis zu zehn Minuten zwischen Optimierungszyklen) zu, um stabilere Voraussagen von Trends zu erzielen. Bei Auslastungen mit konstanteren Speichernutzungsprofilen führt die Speicheroptimierung häufigere Optimierungen durch (in auf bis zu 30 Sekunden verkürzten Abständen zwischen Optimierungszyklen), um die beste Speicherkonfiguration rascher zu erreichen.

Verfolgen des Prozesses der automatischen Speicher-optimierung

Die aktuelle Hauptspeicherkonfiguration kann mit Hilfe des Befehls GET DATABASE CONFIGURATION oder durch eine Momentaufnahme (Snapshot) abgerufen werden. Änderungen, die durch die automatische Leistungsoptimierung erfolgen, werden in den Protokolldateien der Speicheroptimierung im Verzeichnis 'stmmlog' aufgezeichnet. Die Protokolldateien der Speicheroptimierung enthalten Zusammen-

fassungen der Ressourcenbedarfsdaten für jeden Speicherkonsumenten in jedem Optimierungsintervall. Diese Intervalle können anhand der Zeitmarken in den Protokolleinträgen ermittelt werden.

Erwartete Zeit für das Erreichen der besten Konfiguration

Wenn diese Funktion aktiviert bleibt, sollte dies rasch zu einer optimalen Einstellung von Parametern zur Optimierung der Speichernutzung führen. Ein System kann ausgehend von einer Erstkonfiguration bereits innerhalb von einer Stunde optimiert werden. In den meisten Fällen wird die Optimierung in der Regel in maximal 10 Stunden abgeschlossen. Dieser Extremfall tritt auf, wenn Abfragen, die an der Datenbank ausgeführt werden, deutliche Unterschiede in ihren Speicherbedarfsmerkmalen aufweisen.

Einschränkungen der automatischen Speicheroptimierung

In Fällen, in denen nur kleine Speichergrößen verfügbar sind (z. B., weil der Wert des Parameters *database_memory* sehr niedrig eingestellt ist oder weil mehrere Datenbanken, Instanzen oder andere Anwendungen auf dem Server ausgeführt werden), sind die durch die automatische Speicheroptimierung erzielbaren Vorteile begrenzt.

Da bei der automatischen Speicheroptimierung Optimierungsentscheidungen auf der Basis der Datenbankauslastung erfolgen, schränken Auslastungen mit wechselnden Speichernutzungsmerkmalen die Möglichkeiten für eine effektive Optimierung durch die Funktion der automatischen Leistungsoptimierung ein. Wenn sich die Speichernutzungsmerkmale Ihrer Auslastung beständig ändern, führt die automatische Speicheroptimierung weniger häufige Optimierungen durch und richtet diese Optimierungen wiederholt auf wechselnde Zielbedingungen aus. In diesem Fall kann die automatische Speicheroptimierung keine absolute Konvergenz auf eine optimale Speicherkonfiguration erzielen, sondern versucht stattdessen eine Speicherkonfiguration zu erhalten, die auf die aktuelle Auslastung optimal zugeschnitten ist.

Zugehörige Konzepte:

- „Speicher mit automatischer Leistungsoptimierung“ auf Seite 295

Zugehörige Tasks:

- „Inaktivieren des Speichers mit automatischer Leistungsoptimierung“ auf Seite 297
- „Aktivieren des Speichers mit automatischer Leistungsoptimierung“ auf Seite 296

Zugehörige Referenzen:

- „database_memory - Größe des gemeinsam genutzten Datenbankspeichers “ auf Seite 434
- „GET DATABASE CONFIGURATION command“ in *Command Reference*

Konfigurationsparameter - Übersicht

Übersicht über die Konfigurationsparameter des Datenbankmanagers

In der folgenden Tabelle sind die Parameter der Konfigurationsdatei des Datenbankmanagers für Datenbankserver aufgeführt. Beachten Sie beim Ändern der Konfigurationsparameter des Datenbankmanagers die detaillierten Informationen zu jedem Parameter. Informationen zu spezifischen Betriebsumgebungen mit Standardwerten sind in jeder Parameterbeschreibung enthalten.

Für einige Konfigurationsparameter des Datenbankmanagers muss der Datenbankmanager gestoppt (db2stop) und erneut gestartet (db2start) werden, um die neuen Parameterwerte in Kraft zu setzen. Andere Parameter können online geändert werden. Diese werden als *online konfigurierbare Konfigurationsparameter* bezeichnet. Wenn Sie die Einstellung eines online konfigurierbaren Konfigurationsparameters des Datenbankmanagers ändern, während Sie mit einer Instanz verbunden sind, wird durch das Standardverhalten des Befehls UPDATE DBM CFG die Änderung unverzüglich angewendet. Wenn die Änderung nicht sofort angewendet werden soll, verwenden Sie die Option DEFERRED im Befehl UPDATE DBM CFG.

Die Spalte „Auto.“ in der folgenden Tabelle gibt an, ob der Parameter das Schlüsselwort AUTOMATIC im Befehl UPDATE DATABASE MANAGER CONFIGURATION unterstützt. Wenn Sie einen Parameter auf den Wert „AUTOMATIC“ setzen, passt DB2 den Parameter automatisch so an, dass er die aktuellen Ressourcenanforderungen widerspiegelt.

In der Spalte „Leistungsrelevanz“ (Leist.-Relev.) ist vermerkt, in welchem relativen Ausmaß sich jeder Parameter in Bezug auf die Systemleistung auswirken kann. Diese Spalte kann allerdings nicht für alle Arten von Umgebungen gültige Angaben enthalten. Sie sollten die Informationen als allgemeine Hinweise betrachten.

- **Hoch:** Gibt an, dass ein Parameter wesentliche Auswirkungen auf die Leistung haben kann. Die Werte für diese Parameter müssen sehr eingehend überlegt werden. In einigen Fällen kann das bedeuten, dass Sie die vorgegebenen Standardwerte übernehmen sollten.
- **Mittel:** Gibt an, dass der Parameter unter Umständen Auswirkungen auf die Leistung haben kann. Sie sollten von Ihrer spezifischen Umgebung und Ihren Anforderungen ausgehend entscheiden, wie viel Aufwand in die Optimierung dieser Parameter investiert werden sollte.
- **Niedrig:** Gibt an, dass der Parameter weniger allgemeine bzw. weniger bedeutende Auswirkungen auf die Leistung hat.
- **Keine:** Gibt an, dass der Parameter keine direkten Auswirkungen auf die Leistung hat. Da diese Parameter nicht leistungsrelevant sind, müssen sie nicht optimiert werden. Sie können jedoch in anderer Hinsicht für die Systemkonfiguration von Bedeutung sein, zum Beispiel zur Aktivierung der Kommunikationsunterstützung.

Die Spalten „Token“, „Tokenwert“ und „Datentyp“ stellen Informationen bereit, die Sie beim Aufrufen der API `db2CfgGet` oder `db2CfgSet` benötigen. Diese Informationen enthalten Kennungen für Konfigurationsparameter, Einträge für das Element *token* in der Datenstruktur *db2CfgParam* und Datentypen für Werte, die an die Struktur übergeben werden.

Tabelle 62. Konfigurierbare Konfigurationsparameter des Datenbankmanagers

Parameter	Onl. Cfg.	Auto.	Leist.-Relev.	Token	Tokenwert	Datentyp	Zusätzliche Informationen
<i>agent_stack_sz</i>	Nein	Nein	Niedrig	SQLF_KTN_AGENT_STACK_SZ	61	UInt16	„agent_stack_sz - Größe des Agentenstacks “ auf Seite 452
<i>agentpri</i>	Nein	Nein	Hoch	SQLF_KTN_AGENTPRI	26	Sint16	„agentpri - Agentenpriorität “ auf Seite 482
<i>aslheapsz</i>	Nein	Nein	Hoch	SQLF_KTN_ASHEAPSZ	15	UInt32	„aslheapsz - Zwischenspeichergröße für Anwendungsunterstützungsebene “ auf Seite 461
<i>audit_buf_sz</i>	Nein	Nein	Hoch	SQLF_KTN_AUDIT_BUF_SZ	312	Sint32	„audit_buf_sz - Prüfpuffergröße “ auf Seite 466
<i>authentication</i> ¹	Nein	Nein	Niedrig	SQLF_KTN_AUTHENTICATION	78	UInt16	„authentication - Authentifizierungstyp “ auf Seite 575
<i>catalog_noauth</i>	Ja	Nein	Keine	SQLF_KTN_CATALOG_NOAUTH	314	UInt16	„catalog_noauth - Katalogisieren ohne Berechtigung zulässig “ auf Seite 576

Tabelle 62. Konfigurierbare Konfigurationsparameter des Datenbankmanagers (Forts.)

Parameter	Onl. Cfg.	Auto.	Leist.-Relev.	Token	Token-wert	Datentyp	Zusätzliche Informationen
clnt_krb_plugin	Nein	Nein	Keine	SQLF_KTN_CLNT_KRB_PLUGIN	812	char(33)	„clnt_krb_plugin - Client-Kerberos-Plug-in “ auf Seite 577
clnt_pw_plugin	Nein	Nein	Keine	SQLF_KTN_CLNT_PW_PLUGIN	811	char(33)	„clnt_pw_plugin - Client-Plug-in für Benutzer-ID und Kennwort “ auf Seite 578
comm_bandwidth	Ja	Nein	Mittel	SQLF_KTN_COMM_BANDWIDTH	307	float	„comm_bandwidth - Kommunikationsbandbreite “ auf Seite 566
conn_elapse	Ja	Nein	Mittel	SQLF_KTN_CONN_ELAPSE	508	UInt16	„conn_elapse - Antwortzeit für Verbindung “ auf Seite 554
cpuspeed	Ja	Nein	Low ²	SQLF_KTN_CPUSPEED	42	float	„cpuspeed - CPU-Geschwindigkeit “ auf Seite 567
dft_account_str	Ja	Nein	Keine	SQLF_KTN_DFT_ACCOUNT_STR	28	char(25)	„dft_account_str - Standardzeichenfolge für Abrechnung “ auf Seite 568
dft_monswitches • dft_mon_bufpool • dft_mon_lock • dft_mon_sort • dft_mon_stmt • dft_mon_table • dft_mon_timestamp • dft_mon_uow	Ja	Nein	Mittel	SQLF_KTN_DFT_MONSWITCHES ³ • SQLF_KTN_DFT_MON_BUFPOOL • SQLF_KTN_DFT_MON_LOCK • SQLF_KTN_DFT_MON_SORT • SQLF_KTN_DFT_MON_STMT • SQLF_KTN_DFT_MON_TABLE • SQLF_KTN_DFT_MON_TIMESTAMP • SQLF_KTN_DFT_MON_UOW	29 • 33 • 34 • 35 • 31 • 32 • 36 • 30	UInt16 • UInt16 • UInt16 • UInt16 • UInt16 • UInt16 • UInt16 • UInt16	„dft_monswitches - Monitorschalter des Standarddatenbanksystems “ auf Seite 565
dftdbpath	Ja	Nein	Keine	SQLF_KTN_DFTDBPATH	27	char(215)	„dftdbpath - Standarddatenbankpfad “ auf Seite 578
diaglevel	Ja	Nein	Niedrig	SQLF_KTN_DIAGLEVEL	64	UInt16	„diaglevel - Aufzeichnungsebene bei Fehlerdiagnose “ auf Seite 561
diagpath	Ja	Nein	Keine	SQLF_KTN_DIAGPATH	65	char(215)	„diagpath - Verzeichnispfad für Diagnosedaten “ auf Seite 562
dir_cache	Nein	Nein	Mittel	SQLF_KTN_DIR_CACHE	40	UInt16	„dir_cache - Verzeichniscacheunterstützung “ auf Seite 467
discover ⁴	Nein	Nein	Mittel	SQLF_KTN_DISCOVER	304	UInt16	„discover - Discovery-Modus “ auf Seite 552
discover_inst	Ja	Nein	Niedrig	SQLF_KTN_DISCOVER_INST	308	UInt16	„discover_inst - Discovery-Serverinstanz “ auf Seite 553
fcm_num_buffers	Ja	Ja	Mittel	SQLF_KTN_FCM_NUM_BUFFERS	503	UInt32	„fcm_num_buffers - Anzahl FCM-Puffer “ auf Seite 554
fcm_num_channels	Ja	Ja	Mittel	SQLF_KTN_FCM_NUM_CHANNELS	902	UInt32	„fcm_num_channels - Anzahl FCM-Kanäle “ auf Seite 555
fed_noauth	Ja	Nein	Keine	SQLF_KTN_FED_NOAUTH	806	UInt16	„fed_noauth - Authentifizierung bei Servern mit föderierten Datenbanken umgehen “ auf Seite 579
federated	Nein	Nein	Mittel	SQLF_KTN_FEDERATED	604	Sint16	„federated - Unterstützung für Systeme föderierter Datenbanken “ auf Seite 569
fenced_pool	Nein	Nein	Mittel	SQLF_KTN_FENCED_POOL	80	Sint32	„fenced_pool - Maximale Anzahl abgeschirmter Prozesse “ auf Seite 493
group_plugin	Nein	Nein	Keine	SQLF_KTN_GROUP_PLUGIN	810	char(33)	„group_plugin - Gruppen-Plug-in “ auf Seite 580
health_mon	Ja	Nein	Niedrig	SQLF_KTN_HEALTH_MON	804	UInt16	„health_mon - Überwachung mit dem Diagnosemonitor “ auf Seite 563
indexrec ⁵	Ja	Nein	Mittel	SQLF_KTN_INDEXREC	20	UInt16	„indexrec - Zeitpunkt für Indexneuerstellung “ auf Seite 523
instance_memory	Nein	Ja	Mittel	SQLF_KTN_INSTANCE_MEMORY	803	UInt64	„instance_memory - Instanzspeicher “ auf Seite 468
intra_parallel	Nein	Nein	Hoch	SQLF_KTN_INTRA_PARALLEL	306	Sint16	„intra_parallel - Partitionsinterne Parallelität aktivieren “ auf Seite 559
java_heap_sz	Nein	Nein	Hoch	SQLF_KTN_JAVA_HEAP_SZ	310	Sint32	„java_heap_sz - Maximale Zwischenspeichergröße für Java-Interpreter “ auf Seite 469
jdk_path	Nein	Nein	Keine	SQLF_KTN_JDK_PATH	311	char(255)	„jdk_path - Installationspfad für das Software Developer's Kit für Java “ auf Seite 569
keepfenced	Nein	Nein	Mittel	SQLF_KTN_KEEPPFENCED	81	UInt16	„keepfenced - Abgeschirmten Prozess beibehalten “ auf Seite 494
local_gssplugin	Nein	Nein	Keine	SQLF_KTN_LOCAL_GSSPLUGIN	816	char(33)	„local_gssplugin - GSS-API-Plug-in für lokale Berechtigung auf Instanzebene “ auf Seite 580
max_connections	Nein	Nein	Mittel	SQLF_DBTN_MAX_CONNECTIONS	802	Sint32	„max_connections - Maximale Anzahl von Clientverbindungen “ auf Seite 485

Tabelle 62. Konfigurierbare Konfigurationsparameter des Datenbankmanagers (Forts.)

Parameter	Onl. Cfg.	Auto.	Leist.-Relev.	Token	Tokenwert	Datentyp	Zusätzliche Informationen
max_connretries	Ja	Nein	Mittel	SQLF_KTN_MAX_CONNRETRIES	509	UInt16	„max_connretries - Anzahl Wiederholungen für Verbindungsaufbau zum Knoten “ auf Seite 556
max_coordagents	Nein	Nein	Mittel	SQLF_KTN_MAX_COORDAGENTS	501	Sint32	„max_coordagents - Maximale Anzahl koordinierender Agenten “ auf Seite 485
max_querydegree	Ja	Nein	Hoch	SQLF_KTN_MAX_QUERYDEGREE	303	Sint32	„max_querydegree - Maximaler Grad der Parallelität bei Abfragen “ auf Seite 559
max_time_diff	Nein	Nein	Mittel	SQLF_KTN_MAX_TIME_DIFF	510	UInt16	„max_time_diff - Maximale Zeitdifferenz zwischen Knoten “ auf Seite 557
maxagents	Nein	Nein	Mittel	SQLF_KTN_MAXAGENTS	12	UInt32	„maxagents - Maximale Anzahl von Agenten “ auf Seite 487
maxcagents	Nein	Nein	Mittel	SQLF_KTN_MAXCAGENTS	13	Sint32	„maxcagents - Maximale Anzahl gleichzeitig aktiver Agenten “ auf Seite 489
maxtotfilop	Nein	Nein	Mittel	SQLF_KTN_MAXTOTFILOP	45	UInt16	„maxtotfilop - Maximale Anzahl offener Dateien “ auf Seite 491
mon_heap_sz	Nein	Nein	Niedrig	SQLF_KTN_MON_HEAP_SZ	79	UInt16	„mon_heap_sz - Zwischenspeichergröße für Datenbanksystemmonitor “ auf Seite 470
nname	Nein	Nein	Keine	SQLF_KTN_NNAME	7	char(8)	„nname - Name der NetBIOS-Workstation “ auf Seite 550
notifylevel	Ja	Nein	Niedrig	SQLF_KTN_NOTIFYLEVEL	605	Sint16	„notifylevel - Benachrichtigungsstufe “ auf Seite 563
num_initagents	Nein	Nein	Mittel	SQLF_KTN_NUM_INITAGENTS	500	UInt32	„num_initagents - Anfangswert für die Anzahl Agenten im Pool “ auf Seite 491
num_initfenced	Nein	Nein	Mittel	SQLF_KTN_NUM_INITFENCED	601	Sint32	„num_initfenced - Anfangswert für die Anzahl abgeschirmter Prozesse “ auf Seite 496
num_poolagents	Nein	Nein	Hoch	SQLF_KTN_NUM_POOLAGENTS	502	Sint32	„num_poolagents - Agentenpoolgröße “ auf Seite 492
numdb	Nein	Nein	Niedrig	SQLF_KTN_NUMDB	6	UInt16	„numdb - Maximale Anzahl gleichzeitig aktiver Datenbanken einschließlich Host- und iSeries-Datenbanken “ auf Seite 570
query_heap_sz	Nein	Nein	Mittel	SQLF_KTN_QUERY_HEAP_SZ	49	Sint32	„query_heap_sz - Größe des Abfrage-zwischenspeichers “ auf Seite 455
resync_interval	Nein	Nein	Keine	SQLF_KTN_RESYNC_INTERVAL	68	UInt16	„resync_interval - Intervall für Transaktionsresynchronisation “ auf Seite 530
rqrioblkl	Nein	Nein	Hoch	SQLF_KTN_RQRIOBLK	1	UInt16	„rqrioblkl - E/A-Blockgröße für Clients “ auf Seite 464
sheapthres	Nein	Nein	Hoch	SQLF_KTN_SHEAPTHRES	21	UInt32	„sheapthres - Schwellenwert für Sortierspeicher “ auf Seite 456
spm_log_file_sz	Nein	Nein	Niedrig	SQLF_KTN_SPM_LOG_FILE_SZ	90	Sint32	„spm_log_file_sz - Protokolldateigröße für SPM “ auf Seite 530
spm_log_path	Nein	Nein	Mittel	SQLF_KTN_SPM_LOG_PATH	313	char(226)	„spm_log_path - Protokolldateipfad für SPM “ auf Seite 531
spm_max_resync	Nein	Nein	Niedrig	SQLF_KTN_SPM_MAX_RESYNC	91	Sint32	„spm_max_resync - Maximale Anzahl von SPM-Resynchronisationsagenten “ auf Seite 532
spm_name	Nein	Nein	Keine	SQLF_KTN_SPM_NAME	92	char(8)	„spm_name - Name des Synchronisationspunktmanagers “ auf Seite 532
srvcon_auth	Nein	Nein	Keine	SQLF_KTN_SRVCON_AUTH	815	UInt16	„srvcon_auth - Authentifizierungstyp für ankommende Verbindungen auf dem Server “ auf Seite 581
srvcon_gssplugin_list	Nein	Nein	Keine	SQLF_KTN_SRVCON_GSSPLUGIN_LIST	814	char(256)	„srvcon_gssplugin_list - Liste der GSS-API-Plug-ins für ankommende Verbindungen auf dem Server “ auf Seite 581
srv_plugin_mode	Nein	Nein	Keine	SQLF_KTN_SRV_PLUGIN_MODE	809	UInt16	„srv_plugin_mode - Server-Plug-in-Modus “ auf Seite 583
srvcon_pw_plugin	Nein	Nein	Keine	SQLF_KTN_SRVCON_PW_PLUGIN	813	char(33)	„srvcon_pw_plugin - Plug-in für Benutzer-ID/Kennwort für ankommende Verbindungen auf dem Server “ auf Seite 582
start_stop_time	Ja	Nein	Niedrig	SQLF_KTN_START_STOP_TIME	511	UInt16	„start_stop_time - Zeitlimit für DB2START und DB2STOP “ auf Seite 557
svconame	Nein	Nein	Keine	SQLF_KTN_SVCENAME	24	char(14)	„svconame - TCP/IP-Servicename “ auf Seite 551
sysadm_group	Nein	Nein	Keine	SQLF_KTN_SYSADM_GROUP	39	char(30)	„sysadm_group - SYSADM-Gruppenname “ auf Seite 583

Tabelle 62. Konfigurierbare Konfigurationsparameter des Datenbankmanagers (Forts.)

Parameter	Onl. Cfg.	Auto.	Leist.-Relev.	Token	Tokenwert	Datentyp	Zusätzliche Informationen
<i>sysctrl_group</i>	Nein	Nein	Keine	SQLF_KTN_SYSCtrl_GROUP	63	char(30)	„sysctrl_group - SYSCtrl-Gruppenname “ auf Seite 584
<i>sysmaint_group</i>	Nein	Nein	Keine	SQLF_KTN_SYSMAINT_GROUP	62	char(30)	„sysmaint_group - SYSMAINT-Gruppenname “ auf Seite 585
<i>sysmon_group</i>	Nein	Nein	Keine	SQLF_KTN_SYSMON	808	char(30)	„sysmon_group - Berechtigungsgruppenname für Systemmonitor “ auf Seite 586
<i>tm_database</i>	Nein	Nein	Keine	SQLF_KTN_TM_DATABASE	67	char(8)	„tm_database - Name für Transaktionsmanagerdatenbank “ auf Seite 533
<i>tp_mon_name</i>	Nein	Nein	Keine	SQLF_KTN_TP_MON_NAME	66	char(19)	„tp_mon_name - Name des Transaktionsprozessormonitors “ auf Seite 571
<i>trust_allclnts</i> ⁶	Nein	Nein	Keine	SQLF_KTN_TRUST_ALLCLNTS	301	Uint16	„trust_allclnts - Alle Clients akzeptieren “ auf Seite 587
<i>trust_clntauth</i>	Nein	Nein	Keine	SQLF_KTN_TRUST_CLNTAUTH	302	Uint16	„trust_clntauth - Authentifizierung gesicherter Clients “ auf Seite 588
<i>util_impact_lim</i>	Ja	Nein	Hoch	SQLF_KTN_UTIL_IMPACT_LIM	807	Uint32	„util_impact_lim - Richtlinie für Instanzauslastungswirkung “ auf Seite 573
Anmerkungen: 1. Gültige Werte (in <code>sqlenv.h</code> definiert) sind: SQL_AUTHENTICATION_SERVER (0) SQL_AUTHENTICATION_CLIENT (1) SQL_AUTHENTICATION_DCS (2) SQL_AUTHENTICATION_DCE (3) SQL_AUTHENTICATION_SVR_ENCRYPT (4) SQL_AUTHENTICATION_DCS_ENCRYPT (5) SQL_AUTHENTICATION_DCE_SVR_ENC (6) SQL_AUTHENTICATION_KERBEROS (7) SQL_AUTHENTICATION_KRB_SVR_ENC (8) SQL_AUTHENTICATION_GSSPLUGIN (9) SQL_AUTHENTICATION_GSS_SVR_ENC (10) SQL_AUTHENTICATION_DATAENC (11) SQL_AUTHENTICATION_DATAENC_CMP (12) SQL_AUTHENTICATION_NOT_SPEC (255) 2. Der Parameter <i>cpuspeed</i> kann sich wesentlich auf die Leistung auswirken, jedoch sollten Sie den Standardwert verwenden, sofern nicht spezielle Umstände vorliegen, wie sie in der Parameterbeschreibung dokumentiert sind. 3. Bit 1 (xxxx xxx1): dft_mon_uow Bit 2 (xxxx x1x): dft_mon_stmt Bit 3 (xxxx x1xx): dft_mon_table Bit 4 (xxxx 1xxx): dft_mon_buffpool Bit 5 (xxx1 xxxx): dft_mon_lock Bit 6 (xx1x xxxx): dft_mon_sort Bit 7 (x1xx xxxx): dft_mon_timestamp 4. Gültige Werte (in <code>sqlutil.h</code> definiert) sind: SQLF_DSCVR_KNOWN (1) SQLF_DSCVR_SEARCH (2) 5. Gültige Werte (in <code>sqlutil.h</code> definiert) sind: SQLF_INX_REC_SYSTEM (0) SQLF_INX_REC_REFERENCE (1) 6. Gültige Werte (in <code>sqlutil.h</code> definiert) sind: SQLF_TRUST_ALLCLNTS_NO (0) SQLF_TRUST_ALLCLNTS_YES (1) SQLF_TRUST_ALLCLNTS_DRDAONLY (2)							

Tabelle 63. Informative Konfigurationsparameter des Datenbankmanagers

Parameter	Token	Tokenwert	Datentyp	Zusätzliche Informationen
<i>nodetype</i> ¹	SQLF_KTN_NODETYPE	100	Uint16	„nodetype - Knotenart des Systems “ auf Seite 570
<i>release</i>	SQLF_KTN_RELEASE	101	Uint16	„release - Releasestand der Datenbankkonfiguration “ auf Seite 537

Tabelle 63. Informative Konfigurationsparameter des Datenbankmanagers (Forts.)

Parameter	Token	Token- wert	Daten- typ	Zusätzliche Informationen
Anmerkungen: 1. Gültige Werte (in <code>sqlutil.h</code> definiert) sind: SQLF_NT_STANDALONE (0) SQLF_NT_SERVER (1) SQLF_NT_REQUESTOR (2) SQLF_NT_STAND_REQ (3) SQLF_NT_MPP (4) SQLF_NT_SATELLITE (5)				

Übersicht über die Konfigurationsparameter der Datenbank

In der folgenden Tabelle sind die Parameter der Konfigurationsdatei für die Datenbank aufgeführt. Wenn Sie Konfigurationsparameter der Datenbank ändern möchten, lesen Sie die detaillierten Informationen zu den Parametern.

Für einige Datenbankkonfigurationsparameter werden Änderungen erst dann wirksam, wenn die Datenbank erneut aktiviert wird. Dazu müssen zunächst alle Anwendungen ihre Verbindung zur Datenbank trennen. (Wenn die Datenbank aktiviert war, muss sie inaktiviert und anschließend erneut aktiviert werden.) Die Änderungen werden wirksam, sobald das nächste Mal eine Verbindung zur Datenbank hergestellt wird. Andere Parameter können online geändert werden. Diese werden als *online konfigurierbare Konfigurationsparameter* bezeichnet.

Die Spalte „Auto.“ in der folgenden Tabelle gibt an, ob der Parameter das Schlüsselwort AUTOMATIC im Befehl UPDATE DATABASE MANAGER CONFIGURATION unterstützt. Wenn Sie einen Parameter auf den Wert „AUTOMATIC“ setzen, passt DB2 den Parameter automatisch so an, dass er die aktuellen Ressourcenanforderungen widerspiegelt.

In der Spalte „Leistungsrelevanz“ (Leist.-Relev.) ist vermerkt, in welchem relativen Ausmaß sich jeder Parameter in Bezug auf die Systemleistung auswirken kann. Diese Spalte kann allerdings nicht für alle Arten von Umgebungen gültige Angaben enthalten. Sie sollten die Informationen als allgemeine Hinweise betrachten.

- **Hoch:** Gibt an, dass ein Parameter wesentliche Auswirkungen auf die Leistung haben kann. Die Werte für diese Parameter müssen sehr eingehend überlegt werden. In einigen Fällen kann das bedeuten, dass Sie die vorgegebenen Standardwerte übernehmen sollten.
- **Mittel:** Gibt an, dass der Parameter unter Umständen Auswirkungen auf die Leistung haben kann. Sie sollten von Ihrer spezifischen Umgebung und Ihren Anforderungen ausgehend entscheiden, wie viel Aufwand in die Optimierung dieser Parameter investiert werden sollte.
- **Niedrig:** Gibt an, dass der Parameter weniger allgemeine bzw. weniger bedeutende Auswirkungen auf die Leistung hat.
- **Keine:** Gibt an, dass der Parameter keine direkten Auswirkungen auf die Leistung hat. Da diese Parameter nicht leistungsrelevant sind, müssen sie nicht optimiert werden. Sie können jedoch in anderer Hinsicht für die Systemkonfiguration von Bedeutung sein, zum Beispiel zur Aktivierung der Kommunikationsunterstützung.

Die Spalten „Token“, „Tokenwert“ und „Datentyp“ stellen Informationen bereit, die Sie beim Aufrufen der API `db2CfgGet` oder `db2CfgSet` benötigen. Diese Informationen enthalten Kennungen für Konfigurationsparameter, Einträge für das Element `token` in der Datenstruktur `db2CfgParam` und Datentypen für Werte, die an die Struktur übergeben werden.

Tabelle 64. Konfigurierbare Konfigurationsparameter für die Datenbank

Parameter	Onl. Cfg.	Auto.	Leist.-Relev.	Token	Tokenwert	Datentyp	Zusätzliche Informationen
<code>alt_collate</code>	Nein	Nein	Keine	SQLF_DBTN_ALT_COLLATE	809	UInt32	„alt_collate - Alternative Sortierfolge “ auf Seite 535
<code>app_ctl_heap_sz</code>	Nein	Nein	Mittel	SQLF_DBTN_APP_CTL_HEAP_SZ	500	UInt16	„app_ctl_heap_sz - Zwischenspeichergröße für Anwendungssteuerung “ auf Seite 449
<code>appgroup_mem_sz</code>	Nein	Nein	Mittel	SQLF_DBTN_APPGROUP_MEM_SZ	800	UInt32	„appgroup_mem_sz - Maximale Speichergröße für Anwendungsgruppe “ auf Seite 450
<code>applheapsz</code>	Nein	Nein	Mittel	SQLF_DBTN_APPLHEAPSZ	51	UInt16	„applheapsz - Zwischenspeichergröße für Anwendungen “ auf Seite 454
<code>archretrydelay</code>	Ja	Nein	Keine	SQLF_DBTN_ARCHRETRYDELAY	828	UInt16	„archretrydelay - Wiederholungsintervall für Archivierung bei Fehler “ auf Seite 507
• <code>auto_maint</code> • <code>auto_db_backup</code> • <code>auto_tbl_maint</code> • <code>auto_runstats</code> • <code>auto_stats_prof</code> • <code>auto_prof_upd</code> • <code>auto_reorg</code>	Ja	Nein	Mittel	• SQLF_ENABLE_AUTO_MAINT¹ • SQLF_ENABLE_AUTO_DB_BACKUP • SQLF_ENABLE_AUTO_TBL_MAINT • SQLF_ENABLE_AUTO_RUNSTATS • SQLF_ENABLE_AUTO_STATS_PROF • SQLF_ENABLE_AUTO_PROF_UPD • SQLF_ENABLE_AUTO_REORG	• 831 • 833 • 835 • 837 • 839 • 844 • 841	UInt32	„auto_maint - Automatische Verwaltung “ auf Seite 547
<code>autorestart</code>	Ja	Nein	Niedrig	SQLF_DBTN_AUTO_RESTART	25	UInt16	„autorestart - Automatischer Neustart aktiviert “ auf Seite 518
<code>avg_appls</code>	Ja	Ja	Hoch	SQLF_DBTN_AVG_APPLS	47	UInt16	„avg_appls - Durchschnittliche Anzahl aktiver Anwendungen “ auf Seite 484
<code>blk_log_dsk_ful</code>	Ja	Nein	Keine	SQLF_DBTN_BLK_LOG_DSK_FUL	804	UInt16	„blk_log_dsk_ful - Bei voller Protokollplatte blockieren “ auf Seite 508
<code>catalogcache_sz</code>	Ja	Nein	Hoch	SQLF_DBTN_CATALOGCACHE_SZ	56	Sint32	„catalogcache_sz - Katalogcachegröße “ auf Seite 432
<code>chngpgs_thresh</code>	Nein	Nein	Hoch	SQLF_DBTN_CHNGPGS_THRESH	38	UInt16	„chngpgs_thresh - Schwellenwert für geänderte Seiten “ auf Seite 476
<code>database_memory</code>	Nein	Ja	Mittel	SQLF_DBTN_DATABASE_MEMORY	803	UInt64	„database_memory - Größe des gemeinsam genutzten Datenbankspeichers “ auf Seite 434
<code>dbheap</code>	Ja	Nein	Mittel	SQLF_DBTN_DB_HEAP	58	UInt64	„dbheap - Zwischenspeicher für Datenbank “ auf Seite 437
<code>db_mem_thresh</code>	Ja	Nein	Niedrig	SQLF_DBTN_DB_MEM_THRESH	849	UInt16	„db_mem_thresh - Schwellenwert für Datenbankspeicher “ auf Seite 436
<code>dft_degree</code>	Ja	Nein	Hoch	SQLF_DBTN_DFT_DEGREE	301	Sint32	„dft_degree - Standardgrad der Parallelität “ auf Seite 541
<code>dft_extent_sz</code>	Ja	Nein	Mittel	SQLF_DBTN_DFT_EXTENT_SZ	54	UInt32	„dft_extent_sz - Standardwert für EXTENTSIZE bei Tabellenbereichen “ auf Seite 477
<code>dft_loadrec_ses</code>	Ja	Nein	Mittel	SQLF_DBTN_DFT_LOADREC_SES	42	Sint16	„dft_loadrec_ses - Standardanzahl von Sitzungen für Recovery “ auf Seite 518
<code>dft_mttb_types</code>	Nein	Nein	Keine	SQLF_DBTN_DFT_MTTB_TYPES	843	UInt32	„dft_mttb_types - Standardtypen von verwalteten Tabellen zur Optimierung “ auf Seite 541
<code>dft_prefetch_sz</code>	Ja	Ja	Mittel	SQLF_DBTN_DFT_PREFETCH_SZ	40	Sint16	„dft_prefetch_sz - Standardwert für PREFETCHSIZE “ auf Seite 477
<code>dft_queryopt</code>	Ja	Nein	Mittel	SQLF_DBTN_DFT_QUERYOPT	57	Sint32	„dft_queryopt - Standardabfrageoptimierungsklasse “ auf Seite 542
<code>dft_refresh_age</code>	Nein	Nein	Mittel	SQLF_DBTN_DFT_REFRESH_AGE	702	char(22)	„dft_refresh_age - Standardaktualisierungsalter “ auf Seite 542

Tabelle 64. Konfigurierbare Konfigurationsparameter für die Datenbank (Forts.)

Parameter	Onl. Cfg.	Auto.	Leist.- Relev.	Token	Token- wert	Datentyp	Zusätzliche Informationen
dft_sqlmathwarn	Nein	Nein	Keine	SQLF_DBTN_DFT_SQLMATHWARN	309	Sint16	„dft_sqlmathwarn - Bei arithmetischen Ausnahmebedingungen fortsetzen “ auf Seite 543
discover_db	Ja	Nein	Mittel	SQLF_DBTN_DISCOVER	308	Uint16	„discover_db - Discovery-Unterstützung für diese Datenbank “ auf Seite 552
dlchktime	Ja	Nein	Mittel	SQLF_DBTN_DLCHKTIME	9	Uint32	„dlchktime - Zeitintervall für Prüfung von Deadlocks “ auf Seite 471
dyn_query_mgmt	Nein	Nein	Niedrig	SQLF_DBTN_DYN_QUERY_MGMT	604	Uint16	„dyn_query_mgmt - Dynamische SQL- und XQuery-Abfrageverwaltung “ auf Seite 534
failarchpath	Ja	Nein	Keine	SQLF_DBTN_FAILARCHPATH	826	char(243)	„failarchpath - Protokollarchivpfad für Funktionsübernahme “ auf Seite 509
groupheap_ratio	Nein	Nein	Mittel	SQLF_DBTN_GROUPHEAP_RATIO	801	Uint16	„groupheap_ratio - Für Anwendungsgruppenzwischenspeicher vorgesehener Speicher in Prozent “ auf Seite 451
hadr_local_host	Nein	Nein	Keine	SQLF_DBTN_HADR_LOCAL_HOST	811	char(256)	„hadr_local_host - Name des lokalen Hosts für HADR “ auf Seite 520
hadr_local_svc	Nein	Nein	Keine	SQLF_DBTN_HADR_LOCAL_SVC	812	char(41)	„hadr_local_svc - Lokaler HADR-Servicename “ auf Seite 520
hadr_remote_host	Nein	Nein	Keine	SQLF_DBTN_HADR_REMOTE_HOST	813	char(256)	„hadr_remote_host - Name des fernen Hosts für HADR “ auf Seite 521
hadr_remote_inst	Nein	Nein	Keine	SQLF_DBTN_HADR_REMOTE_INST	815	char(9)	„hadr_remote_inst - HADR-Instanzname des fernen Servers “ auf Seite 521
hadr_remote_svc	Nein	Nein	Keine	SQLF_DBTN_HADR_REMOTE_SVC	814	char(41)	„hadr_remote_svc - Ferner HADR-Servicename “ auf Seite 522
hadr_syncmode	Nein	Nein	Keine	SQLF_DBTN_HADR_SYNCMODE	817	Uint32	„hadr_syncmode - HADR-Synchronisationsmodus für Protokollierung im Peerstatus “ auf Seite 522
hadr_timeout	Nein	Nein	Keine	SQLF_DBTN_HADR_TIMEOUT	816	Sint32	„hadr_timeout - HADR-Zeitlimitwert “ auf Seite 523
indexrec ²	Ja	Nein	Mittel	SQLF_DBTN_INDEXREC	30	Uint16	„indexrec - Zeitpunkt für Indexneuerstellung “ auf Seite 523
locklist	Ja	Ja	Hoch bei Einfluss auf die Eskalation	SQLF_DBTN_LOCK_LIST	704	Uint64	„locklist - Maximaler Speicher für Sperrenliste “ auf Seite 438
locktimeout	Nein	Nein	Mittel	SQLF_DBTN_LOCKTIMEOUT	34	Sint16	„locktimeout - Zeitlimit für Sperren “ auf Seite 472
logarchmeth1	Ja	Nein	Keine	SQLF_DBTN_LOGARCHMETH1	822	Uint16	„logarchmeth1 - Primäre Protokollarchivierungsmethode “ auf Seite 509
logarchmeth2	Ja	Nein	Keine	SQLF_DBTN_LOGARCHMETH2	823	Uint16	„logarchmeth2 - Sekundäre Protokollarchivierungsmethode “ auf Seite 510
logarchopt1	Ja	Nein	Keine	SQLF_DBTN_LOGARCHOPT1	824	char(243)	„logarchopt1 - Optionen für primäres Protokollarchiv “ auf Seite 510
logarchopt2	Ja	Nein	Keine	SQLF_DBTN_LOGARCHOPT2	825	char(243)	„logarchopt2 - Optionen für sekundäres Protokollarchiv “ auf Seite 511
logbufsz	Nein	Nein	Hoch	SQLF_DBTN_LOGBUFSZ	33	Uint16	„logbufsz - Protokollpuffergröße “ auf Seite 441
logfilsiz	Nein	Nein	Mittel	SQLF_DBTN_LOGFIL_SIZ	92	Uint32	„logfilsiz - Protokolldateigröße “ auf Seite 497
logindexbuild	Ja	Ja	Keine	SQLF_DBTN_LOGINDEXBUILD	818	Uint32	„logindexbuild - Erstellte Indexseiten protokollieren “ auf Seite 511
logprimary	Nein	Nein	Mittel	SQLF_DBTN_LOGPRIMARY	16	Uint16	„logprimary - Anzahl primärer Protokolldateien “ auf Seite 499
logretain ³	Nein	Nein	Niedrig	SQLF_DBTN_LOG_RETAIN	23	Uint16	„logretain - Beibehalten von Protokollen aktivieren “ auf Seite 511
logsecond	Ja	Nein	Mittel	SQLF_DBTN_LOGSECOND	17	Uint16	„logsecond - Anzahl sekundärer Protokolldateien “ auf Seite 501

Tabelle 64. Konfigurierbare Konfigurationsparameter für die Datenbank (Forts.)

Parameter	Onl. Cfg.	Auto.	Leist.- Relev.	Token	Token- wert	Datentyp	Zusätzliche Informationen
max_log	Ja	Ja		SQLF_DBTN_MAX_LOG	807	Uint16	„max_log - Maximale Protokollgröße pro Transaktion “ auf Seite 502
maxappls	Ja	Ja	Mittel	SQLF_DBTN_MAXAPPLS	6	Uint16	„maxappls - Maximale Anzahl aktiver Anwendungen “ auf Seite 487
maxfilop	Ja	Nein	Mittel	SQLF_DBTN_MAXFILOP	3	Uint16	„maxfilop - Maximale Anzahl offener Datenbankdateien pro Anwendung “ auf Seite 490
maxlocks	Ja	Ja	Hoch bei Einfluss auf die Eskalation	SQLF_DBTN_MAXLOCKS	15	Uint16	„maxlocks - Maximale Anzahl Sperren pro Anwendung “ auf Seite 474
min_dec_div_3	Nein	Nein	Hoch	SQLF_DBTN_MIN_DEC_DIV_3	605	Sint32	„min_dec_div_3 - 3 Kommastellen bei Dezimaldivision “ auf Seite 463
mincommit	Ja	Nein	Hoch	SQLF_DBTN_MINCOMMIT	32	Uint16	„mincommit - Anzahl der Gruppencommits “ auf Seite 512
mirrorlogpath	Nein	Nein	Niedrig	SQLF_DBTN_MIRRORLOGPATH	806	char(242)	„mirrorlogpath - Pfad für Protokollspiegelung “ auf Seite 502
newlogpath	Nein	Nein	Niedrig	SQLF_DBTN_NEWLOGPATH	20	char(242)	„newlogpath - Datenbankprotokollpfad ändern “ auf Seite 504
num_db_backups	Ja	Nein	Keine	SQLF_DBTN_NUM_DB_BACKUPS	601	Uint16	„num_db_backups - Anzahl der Datenbank-Backups “ auf Seite 526
num_freqvalues	Ja	Nein	Niedrig	SQLF_DBTN_NUM_FREQVALUES	36	Uint16	„num_freqvalues - Anzahl der häufigsten Werte “ auf Seite 544
num_iocleaners	Nein	Ja	Hoch	SQLF_DBTN_NUM_IOCLEANERS	37	Uint16	„num_iocleaners - Anzahl asynchroner Seitenlöschfunktionen “ auf Seite 478
num_ioservers	Nein	Ja	Hoch	SQLF_DBTN_NUM_IOSERVERS	39	Uint16	„num_ioservers - Anzahl von E/A-Servern “ auf Seite 480
num_log_span	Ja	Ja		SQLF_DBTN_NUM_LOG_SPAN	808	Uint16	„num_log_span - Anzahl verwendeter Protokolldateien “ auf Seite 505
num_quantiles	Ja	Nein	Niedrig	SQLF_DBTN_NUM_QUANTILES	48	Uint16	„num_quantiles - Anzahl der Quantile für Spalten “ auf Seite 546
numarchretry	Ja	Nein	Keine	SQLF_DBTN_NUMARCHRETRY	827	Uint16	„numarchretry - Anzahl der Wiederholungen bei Fehler “ auf Seite 514
overflowlogpath	Nein	Nein	Mittel	SQLF_DBTN_OVERFLOWLOGPATH	805	char(242)	„overflowlogpath - Überlaufprotokollpfad “ auf Seite 506
pckcachesz	Ja	Ja	Hoch	SQLF_DBTN_PCKCACHE_SZ	505	Uint32	„pckcachesz - Größe des Paketcache “ auf Seite 442
rec_his_retentn	Nein	Nein	Keine	SQLF_DBTN_REC_HIS_RETENTN	43	Sint16	„rec_his_retentn - Aufbewahrungszeitraum für Recoveryprotokoll “ auf Seite 526
self_tuning_mem	Ja	Nein	Hoch	SQLF_DBTN_SELF_TUNING_MEM	848	Uint16	„self_tuning_mem - Speicher mit automatischer Leistungs-optimierung “ auf Seite 444
seqdetect	Ja	Nein	Hoch	SQLF_DBTN_SEQDETECT	41	Uint16	„seqdetect - Markierung für Sequenzerkennung “ auf Seite 481
sheapthres_shr	Nein	Ja	Hoch	SQLF_DBTN_SHEAPTHRES_SHR	802	Uint32	„sheapthres_shr - Schwellenwert für Sortierspeicher für gemeinsame Sortiervorgänge “ auf Seite 446
softmax	Nein	Nein	Mittel	SQLF_DBTN_SOFTMAX	5	Uint16	„softmax - Vor bedingtem Prüfpunkt zu schreibende Protokollsätze “ auf Seite 514
sortheap	Ja	Ja	Hoch	SQLF_DBTN_SORT_HEAP	52	Uint32	„sortheap - Zwischenspeichergröße für Sortierlisten “ auf Seite 458
stat_heap_sz	Nein	Nein	Niedrig	SQLF_DBTN_STAT_HEAP_SZ	45	Uint32	„stat_heap_sz - Größe des Statistikzwischenspeichers “ auf Seite 459
stmthep	Ja	Nein	Mittel	SQLF_DBTN_STMT_HEAP	821	Uint32	„stmthep - Größe des Anweisungszwischenspeichers “ auf Seite 460

Tabelle 64. Konfigurierbare Konfigurationsparameter für die Datenbank (Forts.)

Parameter	Onl. Cfg.	Auto.	Leist.-Relev.	Token	Token-wert	Datentyp	Zusätzliche Informationen
<i>trackmod</i>	Nein	Nein	Niedrig	SQLF_DBTN_TRACKMOD	703	UInt16	„trackmod - Geänderte Seiten protokollieren “ auf Seite 527
<i>tsm_mgmtclass</i>	Ja	Nein	Keine	SQLF_DBTN_TSM_MGMTCLASS	307	char(30)	„tsm_mgmtclass - Tivoli Storage Manager-Verwaltungs-klasse “ auf Seite 527
<i>tsm_nodename</i>	Ja	Nein	Keine	SQLF_DBTN_TSM_NODENAME	306	char(64)	„tsm_nodename - TSM-Knotenname “ auf Seite 528
<i>tsm_owner</i>	Ja	Nein	Keine	SQLF_DBTN_TSM_OWNER	305	char(64)	„tsm_owner - TSM-Eigenername “ auf Seite 528
<i>tsm_password</i>	Ja	Nein	Keine	SQLF_DBTN_TSM_PASSWORD	501	char(64)	„tsm_password - TSM-Kennwort “ auf Seite 529
<i>userexit</i>	Nein	Nein	Niedrig	SQLF_DBTN_USER_EXIT	24	UInt16	„userexit - Benutzerexit aktivieren “ auf Seite 516
<i>util_heap_sz</i>	Ja	Nein	Niedrig	SQLF_DBTN_UTIL_HEAP_SZ	55	UInt32	„util_heap_sz - Zwischen-speichergröße für Dienstprogramme “ auf Seite 448
<i>vendoropt</i>	Ja	Nein	Keine	SQLF_DBTN_VENDOROPT	829	char(242)	„vendoropt - Lieferantenoptionen “ auf Seite 517
Anmerkungen: 1. Default => Bit 1 on (xxxx xxxx xxxx xxx1): auto_maint Bit 2 off (xxxx xxxx xxxx xx0x): auto_db_backup Bit 3 on (xxxx xxxx xxxx x0xx): auto_tbl_maint Bit 4 on (xxxx xxxx xxxx lxxx): auto_runstats Bit 5 off (xxxx xxxx xxx1 xxxx): auto_stats_prof Bit 6 off (xxxx xxxx xx0x xxxx): auto_prof_upd Bit 7 off (xxxx xxxx x0xx xxxx): auto_reorg 0 0 1 9 Maximum => Bit 1 on (xxxx xxxx xxxx xxx1): auto_maint Bit 2 off (xxxx xxxx xxxx x1xx): auto_db_backup Bit 3 on (xxxx xxxx xxxx x1xx): auto_tbl_maint Bit 4 on (xxxx xxxx xxxx lxxx): auto_runstats Bit 5 off (xxxx xxxx xxx1 xxxx): auto_stats_prof Bit 6 off (xxxx xxxx x1xx xxxx): auto_prof_upd Bit 7 off (xxxx xxxx x1xx xxxx): auto_reorg 0 0 7 F 2. Gültige Werte (in sqlutil.h definiert) sind: SQLF_INX_REC_SYSTEM (0) SQLF_INX_REC_REFERENCE (1) SQLF_INX_REC_RESTART (2) 3. Gültige Werte (in sqlutil.h definiert) sind: SQLF_LOGRETAIN_NO (0) SQLF_LOGRETAIN_RECOVERY (1) SQLF_LOGRETAIN_CAPTURE (2)							

Tabelle 65. Informative Konfigurationsparameter für die Datenbank

Parameter	Token	Token-wert	Datentyp	Zusätzliche Informationen
<i>backup_pending</i>	SQLF_DBTN_BACKUP_PENDING	112	UInt16	„backup_pending - Backup anstehend “ auf Seite 538
<i>codepage</i>	SQLF_DBTN_CODEPAGE	101	UInt16	„codepage - Codepage für die Datenbank “ auf Seite 535
<i>codeset</i>	SQLF_DBTN_CODESET	120	char(9) ¹	„codeset - Codierter Zeichensatz für die Datenbank “ auf Seite 535
<i>collate_info</i>	SQLF_DBTN_COLLATE_INFO	44	char(260)	„collate_info - Informationen zur Sortierfolge “ auf Seite 536
<i>country/region</i>	SQLF_DBTN_COUNTRY	100	UInt16	„country/region - Datenbankgebietscode “ auf Seite 537

Tabelle 65. Informative Konfigurationsparameter für die Datenbank (Forts.)

Parameter	Token	Tokenwert	Datentyp	Zusätzliche Informationen
<i>database_consistent</i>	SQLF_DBTN_CONSISTENT	111	Uint16	„database_consistent - Datenbank ist konsistent “ auf Seite 538
<i>database_level</i>	SQLF_DBTN_DATABASE_LEVEL	124	Uint16	„database_level - Releasestand der Datenbank “ auf Seite 537
<i>hadr_db_role</i>	SQLF_DBTN_HADR_DB_ROLE	810	Uint32	„hadr_db_role - Rolle der HADR-Datenbank “ auf Seite 519
<i>log_retain_status</i>	SQLF_DBTN_LOG_RETAIN_STATUS	114	Uint16	„log_retain_status - Statusanzeiger für Beibehalten der Protokolle “ auf Seite 539
<i>loghead</i>	SQLF_DBTN_LOGHEAD	105	char(12)	„loghead - Erste aktive Protokolldatei “ auf Seite 498
<i>logpath</i>	SQLF_DBTN_LOGPATH	103	char(242)	„logpath - Pfad zu Protokolldateien “ auf Seite 498
<i>multipage_alloc</i>	SQLF_DBTN_MULTIPAGE_ALLOC	506	Uint16	„multipage_alloc - Zuordnung aus mehreren Seiten bestehender Datei aktiv “ auf Seite 539
<i>numsegs</i>	SQLF_DBTN_NUMSEGS	122	Uint16	„numsegs - Standardanzahl von SMS-Containern “ auf Seite 481
<i>pagesize</i>	SQLF_DBTN_PAGESIZE	846	Uint32	„pagesize - Standardgröße der Datenbankseite“ auf Seite 537
<i>release</i>	SQLF_DBTN_RELEASE	102	Uint16	„release - Releasestand der Datenbankkonfiguration “ auf Seite 537
<i>restore_pending</i>	SQLF_DBTN_RESTORE_PENDING	503	Uint16	„restore_pending - Restore anstehend “ auf Seite 540
<i>restrict_access</i>	SQLF_DBTN_RESTRICT_ACCESS	852	Sint32	„restrict_access - Eingeschränkter Datenbankzugriff (Konfigurationsparameter)“ auf Seite 589
<i>rollfwd_pending</i>	SQLF_DBTN_ROLLFWD_PENDING	113	Uint16	„rollfwd_pending - Aktualisierende Recovery anstehend “ auf Seite 540
<i>territory</i>	SQLF_DBTN_TERRITORY	121	char(5) ²	„territory - Datenbankgebiet “ auf Seite 538
<i>user_exit_status</i>	SQLF_DBTN_USER_EXIT_STATUS	115	Uint16	„user_exit_status - Statusanzeiger für Benutzerexit “ auf Seite 540

Tabelle 65. Informative Konfigurationsparameter für die Datenbank (Forts.)

Parameter	Token	Tokenwert	Datentyp	Zusätzliche Informationen
Anmerkungen: 1. char(17) unter HP-UX und /. 2. char(33) unter HP-UX und /.				

Übersicht über die Konfigurationsparameter für den DB2-Verwaltungsserver (DAS)

Tabelle 66. DAS-Konfigurationsparameter

Parameter	Parametertyp	Zusätzliche Informationen
<i>authentication</i>	Konfigurierbar	„authentication - DAS-Authentifizierungstyp “ auf Seite 589
<i>contact_host</i>	Online konfigurierbar	„contact_host - Speicherposition der Liste mit Ansprechpartnern “ auf Seite 590
<i>das_codepage</i>	Online konfigurierbar	„das_codepage - DAS-Codepage “ auf Seite 591
<i>das_territory</i>	Online konfigurierbar	„das_territory - DAS-Gebiet “ auf Seite 591
<i>dasadm_group</i>	Konfigurierbar	„dasadm_group - DASADM-Gruppenname “ auf Seite 591
<i>db2system</i>	Online konfigurierbar	„db2system - Name des DB2-Serversystems “ auf Seite 592
<i>discover</i>	Online konfigurierbar	„discover - DAS-Discovery-Modus “ auf Seite 593
<i>exec_exp_task</i>	Konfigurierbar	„exec_exp_task - Verfallene Tasks ausführen “ auf Seite 593
<i>jdk_64_path</i>	Online konfigurierbar	„jdk_64_path - Installationspfad für Software Developer's Kit (64 Bit) für Java auf DAS “ auf Seite 594
<i>jdk_path</i>	Online konfigurierbar	„jdk_path - Installationspfad für Software Developer's Kit für Java auf DAS “ auf Seite 595
<i>sched_enable</i>	Konfigurierbar	„sched_enable - Schedulermodus “ auf Seite 595
<i>sched_userid</i>	Informativ	„sched_userid - Benutzer-ID für Scheduler “ auf Seite 596
<i>smtp_server</i>	Online konfigurierbar	„smtp_server - SMTP-Server “ auf Seite 596
<i>toolscat_db</i>	Konfigurierbar	„toolscat_db - Toolskatalogdatenbank “ auf Seite 597
<i>toolscat_inst</i>	Konfigurierbar	„toolscat_inst - Instanz der Toolskatalogdatenbank “ auf Seite 597
<i>toolscat_schema</i>	Konfigurierbar	„toolscat_schema - Schema der Toolskatalogdatenbank “ auf Seite 598

Teil 3. Erhalten der Systemleistung

Kapitel 12. Erfassen von Statistikdaten

Katalogstatistiken

Wenn der Abfragecompiler die Abfragepläne optimiert, werden die Entscheidungen in hohem Maße von statistischen Informationen über die Größe der Tabellen, Indizes und statistischen Sichten der Datenbank beeinflusst. Das Optimierungsprogramm verwendet außerdem Informationen über die Verteilung von Daten in bestimmten Spalten von Tabellen, Indizes und statistischen Sichten, sofern diese Spalten zur Auswahl von Zeilen oder für den Join von Tabellen herangezogen werden. Das Optimierungsprogramm schätzt anhand dieser Informationen den Aufwand für alternative Zugriffspläne für jede Abfrage ab.

Über die Informationen zu Tabellengrößen und Datenverteilungen hinaus können Sie auch statistische Informationen über das Clusterverhältnis (Cluster ratio) von Indizes, die Anzahl von Blattseiten (Leaf pages) in Indizes, die Anzahl von Tabellenzeilen, die aus ihren ursprünglichen Seiten überlaufen, sowie die Anzahl gefüllter und leerer Seiten in einer Tabelle erfassen. Diese Informationen helfen Ihnen bei der Entscheidung, wann eine Reorganisation von Tabellen und Indizes durchzuführen ist.

Wenn Sie Statistikdaten für eine Tabelle in einer Umgebung mit partitionierten Datenbanken sammeln, werden nur Statistikdaten für den Teil der Tabelle gesammelt, der sich in der Datenbankpartition befindet, in der das Dienstprogramm ausgeführt wird; oder es werden nur Statistikdaten für die erste Datenbankpartition in der Datenbankpartitionsgruppe gesammelt, die die Tabelle enthält. Wenn Sie Statistikdaten für eine statistische Sicht erfassen, werden für alle Datenbankpartitionen Statistikdaten erfasst.

Wenn Sie das Dienstprogramm RUNSTATS für eine Tabelle, eine statistische Sicht oder für eine Tabelle mit zugeordneten Indizes ausführen, werden die folgenden Arten von Statistikinformationen in den Systemkatalogtabellen gespeichert:

Für eine Tabelle und einen Index:

- Die Anzahl der verwendeten Seiten
- Die Anzahl der Seiten, die Zeilen enthalten
- Die Anzahl der Zeilen, die überlaufen
- Die Anzahl von Zeilen in der Tabelle (Kardinalität)
- Für MDC-Tabellen: die Anzahl von Blöcken, die Daten enthalten
- Für partitionierte Tabellen: der Grad der Datenclusterbildung in einer einzigen Datenpartition

Für jede Spalte in der Tabelle oder der statistischen Sicht und für die erste Spalte im Indexschlüssel:

- Die Kardinalität der Spalte
- Die Durchschnittslänge der Spalte
- Der zweithöchste Wert in der Spalte
- Der zweitniedrigste Wert in der Spalte
- Die Anzahl von Nullwerten (NULL) in der Spalte

Für jede XML-Spalte werden die folgenden Statistikdaten erfasst. In jeder Zeile einer XML-Spalte wird ein XML-Dokument gespeichert. Der Knotenzähler eines

angegebenen Pfads oder eines Pfad-Wert-Paars bezieht sich auf die Anzahl an Knoten, die durch den Pfad oder das Pfad-Wert-Paar erreichbar sind. Der Dokumentzähler eines angegebenen Pfads oder eines Pfad-Wert-Paars bezieht sich auf die Anzahl an Dokumenten, die der angegebene Pfad oder das angegebene Pfad-Wert-Paar enthält.

- Die Anzahl von XML-Dokumenten mit dem Wert NULL
- Die Anzahl von XML-Dokumenten mit dem Wert ungleich NULL
- Die Anzahl einzigartiger Pfade
- Die Summe des Knotenzählers für die einzelnen einzigartigen Pfade
- Die Summe des Dokumentzählers für die einzelnen einzigartigen Pfade
- Die k Paare aus Pfad und Knotenzähler für den größten Knotenzähler
- Die k Paare aus Pfad und Dokumentzähler für den größten Dokumentzähler
- Die k Triplets aus Pfad, Wert und Knotenzähler für den größten Knotenzähler
- Die k Triplets aus Pfad, Wert und Dokumentzähler für den größten Dokumentzähler
- Für jeden einzigartigen Pfad, der zu einem Text- oder Attributwert führt, gilt Folgendes:
 - Die Anzahl der unterschiedliche Werte dieses Pfads kann folgende Werte annehmen:
 - Den größten Wert
 - Den niedrigsten Wert
 - Die Anzahl an Text- oder Attributknoten
 - Die Anzahl an Dokumenten mit den Text- oder Attributknoten

Für Gruppen von Spalten, die Sie angeben:

- Ein auf einer Zeitmarke basierender Name für die Spaltengruppe
- Die Kardinalität der Spaltengruppe

Nur für Indizes:

- Die Anzahl von Indexeinträgen (Indexkardinalität)
- Die Anzahl von Blattseiten (leaf pages)
- Die Anzahl von Indexstufen
- Der Grad der Clusterbildung der Tabellendaten in Bezug auf den Index
- Der Grad der Clusterbildung der Indexschlüssel in Bezug auf die Datenpartitionen
- Das Verhältnis von Blattseiten auf dem Datenträger in der Reihenfolge des Indexschlüssels zur Anzahl von Seiten in dem Bereich von Seiten, die durch den Index belegt werden
- Die Anzahl unterschiedlicher Werte in der Spalte des Index
- Die Anzahl unterschiedlicher Werte in den ersten zwei, drei und vier Spalten des Index
- Die Anzahl unterschiedlicher Werte in allen Spalten des Index
- Die Anzahl von Blattseiten, die sich auf dem Datenträger in der Reihenfolge des Indexschlüssels mit wenigen oder keinen dazwischen liegenden Lücken befinden
- Die Anzahl von Seiten, in denen alle Satz-IDs (RIDs) als gelöscht markiert sind
- Die Anzahl als gelöscht markierter Satz-IDs in Seiten, in denen nicht alle Satz-IDs als gelöscht markiert sind

Wenn Sie detaillierte Statistiken für einen Index anfordern, speichern Sie auch feinere Informationen über den Grad der Clusterbildung der Tabelle in Bezug auf den Index sowie die Seitenabrufschätzwerte für verschiedene Puffergrößen.

Darüber hinaus können Sie die folgenden Arten von Statistiken über Tabellen und Indizes erfassen:

- Statistiken zur Datenverteilung
 - Das Optimierungsprogramm nutzt die Statistiken über die Datenverteilung zur

Abschätzung effizienter Zugriffspläne für Tabellen und statistische Sichten, in denen Daten nicht gleichmäßig verteilt sind und Spalten eine beträchtliche Anzahl mehrfach auftretender Werte aufweisen.

- Detaillierte Statistiken zu Indizes
Das Optimierungsprogramm nutzt detaillierte Indexstatistiken zur Ermittlung, wie effizient der Zugriff auf eine Tabelle über einen Index realisiert werden kann.
- Statistiken zu Unterelementen
Das Optimierungsprogramm nutzt Unterelementstatistiken für LIKE-Vergleichselemente, insbesondere für solche, die nach einem in eine Zeichenfolge eingebetteten Muster suchen, wie zum Beispiel LIKE %disk%.

Verteilungsstatistiken werden in folgenden Fällen nicht erfasst:

- Wenn die Konfigurationsparameter *num_freqvalues* und *num_quantiles* auf den Wert null (0) gesetzt sind.
- Wenn die Verteilung von Daten bekannt ist, zum Beispiel wenn jeder Datenwert nur einmal vorkommt.
- Wenn die Spalte einen Datentyp besitzt, für den keine Statistikdaten erfasst werden. Solche Datentypen sind LONG, LOB oder strukturierte Spalten.
- Wenn es sich um Zeilentypen in untergeordneten Tabellen handelt, werden die Statistiken für NPAGES, FPAGES und OVERFLOW der Tabellenebene nicht erfasst.
- Wenn Quantilverteilungsdaten angefordert werden, jedoch nur ein Nichtnullwert in der Spalte vorhanden ist.
- Wenn es sich um erweiterte Indizes oder deklarierte temporäre Tabellen handelt.

Anmerkung: Sie können das Dienstprogramm RUNSTATS für eine deklarierte temporäre Tabelle ausführen. Die Ergebnisstatistikdaten werden jedoch nicht in den Systemkatalogen gespeichert, weil deklarierte Tabellen keine Katalogeinträge besitzen. Allerdings werden die Statistikdaten in Speicherstrukturen abgelegt, welche die Kataloginformationen für deklarierte temporäre Tabellen darstellen. In bestimmten Fällen kann eine Ausführung von RUNSTATS für diese Tabellen daher sinnvoll sein.

Zugehörige Konzepte:

- „Katalogstatistiken zu Modellierung und Fallstudien“ auf Seite 349
- „Katalogstatistiktabellen“ auf Seite 330
- „Verteilungsstatistiken“ auf Seite 335
- „Allgemeine Regeln zur manuellen Aktualisierung von Katalogstatistiken“ auf Seite 353
- „Statistische Sichten“ auf Seite 211
- „Statistiken zur Modellierung von Produktionsdatenbanken“ auf Seite 350

Zugehörige Tasks:

- „Erfassen von Katalogstatistiken“ auf Seite 322

Zugehörige Referenzen:

- „RUNSTATS command“ in *Command Reference*
- „num_freqvalues - Anzahl der häufigsten Werte “ auf Seite 544
- „num_quantiles - Anzahl der Quantile für Spalten “ auf Seite 546

Erfassen von Katalogstatistiken

Die Erfassung von Katalogstatistiken über Tabellen, Indizes und statistische Sichten dient der Bereitstellung von Informationen, die das Optimierungsprogramm zur Auswahl des besten Zugriffsplans für Abfragen verwendet.

Voraussetzungen:

Für Tabellen und Indizes müssen Sie über eine der folgenden Berechtigungen verfügen:

- SYSADM
- SYSCTRL
- SYSMANT
- DBADM
- Zugriffsrecht CONTROL für die Tabelle
- Berechtigung LOAD

Für statistische Sichten müssen Sie über eine der folgenden Berechtigungen verfügen:

- SYSADM
- SYSCTRL
- SYSMANT
- DBADM
- Zugriffsrecht CONTROL für die Tabelle

Außerdem müssen Sie über Zugriffsrechte zum Zugreifen auf Zeilen der statistischen Sicht verfügen. Sie müssen insbesondere für die einzelnen Tabellen, Sichten oder Kurznamen, auf die in der Definition der statistischen Sicht verwiesen wird, über eines der folgenden Zugriffsrechte verfügen:

- SYSADM oder DBADM
- CONTROL
- SELECT

Vorgehensweise:

Gehen Sie wie folgt vor, um Katalogstatistiken zu erfassen:

1. Stellen Sie eine Verbindung zu der Datenbank her, in der die Tabellen, Indizes und statistischen Sichten enthalten sind, für die Sie statistischen Informationen erfassen möchten.
2. Führen Sie über die DB2-Befehlszeile den Befehl RUNSTATS mit den gewünschten Optionen aus. Über diese Optionen können Sie die Statistikdaten anpassen, die für Abfragen erfasst werden, die für Tabellen, Indizes und statistische Sichten ausgeführt werden.
3. Setzen Sie im Anschluss an die Ausführung von RUNSTATS eine Anweisung COMMIT ab, um die Sperren freizugeben.
4. Führen Sie für die Pakete einen Rebind durch, die auf Tabellen, Indizes und statistische Sichten zugreifen, für die Sie statistische Informationen neu generiert haben.

Verwenden Sie die Steuerzentrale, wenn Sie eine grafische Benutzerschnittstelle zur Angabe von Optionen und zur Erfassung von Statistiken nutzen wollen.

Anmerkungen:

1.

Anmerkung: Da das Dienstprogramm RUNSTATS keine Verwendung von Kurznamen unterstützt, gehen Sie bei der Aktualisierung von Statistiken für Abfragen in föderierten Datenbanken anders vor. Wenn Abfragen auf eine föderierte Datenbank zugreifen, führen Sie RUNSTATS für die Tabellen in allen Datenbanken aus, löschen die Kurznamen, über die auf ferne Tabellen zugegriffen wird, und erstellen diese Kurznamen erneut, um die neuen Statistiken für das Optimierungsprogramm verfügbar zu machen.

2.

Anmerkung:

Wenn Sie Statistikdaten für eine Tabelle in einer Umgebung mit partitionierten Datenbanken sammeln möchten, erfasst das Dienstprogramm RUNSTATS nur Statistikdaten für Tabellen in der Datenbankpartition, in der Sie es ausführen. Die RUNSTATS-Ergebnisse aus dieser Datenbankpartition werden für die anderen Datenbankpartitionen extrapoliert. Wenn die Datenbankpartition, in der Sie das Dienstprogramm RUNSTATS ausführen, keinen Teil einer bestimmten Tabelle enthält, wird die Anforderung an die erste Datenbankpartition in der Datenbankpartitionsgruppe gesendet, die einen Teil der Tabelle enthält.

Wenn Sie Statistikdaten für eine statistische Sicht erfassen, werden für alle Datenbankpartitionen Statistikdaten erfasst, die Basistabellen enthalten, auf die die Sicht verweist.

Zugehörige Konzepte:

- „Katalogstatistiken“ auf Seite 319
- „Richtlinien für die Erfassung und Aktualisierung von Statistiken“ auf Seite 326
- „Statistische Sichten“ auf Seite 211

Zugehörige Tasks:

- „Erfassen von Verteilungsstatistiken für bestimmte Spalten“ auf Seite 324
- „Erfassen von Indexstatistiken“ auf Seite 325
- „Feststellen des Zeitpunkts zur Reorganisation von Tabellen“ auf Seite 369

Zugehörige Referenzen:

- „RUNSTATS command“ in *Command Reference*
- „RUNSTATS command using the ADMIN_CMD procedure“ in *Administrative SQL Routines and Views*

Erfassen von Verteilungsstatistiken für bestimmte Spalten

Zur Gewährleistung der Effizienz sowohl des Dienstprogramms RUNSTATS als auch der nachfolgenden Abfrageplananalyse ist es sinnvoll, Verteilungsstatistikdaten nur für die Spalten zu erfassen, die von Abfragen in Klauseln wie WHERE, GROUP BY und ähnlichen verwendet werden. Darüber hinaus kann es auch nützlich sein, Statistikdaten zur Kardinalität für kombinierte Gruppen von Spalten zu erfassen. Das Optimierungsprogramm verwendet solche Informationen zur Erkennung von Spaltenkorrelationen bei der Abschätzung der Selektivität von Abfragen, die auf die Spalten in der Gruppe verweisen.

Die Beschreibung der folgenden Schritte geht von der Beispielannahme aus, dass die Datenbank **sales** die Tabelle **customers** mit den Indizes **custidx1** und **custidx2** enthält.

Voraussetzungen:

Sie müssen eine Verbindung zu der Datenbank mit den Tabellen und Indizes herstellen und über eine der folgenden Berechtigungsstufen verfügen:

- SYSADM
- SYSCTRL
- SYSMANT
- DBADM
- Zugriffsrecht CONTROL für die Tabelle

Anmerkung: Das Dienstprogramm RUNSTATS erfasst nur Statistikdaten für Tabellen in der Datenbankpartition, in der Sie es ausführen. Die RUNSTATS-Ergebnisse aus dieser Datenbankpartition werden für die anderen Datenbankpartitionen extrapoliert. Wenn die Datenbankpartition, in der Sie das Dienstprogramm RUNSTATS ausführen, keinen Teil einer Tabelle enthält, wird die Anforderung an die erste Datenbankpartition in der Datenbankpartitionsgruppe gesendet, die diesen Teil der Tabelle enthält.

Vorgehensweise:

Gehen Sie wie folgt vor, um Statistiken für bestimmte Spalten zu erfassen:

1. Stellen Sie eine Verbindung zur Datenbank **sales** her.
2. Führen Sie in Abhängigkeit von Ihren Anforderungen einen der folgenden Befehle über die DB2-Befehlszeile aus:
 - Zur Erfassung der Verteilungsstatistik für die Spalten **zip** und **ytdtotal**:

```
RUNSTATS ON TABLE sales.customers
  WITH DISTRIBUTION ON COLUMNS (zip, ytdtotal)
```
 - Zur Erfassung der Verteilungsstatistik für die gleichen Spalten, jedoch mit Anpassung der Verteilungsstandardwerte:

```
RUNSTATS ON TABLE sales.customers
  WITH DISTRIBUTION ON
  COLUMNS (zip, ytdtotal NUM_FREQVALUES 50 NUM_QUANTILES 75)
```
 - Zur Erfassung der Verteilungsstatistik für die Spalten, die in **custidx1** und **custidx2** indexiert sind:

```
RUNSTATS ON TABLE sales.customer
  ON KEY COLUMNS
```

- Zur Erfassung von Spaltenstatistiken nur für die Spalten **zip** und **ytdtotal** in der Tabelle sowie für eine Spaltengruppe, die die Spalten **region** und **territory** umfasst:

```
RUNSTATS ON TABLE sales.customers
ON COLUMNS (zip, (region, territory), ytdtotal)
```

- Angenommen, die Statistikdaten für andere Spalten (nicht XML) wurde mit dem Befehl LOAD zusammen mit der Option STATISTICS gesammelt. Zum Ergänzen der anderen Statistikdaten (nicht XML) durch die Statistikdaten der XML-Spalte **miscinfo**:

```
RUNSTATS ON TABLE sales.customers
ON COLUMNS (miscinfo)
```

- Zur Erfassung der Spaltenstatistikdaten in der Tabelle nur für andere Spalten (nicht XML) (die Option EXCLUDING XML COLUMNS hat eine Vorrangstellung vor allen anderen Klauseln, die XML-Spalten angeben können):

```
RUNSTATS ON TABLE sales.customers
EXCLUDING XML COLUMNS
```

Sie können Verteilungsstatistiken auch über die Steuerzentrale erfassen.

Zugehörige Konzepte:

- „Katalogstatistiktabellen“ auf Seite 330
- „Spaltenkorrelation für mehrere Vergleichselemente“ auf Seite 227
- „Berechnen von Gruppierungsschlüsselkardinalitäten mit Index- und Spaltengruppenstatistikdaten“ auf Seite 204

Zugehörige Tasks:

- „Erfassen von Katalogstatistiken“ auf Seite 322
- „Erfassen von Indexstatistiken“ auf Seite 325

Erfassen von Indexstatistiken

Durch die Erfassung von Indexstatistikdaten erhält das Optimierungsprogramm die Möglichkeit, zu beurteilen, ob ein Index zur Erfüllung einer Abfrage verwendet werden sollte.

Die Beschreibung der folgenden Schritte geht von der Beispielannahme aus, dass die Datenbank **sales** die Tabelle **customers** mit den Indizes **custidx1** und **custidx2** enthält.

Voraussetzungen:

Sie müssen eine Verbindung zu der Datenbank mit den Tabellen und Indizes herstellen und über eine der folgenden Berechtigungsstufen verfügen:

- SYSADM
- SYSCTRL
- SYSMAINT
- DBADM
- Zugriffsrecht CONTROL für die Tabelle

Die Ausführung von RUNSTATS mit der Option SAMPLED DETAILED erfordert 2 MB Statistikzwischenspeicher. Ordnen Sie wegen dieses zusätzlichen Speicherbedarfs dem Datenbankkonfigurationsparameter *stat_heap_sz* weitere 488 4-KB-Sei-

ten zu. Wenn der Zwischenspeicher zu klein erscheint, gibt RUNSTATS einen Fehler zurück, bevor das Dienstprogramm versucht, Statistikdaten zu erfassen.

Vorgehensweise:

Gehen Sie wie folgt vor, um detaillierte Statistikdaten für einen Index zu erfassen:

1. Stellen Sie eine Verbindung zur Datenbank **sales** her.
2. Führen Sie in Abhängigkeit von Ihren Anforderungen einen der folgenden Befehle über die DB2-Befehlszeile aus:
 - Zur Erstellung detaillierter Statistikdaten für die beiden Indizes **custidx1** und **custidx2**:

```
RUNSTATS ON TABLE sales.customers AND DETAILED INDEXES ALL
```
 - Zur Erstellung detaillierter Statistikdaten für die beiden Indizes, jedoch mit Stichprobenwerten (Sampling) anstelle detaillierter Berechnungen für jeden einzelnen Indexeintrag:

```
RUNSTATS ON TABLE sales.customers AND SAMPLED DETAILED INDEXES ALL
```
 - Zur Erstellung detaillierter Stichprobenstatistiken für Indizes sowie zur Erstellung von Verteilungsstatistiken für die Tabelle, so dass die Statistikdaten für Indizes und Tabelle konsistent sind:

```
RUNSTATS ON TABLE sales.customers  
      WITH DISTRIBUTION ON KEY COLUMNS  
      AND SAMPLED DETAILED INDEXES ALL
```

Sie können Index- und Tabellenstatistiken auch über die Steuerzentrale erfassen.

Zugehörige Konzepte:

- „Katalogstatistiken“ auf Seite 319
- „Katalogstatistiktabellen“ auf Seite 330
- „Detaillierte Indexstatistiken“ auf Seite 338

Zugehörige Tasks:

- „Erfassen von Katalogstatistiken“ auf Seite 322

Richtlinien für die Erfassung und Aktualisierung von Statistiken

Der Befehl RUNSTATS erfasst Statistikdaten für Tabellen, Indizes und statistische Sichten, um genaue Informationen bereitzustellen, die das Optimierungsprogramm zur Auswahl von Zugriffsplänen auswerten kann.

Verwenden Sie das Dienstprogramm RUNSTATS zur Erfassung von Statistiken in folgenden Situationen:

- Wenn Daten in eine Tabelle geladen und die geeigneten Indizes erstellt wurden.
- Wenn Sie einen neuen Index für eine Tabelle erstellen. Sie müssen das Dienstprogramm RUNSTATS nur für den neuen Index ausführen, wenn die Tabelle seit der letzten Ausführung von RUNSTATS für sie nicht modifiziert wurde.
- Wenn eine Tabelle mit dem Dienstprogramm REORG reorganisiert wurde.
- Wenn die Tabelle und ihre Indizes durch Änderungen, Löschungen und Einfügungen von Daten umfangreich aktualisiert wurden. („Umfangreich“ heißt in diesem Fall, dass 10 bis 20 Prozent der Tabellen- und Indexdaten von den Änderungen betroffen sind.)
- Vor dem Binden von Anwendungsprogrammen, deren Leistung von kritischer Bedeutung ist.

- Wenn Sie aktuelle und frühere Statistiken vergleichen wollen. Wenn Sie Statistiken in regelmäßigen Abständen aktualisieren, können Sie Leistungsprobleme frühzeitig erkennen.
- Wenn die Menge der vorab gelesenen Daten (PREFETCHSIZE) geändert wird.
- Wenn das Dienstprogramm REDISTRIBUTE DATABASE PARTITION GROUP zur Umverteilung der Daten verwendet wurde.

Anmerkung: In früheren Versionen von DB2 arbeitete dieser Befehl mit dem Schlüsselwort NODEGROUP an Stelle der Schlüsselwörter DATABASE PARTITION GROUP.

- Verwenden Sie das Dienstprogramm RUNSTATS zur Erfassung von Statistiken zu XML-Spalten: Wenn RUNSTATS nur zum Sammeln von Statistikdaten für XML-Spalten verwendet wird, werden Statistikdaten für nicht-XML-Spalten, die mit LOAD oder durch eine frühere Ausführung des Dienstprogramms RUNSTATS gesammelt wurden, behalten. In dem Fall, in dem Statistikdaten für einige XML-Spalten zuvor gesammelt wurden, werden diese Statistikdaten entweder gelöscht, falls keine Statistikdaten für diese XML-Spalte über den aktuellen Befehl gesammelt werden, oder sie werden ersetzt, falls Statistikdaten für diese XML-Spalte mit Hilfe des aktuellen Befehls gesammelt werden.

Zur Verbesserung der Leistung von RUNSTATS und zur Einsparung von Platten Speicherplatz für Statistiken sollten Sie in Betracht ziehen, nur die Spalten anzugeben, für die Datenverteilungsstatistiken erfasst werden sollten.

Im Idealfall sollte für Anwendungsprogramme nach der Ausführung von RUNSTATS ein Rebind durchgeführt werden. Das Abfrageoptimierungsprogramm könnte aufgrund der neuen Statistiken einen anderen Zugriffsplan auszuwählen.

Wenn Sie nicht über genügend Zeit verfügen, alle gewünschten Statistikdaten auf einmal zu erfassen, bietet es sich an, RUNSTATS jeweils nur zur Aktualisierung einiger weniger Tabellen, Indizes oder statistischen Sichten nach dem Rotationsprinzip für eine Gruppe von Objekten auszuführen. Wenn aufgrund der Aktivitäten an den Tabellen zwischen den Zeitpunkten, zu denen Sie RUNSTATS jeweils zu einer Teilaktualisierung ausführen, Inkonsistenzen auftreten, wird während der Abfrageoptimierung eine Warnung (SQL0437W, Ursachencode 6) ausgegeben. Beispiel: Zu dieser Situation kann es kommen, wenn Sie RUNSTATS ausführen, um Statistikdaten zur Tabellenverteilung zu erfassen und RUNSTATS, nach einer Tabellenaktivität, erneut ausführen, um Indexstatistikdaten zu dieser Tabelle zu sammeln. Wenn Inkonsistenzen nach der Aktivität an der Tabelle auftreten und während der Abfrageoptimierung erkannt werden, wird die Warnung ausgegeben. Wenn dieser Fall eintritt, sollten Sie RUNSTATS erneut ausführen, um die Verteilungsstatistikdaten zu aktualisieren.

Um sicherzustellen, dass die Indexstatistikdaten mit den Tabellenstatistikdaten synchron sind, sollten Sie RUNSTATS so ausführen, dass Tabellen- und Indexstatistikdaten zu gleicher Zeit erfasst werden. Indexstatistikdaten behalten die Mehrheit der bei der letzten Ausführung von RUNSTATS erfassten Tabellen- und Spaltenstatistikdaten bei. Wenn seit der letzten Erfassung von Tabellenstatistikdaten umfangreiche Änderungen an der Tabelle vorgenommen wurden, geht durch die Erfassung nur der Indexstatistikdaten für diese Tabelle die Synchronisierung der beiden Arten von Statistikdaten auf allen Knoten verloren.

Das Aufrufen des Dienstprogramms RUNSTATS auf einem Produktionssystem kann sich negativ auf die Leistung der Auslastung im Produktionsbetrieb auswirken. Das Dienstprogramm RUNSTATS unterstützt jetzt eine Drosselungsoption, die

zur Begrenzung der Leistungsbeeinträchtigung durch die Ausführung von RUNSTATS bei hohem Aufkommen an Datenbankaktivitäten eingesetzt werden kann.

Wenn Sie Statistikdaten für eine Tabelle in einer Umgebung mit partitionierten Datenbanken sammeln möchten, erfasst das Dienstprogramm RUNSTATS nur Statistikdaten für Tabellen in der Datenbankpartition, in der Sie es ausführen. Die RUNSTATS-Ergebnisse aus dieser Datenbankpartition werden für die anderen Datenbankpartitionen extrapoliert. Wenn die Datenbankpartition, in der Sie das Dienstprogramm RUNSTATS ausführen, keinen Teil einer bestimmten Tabelle enthält, wird die Anforderung an die erste Datenbankpartition in der Datenbank-partitionsgruppe gesendet, die einen Teil der Tabelle enthält.

Wenn Sie Statistikdaten für eine statistische Sicht erfassen, wenden für alle Datenbankpartitionen Statistikdaten erfasst, die Basistabellen enthalten, auf die die Sicht verweist.

Beachten Sie die folgenden Hinweise zur Verbesserung der Effizienz von RUNSTATS und der Nützlichkeit der erfassten Statistikdaten:

- Erfassen Sie Statistikdaten nur für die Spalten, die für den Join von Tabellen bzw. in Klauseln WHERE, GROUP BY und ähnlichen von Abfragen verwendet werden. Wenn diese Spalten indexiert sind, können Sie die Spalten mit der Klausel ONLY ON KEY COLUMNS des Befehls RUNSTATS angeben.
- Passen Sie die Werte für die Parameter *num_freqvalues* und *num_quantiles* für bestimmte Tabellen und bestimmte Spalten in Tabellen an.
- Erfassen Sie detaillierte Indexstatistikdaten mit der Klausel SAMPLE DETAILED, um den Umfang der im Hintergrund für detaillierte Indexstatistiken durchgeführten Berechnungen zu verringern. Die Klausel SAMPLE DETAILED verkürzt die Zeit, die zur Erfassung von Statistikdaten benötigt wird, und liefert in den meisten Fällen eine angemessene Genauigkeit.
- Wenn Sie einen Index für eine mit Daten gefüllte Tabelle erstellen, fügen Sie die Klausel COLLECT STATISTICS hinzu, um die Statistikdaten bei der Erstellung des Index zu erfassen.
- Wenn Tabellenzeilen in größerem Umfang hinzugefügt oder gelöscht werden oder wenn Daten in Spalten, für die Statistiken erfasst werden, aktualisiert werden, sollten Sie RUNSTATS erneut ausführen, um die Statistiken zu aktualisieren.
- Da RUNSTATS nur Statistikdaten in einer einzigen Datenbankpartition sammelt, sind die Statistikdaten weniger genau, wenn die Daten nicht gleichmäßig über alle Datenbankpartitionen verteilt sind. Wenn Sie eine ungleichmäßige Datenverteilung vermuten, kann es sinnvoll sein, die Daten vor der Ausführung von RUNSTATS mit Hilfe des Befehls REDISTRIBUTE DATABASE PARTITION GROUP erneut auf Datenbankpartitionen zu verteilen.

Zugehörige Konzepte:

- „Automatische Statistikerfassung“ auf Seite 358
- „Katalogstatistiken“ auf Seite 319
- „Verwendung der Verteilungsstatistiken durch das Optimierungsprogramm“ auf Seite 343

Zugehörige Tasks:

- „Erfassen von Katalogstatistiken“ auf Seite 322
- „Erfassen von Verteilungsstatistiken für bestimmte Spalten“ auf Seite 324
- „Erfassen von Indexstatistiken“ auf Seite 325

Erfassen von Statistiken an einer Stichprobe der Tabellendaten

Tabellenstatistiken werden vom Abfrageoptimierungsprogramm bei der Auswahl des besten Zugriffsplans für eine gegebene Abfrage verwendet. Daher spielt es eine wichtige Rolle, dass Statistikdaten aktuell bleiben, um den Status einer Tabelle zu einem beliebigen Zeitpunkt adäquat wiederzugeben. Mit steigendem Aktivitätsvolumen für eine Tabelle sollte auch die Häufigkeit der Statistikerfassung erhöht werden. Mit wachsender Größe von Datenbanken wird es zunehmend wichtiger, effiziente Methoden zur Erfassung von Statistiken zu entwickeln. Zufällige Stichproben, die aus den Tabellendaten erstellt werden und über die Statistiken ermittelt werden, können die Leistung von RUNSTATS verbessern. Für E/A- oder CPU-gebundene Systeme können die Leistungsvorteile enorm sein. Je kleiner die Stichprobe, desto schneller wird RUNSTATS fertig.

Mit Version 8.2 stellt der Befehl RUNSTATS die Option TABLESAMPLE zur Erfassung von Statistiken über eine Stichprobe der Daten in der Tabelle zur Verfügung. Diese Funktion kann die Effizienz der Statistikerfassung erhöhen, da Stichproben nur eine Teilmenge der Daten verwenden. Gleichzeitig stellen Stichprobenmethoden einen hohen Grad an Genauigkeit sicher.

Es gibt zwei Möglichkeiten, die Art und Weise der Stichprobenerstellung anzugeben. Die BERNOULLI-Methode wählt die Stichprobendaten auf der Zeilenebene aus. Bei einer vollständigen Tabellensuche der Datenseite wird jede Zeile für sich geprüft und nach der Wahrscheinlichkeit P , ausgewählt, die durch den numerischen Parameter angegeben ist. Die Statistiken werden nur über diese ausgewählten Zeilen erfasst. In ähnlicher Weise erstellt die Methode SYSTEM eine Stichprobe der Daten auf Seitenebene. Das heißt, jede Seite wird mit einer Wahrscheinlichkeit $1-P/100$ ausgewählt oder zurückgewiesen.

Die Leistung für Stichproben auf Seitenebene ist hervorragend, da nur eine E/A-Operation für jede ausgewählte Seite erforderlich ist. Bei Stichproben auf Zeilenebene ist der Ein-/Ausgabeaufwand nicht geringer, da jede Tabellenseite in einer vollständigen Tabellensuche abgerufen wird. Jedoch bieten Stichproben auf Zeilenebene erhebliche Verbesserungen, selbst wenn der Umfang der E/A-Operationen nicht kleiner wird, weil die Erfassung von Statistiken sehr rechenintensiv ist.

Stichproben auf Zeilenebene bieten Stichproben auf Seitenebene gegenüber Vorteilen, wenn die Datenwerte eine hohe Clusterbildung aufweisen. Im Vergleich zur Seitenstichprobe vermittelt die Zeilenstichprobe einen genaueren Eindruck von den Daten, da sie P Prozent Zeilen von jeder Datenseite enthält. Bei der Seitenstichprobe befinden jeweils alle Zeilen der P Prozent Seiten in der Stichprobenmenge. Wenn die Zeilen nach dem Zufallsprinzip über die Tabelle verteilt sind, sind die Genauigkeiten von Statistiken über Zeilenstichproben und Seitenstichproben ähnlich.

Jede Stichprobe wird bei jeder Ausführung des Befehls RUNSTATS nach dem Zufallsprinzip neu generiert, sofern nicht die Option REPEATABLE verwendet wird. Mit der Klausel REPEATABLE wird die gleiche Stichprobe generiert wie bei der letzten Ausführung des Befehls RUNSTATS mit der Option TABLESAMPLE. Für Benutzer kann dies in solchen Fällen nützlich sein, in denen eine Generierung konsistenter Statistiken für Tabellen mit konstanten Daten wünschenswert ist.

Zugehörige Konzepte:

- „Stichprobendaten in SQL- und XQuery-Abfragen“ auf Seite 128

Zugehörige Referenzen:

- „RUNSTATS command“ in *Command Reference*

Katalogstatistiktabellen

Die folgenden Tabellen bieten Informationen zu den Systemkatalogtabellen, die Katalogstatistiken enthalten, und zeigen die RUNSTATS-Optionen, die zur Erfassung bestimmter Statistiken verwendet werden.

Tabelle 67. Tabellenstatistiken (SYSCAT.TABLES und SYSSTAT.TABLES)

Statistik	Beschreibung	RUNSTATS-Option	
		Tabelle	Indizes
FPAGES	Anzahl der von einer Tabelle verwendeten Seiten	Ja	Ja
NPAGES	Anzahl der Zeilen enthaltenden Seiten	Ja	Ja
OVERFLOW	Anzahl der Überlaufzeilen	Ja	Nein
CARD	Anzahl der Zeilen in der Tabelle (Kardinalität)	Ja	Ja (Anm. 1)
ACTIVE_BLOCKS	Für MDC-Tabellen: die Gesamtzahl belegter Blöcke	Ja	Nein
Anmerkung: 1. Wenn für die Tabelle keine Indizes erstellt wurden und Sie die Indexstatistik anfordern, wird die CARD-Statistik mit keinem neuen Wert aktualisiert. Die vorige CARD-Statistik wird beibehalten.			

Tabelle 68. Spaltenstatistiken (SYSCAT.COLUMNS und SYSSTAT.COLUMNS)

Statistik	Beschreibung	RUNSTATS-Option	
		Tabelle	Indizes
COLCARD	Anzahl der unterschiedlichen Werte der Spalte (Spaltenkardinalität)	Ja	Ja (Anm. 1)
AVGCOLLEN	Durchschnittslänge der Spalte	Ja	Ja (Anm. 1)
HIGH2KEY	Zweithöchster Wert der Spalte	Ja	Ja (Anm. 1)
LOW2KEY	Zweitniedrigster Wert der Spalte	Ja	Ja (Anm. 1)
NUMNULLS	Die Anzahl Nullwerte (NULLs) in einer Spalte	Ja	Ja (Anm. 1)
SUB_COUNT	Die durchschnittliche Anzahl von Unter-elementen	Ja	Nein (Anm. 2)
SUB_DELIM_LENGTH	Durchschnittslänge jedes Begrenzers, der ein Unter-element trennt	Ja	Nein (Anm. 2)

Tabelle 68. Spaltenstatistiken (SYSCAT.COLUMNS und SYSSTAT.COLUMNS) (Forts.)

Statistik	Beschreibung	RUNSTATS-Option	
		Tabelle	Indizes
Anmerkung: 1. Spaltenstatistiken werden für die erste Spalte im Indexschlüssel erfasst. 2. Diese Statistiken stellen Informationen zu Daten in Spalten bereit, die eine Reihe von Unterfeldern bzw. Unterelementen enthalten, die durch Leerzeichen begrenzt werden. Die Werte SUB_COUNT und SUB_DELIM_LENGTH werden nur für Einzelbyte-Zeichenfolgespalten der Typen CHAR, VARCHAR, GRAPHIC und VARGRAPHIC erfasst.			

Tabelle 69. Spaltengruppenstatistiken (SYSCAT.COLGROUPS und SYSSTAT.COLGROUPS)

Statistik	Beschreibung	RUNSTATS-Option	
		Tabelle	Indizes
COLGROUPECARD	Kardinalität der Spalten- gruppe	Ja	Nein

Anmerkung: Die Verteilungsstatistiken für Spaltengruppen in den folgenden beiden Tabellen werden nicht von RUNSTATS erfasst. Sie können nicht manuell aktualisiert werden.

Tabelle 70. Verteilungsstatistiken für Spaltengruppen (SYSCAT.COLGROUPDIST und SYSSTAT.COLGROUPDIST)

Statistik	Beschreibung	RUNSTATS-Option	
		Tabelle	Indizes
TYPE	F = Häufigkeitswert	Ja	Nein
	Q = Quantilwert		
ORDINAL	Ordinalzahl der Spalte in der Gruppe	Ja	Nein
SEQNO	Folgenummer <i>n</i> , die den <i>n</i> -ten TYPE-Wert darstellt	Ja	Nein
COLVALUE	Der Datenwert als Zeichenliteral oder Null- wert	Ja	Nein

Tabelle 71. Verteilungsstatistiken für Spaltengruppen 2 (SYSCAT.COLGROUPDISTCOUNTS und SYSSTAT.COLGROUPDISTCOUNTS)

Statistik	Beschreibung	RUNSTATS-Option	
		Tabelle	Indizes
TYPE	F = Häufigkeitswert	Ja	Nein
	Q = Quantilwert		
SEQNO	Folgenummer <i>n</i> , die den <i>n</i> -ten TYPE-Wert darstellt	Ja	Nein

Tabelle 71. Verteilungsstatistiken für Spaltengruppen 2 (SYSCAT.COLGROUPDISTCOUNTS und SYSSTAT.COLGROUPDISTCOUNTS) (Forts.)

Statistik	Beschreibung	RUNSTATS-Option	
		Tabelle	Indizes
VALCOUNT	Bei TYPE = F ist VALCOUNT die Anzahl Vorkommen von COLVALUE-Werten für die Spaltengruppe, die durch diese Folgenummer (SEQNO) angegeben ist. Bei TYPE = Q ist VALCOUNT die Anzahl Zeilen, deren Wert kleiner oder gleich COLVALUE-Werten für die Spaltengruppe mit dieser Folgenummer ist.	Ja	Nein
DISTCOUNT	Bei TYPE = Q enthält diese Spalte die Anzahl unterschiedlicher Werte, die kleiner oder gleich COLVALUE-Werten für die Spaltengruppe mit dieser Folgenummer (SEQNO) sind. Null, falls nicht verfügbar.	Ja	Nein

Tabelle 72. Indexstatistiken (SYSCAT.INDEXES und SYSSTAT.INDEXES)

Statistik	Beschreibung	RUNSTATS-Option	
		Tabelle	Indizes
NLEAF	Anzahl der Indexblattseiten (leaf pages)	Nein	Ja
NLEVELS	Anzahl der Indexstufen	Nein	Ja
CLUSTERRATIO	Grad der Clusterbildung der Tabellendaten	Nein	Ja (Anm. 2)
CLUSTERFACTOR	Feinerer Grad der Clusterbildung	Nein	Detailliert (Anm. 1,2)
DENSITY	Verhältnis (Prozentsatz) von SEQUENTIAL_ PAGES zur Anzahl der Seiten im Bereich der vom Index belegten Seiten (Anm. 3)	Nein	Ja
FIRSTKEYCARD	Anzahl der unterschiedlichen Werte in der ersten Spalte des Index	Nein	Ja
FIRST2KEYCARD	Anzahl der unterschiedlichen Werte in den ersten beiden Spalten des Index	Nein	Ja
FIRST3KEYCARD	Anzahl der unterschiedlichen Werte in den ersten drei Spalten des Index	Nein	Ja

Tabelle 72. Indexstatistiken (SYSCAT.INDEXES und SYSSTAT.INDEXES) (Forts.)

Statistik	Beschreibung	RUNSTATS-Option	
		Tabelle	Indizes
FIRST4KEYCARD	Anzahl der unterschiedlichen Werte in den ersten vier Spalten des Index	Nein	Ja
FULLKEYCARD	Anzahl der unterschiedlichen Werte in allen Spalten des Index, mit Ausnahme der Schlüsselwerte in einem Index des Typs 2, für die alle Satz-IDs (RIDs) als gelöscht markiert sind	Nein	Ja
PAGE_FETCH_PAIRS	Geschätzte Anzahl der Seitenabrufe für verschiedene Puffergrößen	Nein	Detailliert (Anm. 1,2)
AVGPARTITION_CLUSTERRATIO	Der Grad der Datenclusterbildung in einer einzigen Datenpartition.	Nein	Ja (Anm. 2)
AVGPARTITION_CLUSTERFACTOR	Feinerer Messwert des Grades der Clusterbildung in einer einzigen Datenpartition.	Nein	Detailliert (Anm. 1,2)
AVGPARTITION_PAGE_FETCH_PAIRS	Geschätzte Anzahl der Seitenabrufe für verschiedene Puffergrößen, die auf der Basis einer einzigen Datenpartition generiert wurden.	Nein	Detailliert (Anm. 1,2)
DATAPARTITION_CLUSTERFACTOR	Die Anzahl an Datenpartitionsverweisen bei einer Indexsuche.	Nein (Anm. 6)	Ja (Anm. 6)
SEQUENTIAL_PAGES	Anzahl der Blattseiten (äußersten Seiten), die auf der Platte in der durch den Indexschlüssel definierten Reihenfolge mit wenigen oder keinen großen dazwischen liegenden Lücken gespeichert sind	Nein	Ja
AVERAGE_SEQUENCE_PAGES	Durchschnittliche Anzahl von Indexseiten, auf die in Reihenfolge zugegriffen werden kann. Dies ist die Anzahl von Indexseiten, die von Vorablesefunktionen als in der richtigen Reihenfolge vorliegend erkannt werden können.	Nein	Ja
AVERAGE_RANDOM_PAGES	Durchschnittliche Anzahl von Indexseiten, auf die zwischen den sequenziellen Seitenzugriffen wahlfrei zugegriffen werden muss	Nein	Ja
AVERAGE_SEQUENCE_GAP	Lücke zwischen den Seitenfolgen	Nein	Ja
AVERAGE_SEQUENCE_FETCH_PAGES	Durchschnittliche Anzahl von Tabellenseiten, auf die in Reihenfolge zugegriffen werden kann. Dies ist die Anzahl von Tabellenseiten, die von Vorablesefunktionen als in der richtigen Reihenfolge vorliegend erkannt werden können, wenn sie versuchen, Tabellenzeilen über den Index abzurufen.	Nein	Ja (Anm. 4)

Tabelle 72. Indexstatistiken (SYSCAT.INDEXES und SYSSTAT.INDEXES) (Forts.)

Statistik	Beschreibung	RUNSTATS-Option	
		Tabelle	Indizes
AVERAGE_RANDOM_FETCH_PAGES	Durchschnittliche Anzahl von Tabellenseiten, auf die ein wahlfreier Zugriff erfolgt, zwischen sequenziellen Seitenzugriffen beim Abrufen von Tabellenzeilen über den Index	Nein	Ja (Anm. 4)
AVERAGE_SEQUENCE_FETCH_GAP	Lücke zwischen sequenziellen Zugriffen beim Abrufen von Tabellenzeilen über den Index	Nein	Ja (Anm. 4)
NUMRIDS	Anzahl von Satz-IDs (RIDs) im Index, einschließlich gelöschter Satz-IDs in Indizes des Typs 2	Nein	Ja
NUMRIDS_DELETED	Gesamtanzahl von Satz-IDs (RIDs), die im Index als gelöscht markiert sind, außer den Satz-IDs in Blattseiten, in denen alle Satz-IDs als gelöscht markiert sind	Nein	Ja
NUM_EMPTY_LEAFS	Gesamtanzahl von Blattseiten, in denen alle Satz-IDs als gelöscht markiert sind	Nein	Ja
INDCARD	Anzahl von Indexeinträgen (Indexkardinalität)	Nein	Ja
Anmerkung: 1. Detaillierte Indexstatistikdaten werden erfasst, indem die Klausel DETAILED im Befehl RUNSTATS angegeben wird. 2. CLUSTERFACTOR und PAGE_FETCH_PAIRS werden mit der Klausel DETAILED nur dann erfasst, wenn die Tabelle eine beträchtliche Größe aufweist. Wenn die Tabelle größer als ca. 25 Seiten ist, werden die Statistikdaten für die Spalten CLUSTERFACTOR und PAGE_FETCH_PAIRS gesammelt. In diesem Fall ist CLUSTERRATIO (Clusterverhältnis) -1 (d. h. wird nicht gesammelt). Wenn die Tabelle relativ klein ist, wird nur die Spalte CLUSTERRATIO vom Dienstprogramm RUNSTATS ausgefüllt, während die Spalten CLUSTERFACTOR und PAGE_FETCH_PAIRS nicht berechnet werden. Wenn die Klausel DETAILED nicht angegeben wird, werden nur die Statistikdaten für CLUSTERRATIO erfasst. 3. Diese Statistik ermittelt den Prozentsatz von Seiten in der Datei, die den Index enthalten, der zu dieser Tabelle gehört. Für eine Tabelle, die nur einen definierten Index hat, sollte DENSITY normalerweise gleich 100 sein. DENSITY wird vom Optimierungsprogramm zur Abschätzung verwendet, wie viele irrelevante Seiten von anderen Indizes durchschnittlich vielleicht gelesen werden, wenn die Indexseiten vorabgelesen würden. 4. Diese Statistiken können nicht berechnet werden, wenn sich die Tabelle in einem DMS-Tabellenbereich befindet. 5. Statistiken über Vorabsehzugriffe werden beim Laden von Indizes (LOAD INDEX) bzw. beim Erstellen von Indizes (CREATE INDEX) nicht erfasst, und zwar auch dann nicht, wenn die Funktion zum Erfassen von Statistiken beim Aufrufen des Befehls angegeben wird. Statistiken über Vorabsehzugriffe werden ebenfalls nicht erfasst, wenn der Konfigurationsparameter für die Markierung der Sequenzerkennung (seqdetect) ausgeschaltet ist. 6. Wenn die RUNSTATS-Optionen für die Tabelle auf "Nein" gesetzt sind, bedeutet dies, dass bei der Erfassung der Tabellenstatistik keine Statistikdaten erfasst werden; "Ja" für Indizes bedeutet, dass Statistikdaten erfasst werden, wenn der Befehl RUNSTATS mit den INDEXES-Optionen verwendet wird.			

Tabelle 73. Spaltenverteilungsstatistiken (SYSCAT.COLDIST und SYSSTAT.COLDIST)

Statistik	Beschreibung	RUNSTATS-Option	
		Tabelle	Indizes
DISTCOUNT	Wenn TYPE den Wert Q hat, die Anzahl unterschiedlicher Werte, die kleiner oder gleich dem Statistikwert COLVALUE sind	Verteilung (Anm. 2)	Nein
TYPE	Gibt an, ob die Zeile statistische Daten über die Häufigkeit der Werte oder über Quantilwerte enthält	Verteilung	Nein
SEQNO	Die Stelle in der Häufigkeitsrangfolge einer Folgenummer, die als Hilfe zur eindeutigen Bestimmung der Zeile in der Tabelle verwendet werden kann	Verteilung	Nein
COLVALUE	Datenwert, für den Statistikdaten zur Häufigkeit oder zu Quantilwerten erfasst werden	Verteilung	Nein
VALCOUNT	Häufigkeit, mit der der Datenwert in der Spalte auftritt, oder bei Quantilwerten die Anzahl der Werte, die kleiner oder gleich dem Datenwert (COLVALUE) sind	Verteilung	Nein
Anmerkung: 1. Verteilungsstatistikdaten über Spalten werden gesammelt, indem die Klausel WITH DISTRIBUTION im Befehl RUNSTATS angegeben wird. Beachten Sie, dass Verteilungsstatistikdaten nicht gesammelt werden, wenn das Ausmaß der Ungleichmäßigkeit der Werteverteilung in den Spaltenwerten nicht hoch genug ist. 2. DISTCOUNT wird nur für Spalten erfasst, die die erste Schlüsselspalte in einem Index bilden.			

Zugehörige Konzepte:

- „Katalogstatistiken“ auf Seite 319
- „Statistiken zur Modellierung von Produktionsdatenbanken“ auf Seite 350
- „Statistiken für benutzerdefinierte Funktionen“ auf Seite 341

Verteilungsstatistiken

Sie können zwei Arten von Statistiken über die Datenverteilung erfassen:

- Statistiken zur Häufigkeit von Datenwerten

Diese Statistiken stellen Informationen über die Spalte und den Datenwert mit der höchsten Anzahl mehrfacher Vorkommen, die nächsthöchste Anzahl mehrfach vorkommender Werte usw. bis zu der Stufe, die durch den Wert des Datenbankkonfigurationsparameters *num_freqvalues* angegeben wird, bereit. Um die Erfassung der Statistik über die Häufigkeit von Werten zu inaktivieren, müssen Sie den Parameter *num_freqvalues* auf den Wert 0 setzen.

Sie können den Wert für *num_freqvalues* auch als RUNSTATS-Optionen für jede Tabelle oder statistische Sicht und bestimmte Spalten festlegen.

- Quantilstatistiken

Diese Statistiken liefern Informationen darüber, wie Datenwerte in Relation zu anderen Werten verteilt sind. Die Statistiken der so genannten K-Quantile stellen den Wert V dar, bei oder unter dem mindestens K Werte liegen. Ein K-Quantil lässt sich durch Sortieren der Werte in aufsteigender Reihenfolge ermitteln. Der K-Quantilwert ist der Wert an der K-ten Position vom unteren Ende des Wertebereichs betrachtet.

Um die Anzahl der Abschnitte anzugeben, in die die Spaltendatenwerte gruppiert werden sollen, setzen Sie den Datenbankkonfigurationsparameter *num_quantiles* auf einen Wert zwischen 2 und 32.767. Der Standardwert ist 20. Dieser Wert stellt einen Schätzfehler durch das Optimierungsprogramm von maximal plus/minus 2,5% für ein beliebiges Vergleichselement mit einem der Operatoren „=“, „>“ oder „<“ sowie einen maximalen Fehler von plus/minus 5% für ein beliebiges BETWEEN-Vergleichselement sicher. Zur Inaktivierung der Erfassung der Quantilstatistiken setzen Sie den Parameter *num_quantiles* auf den Wert 0 oder 1.

Sie können den Parameter *num_quantiles* für jede Tabelle oder statistische Sicht und für bestimmte Spalten definieren.

Anmerkung: Wenn Sie höhere Werte für die Parameter *num_freqvalues* und *num_quantiles* angeben, werden bei der Ausführung von RUNSTATS mehr CPU-Kapazitäten und Speicher entsprechend dem Datenbankkonfigurationsparameter *stat_heap_sz* benötigt.

Wann sind Verteilungsstatistiken zu erfassen? Bei der Entscheidung, ob Verteilungsstatistikdaten für eine bestimmte Tabelle oder statistische Sicht erstellt und aktualisiert werden sollten, spielen zwei Faktoren eine ausschlaggebende Rolle:

- Ob Anwendungen mit statischen oder dynamischen SQL- oder XQuery-Anweisungen arbeiten.

Verteilungsstatistiken sind am besten für dynamische und statische Abfragen geeignet, die keine Hostvariablen verwenden. Wenn Abfragen mit Hostvariablen verwendet werden, nutzt das Optimierungsprogramm die Verteilungsstatistikdaten nur im begrenzten Umfang.

- Ob die Datenwerte in Spalten gleichmäßig verteilt sind.

Erstellen Verteilungsstatistiken, wenn mindestens in einer Spalte der Tabelle die Datenwerte höchst *ungleichmäßig* verteilt sind und diese Spalte häufig in Gleichheits- bzw. Bereichsvergleichselementen auftritt, wie zum Beispiel in folgenden Klauseln:

```
WHERE C1 = KEY;  
WHERE C1 IN (KEY1, KEY2, KEY3);  
WHERE (C1 = KEY1) OR (C1 = KEY2) OR (C1 = KEY3);  
WHERE C1 <= KEY;  
WHERE C1 BETWEEN KEY1 AND KEY2;
```

Zwei Arten ungleichmäßiger Datenverteilungen treten möglicherweise zusammen auf:

- Datenwerte können in einem oder mehreren Unterintervallen gehäuft auftreten, anstatt gleichmäßig zwischen dem höchsten und niedrigsten Datenwert verteilt zu sein. Betrachten Sie die folgende Spalte, in der die Datenwerte im Bereich (5,10) eine Häufung aufweisen:

C1
0,0
5,1
6,3
7,1
8,2
8,4
8,5
9,1
93,6
100,0

Quantilstatistiken helfen dem Optimierungsprogramm bei der Einschätzung dieser Art von Datenverteilung.

Zur Ermittlung, ob Spaltendaten nicht gleichmäßig verteilt sind, können Sie eine Abfrage wie in folgendem Beispiel ausführen:

```
SELECT C1, COUNT(*) AS OCCURRENCES
FROM T1
GROUP BY C1
ORDER BY OCCURRENCES DESC;
```

- Mehrfach auftretende Datenwerte können häufig vorkommen. Betrachten Sie eine Spalte, in der die Daten mit den folgenden Häufigkeiten verteilt sind:

Datenwert	Häufigkeit
20	5
30	10
40	10
50	25
60	25
70	20
80	5

Zur Unterstützung des Optimierungsprogramms bei der Einschätzung mehrfach auftretender Werte sollten Sie sowohl Quantilstatistiken als auch Statistiken zur Worthäufigkeit erstellen.

Wann sind nur Indexstatistiken zu erfassen

Sie können Statistiken nur für Indexdaten in folgenden Situationen erfassen:

- Seit der letzten Ausführung des Dienstprogramms RUNSTATS wurde ein neuer Index erstellt, und Sie möchten nicht erneut statistische Daten für die Tabellendaten sammeln.
- Es wurden viele Änderungen an den Daten vorgenommen, die die erste Spalte eines Index betreffen.

Welche Stufe an statistischer Genauigkeit ist anzugeben

Zur Festlegung der Genauigkeit, mit der Verteilungsstatistiken erfasst werden, definieren Sie die Datenbankkonfigurationsparameter *num_quantiles* und *num_freqvalues* mit den entsprechenden Werten. Sie können diese Parameter auch als RUNSTATS-Optionen angeben, wenn Sie Statistiken für eine Tabelle bzw. für Spalten erfassen. Je höher Sie diese Werte setzen, desto größer ist die Genauigkeit, mit der RUNSTATS Verteilungsstatistiken erstellt und aktualisiert. Allerdings erfordert eine größere Genauigkeit auch mehr Ressourcen, sowohl während der Ausführung von RUNSTATS als auch für die Speicherung in den Katalogtabellen.

Für die meisten Datenbanken empfiehlt sich ein Wert zwischen 10 und 100 für den Datenbankkonfigurationsparameter *num_freqvalues*. Im Idealfall sollten Häufigkeitsstatistikdaten so erstellt werden, dass die Häufigkeiten der verbleibenden Werte entweder einander annähernd gleich sind oder im Vergleich zu den Häufigkeiten der häufigsten Werte vernachlässigbar sind. Der Datenbankmanager erfasst unter Umständen weniger als diese Anzahl, da diese Statistikdaten nur für Datenwerte erhoben werden, die mehr als einmal auftreten. Wenn Sie nur Quantilstatistiken erfassen müssen, setzen Sie den Parameter *num_freqvalues* auf den Wert 0.

Zur Einstellung der Anzahl von Quantilen geben Sie einen Wert zwischen 20 und 50 für den Datenbankkonfigurationsparameter *num_quantiles* an. Als grobe Faustregel zur Bestimmung der Anzahl von Quantilen gilt:

- Bestimmen Sie den maximalen Fehler, der bei der Schätzung der Anzahl von Zeilen jeder Bereichsabfrage tolerierbar ist, als Prozentwert P.
- Die Anzahl der Quantile sollte ca. $100/P$ sein, wenn es sich um ein BETWEEN-Vergleichselement handelt, und $50/P$, wenn das Vergleichselement eine andere Art von Bereichsvergleichselement (<, <=, > oder >=) ist.

Zum Beispiel ergeben 25 Quantile einen maximalen Schätzfehler von 4% bei BETWEEN-Vergleichselementen und 2% bei Vergleichselementen mit ">". Allgemein sollten mindestens 10 Quantile angegeben werden. Mehr als 50 Quantile sind nur bei extrem ungleichmäßig verteilten Daten erforderlich. Wenn Sie nur Statistiken über die häufigsten Werte benötigen, setzen Sie den Parameter *num_quantiles* auf den Wert 0. Wenn Sie diesen Parameter auf den Wert „1“ setzen, werden keine Quantilstatistikdaten erfasst, da der gesamte Wertebereich in diesem Fall in nur einem Quantil liegt.

Zugehörige Konzepte:

- „Katalogstatistiken“ auf Seite 319
- „Katalogstatistiktabellen“ auf Seite 330
- „Erweiterte Beispiele zur Verwendung von Verteilungsstatistiken“ auf Seite 345
- „Verwendung der Verteilungsstatistiken durch das Optimierungsprogramm“ auf Seite 343

Zugehörige Tasks:

- „Erfassen von Katalogstatistiken“ auf Seite 322
- „Erfassen von Verteilungsstatistiken für bestimmte Spalten“ auf Seite 324

Zugehörige Referenzen:

- „num_freqvalues - Anzahl der häufigsten Werte “ auf Seite 544
- „num_quantiles - Anzahl der Quantile für Spalten “ auf Seite 546

Detaillierte Indexstatistiken

Wenn Sie RUNSTATS für Indizes mit der Klausel DETAILED ausführen, erfassen Sie statistische Informationen zu Indizes, mit deren Hilfe das Optimierungsprogramm abschätzen kann, wie viele Datenseitenabrufe für die unterschiedlichen Pufferpoolgrößen erforderlich würden. Diese Zusatzinformationen unterstützen das Optimierungsprogramm bei einer besseren Abschätzung des Aufwands für den Zugriff auf eine Tabelle über einen Index.

Anmerkung: Wenn Sie detaillierte Indexstatistiken erfassen, benötigt RUNSTATS mehr Zeit und erfordert mehr Speicher und CPU-Kapazitäten. Für

die Option SAMPLED DETAILED, bei der eine Berechnung von Daten nur für eine statistisch relevante Anzahl von Einträgen erfolgt, sind 2 MB Statistikzwischenspeicher erforderlich. Ordnen Sie wegen dieses zusätzlichen Speicherbedarfs dem Datenbankkonfigurationsparameter *stat_heap_sz* weitere 488 4-KB-Seiten zu. Wenn der Zwischenspeicher zu klein erscheint, gibt RUNSTATS einen Fehler zurück, bevor das Dienstprogramm versucht, Statistikdaten zu erfassen.

Die DETAILED-Statistiken PAGE_FETCH_PAIRS und CLUSTERFACTOR werden nur erfasst, wenn die Tabelle eine ausreichende Größe hat (ca. 25 Seiten). In diesem Fall hat CLUSTERFACTOR einen Wert zwischen 0 und 1, während CLUSTER-RATIO den Wert -1 (nicht erfasst) hat. Für Tabellen, die kleiner als 25 Seiten sind, hat CLUSTERFACTOR den Wert -1 (nicht erfasst) und CLUSTERRATIO einen Wert zwischen 0 und 100, auch wenn die Klausel DETAILED für einen Index für diese Tabelle angegeben wird.

Mit Hilfe der Klausel DETAILED werden komprimierte Informationen über die Anzahl der physischen E/A-Operationen bereitgestellt, die für den Zugriff auf die Datenseiten einer Tabelle erforderlich werden, wenn eine vollständige Indexsuche bei verschiedenen Puffergrößen ausgeführt wird. Das Dienstprogramm RUNSTATS sucht die Seiten des Index ab und modelliert dabei die verschiedenen Puffergrößen und sammelt Schätzwerte darüber, wie häufig eine Fehlseitenbedingung auftritt. Wenn zum Beispiel nur eine Pufferseite verfügbar ist, führt jede neue Seite, auf die der Index verweist, zu einer Fehlseite. In einem ungünstigeren Fall könnte jede Zeile auf eine andere Seite verweisen, was zur gleichen Anzahl von E/A-Operationen wie Zeilen in der indexierten Tabelle führt. Im entgegengesetzten Extremfall, wenn der Puffer groß genug ist, um die gesamte Tabelle (abhängig von der maximalen Puffergröße) aufzunehmen, werden alle Tabellenseiten nur einmal gelesen. Die Anzahl der physischen E/A-Operationen ist infolgedessen eine monotone, nicht steigende Funktion der Puffergröße.

Die statistischen Informationen ermöglichen außerdem feinere Schätzwerte für den Grad der Clusterbildung der Tabellenzeilen in Bezug auf die Indexreihenfolge. Je geringer die Clusterbildung der Tabellenzeilen in Relation zum Index ist, desto mehr E/A-Operationen sind für den Zugriff auf Tabellenzeilen über den Index erforderlich. Das Optimierungsprogramm zieht sowohl die Puffergröße als auch den Grad der Clusterbildung bei der Abschätzung des Aufwands für den Zugriff auf eine Tabelle über einen Index in Betracht.

Die detaillierten Indexstatistiken sollten erfasst werden, wenn Abfragen auf Spalten zugreifen, die nicht im Index enthalten sind. Darüber hinaus sollten detaillierte Indexdaten in folgenden Fällen verwendet werden:

- Die Tabelle hat mehrere Indizes ohne Clustering mit verschiedenen Graden von Clusterbildung.
- Der Grad der Clusterbildung unter den Schlüsselwerten ist nicht gleichmäßig.
- Die Werte im Index werden ungleichmäßig aktualisiert.

Eine Beurteilung dieser Bedingungen ist ohne Vorwissen bzw. ohne eine zwangsweise durchgeführte Indexsuche unter verschiedenen Puffergrößen und eine Überwachung der sich ergebenden physischen E/A-Operationen schwierig. Die Methode des geringsten Aufwands zur Feststellung, ob eine dieser Situationen auftritt, besteht wahrscheinlich darin, die DETAILED-Statistikdaten für einen Index zu erfassen, zu untersuchen und zu behalten, wenn die resultierenden Wertepaare für PAGE_FETCH_PAIRS nicht linear sind.

Zugehörige Konzepte:

- „Katalogstatistiken“ auf Seite 319
- „Katalogstatistiktabellen“ auf Seite 330

Zugehörige Tasks:

- „Erfassen von Katalogstatistiken“ auf Seite 322
- „Erfassen von Indexstatistiken“ auf Seite 325

Unterelementstatistiken

Wenn Tabellen Spalten enthalten, die durch Leerzeichen getrennte Unterfelder oder Unterelemente enthalten, und Abfragen auf diese Spalten in WHERE-Klauseln verweisen, sollten Sie Unterelementstatistiken erfassen, um die Auswahl der besten Zugriffspläne zu gewährleisten.

Nehmen Sie zum Beispiel an, eine Datenbank enthält eine Tabelle namens DOCUMENTS, deren Zeilen jeweils ein Dokument beschreiben, und nehmen Sie ferner an, DOCUMENTS enthält eine Spalte namens KEYWORDS, die eine Liste relevanter Schlüsselwörter für das Dokument enthält, die zum Abrufen von Texten verwendet werden. Die Spalte KEYWORDS könnte z. B. folgende Werte enthalten:

```
'Datenbank Simulation Analytisch Business Intelligence'  
'Simulation Modell Fruchtfliege Reproduktion Temperatur'  
'Forstwirtschaft Fichte Boden Erosion Regen'  
'Wald Temperatur Boden Niederschlag Brand'
```

In diesem Beispiel besteht jede Spalte aus 5 Unterelementen, von denen jedes ein Wort (das Schlüsselwort) ist und durch ein Leerzeichen von den anderen Wörtern getrennt ist.

Bei Abfragen, die für solche Spalten das Vergleichselement LIKE mit dem Universalplatzhalterzeichen % verwenden:

```
SELECT .... FROM DOCUMENTS WHERE KEYWORDS LIKE '%Simulation%'
```

ist es für das Optimierungsprogramm häufig von Nutzen, wenn es einige grundlegende Statistikdaten zur Unterelementstruktur der Spalte zur Verfügung hat.

Die folgenden Statistiken werden erfasst, wenn Sie das Dienstprogramm RUNSTATS mit der Klausel LIKE STATISTICS ausführen:

SUB_COUNT

Die durchschnittliche Anzahl der Unterelemente.

SUB_DELIM_LENGTH

Die durchschnittliche Länge der einzelnen Begrenzer, die Unterelemente voneinander trennen. Unter einem Begrenzer ist in diesem Zusammenhang ein Leerzeichen oder mehrere aufeinander folgende Leerzeichen zu verstehen.

Im Beispiel der Spalte KEYWORDS hat SUB_COUNT den Wert 5, und SUB_DELIM_LENGTH den Wert 1, da jeder Begrenzer ein einzelnes Leerzeichen ist.

Die Registrierdatenbankvariable DB2_LIKE_VARCHAR beeinflusst die Art und Weise, wie das Optimierungsprogramm ein Vergleichselement des folgenden Formats behandelt:

```
COLUMN LIKE '%xxxxxx'
```

(Dabei steht `xxxxxx` für eine beliebige Zeichenfolge) Das heißt, ein beliebiges LIKE-Vergleichselement, dessen Suchwert mit einem Prozentzeichen (%) beginnt. (Der Suchwert kann, muss aber nicht mit % enden.) Solche Vergleichselemente werden in der Folge als LIKE-Vergleichselemente mit Platzhalter bezeichnet. Das Optimierungsprogramm muss für alle Vergleichselemente abschätzen, wie viele Zeilen dem Vergleichselement entsprechen. Bei LIKE-Vergleichselementen mit Platzhaltern nimmt das Optimierungsprogramm an, dass die verglichene Spalte (COLUMN) eine Reihe miteinander verketteter Elemente enthält, und schätzt die Länge jedes Elements ausgehend von der Länge der Zeichenfolge ohne führende und abschließende %-Zeichen ab.

Sie können die Werte der Unterelementstatistiken durch eine Abfrage der Tabelle SYSIBM.SYSCOLUMNS untersuchen. Zum Beispiel:

```
select substr(NAME,1,16), SUB_COUNT, SUB_DELIM_LENGTH
from sysibm.syscolumns where tname = 'DOCUMENTS'
```

Anmerkung: Bei Verwendung der Klausel LIKE STATISTICS benötigt das Dienstprogramm eventuell mehr Zeit. Bei einer Tabelle mit fünf Zeichenspalten zum Beispiel kann die Ausführung von RUNSTATS zwischen 15 und 40 % länger dauern, wenn die Optionen DETAILED und DISTRIBUTION nicht verwendet werden. Wird die Option DETAILED oder DISTRIBUTION angegeben, ist der Mehraufwand prozentual u. U. geringer, auch wenn der Mehraufwand absolut betrachtet derselbe ist. Wenn Sie die Verwendung dieser Option beabsichtigen, sollten Sie zwischen diesem Mehraufwand und den Verbesserungen der Abfrageleistung abwägen.

Zugehörige Konzepte:

- „Katalogstatistiken“ auf Seite 319
- „Katalogstatistiktabellen“ auf Seite 330

Zugehörige Tasks:

- „Erfassen von Katalogstatistiken“ auf Seite 322
- „Erfassen von Verteilungsstatistiken für bestimmte Spalten“ auf Seite 324

Statistiken für benutzerdefinierte Funktionen

Zur Erstellung statistischer Informationen für benutzerdefinierte Funktionen (UDFs) editieren Sie die Katalogsicht SYSSTAT.FUNCTIONS. Wenn UDF-Statistiken verfügbar sind, können sie vom Optimierungsprogramm zur Abschätzung des Aufwands für verschiedene Zugriffspläne verwendet werden. Das Dienstprogramm RUNSTATS erfasst keine Statistiken für benutzerdefinierte Funktionen. Sind keine Statistiken verfügbar, enthalten die Statistikspalten den Wert -1, und das Optimierungsprogramm verwendet Standardwerte, die von einer einfachen benutzerdefinierten Funktion ausgehen.

Die folgende Tabelle enthält Informationen zu den Statistikspalten, für die Sie Schätzwerte angeben können, um die Leistung zu verbessern:

Tabelle 74. Funktionsstatistiken (SYSCAT.FUNCTIONS und SYSSTAT.FUNCTIONS)

Statistik	Beschreibung
IOS_PER_INVOC	Geschätzte Anzahl der Schreib-/Leseanforderungen, die jedes Mal ausgeführt werden, wenn eine Funktion ausgeführt wird
INSTS_PER_INVOC	Geschätzte Anzahl der Maschineninstruktionen, die jedes Mal ausgeführt werden, wenn eine Funktion ausgeführt wird
IOS_PER_ARGBYTE	Geschätzte Anzahl der Schreib-/Leseanforderungen, die für jedes Eingabeargumentbyte ausgeführt werden
INSTS_PER_ARGBYTES	Geschätzte Anzahl der Maschineninstruktionen, die für jedes Eingabeargumentbyte ausgeführt werden
PERCENT_ARGBYTES	Geschätzter Durchschnittsprozentwert der Eingabeargumentbyte, die von der Funktion tatsächlich verarbeitet werden
INITIAL_IOS	Geschätzte Anzahl der Schreib-/Leseanforderungen, die nur beim ersten/letzten Aufruf der Funktion ausgeführt werden
INITIAL_INSTS	Geschätzte Anzahl der Maschineninstruktionen, die nur beim ersten/letzten Aufruf der Funktion ausgeführt werden
CARDINALITY	Geschätzte Anzahl von Zeilen, die von einer Tabellenfunktion generiert werden

Betrachten Sie zum Beispiel eine benutzerdefinierte Funktion (EU_SHOE), die eine amerikanische Schuhgröße in die entsprechende europäische Schuhgröße umwandelt. (Diese beiden Schuhgrößen könnten benutzerdefinierte Datentypen (UDTs) haben.) Für diese UDF könnten Sie die Statistikspalten mit folgenden Werten versehen:

- Für INSTS_PER_INVOC könnte als Wert die geschätzte Anzahl der Maschineninstruktionen festgelegt werden, die zu folgenden Operationen erforderlich sind:
 - Aufrufen der Funktion EU_SHOE
 - Initialisieren der Ausgabezeichenfolge
 - Rückgabe des Ergebnisses
- Für INSTS_PER_ARGBYTE könnte als Wert die geschätzte Anzahl der Maschineninstruktionen festgelegt werden, die zur Umwandlung der Eingabezeichenfolge in die europäische Schuhgröße erforderlich ist.
- Der Wert für PERCENT_ARGBYTES könnte auf 100 gesetzt werden, was anzeigt, dass die gesamte Eingabezeichenfolge umzuwandeln ist.
- Die Werte für INITIAL_INSTS, IOS_PER_INVOC, IOS_PER_ARGBYTE und INITIAL_IOS könnten jeweils auf 0 gesetzt werden, da diese benutzerdefinierte Funktion lediglich Berechnungen ausführt.

PERCENT_ARGBYTES würde für eine Funktion verwendet, die nicht immer die gesamte Eingabezeichenfolge verarbeitet. Ein Beispiel wäre eine benutzerdefinierte Funktion (LOCATE), an die zwei Argumente als Eingabe übergeben werden und die die Anfangsposition des ersten Vorkommens des ersten Arguments innerhalb des zweiten Arguments als Ergebnis zurückliefert. Nehmen Sie an, dass die Länge des ersten Arguments ausreichend klein ist, um im Vergleich zum zweiten Argument kaum eine Rolle zu spielen, und im Durchschnitt 75% des zweiten Arguments zum Auffinden des ersten durchsucht werden. Aufgrund dieser Informatio-

nen sollte der Wert für PERCENT_ARGBYTES auf 75 gesetzt werden. Die obige Abschätzung eines Durchschnitts von 75% fußt dabei auf folgenden zusätzlichen Annahmen:

- In der Hälfte der Fälle wird das erste Argument gar nicht gefunden, d. h., das gesamte zweite Argument wird durchsucht.
- Die Wahrscheinlichkeit, dass das erste Argument innerhalb des zweiten Arguments auftritt, ist an allen Stellen gleich groß, d. h., in den Fällen, in denen das erste Argument überhaupt gefunden wird, muss im Durchschnitt die Hälfte des zweiten Arguments durchsucht werden.

Sie können die Spalten INITIAL_INSTS bzw. INITIAL_IOS zur Eintragung der geschätzten Anzahl von Maschineninstruktionen bzw. Schreib-/Leseanforderungen verwenden, die nur beim ersten bzw. letzten Aufruf der Funktion ausgeführt werden, um zum Beispiel den Aufwand zur Einrichtung eines Arbeitspufferbereichs zu vermerken.

Informationen über Ein-/Ausgaben und Instruktionen, die von einer benutzerdefinierten Funktion verursacht werden, erhalten Sie über die Ausgaben des Compilers für die Programmiersprache bzw. der vom Betriebssystem bereitgestellten Überwachungsprogramme.

Zugehörige Konzepte:

- „Katalogstatistiktabellen“ auf Seite 330
- „Allgemeine Regeln zur manuellen Aktualisierung von Katalogstatistiken“ auf Seite 353

Verwendung der Verteilungsstatistiken durch das Optimierungsprogramm

Das Optimierungsprogramm nutzt Verteilungsstatistiken zu besseren Abschätzungen des Aufwands für verschiedene mögliche Zugriffspläne zur Erfüllung von Abfragen.

Wenn Sie das Dienstprogramm RUNSTATS nicht mit der Klausel WITH DISTRIBUTION ausführen, enthalten die Katalogstatistiktabellen nur Informationen über die Größe der Tabelle oder der statistischen Sicht, die höchsten und niedrigsten Werte in der Tabelle oder der statistischen Sicht, den Grad der Clusterbildung der Tabelle in Bezug auf ihre Indizes und die Anzahl der unterschiedlichen Werte in den indextierten Spalten.

Sofern keine zusätzlichen Informationen über die Verteilung von Werten zwischen dem niedrigsten und dem höchsten Wert vorliegen, geht das Optimierungsprogramm davon aus, dass die Datenwerte gleichmäßig verteilt sind. Wenn die Datenwerte erhebliche Abweichungen voneinander zeigen, in bestimmten Abschnitten des Wertebereichs Häufungen aufweisen oder einzelne Werte besonders zahlreich vorkommen, wählt das Optimierungsprogramm möglicherweise einen nicht optimalen Zugriffsplan aus.

Betrachten Sie das folgende Beispiel:

Das Optimierungsprogramm muss die Anzahl von Zeilen abschätzen, die einen Spaltenwert enthalten, der ein Gleichheits- oder Bereichsvergleichselement erfüllt, um den aufwandsgünstigsten Zugriffsplan auszuwählen. Je genauer die Schätzung

ist, desto größer ist auch die Wahrscheinlichkeit, dass das Optimierungsprogramm den optimalen Zugriffsplan wählt. Betrachten Sie zum Beispiel die folgende Abfrage:

```
SELECT C1, C2
FROM TABLE1
WHERE C1 = 'NEW YORK'
AND C2 <= 10
```

Nehmen Sie an, dass es einen Index sowohl für C1 als auch für C2 gibt. Ein möglicher Zugriffsplan besteht darin, über den Index für C1 alle Zeilen mit C1 = 'NEW YORK' abzurufen und anschließend jede abgerufene Zeile daraufhin zu überprüfen, ob C2 <= 10 gilt. Ein alternativer Plan wäre, über den Index für C2 alle Zeilen mit C2 <= 10 abzurufen und anschließend jede abgerufene Zeile daraufhin zu überprüfen, ob C1 = 'NEW YORK' gilt. Da der Hauptaufwand bei der Ausführung einer Abfrage in der Regel im Zusammenhang mit dem Abrufen der Zeilen anfällt, ist der Plan der beste, der die wenigsten Abrufe erfordert. Zur Auswahl dieses Plans ist eine Abschätzung der Anzahl von Zeilen erforderlich, die das jeweilige Vergleichselement erfüllen.

Wenn keine Verteilungsstatistiken verfügbar sind, jedoch RUNSTATS für eine Tabelle oder eine statistische Sicht ausgeführt wurde, sind die einzigen Informationen, die dem Optimierungsprogramm zur Verfügung stehen, der zweithöchste Datenwert (HIGH2KEY), der zweitniedrigste Datenwert (LOW2KEY), die Anzahl unterschiedlicher Werte (COLCARD) und die Anzahl von Zeilen für eine Spalte (CARD). Die Anzahl der Zeilen, die ein Gleichheitsvergleichselement oder ein Bereichsvergleichselement erfüllen, wird dann unter der Annahme abgeschätzt, dass die Häufigkeiten der Datenwerte in einer Spalte alle gleich und die Datenwerte gleichmäßig über das Intervall (LOW2KEY, HIGH2KEY) verteilt sind. Im Einzelnen wird die Anzahl der Zeilen, die ein Gleichheitsvergleichselement C1 = KEY erfüllen, mit dem Wert CARD/COLCARD abgeschätzt. Die Anzahl der Zeilen, die ein Bereichsvergleichselement C1 BETWEEN KEY1 AND KEY2 erfüllen, wird nach folgender Formel abgeschätzt:

$$\frac{\text{KEY2} - \text{KEY1}}{\text{HIGH2KEY} - \text{LOW2KEY}} \times \text{CARD} \quad (1)$$

Diese Schätzwerte sind nur dann realistisch, wenn die tatsächliche Verteilung der Datenwerte weitgehend gleichmäßig ist. Wenn keine Verteilungsstatistikdaten verfügbar sind und entweder die Häufigkeiten der Datenwerte grob von einander abweichen oder die Datenwerte in einigen wenigen Unterbereichen des Intervalls (LOW_KEY, HIGH_KEY) Häufungen bilden, können die Schätzwerte um Größenordnungen von der Realität abweichen, so dass das Optimierungsprogramm möglicherweise einen nicht optimalen Zugriffsplan wählt.

Wenn Verteilungsstatistikdaten verfügbar sind, können die oben beschriebenen Schätzfehler wesentlich verringert werden, indem die statistischen Daten zur Häufigkeit der Werte zur Berechnung der Anzahl von Zeilen, die ein Gleichheitsvergleichselement erfüllen, und sowohl die statistischen Daten zur Häufigkeit der Werte als auch die Quantilstatistik zur Berechnung der Anzahl von Zeilen, die ein Bereichsvergleichselement erfüllen, herangezogen werden.

Zugehörige Konzepte:

- „Katalogstatistiken“ auf Seite 319
- „Verteilungsstatistiken“ auf Seite 335
- „Erweiterte Beispiele zur Verwendung von Verteilungsstatistiken“ auf Seite 345

Zugehörige Tasks:

- „Erfassen von Verteilungsstatistiken für bestimmte Spalten“ auf Seite 324

Erweiterte Beispiele zur Verwendung von Verteilungsstatistiken

Zur Erläuterung, wie das Optimierungsprogramm Verteilungsstatistiken nutzen kann, soll zunächst eine Abfrage betrachtet werden, die ein Gleichheitsvergleichselement der Form $C1 = KEY$ enthält.

Beispiel für Statistiken über die Worthäufigkeit

Wenn Statistiken zur Worthäufigkeit verfügbar sind, kann das Optimierungsprogramm diese wie folgt zur Auswahl eines geeigneten Zugriffsplans verwenden:

- Wenn KEY einer der N häufigsten Werte ist, dann verwendet das Optimierungsprogramm die Häufigkeit von KEY, die im Katalog gespeichert ist.
- Wenn KEY nicht einer der N häufigsten Werte ist, schätzt das Optimierungsprogramm die Anzahl der Zeilen, die das Vergleichselement erfüllen, unter der Annahme ab, dass die nicht häufigen Werte ($COLCARD - N$) eine gleichmäßige Verteilung aufweisen. Das heißt, die Anzahl der Zeilen wird folgendermaßen abgeschätzt:

$$\frac{CARD - NUM\_FREQ\_ROWS}{COLCARD - N} \quad (2)$$

Dabei ist CARD die Anzahl von Zeilen in der Tabelle, COLCARD ist die Kardinalität der Spalte und COLCARD ist die Gesamtanzahl von Zeilen mit einem Wert, der gleich einem der N häufigsten Werte ist.

Betrachten Sie zum Beispiel eine Spalte (C1), für die sich die Häufigkeit der Datenwerte folgendermaßen darstellt:

Datenwert	Häufigkeit
1	2
2	3
3	40
4	4
5	1

Wenn Häufigkeitsstatistiken nur für die häufigsten Werte (d. h. $N = 1$) dieser Spalte verfügbar sind, ist die Anzahl der Zeilen in der Tabelle gleich 50 und die Spaltenkardinalität gleich 5. Das Vergleichselement $C1 = 3$ wird exakt von 40 Zeilen erfüllt. Wenn das Optimierungsprogramm annimmt, dass die Datenwerte gleichmäßig verteilt sind, schätzt es die Anzahl der Zeilen, die das Vergleichselement erfüllen mit $50/5 = 10$ ab, was einen Schätzfehler von -75% bedeutet. Wenn das Optimierungsprogramm auf Häufigkeitsstatistiken zurückgreifen kann, wird die Anzahl der Zeilen mit 40 abgeschätzt und es entsteht in diesem Fall kein Fehler.

Betrachten Sie ein anderes Beispiel, in dem zwei Zeilen das Vergleichselement $C1 = 1$ erfüllen. Ohne Häufigkeitsstatistiken wird die Anzahl der Zeilen, die das Vergleichselement erfüllen, auf 10 geschätzt. Mithin entsteht ein Fehler von 400 %. Die folgende Formel kann zur Berechnung des Schätzfehlers (in Prozent) herangezogen werden:

$$\frac{\text{geschätzte Zeilen} - \text{tatsächliche Zeilen}}{\text{tatsächliche Zeilen}} \times 100$$

Bei Verwendung der Häufigkeitsstatistik ($N = 1$) schätzt das Optimierungsprogramm die Anzahl der Zeilen, die diesen Wert enthalten, anhand der oben gezeigten Formel (2). Zum Beispiel:

$$\frac{(50 - 40)}{(5 - 1)} = 3$$

Der Fehler wird dabei um eine Größenordnung verringert, wie folgende Berechnung zeigt:

$$\frac{3 - 2}{2} = 50\%$$

Beispiel für Quantilstatistiken

In den folgenden Erläuterungen der Quantilstatistiken wird der Begriff „K-Quantile“ verwendet. Das *K-Quantil* für eine Spalte ist der kleinste Datenwert V , so dass mindestens „K“ Zeilen Datenwerte enthalten, die kleiner oder gleich V sind. Ein K-Quantil lässt sich berechnen, indem die Zeilen in der Spalte nach aufsteigenden Datenwerten sortiert werden. Das K-Quantil ist der Datenwert in der K-ten Zeile der sortierten Spalte.

Wenn Quantilstatistiken verfügbar sind, kann das Optimierungsprogramm die Anzahl von Zeilen besser abschätzen, die ein Bereichsvergleichselement erfüllen, wie in folgenden Beispielen veranschaulicht wird. Betrachten Sie eine Spalte (C), die folgende Werte enthält:

C
0,0
5,1
6,3
7,1
8,2
8,4
8,5
9,1
93,6
100,0

Nehmen Sie an, dass K-Quantile für $K = 1, 4, 7$ und 10 wie folgt zur Verfügung stehen:

K	K-Quantil
1	0,0
4	7,1
7	8,5
10	100,0

Betrachten Sie zunächst das Vergleichselement $C \leq 8,5$. Für die oben angegebenen Daten erfüllen exakt sieben Zeilen dieses Vergleichselement. Unter der Annahme einer gleichmäßigen Datenverteilung und unter Verwendung der oben angegebenen

Formel (1), wobei KEY1 durch LOW2KEY ersetzt wird, wird die Anzahl der Zeilen, die das Vergleichselement erfüllen, folgendermaßen abgeschätzt:

$$\frac{8,5 - 5,1}{93,6 - 5,1} \times 10 \approx 0$$

Die Notation $\approx$ bedeutet „annähernd gleich“. Der Fehler bei dieser Schätzung ist annähernd -100%.

Wenn Quantilstatistiken verfügbar sind, schätzt das Optimierungsprogramm die Anzahl der Zeilen, die das gleiche Vergleichselement ($C \leq 8,5$) erfüllen ab, indem es 8,5 als höchsten Wert eines der K-Quantile feststellt und die Anzahl der Zeilen ermittelt, indem es den entsprechenden Wert von K, also 7, verwendet. In diesem Fall reduziert sich der Fehler auf 0.

Betrachten Sie nun das Vergleichselement $C \leq 10$. Dieses Vergleichselement wird von exakt acht Zeilen erfüllt. Wenn das Optimierungsprogramm von einer gleichmäßigen Datenwertverteilung ausgehen und Formel (1) verwenden muss, schätzt es die Anzahl von Zeilen, die das Vergleichselement erfüllen, auf 1, was einen Fehler von -87,5% ergibt.

Im Unterschied zum vorigen Beispiel ist der Wert 10 keiner der gespeicherten K-Quantile. Allerdings kann das Optimierungsprogramm mit Hilfe von Quantilen die Anzahl von Zeilen, die das Vergleichselement erfüllen, als $r_1 + r_2$ abschätzen, wobei r_1 die Anzahl von Zeilen ist, die das Vergleichselement $C \leq 8,5$ erfüllen, und r_2 die Anzahl von Zeilen ist, die das Vergleichselement $C > 8,5$ AND $C \leq 10$ erfüllen. Wie im vorigen Beispiel gilt $r_1 = 7$. Zur Abschätzung von r_2 verwendet das Optimierungsprogramm die lineare Interpolation:

$$\begin{aligned} r_2 &\approx \frac{10 - 8,5}{100 - 8,5} \times (\text{Anzahl Zeilen mit Wert} > 8,5 \text{ und} \leq 100,0) \\ r_2 &\approx \frac{10 - 8,5}{100 - 8,5} \times (10 - 7) \\ r_2 &\approx \frac{1,5}{91,5} \times (3) \\ r_2 &\approx 0 \end{aligned}$$

Die abschließende Schätzung ist $r_1 + r_2 \approx 7$, und der Fehler beträgt nur -12,5 %.

Quantile verbessern die Schätzgenauigkeit in den obigen Beispielen deswegen, weil die realen Datenwerte „Häufungen“ im Bereich von 5 bis 10 bilden, aber die Standardformeln zur Schätzung von einer gleichmäßigen Verteilung der Werte zwischen 0 und 100 ausgehen. Die Verwendung von Quantilen erhöht auch die Genauigkeit, wenn es wesentliche Unterschiede in den Häufigkeiten verschiedener Datenwerte gibt. Betrachten Sie eine Spalte, die Datenwerte mit den folgenden Häufigkeiten enthält:

Datenwert	Häufigkeit
20	5
30	5
40	15
50	50
60	15
70	5
80	5

Angenommen, es stehen K-Quantile zur Verfügung für K = 5, 25, 75, 95 und 100:

K	K-Quantil
5	20
25	40
75	50
95	70
100	80

Nehmen Sie außerdem an, dass statistische Häufigkeitsdaten für die 3 häufigsten Werte verfügbar sind.

Betrachten Sie das Vergleichselement C BETWEEN 20 AND 30. An der Verteilung der Datenwerte können Sie sehen, dass genau 10 Zeilen das Vergleichselement erfüllen. Unter der Annahme einer gleichmäßigen Datenverteilung und unter Verwendung der Formel (1) wird die Anzahl der Zeilen, die das Vergleichselement erfüllen, wie folgt abgeschätzt:

$$\frac{30 - 20}{70 - 30} \times 100 = 25$$

Diese Abschätzung enthält einen Fehler von 150%.

Unter Verwendung der Häufigkeitsstatistik und der Quantile wird die Anzahl der Zeilen, die das Vergleichselement erfüllen, als $r_1 + r_2$ abgeschätzt, wobei r_1 die Anzahl der Zeilen ist, die das Vergleichselement C = 20 erfüllen, und r_2 die Anzahl der Zeilen ist, die das Vergleichselement C > 20 AND C <= 30 erfüllen. Bei Verwendung der Formel (2) wird r_1 folgendermaßen abgeschätzt:

$$\frac{100 - 80}{7 - 3} = 5$$

r_2 wird mit linearer Interpolation folgendermaßen abgeschätzt:

$$\begin{aligned} & \frac{30 - 20}{40 - 20} \times (\text{Anzahl Zeilen mit Wert} > 20 \text{ und} \leq 40) \\ & \frac{30 - 20}{40 - 20} \times (25 - 5) \\ & = 10, \end{aligned}$$

Als endgültiger Schätzwert ergibt sich 15, wodurch der Schätzfehler um den Faktor 3 verringert wird.

Zugehörige Konzepte:

- „Katalogstatistiken“ auf Seite 319
- „Verteilungsstatistiken“ auf Seite 335
- „Verwendung der Verteilungsstatistiken durch das Optimierungsprogramm“ auf Seite 343
- „Regeln zur manuellen Aktualisierung von Verteilungsstatistiken“ auf Seite 355

Zugehörige Tasks:

- „Erfassen von Verteilungsstatistiken für bestimmte Spalten“ auf Seite 324

Katalogstatistiken zu Modellierung und Fallstudien

Sie können die statistischen Informationen in den Systemkatalogen abweichend vom tatsächlichen Status von Tabellen und Indizes ändern, um verschiedene mögliche Änderungen an der Datenbank zu Planungszwecken zu untersuchen. Aufgrund dieser Aktualisierbarkeit ausgewählter Systemkatalogstatistiken haben Sie folgende Möglichkeiten:

- Sie können die Abfrageleistung auf einem Entwicklungssystem unter Verwendung wirklichkeitsnaher Systemstatistiken eines geschäftlich genutzten Systems nachmodellieren.
- Sie können Fallstudien („Was wäre, wenn?“) für die Abfrageleistung durchführen und die Ergebnisse analysieren.

Nehmen Sie keine manuellen Aktualisierungen an den Statistiken eines Produktionssystems vor. Anderenfalls wählt das Optimierungsprogramm möglicherweise nicht den besten Zugriffsplan für Abfragen in der Produktionsumgebung aus, die dynamische SQL- oder XQuery-Anweisungen enthalten.

Voraussetzungen

Sie müssen über eine explizite DBADM-Berechtigung für die Datenbank verfügen, um Statistiken für Tabellen und Indizes und ihre Komponenten modifizieren zu können. Das heißt, Ihre Benutzer-ID muss in der Tabelle SYSCAT.DBAUTH mit der Berechtigung DBADM eingetragen sein. Durch die Zugehörigkeit zu einer DBADM-Gruppe wird diese Berechtigung nicht explizit erteilt. Ein DBADM-Benutzer kann die Statistikzeilen für alle Benutzer abrufen und kann SQL-Anweisungen UPDATE an den für das Schema SYSSTAT definierten Sichten ausführen, um die Werte dieser Statistikspalten zu aktualisieren.

Ein Benutzer ohne DBADM-Berechtigung kann nur die Zeilen sehen, die statistische Daten für Objekte enthalten, für die er das Zugriffsrecht CONTROL hat. Wenn Sie keine DBADM-Berechtigung besitzen, können Sie die Statistiken für einzelne Datenbankobjekte ändern, wenn Sie die folgenden Zugriffsrechte für jedes Objekt haben:

- Explizites Zugriffsrecht CONTROL für Tabellen. Sie können auch die Statistiken für Spalten und Indizes dieser Tabellen aktualisieren.
- Explizites Zugriffsrecht CONTROL für Kurznamen in einem System föderierter Datenbanken. Sie können auch Statistiken für Spalten und Indizes für diese Kurznamen aktualisieren. Beachten Sie, dass die Aktualisierung nur lokale Metadaten betrifft (Tabellenstatistiken der Datenquelle werden nicht geändert). Diese Aktualisierungen betreffen nur die globale Zugriffsstrategie, die vom DB2-Optimierungsprogramm generiert wird.
- Eigentumsrecht für benutzerdefinierte Funktionen (UDFs).

Das folgende Beispiel zeigt, wie die Tabellenstatistik für die Tabelle EMPLOYEE aktualisiert werden kann:

```
UPDATE SYSSTAT.TABLES
SET CARD    = 10000,
    NPAGES  = 1000,
    FPAGES  = 1000,
    OVERFLOW = 2
WHERE TABSCHEMA = 'benutzer-id'
AND TABNAME    = 'EMPLOYEE'
```

Bei der manuellen Aktualisierung von Katalogstatistiken ist Vorsicht geboten. Willkürliche Änderungen können die Leistung nachfolgender Abfragen ernsthaft beein-

flussen. Auch in einer nicht in der Produktion eingesetzten Datenbank, die Sie zum Testen oder Modellieren verwenden, können Sie jede der folgenden Methoden anwenden, um Aktualisierungen, die Sie auf diese Tabellen angewendet haben, zu erneuern und die Statistiken in einen konsistenten Status zu bringen:

- Rollback der UOW (Unit of Work, Arbeitseinheit), in der Änderungen durchgeführt wurden (unter der Annahme, dass die UOW noch nicht mit COMMIT festgeschrieben wurde).
- Ausführen des Dienstprogramms RUNSTATS, um die Katalogstatistiken neu berechnen und aktualisieren zu lassen.
- Aktualisieren der Katalogstatistiken, um anzugeben, dass keine Statistikdaten gesammelt wurden. (Zum Beispiel wird durch den Wert -1 in der Spalte NPA-GES angezeigt, dass keine Statistik zur Anzahl von Seiten erfasst wurde.)
- Ersetzen der Katalogstatistiken durch die Daten, die sie enthielten, bevor Änderungen durchgeführt wurden. Diese Methode ist nur möglich, wenn Sie das Tool *db2look* zum Erfassen der Statistiken vor Ihren Änderungen ausgeführt haben.

In einigen Fällen kann es vorkommen, dass das Optimierungsprogramm einen bestimmten statistischen Wert oder eine Kombination von Werten als ungültig erkennt. In diesem Fall verwendet es die Standardwerte und gibt eine Warnung aus. Situationen dieser Art sind jedoch selten, da der Hauptteil der Gültigkeitsprüfungen bei der Aktualisierung der Statistiken durchgeführt wird.

Zugehörige Konzepte:

- „Katalogstatistiken“ auf Seite 319
- „Katalogstatistiktabellen“ auf Seite 330
- „Statistiken zur Modellierung von Produktionsdatenbanken“ auf Seite 350
- „Statistiken für benutzerdefinierte Funktionen“ auf Seite 341

Statistiken zur Modellierung von Produktionsdatenbanken

Manchmal ist es wünschenswert, auf einem Testsystem einen Teil der Daten eines tatsächlich geschäftlich genutzten Systems nachzubilden. Jedoch sind die Zugriffspläne, die auf einem solchen Testsystem ausgewählt werden, nicht unbedingt dieselben wie die, die auf dem tatsächlich genutzten System gewählt würden, sofern nicht die Katalogstatistiken und die Konfigurationsparameter auf dem Testsystem so aktualisiert werden, dass sie mit denen auf dem Produktionssystem übereinstimmen.

Das Tool *db2look* kann für die Produktionsdatenbank ausgeführt werden, um die Aktualisierungsanweisungen (UPDATE-Anweisungen) zu generieren, die erforderlich sind, um die Katalogstatistiken der Testdatenbank in Übereinstimmung mit denen der Produktionsdatenbank zu bringen. Diese Aktualisierungsanweisungen können mit Hilfe des Programms *db2look* mit der Option *-m* (mimic-Modus) generiert werden. In diesem Fall generiert das Tool *db2look* ein Befehlsprozessorscript, das alle Anweisungen enthält, die erforderlich sind, um die Katalogstatistiken der Produktionsdatenbank nachzubilden. Dies kann bei der Analyse von SQL- oder XQuery-Anweisungen mit Hilfe von Visual Explain in einer Testumgebung nützlich sein.

Sie können Datenbankdatenobjekte einschließlich Tabellen, Sichten, Indizes und andere Objekte in einer Datenbank wieder erstellen, indem Sie die DDL-Anweisungen mit dem Befehl *db2look -e* extrahieren. Sie können das Befehlsprozessorscript, das mit diesem Befehl erstellt wurde, in einer anderen Datenbank ausführen, um

die Datenbank neu zu erstellen. Sie können die Option `-e` und die Option `-m` zusammen in einem Script verwenden, das die Datenbank neu erstellt und die Statistikdaten festlegt.

Nach der Ausführung der von `db2look` erstellten Aktualisierungsanweisungen im Testsystem kann das Testsystem zur Prüfung der Zugriffspläne verwendet werden, die in der Produktionsdatenbank generiert werden sollen. Da das Optimierungsprogramm die Art und Konfiguration der Tabellenbereiche zur Abschätzung der E/A-Aufwände verwendet, muss das Testsystem über dieselbe Tabellenbereichsanordnung bzw. -konfiguration verfügen. Das heißt, sie muss dieselbe Anzahl von Containern desselben Typs, SMS oder DMS, haben.

Das Tool *db2look* befindet sich im Unterverzeichnis *bin*.

Weitere Informationen zur Verwendung dieses Tools erhalten Sie, wenn Sie folgenden Befehl in eine Befehlszeile eingeben: `db2look -h`

Darüber hinaus stellt die Steuerzentrale eine Schnittstelle zum Dienstprogramm *db2look* mit dem Namen „DDL generieren“ bereit. Die Steuerzentrale ermöglicht eine Integration der Ergebnisdatei aus dem Dienstprogramm in die Scriptzentrale. Sie können außerdem den Befehl *db2look* über die Steuerzentrale zeitlich terminieren. Ein Unterschied bei der Verwendung der Steuerzentrale besteht darin, dass eine Analyse nur einer Tabelle durchgeführt werden kann, während in einem Aufruf des Befehls *db2look* bis zu 30 Tabellen analysiert werden können. Beachten Sie darüber hinaus, dass LaTeX- und Graphical-Ausgaben über die Steuerzentrale nicht unterstützt werden.

Sie können das Dienstprogramm `db2look` auch für eine OS/390- oder z/OS-Datenbank ausführen. Das Dienstprogramm `db2look` extrahiert die DDL- und UPDATE-Anweisungen für die Statistiken von OS/390-Objekten. Diese Funktion ist sehr nützlich, wenn Sie OS/390- oder z/OS-Objekte extrahieren und in DB2 Database für Linux, UNIX und Windows neu erstellen möchten.

Es gibt einige Unterschiede zwischen den Statistikdaten für DB2 Database für Linux, UNIX und Windows und den Statistikdaten für OS/390. Das Dienstprogramm `db2look` führt die entsprechenden Umsetzungen von DB2 für OS/390 oder z/OS in DB2 Database für Linux, UNIX und Windows aus, wenn dies zutreffend ist, und setzt die Statistikdaten von DB2 Database für Linux, UNIX und Windows, für die in DB2 für OS/390 keine Entsprechung vorhanden ist, auf den Standardwert (-1). Im Folgenden wird beschrieben, wie das Dienstprogramm `db2look` die Statistikdaten von DB2 für OS/390 oder z/OS den Statistikdaten von DB2 Database für Linux, UNIX und Windows zuordnet. In den nachfolgenden Erläuterungen steht jeweils „UDB_x“ für eine Statistikdatenspalte von DB2 Database für Linux, UNIX und Windows; „S390_x“ steht für eine Statistikdatenspalte von DB2 für OS/390 oder z/OS.

1. Statistikdaten auf Tabellenebene.

UDB_CARD = S390_CARDF
UDB_NPAGES = S390_NPAGES

Es gibt keinen Parameter `S390_FPAGES`. DB2 für OS/390 oder z/OS verfügt jedoch über einen anderen Parameter für Statistikdaten mit dem Namen `PCTPAGES`, der den Prozentsatz von aktiven Tabellenbereichsseiten darstellt, die Zeilen der Tabelle enthalten. Daher kann der Wert des Parameters `UDB_FPAGES` auf der Basis von `S390_NPAGES` und `S390_PCTPAGES` wie folgt berechnet werden:

UDB_FPAGES=(S390_NPAGES * 100)/S390_PCTPAGES

Es gibt keinen Parameter S390_OVERFLOW, der UDB_OVERFLOW zugeordnet werden kann. Daher wird dieser Wert vom Dienstprogramm db2look einfach auf den Standardwert gesetzt:

UDB_OVERFLOW=-1

2. Statistikdaten auf Spaltenebene.

UDB_COLCARD = S390_COLCARDF

UDB_HIGH2KEY = S390_HIGH2KEY

UDB_LOW2KEY = S390_LOW2KEY

Es gibt keinen Parameter S390_AVGCOLLEN, der UDB_AVGCOLLEN zugeordnet werden kann. Daher wird dieser Wert vom Dienstprogramm db2look einfach auf den Standardwert gesetzt:

UDB_AVGCOLLEN=-1

3. Statistikdaten auf Indexebene.

UDB_NLEAF = S390_NLEAF

UDB_NLEVELS = S390_NLEVELS

UDB_FIRSTKEYCARD= S390_FIRSTKEYCARD

UDB_FULLKEYCARD = S390_FULLKEYCARD

UDB_CLUSTERRATIO= S390_CLUSTERRATIO

Die anderen Statistikdaten, für die es keine OS/390- oder z/OS-Entsprechungen gibt, werden einfach auf den Standardwert gesetzt. Dies heißt Folgendes:

UDB_FIRST2KEYCARD = -1

UDB_FIRST3KEYCARD = -1

UDB_FIRST4KEYCARD = -1

UDB_CLUSTERFACTOR = -1

UDB_SEQUENTIAL_PAGES = -1

UDB_DENSITY = -1

4. Spaltenverteilungsstatistiken.

In der Tabelle SYSIBM.SYSCOLUMNS unter DB2 für OS/390 oder z/OS gibt es zwei Typen von Statistikdaten. Der Typ „F“ für häufige (Frequent) Werte und der Typ „C“ für Kardinalität (Cardinality). Nur Einträge des Typs „F“ gelten für DB2 Database für Linux, UNIX und Windows. Dies sind auch die Einträge, die berücksichtigt werden.

UDB_COLVALUE = S390_COLVALUE

UDB_VALCOUNT = S390_FrequencyF * S390_CARD

Außerdem gibt es in der Tabelle SYSIBM.SYSCOLUMNS unter DB2 für OS/390 keine Spalte SEQNO. Da eine solche Spalte für DB2 für UDB erforderlich ist, generiert das Dienstprogramm db2look eine solche Spalte automatisch.

Zugehörige Konzepte:

- „Katalogstatistiken“ auf Seite 319
- „Katalogstatistiken zu Modellierung und Fallstudien“ auf Seite 349
- „Katalogstatistiktabellen“ auf Seite 330
- „Allgemeine Regeln zur manuellen Aktualisierung von Katalogstatistiken“ auf Seite 353

Zugehörige Referenzen:

- „db2look - DB2 statistics and DDL extraction tool command“ in *Command Reference*

Manuelles Aktualisieren von Katalogstatistiken

Allgemeine Regeln zur manuellen Aktualisierung von Katalogstatistiken

Die wichtigste allgemeine Regel, die bei einer Aktualisierung der Katalogstatistik zu beachten ist, ist die, dass gültige Werte, Wertebereiche und Formate der verschiedenen Statistikdaten in den Sichten für die Statistikdaten gespeichert werden. Darüber hinaus muss die Konsistenz der Beziehungen zwischen verschiedenen Statistiken gewahrt bleiben.

Zum Beispiel muss der Wert für COLCARD in der Sicht SYSSTAT.COLUMNS kleiner sein als der für CARD in der Sicht SYSSTAT.TABLES (die Anzahl der unterschiedlichen Werte in einer Spalte darf die Anzahl der Zeilen nicht überschreiten). Nehmen Sie an, Sie wollen den Wert für COLCARD von 100 auf 25 und den Wert für CARD von 200 auf 50 verringern. Wenn Sie die Sicht SYSCAT.TABLES zuerst aktualisieren, sollten Sie eine Fehlermeldung empfangen (da CARD kleiner als COLCARD würde). Die richtige Reihenfolge wäre, zuerst den Wert für COLCARD in der Sicht SYSCAT.COLUMNS und anschließend den Wert für CARD in der Sicht SYSSTAT.TABLES zu aktualisieren. Der Fall stellt sich umgekehrt dar, wenn Sie den Wert von COLCARD von 100 auf 250 und den Wert für CARD von 200 auf 300 erhöhen wollen. In diesem Fall müssten Sie zuerst den Wert CARD und anschließend den Wert COLCARD aktualisieren.

Wenn ein Konflikt zwischen einer aktualisierten Statistik und einer anderen Statistik festgestellt wird, wird eine Fehlermeldung ausgegeben. Jedoch werden vielleicht nicht immer Fehler gemeldet, wenn Konflikte auftreten. In einigen Fällen können die Konflikte nur schwer festgestellt und als Fehler gemeldet werden, besonders wenn die beiden zusammengehörigen Statistiken in verschiedenen Katalogen gespeichert sind. Aus diesem Grund sollten Sie solche Konflikte umsichtig vermeiden.

Die folgenden allgemeinen Prüfungen sollten vor der Aktualisierung einer Katalogstatistik durchgeführt werden:

1. Numerische Statistikdaten müssen -1 bzw. größer oder gleich null (0) sein.
2. Numerische Statistikdaten, die Prozentwerte darstellen (z. B. CLUSTERRATIO in der Katalogsicht SYSSTAT.INDEXES), müssen zwischen 0 und 100 liegen.

Anmerkung: Bei Zeilentypen sind die statistischen Daten auf Tabellenebene für NPAGES, FPAGES und OVERFLOW für eine untergeordnete Tabelle nicht aktualisierbar.

Wenn eine Tabelle erstmals erstellt wird, werden die Systemkatalogstatistikdaten auf -1 gesetzt; dies bedeutet, dass die Tabelle über keine Statistikdaten verfügt. DB2 verwendet für die Kompilierung und Optimierung von SQL- oder XQuery-Anweisungen Standardwerte, bis die Statistikdaten erfasst worden sind. Die Aktualisierung der Tabelle oder der Indexstatistik schlägt möglicherweise fehl, wenn die neuen Werte mit den Standardwerten nicht übereinstimmen. Aus diesem Grund ist es empfehlenswert, RUNSTATS nach der Erstellung einer Tabelle auszuführen, bevor Sie die Tabelle oder Indexstatistik aktualisieren.

Zugehörige Konzepte:

- „Katalogstatistiken zu Modellierung und Fallstudien“ auf Seite 349
- „Katalogstatistiktabellen“ auf Seite 330
- „Regeln zur manuellen Aktualisierung von Spaltenstatistiken“ auf Seite 354
- „Regeln zur manuellen Aktualisierung von Verteilungsstatistiken“ auf Seite 355
- „Regeln zur manuellen Aktualisierung von Indexstatistiken“ auf Seite 357
- „Regeln zur manuellen Aktualisierung von Tabellen- und Kurznamenstatistiken“ auf Seite 356
- „Statistiken zur Modellierung von Produktionsdatenbanken“ auf Seite 350
- „Statistiken für benutzerdefinierte Funktionen“ auf Seite 341

Regeln zur manuellen Aktualisierung von Spaltenstatistiken

Bei der Aktualisierung von Statistikdaten in der Katalogsicht SYSSTAT.COLUMNS sind folgende Regeln zu beachten.

- Folgen Sie bei der manuellen Aktualisierung der Werte für die Spalten HIGH2KEY und LOW2KEY in der Sicht SYSSTAT.COLUMNS der Funktionsweise der generierten Werte:
 - Die Werte für HIGH2KEY und LOW2KEY müssen gültige Werte für den Datentyp der entsprechenden Benutzerspalte sein.
 - Die Länge der Werte für HIGH2KEY, LOW2KEY muss entweder 33 oder die maximale Länge des Datentyps der Zielspalte betragen, je nachdem, welcher der beiden Werte kleiner ist. Dies schließt zusätzliche Anführungszeichen nicht mit ein, durch die die Länge der Zeichenfolge auf 68 anwachsen kann. Dies bedeutet, dass nur die ersten 33 Zeichen des Werts in der entsprechenden Benutzerspalte bei der Bestimmung der Werte für HIGH2KEY, LOW2KEY in Betracht gezogen werden.
 - Die HIGH2KEY/LOW2KEY-Werte werden so gespeichert, dass sie in der SET-Klausel einer UPDATE-Anweisung und ohne Manipulation an Aufwandsberechnungen verwendet werden können. Für Zeichenfolgen bedeutet dies, dass einfache Anführungszeichen am Anfang und am Ende der Zeichenfolge angefügt und ein zusätzliches Anführungszeichen für jedes bereits in der Zeichenfolge vorhandene Anführungszeichen hinzugefügt wird. Beispiele von Benutzerspaltenwerten und ihren entsprechenden Werten für HIGH2KEY, LOW2KEY finden Sie in der folgenden Tabelle.

Tabelle 75. HIGH2KEY- und LOW2KEY-Werte für Datentypen

Datentyp in Benutzerspalte	Benutzerdaten	Entsprechender HIGH2KEY-, LOW2KEY-Wert
INTEGER	-12	-12
CHAR	abc	'abc'
CHAR	ab'c	'ab"c'

- Der Wert für HIGH2KEY sollte größer als der Wert für LOW2KEY sein, wenn es mehr als drei unterschiedliche Werte in der entsprechenden Spalte gibt.
- Die Kardinalität einer Spalte (Statistik COLCARD in der Katalogsicht SYSSTAT.COLUMNS) darf nicht größer sein als die Kardinalität der zugehörigen Tabelle oder statistischen Sicht (Statistik CARD in der Katalogsicht SYSSTAT.TABLES).

- Die Anzahl der Nullwerte in einer Spalte (Statistik NUMNULLS in der Katalogsicht SYSSTAT.COLUMNS) darf nicht größer sein als die Kardinalität der zugehörigen Tabelle oder statistischen Sicht (Statistik CARD in der Katalogsicht SYSSTAT.TABLES).
- Für Spalten mit folgenden Datentypen werden keine Statistikdaten unterstützt: LONG VARCHAR, LONG VARGRAPHIC, BLOB, CLOB, DBCLOB.

Zugehörige Konzepte:

- „Katalogstatistiken“ auf Seite 319
- „Katalogstatistiken zu Modellierung und Fallstudien“ auf Seite 349
- „Katalogstatistiktabellen“ auf Seite 330
- „Allgemeine Regeln zur manuellen Aktualisierung von Katalogstatistiken“ auf Seite 353

Regeln zur manuellen Aktualisierung von Verteilungsstatistiken

Manuelle Aktualisierungen von Verteilungsstatistiken nehmen Sie nur zu dem Zweck vor, eine Produktionsdatenbank nachzumodellieren oder Fallstudien für eine künstlich konstruierte Datenbank durchzuführen. Nehmen Sie keine manuellen Aktualisierungen an den Verteilungsstatistiken einer Produktionsdatenbank vor.

Stellen Sie sicher, dass alle Statistiken im Katalog konsistent sind. Im Einzelnen müssen die Katalogeinträge für Häufigkeitsstatistiken und Quantildaten für jede Spalte die folgenden Bedingungen erfüllen:

- Häufigkeitsstatistiken (in der Katalogsicht SYSSTAT.COLDIST). Dazu gehören:
 - Die Werte in der Spalte VALCOUNT müssen bei steigenden Werten für SEQNO unverändert bleiben oder abnehmen.
 - Die Anzahl der Werte in der Spalte COLVALUE muss kleiner oder gleich der Anzahl der unterschiedlichen Werte in der Spalte sein. Diese Anzahl wird in der Spalte COLCARD der Katalogsicht SYSSTAT.COLUMNS gespeichert.
 - Die Summe der Werte in der Spalte VALCOUNT muss kleiner oder gleich der Anzahl der Zeilen in der Spalte sein. Diese Anzahl wird in der Spalte CARD der Katalogsicht SYSSTAT.TABLES gespeichert.
 - In der Regel sollten die Werte in der Spalte COLVALUE zwischen dem zweithöchsten und dem zweitniedrigsten Datenwert für die Spalte liegen. Diese beiden Werte werden in der Spalte HIGH2KEY bzw. LOW2KEY der Katalogsicht SYSSTAT.COLUMNS gespeichert. Es kann nur einen häufigen Wert geben, der größer als der Wert für HIGH2KEY ist, und einen, der kleiner als der Wert für LOW2KEY ist.
- Quantile (in der Katalogsicht SYSSTAT.COLDIST). Dazu gehören:
 - Die Werte in der Spalte COLVALUE müssen bei steigenden Werten für SEQNO unverändert bleiben oder abnehmen.
 - Die Werte in der Spalte VALCOUNT müssen bei steigenden Werten für SEQNO steigen.
 - Der größte Wert in der Spalte COLVALUE muss einen entsprechenden Eintrag in der Spalte VALCOUNT haben, der gleich der Anzahl von Zeilen in der Spalte ist.
 - In der Regel sollten die Werte in der Spalte COLVALUE zwischen dem zweithöchsten und dem zweitniedrigsten Datenwert für die Spalte liegen. Diese beiden Werte werden in der Spalte HIGH2KEY bzw. LOW2KEY der Katalogsicht SYSSTAT.COLUMNS gespeichert.

Nehmen Sie an, es stehen Verteilungsstatistikdaten für eine Spalte C1 mit „Z“ Zeilen zur Verfügung, und Sie möchten die Statistiken so modifizieren, dass sie einer Spalte mit identischen relativen Proportionen der Datenwerte, aber mit „(F × Z)“ Zeilen entsprechen. Um die Größenordnung der Häufigkeitsstatistiken um einen Faktor F zu erhöhen, muss jeder Eintrag der Spalte VALCOUNT mit F multipliziert werden. Um die Quantilwerte ebenfalls um einen Faktor F zu vergrößern, muss jeder Eintrag der Spalte VALCOUNT mit F multipliziert werden. Wenn Sie diese Regeln nicht beachten, kann das Optimierungsprogramm den falschen Filterfaktor verwenden, was bei der Ausführung der Abfrage zu nicht vorhersehbaren Auswirkungen auf die Leistung führen kann.

Zugehörige Konzepte:

- „Katalogstatistiken“ auf Seite 319
- „Katalogstatistiken zu Modellierung und Fallstudien“ auf Seite 349
- „Katalogstatistiktabellen“ auf Seite 330
- „Allgemeine Regeln zur manuellen Aktualisierung von Katalogstatistiken“ auf Seite 353

Regeln zur manuellen Aktualisierung von Tabellen- und Kurznamenstatistiken

Die einzigen statistischen Werte, die Sie in der Katalogsicht SYSTAT.TABLES aktualisieren können, sind CARD, FPAGES, NPAGES und OVERFLOW sowie für MDC-Tabellen ACTIVE_BLOCKS. Beachten Sie dabei Folgendes:

1. Der Wert für CARD der Tabelle muss größer als oder gleich allen auf diese Tabelle bezogenen Werten für COLCARD in der Katalogsicht SYSSTAT.COLUMNS sein.
2. Der Wert für CARD muss größer als der Wert für NPAGES sein.
3. Der Wert für FPAGES muss größer als der Wert für NPAGES sein.
4. Der Wert für NPAGES muss kleiner oder gleich jedem Wert für den Abrufteil der Wertepaare in der Spalte PAGE_FETCH_PAIRS für jeden Index sein (unter der Annahme, dass diese Statistik für den Index relevant ist).
5. Der Wert für CARD darf nicht kleiner oder gleich irgendeinem Wert für den Seitenabruf in den Wertepaaren der Spalte PAGE_FETCH_PAIRS für jeden Index sein (unter der Annahme, dass diese Statistik für den Index relevant ist).

Wenn Sie in einem System föderierter Datenbanken arbeiten, gehen Sie bei der Erstellung bzw. Aktualisierung von Statistiken für einen Kurznamen über eine ferne Sicht sehr vorsichtig vor. Die statistischen Informationen, wie zum Beispiel die Anzahl von Zeilen, die von diesem Kurznamen zurückgegeben werden, entsprechen möglicherweise nicht dem realen Aufwand zur Auswertung dieser Sicht und können das DB2-Optimierungsprogramm irreführen. Fälle, die von Aktualisierungen der Statistiken profitieren können, sind ferne Sichten, die für eine einzelne ferne Tabelle ohne Anwendung von Spaltenfunktionen in der SELECT-Liste definiert wurden. Komplexe Sichten erfordern möglicherweise einen komplexen Optimierungsprozess, der die Optimierung jeder einzelnen Abfrage erforderlich macht. Ziehen Sie stattdessen die Erstellung lokaler Sichten über Kurznamen in Betracht, damit das DB2-Optimierungsprogramm in der Lage ist, den Aufwand für die Sicht genauer abzuschätzen.

Zugehörige Konzepte:

- „Katalogstatistiken“ auf Seite 319
- „Katalogstatistiken zu Modellierung und Fallstudien“ auf Seite 349

- „Katalogstatistiktabellen“ auf Seite 330
- „Allgemeine Regeln zur manuellen Aktualisierung von Katalogstatistiken“ auf Seite 353

Regeln zur manuellen Aktualisierung von Indexstatistiken

Bei der Aktualisierung der Statistiken in der Katalogsicht SYSSTAT.INDEXES sind folgende Regeln zu beachten:

1. Die Werte für PAGE_FETCH_PAIRS (in der Katalogsicht SYSSTAT.INDEXES) müssen folgende Regeln erfüllen:
 - Einzelne Werte in der Statistik PAGE_FETCH_PAIRS müssen durch eine Folge von Leerzeichenbegrenzern getrennt werden.
 - Einzelne Werte in der Statistik PAGE_FETCH_PAIRS dürfen eine Länge von 10 Stellen nicht überschreiten und müssen kleiner als der größte ganzzahlige Wert (MAXINT = 2147483647) sein.
 - Es muss immer einen gültigen Wert für PAGE_FETCH_PAIRS geben, wenn für CLUSTERFACTOR ein Wert größer null (0) angegeben ist.
 - Es müssen sich genau 11 Wertepaare in einer einzelnen Statistik PAGE_FETCH_PAIR befinden.
 - Die Einträge für die Puffergrößen in PAGE_FETCH_PAIRS müssen eine aufsteigende Wertefolge bilden.
 - Kein Wert für die Puffergröße in einem Eintrag für PAGE_FETCH_PAIRS darf den Wert $\text{MIN}(\text{NPAGES}, 524287)$ bei 32-Bit-Betriebssystemen bzw. $\text{MIN}(\text{NPAGES}, 2147483647)$ bei 64-Bit-Betriebssystemen überschreiten, wobei NPAGES die Anzahl der Seiten in der entsprechenden Tabelle (in der Katalogsicht SYSSTAT.TABLES) ist.
 - Einträge für „Seitenabrufe“ in PAGE_FETCH_PAIRS müssen in absteigender Wertefolge angegeben werden, wobei kein einzelner Eintrag für „Seitenabrufe“ kleiner als NPAGES sein darf. Einträge für „Seitenabrufe“ in PAGE_FETCH_PAIRS können nicht größer als der Wert der Statistik CARD (Kardinalität) der zugehörigen Tabelle sein.
 - Wenn der Wert für die Puffergröße in zwei aufeinander folgenden Paaren übereinstimmt, muss der Wert für den Seitenabruf (Page fetch) in beiden Paaren ebenfalls übereinstimmen (in der Katalogsicht SYSSTAT.TABLES).

Eine gültige Aktualisierung der Wertefolge für PAGE_FETCH_PAIRS ist zum Beispiel:

```
PAGE_FETCH_PAIRS =
'100 380 120 360 140 340 160 330 180 320 200 310 220 305 240 300
260 300 280 300 300 300'
```

mit

```
NPAGES = 300
CARD    = 10000
CLUSTERRATIO = -1
CLUSTERFACTOR = 0.9
```

2. Die Werte für CLUSTERRATIO und CLUSTERFACTOR (in der Katalogsicht SYSSTAT.INDEXES) müssen folgende Regeln erfüllen:
 - Gültige Werte für CLUSTERRATIO (Clusterverhältnis) sind -1 bzw. Werte zwischen 0 und 100.
 - Gültige Werte für CLUSTERFACTOR (Clusterfaktor) sind -1 bzw. Werte zwischen 0 und 1.
 - Es muss immer mindestens einer der Werte für CLUSTERRATIO und CLUSTERFACTOR gleich -1 sein.

- Wenn für CLUSTERFACTOR ein positiver Wert angegeben ist, müssen gültige Statistikdaten in der Spalte PAGE_FETCH_PAIR enthalten sein.
- 3. Bei relationalen Indizes gelten die folgenden Regeln für FIRSTKEYCARD, FIRST2KEYCARD, FIRST3KEYCARD, FIRST4KEYCARD, FULLKEYCARD und INDCARD:
 - Der Wert für FIRSTKEYCARD muss gleich dem Wert für FULLKEYCARD für einen einspaltigen Index sein.
 - Der Wert für FIRSTKEYCARD muss gleich dem Wert COLCARD (in SYS-STAT.COLUMNS) für die entsprechende Spalte sein.
 - Wenn einige dieser Indexstatistikwerte nicht relevant sind, sollten Sie sie auf den Wert -1 setzen. Wenn Sie beispielsweise einen Index mit nur drei Spalten haben, setzen Sie FIRST4KEYCARD auf den Wert -1.
 - Für mehrspaltige Indizes gilt die folgende Beziehung, wenn alle Statistikwerte relevant sind:

$$\text{FIRSTKEYCARD} \leq \text{FIRST2KEYCARD} \leq \text{FIRST3KEYCARD} \leq \text{FIRST4KEYCARD} \leq \text{FULLKEYCARD} \leq \text{INDCARD} = \text{CARD}$$
- 4. Bei Indizes für XML-Daten gelten die folgenden Regeln für FIRSTKEYCARD, FIRST2KEYCARD, FIRST3KEYCARD, FIRST4KEYCARD, FULLKEYCARD und INDCARD:
 - Die Beziehung muss wie folgt aussehen:

$$\text{FIRSTKEYCARD} \leq \text{FIRST2KEYCARD} \leq \text{FIRST3KEYCARD} \leq \text{FIRST4KEYCARD} \leq \text{FULLKEYCARD} \leq \text{INDCARD}$$
- 5. Die folgenden Regeln gelten für SEQUENTIAL_PAGES und DENSITY:
 - Gültige Werte für SEQUENTIAL_PAGES sind -1 bzw. Werte zwischen 0 und NLEAF.
 - Gültige Werte für DENSITY sind -1 bzw. Werte zwischen 0 und 100.

Zugehörige Konzepte:

- „Katalogstatistiken“ auf Seite 319
- „Katalogstatistiken zu Modellierung und Fallstudien“ auf Seite 349
- „Katalogstatistiktabellen“ auf Seite 330
- „Allgemeine Regeln zur manuellen Aktualisierung von Katalogstatistiken“ auf Seite 353

Automatische Statistikerfassung

Automatische Statistikerfassung

Das DB2-Optimierungsprogramm verwendet Katalogstatistiken, um den effizientesten Zugriffsplan für eine gegebene Abfrage zu ermitteln. Wenn die Statistiken für eine Tabelle oder einen Index nicht auf dem neuesten Stand oder unvollständig sind, könnte das Optimierungsprogramm einen Plan auswählen, der nicht optimal ist, so dass die Abfrageausführung verlangsamt würde. Allerdings ist die Entscheidung, welche Statistiken für eine gegebene Auslastung zu erfassen sind, recht komplex und die Pflege aktueller Statistiken zeitaufwendig.

Mit der automatischen Statistikerfassung, die Teil der Funktion zur automatischen Tabellenverwaltung von DB2 ist, können Sie DB2 die Entscheidung überlassen, welche Statistiken für Ihre Auslastung erforderlich sind und welche Statistiken aktualisiert werden müssen. Bei aktivierter automatischer Statistikerfassung führt DB2 das Dienstprogramm RUNSTATS automatisch im Hintergrund aus, um sicherzustellen, dass die korrekten Statistiken erfasst und gepflegt werden.

Ab DB2 Version 9 ist die automatische Statistikerfassung standardmäßig aktiviert, wenn eine neue Datenbank erstellt wird.

Die Leistungsbeeinträchtigung durch die automatische Statistikerfassung wird auf mehrere Arten minimiert:

- Die Statistikerfassung erfolgt durch eine gedrosselte Ausführung von RUNSTATS. Die Drosselung begrenzt je nach aktueller Datenbankaktivität die Menge der Ressourcen, die vom Dienstprogramm RUNSTATS beansprucht werden: wenn die Datenbankaktivitäten zunehmen, läuft das Dienstprogramm RUNSTATS langsamer, so dass es weniger Ressourcen benötigt.
- Es wird nur eine minimale Gruppe von Statistiken zur Optimierung der Leistung erfasst. Dies wird durch die Verwendung von Statistikprofilen erreicht, bei der Informationen über vergangene Datenbankaktivitäten genutzt werden, um zu bestimmen, welche Statistiken für die Datenbankauslastung erforderlich sind und wie schnell diese Statistiken in Anbetracht der Arten von Aktivitäten in der Datenbank veralten.
- Nur Tabellen mit hohen Graden an Aktivität (gemessen an der Anzahl der Aktualisierungen, Löschungen und Einfügungen) werden für die Statistikerfassung in Betracht gezogen. Für große Tabellen (die aus mehr als 4000 Seiten bestehen) werden außerdem Stichproben erstellt, um festzustellen, ob die Statistiken durch die intensive Tabellenaktivität tatsächlich beeinflusst wurden. Statistiken für solch große Tabellen werden nur erfasst, wenn dies gerechtfertigt ist.
- Das Dienstprogramm RUNSTATS wird automatisch zur Ausführung während des optimalen Verwaltungsfensters terminiert, das in der Definition Ihrer Verwaltungsrichtlinie angegeben ist. Diese Richtlinie gibt außerdem die Gruppe von Tabellen an, die zum Umfang der automatischen Statistikerfassung gehören, was zusätzlich eine unnötige Ressourcennutzung minimiert.
- Während der Ausführung der automatischen Statistikerfassung bleiben die betroffenen Tabellen weiterhin für reguläre Datenbankaktivitäten (Aktualisierungen, Einfügungen, Löschungen) verfügbar, so als ob RUNSTATS nicht für die Tabelle ausgeführt würde.

Zugehörige Konzepte:

- „Standardmäßig aktivierte automatische Funktionen“ in *Systemverwaltung: Konzept*
- „Erfassen von Statistiken mit einem Statistikprofil“ auf Seite 361

Zugehörige Tasks:

- „Verwenden der automatischen Statistikerfassung“ auf Seite 359

Zugehörige Referenzen:

- „RUNSTATS command using the ADMIN_CMD procedure“ in *Administrative SQL Routines and Views*
- „auto_maint - Automatische Verwaltung “ auf Seite 547
- „util_impact_lim - Richtlinie für Instanzauslastungswirkung “ auf Seite 573
- „RUNSTATS command“ in *Command Reference*

Verwenden der automatischen Statistikerfassung

Über genaue und vollständige Datenbankstatistiken zu verfügen, ist von kritischer Bedeutung für einen effizienten Datenzugriff und eine optimale Auslastungsleistung. Verwenden Sie die Funktion zur automatischen Statistikerfassung der Funktionalität zur automatischen Tabellenverwaltung, um relevante Datenbank-

statistiken zu aktualisieren und zu pflegen. In einer seriellen Umgebung können Sie diese Funktionalität optional noch erweitern, indem Sie Abfragedaten sammeln und Statistikprofile generieren, die DB2 bei der automatischen Erfassung der exakten Gruppe der für Ihre Auslastung erforderlichen Statistiken unterstützen. Diese Option ist in föderierten bzw. MPP-Umgebungen oder bei aktivierter partitions-interner Parallelität nicht verfügbar.

Vorgehensweise:

Sie können diese Funktion entweder über die Tools der grafischen Benutzerschnittstelle oder über die Befehlszeilenschnittstelle aktivieren.

- Gehen Sie wie folgt vor, um Ihre Datenbank für die automatische Statistikerfassung über die Tools der grafischen Benutzerschnittstelle zu konfigurieren:
 1. Öffnen Sie den Assistenten „Automatische Verwaltung konfigurieren“ entweder über die Steuerzentrale, indem Sie ein Datenbankobjekt mit der rechten Maustaste anklicken, oder über die Diagnosezentrale, indem Sie die Datenbankinstanz, die Sie zur automatischen Statistikerfassung konfigurieren wollen, mit der rechten Maustaste anklicken. Wählen Sie **Automatische Verwaltung konfigurieren** im Kontextmenü aus.
 2. In diesem Assistenten können Sie die automatische Statistikerfassung aktivieren, die Tabellen angeben, aus denen die Statistiken automatisch zu erfassen sind, und ein Verwaltungsfenster für die Ausführung des Dienstprogramms RUNSTATS definieren.
 3. OPTIONAL: Zur Aktivierung der automatischen Statistikprofilerstellung setzen Sie die folgenden beiden Konfigurationsparameter über die Befehlszeilenschnittstelle auf ON:
 - AUTO_STATS_PROF
 - AUTO_PROF_UPD
- Gehen Sie wie folgt vor, um Ihre Datenbank für die automatische Statistikerfassung über die Befehlszeilenschnittstelle zu konfigurieren:
 1. Setzen Sie jeden der folgenden Konfigurationsparameter auf den Wert ON:
 - AUTO_MAINT
 - AUTO_TBL_MAINT
 - AUTO_RUNSTATS
 2. OPTIONAL: Zur Aktivierung der automatischen Statistikprofilerstellung setzen Sie die folgenden beiden Konfigurationsparameter auf ON:
 - AUTO_STATS_PROF
 - AUTO_PROF_UPD

Zugehörige Konzepte:

- „Automatische Statistikerfassung“ auf Seite 358
- „Erfassen von Statistiken mit einem Statistikprofil“ auf Seite 361

Zugehörige Referenzen:

- „auto_maint - Automatische Verwaltung “ auf Seite 547

Automatische Statistikprofilerstellung

Das Dienstprogramm RUNSTATS bietet eine Option zur Registrierung und Verwendung eines Statistikprofils, bei dem es sich um eine Gruppe von Optionen handelt, die angeben, welche Statistiken für eine bestimmte Tabelle zu erfassen sind, wie zum Beispiel Tabellenstatistiken, Indexstatistiken oder Verteilungsstatistiken.

Statistikprofile können nun automatisch durch die DB2-Funktion zur automatischen Statistikprofilerstellung generiert werden. Wenn diese Funktion aktiviert ist, werden Informationen zur Datenbankaktivität gesammelt und in einem Abfrage-Feedback-Warehouse gespeichert. Auf der Grundlage dieser Daten wird ein Statistikprofil generiert. Die Aktivierung dieser Funktion kann das Problem der Unsicherheit, welche Statistiken für eine bestimmte Auslastung relevant sind, etwas mildern und die Erfassung einer minimalen Gruppe von Statistiken ermöglichen, um eine optimale Leistung für die Auslastung der Datenbank zu erzielen.

Diese Funktion kann zusammen mit der Funktion zur automatischen Statistikerfassung verwendet werden, damit entschieden werden kann, welche Statistikdaten für eine bestimmte Tabelle auf der Basis vorheriger Tabellenaktivitäten erfasst werden sollen.

Zugehörige Konzepte:

- „Automatisieren der Datenbankverwaltung“ in den *Themen aus Vorversionen*
- „Automatische Erstellung von Statistikprofilen mit Hilfe der automatischen Statistikerfassung“ in *Systemverwaltung: Konzept*
- „Erfassen von Statistiken mit einem Statistikprofil“ auf Seite 361
- „Richtlinien für die Erfassung und Aktualisierung von Statistiken“ auf Seite 326

Erfassen von Statistiken mit einem Statistikprofil

Das Dienstprogramm RUNSTATS bietet eine Option zur Registrierung und Verwendung eines Statistikprofils, bei dem es sich um eine Gruppe von Optionen handelt, die angeben, welche Statistiken für eine bestimmte Tabelle zu erfassen sind, wie zum Beispiel Tabellenstatistiken, Indexstatistiken oder Verteilungsstatistiken.

Diese Einrichtung vereinfacht die Erfassung von Statistiken, indem es Ihnen die Möglichkeit gibt, die Optionen, die Sie bei der Ausführung des Befehls RUNSTATS angeben, zu speichern, so dass Sie die gleichen Statistiken wiederholt für eine Tabelle erfassen können, ohne die Befehlsoptionen erneut eingeben zu müssen.

Sie können ein Statistikprofil mit und ohne gleichzeitige Erfassung von Statistiken registrieren oder aktualisieren. Um zum Beispiel ein Profil zu registrieren und gleichzeitig Statistiken zu erfassen, führen Sie den Befehl RUNSTATS mit der Option SET PROFILE aus. Soll nur ein Profil registriert werden, ohne tatsächlich Statistiken zu erfassen, führen Sie den Befehl RUNSTATS mit der Option SET PROFILE ONLY aus.

Zur Erfassung von Statistiken unter Verwendung eines bereits registrierten Statistikprofils führen Sie den Befehl RUNSTATS aus, indem Sie nur den Namen der Tabelle und die Option USE PROFILE angeben.

Wenn Sie prüfen wollen, welche Optionen zurzeit in dem Statistikprofil für eine bestimmte Tabelle angegeben sind, können Sie die Katalogtabellen mit der folgenden SELECT-Anweisung abfragen, wobei tabellenname der Name der Tabelle ist, für die Sie das Profil abrufen wollen:

```
SELECT STATISTICS_PROFILE FROM SYSIBM.SYSTABLES WHERE NAME = tabellenname
```

Automatische Statistikprofilerstellung

Statistikprofile können außerdem automatisch durch die DB2-Funktion zur automatischen Statistikprofilerstellung generiert werden. Wenn diese Funktion aktiviert ist, werden Informationen zur Datenbankaktivität gesammelt und in einem Abfrage-Feedback-Warehouse gespeichert. Auf der Grundlage dieser Daten wird ein

Statistikprofil generiert. Die Aktivierung dieser Funktion kann das Problem der Unsicherheit, welche Statistiken für eine bestimmte Auslastung relevant sind, etwas mildern und die Erfassung einer minimalen Gruppe von Statistiken ermöglichen, um eine optimale Leistung für die Auslastung der Datenbank zu erzielen.

Diese Funktion kann zusammen mit der Funktion zur automatischen Statistikerfassung verwendet werden, die wiederum automatisch Statistikpflegeoperationen auf der Basis der Informationen terminiert, die in dem automatisch generierten Statistikprofil enthalten sind.

Zur Aktivierung dieser Funktion müssen Sie die automatische Tabellenpflege bereits mit Hilfe der entsprechenden Konfigurationsparameter aktiviert haben. Der Konfigurationsparameter `AUTO_STATS_PROF` aktiviert die Erfassung von Abfragefeedbackdaten, während der Konfigurationsparameter `AUTO_PROF_UPD` die Generierung eines Statistikprofils zur Verwendung durch die automatische Statistikerfassung aktiviert.

Anmerkung: Die automatische Statistikprofilerstellung kann nur im seriellen DB2-Modus aktiviert werden und wird für Abfragen in föderierten Umgebungen oder in MPP-Umgebungen mit mehreren Partitionen sowie bei aktivierter partitionsinterner Parallelität blockiert.

Die Statistikprofilgenerierung eignet sich am besten für Umgebungen, in denen große, komplexe Abfragen ausgeführt werden, die viele Vergleichselemente anwenden und häufig Korrelationen in den Daten der Vergleichselementspalten haben und Joins und Gruppierungen mehrerer Tabellen erfordern. Sie eignet sich weniger für Umgebungen mit einer in erster Linie transaktionsorientierten Auslastung.

Es gibt zwei unterschiedliche Möglichkeiten, diese Funktion zu verwenden:

- In einer Testumgebung. Setzen Sie `AUTO_STATS_PROF` und `AUTO_PROF_UPD` auf ON in Testsystemen, in denen der Leistungsaufwand der Laufzeitüberwachung leicht toleriert werden kann. Wenn das Testsystem mit realistischen Daten und Abfragen arbeitet, ermöglicht dies eine Ermittlung der tatsächlichen Korrelationen und Einstellungen der Statistikparameter für RUNSTATS, die wiederum im Statistikprofil gespeichert werden. Diese Profile können anschließend auf das Produktionssystem übertragen werden, auf dem die Abfragen von ihnen profitieren können, ohne den Überwachungsaufwand zu verursachen.
- Zur Untersuchung von Leistungsproblemen für bestimmte Abfragen in einer Produktionsumgebung. Wenn Leistungsprobleme für eine bestimmte Gruppe von Abfragen festgestellt werden, die fehlerhaften Statistiken oder Korrelationen zugeschrieben werden können, können Sie `AUTO_STATS_PROF` aktivieren und die Zielauslastung über einen Zeitraum ausführen. Die automatische Statistikprofilgenerierung analysiert das Abfragefeedback und erstellt Empfehlungen in den Tabellen `SYSTOOLS.OPT_FEEDBACK_RANKING*`. Sie können diese Empfehlungen einsehen und die Statistikprofile manuell auf der Grundlage der Empfehlungen verfeinern. Wenn DB2 die Statistikprofile auf der Grundlage dieser Empfehlungen automatisch aktualisieren soll, aktivieren Sie `AUTO_PROF_UPD`, wenn Sie `AUTO_STATS_PROF` aktivieren.

Anmerkung: Die Überwachung der Abfragen und das Speichern von Abfragefeedbackdaten im Feedback-Warehouse ist mit einigem Leistungsaufwand verbunden.

Erstellen des Warehouse für Abfragefeedback: Das Feedback-Warehouse besteht aus fünf Tabellen im Schema SYSTOOLS, in denen Informationen zu den Vergleichselementen, die bei der Ausführung von Abfragen angetroffen werden, sowie Empfehlungen für die Statistikerfassung gespeichert werden. Diese fünf Tabellen heißen OPT_FEEDBACK_QUERY, OPT_FEEDBACK_PREDICATE, OPT_FEEDBACK_PREDICATE_COLUMN, OPT_FEEDBACK_RANKING und OPT_FEEDBACK_RANKING_COLUMN.

Zur Verwendung der automatischen Statistikprofilgenerierung müssen Sie zunächst das Abfrage-Feedback-Warehouse mit Hilfe der gespeicherten Prozedur SYSINSTALLOBJECTS erstellen. Diese gespeicherte Prozedur ist die allgemeine gespeicherte Prozedur zum Erstellen und Löschen von Objekten im Schema SYSTOOLS.

Rufen Sie die gespeicherte Prozedur SYSINSTALLOBJECTS wie folgt auf:

```
call SYSINSTALLOBJECTS ( toolname, aktion, tabellenbereichsname, schemaname)
```

Dabei gilt Folgendes:

toolname

Gibt den Namen des Tools an, dessen Objekte zu erstellen oder löschen sind. In diesem Fall 'ASP' bzw. 'AUTO STATS PROFILING'.

aktion Gibt die auszuführende Aktion an: 'C' für Erstellen (Create) und 'D' für Löschen (Drop).

tabellenbereichsname

Der Name des Tabellenbereichs, in dem die Tabellen für das Feedback-Warehouse erstellt werden sollen. Dieser Eingabeparameter ist optional. Wenn er nicht angegeben wird, wird der Standardbenutzertabellenbereich verwendet.

schemaname

Der Name des Schemas, in dem die Objekte erstellt oder gelöscht werden. Dieser Parameter wird momentan nicht verwendet.

Wenn Sie zum Beispiel das Feedback-Warehouse in Tabellenbereich "A" erstellen wollen, geben Sie Folgendes ein: `call SYSINSTALLOBJECTS ('ASP', 'C', 'A', '')`

Zugehörige Konzepte:

- „Automatische Statistikerfassung“ auf Seite 358

Zugehörige Tasks:

- „Verwenden der automatischen Statistikerfassung“ auf Seite 359

Kapitel 13. Tabellen- und Indexverwaltung

Tabellenreorganisation

Nach zahlreichen Änderungen an Tabellendaten können sich logisch sequenzielle Daten auf physisch nicht sequenziellen Datenseiten befinden, so dass der Datenbankmanager zusätzliche Leseoperationen ausführen muss, um auf die Daten zuzugreifen. Darüber hinaus sind zusätzliche Leseoperationen auch erforderlich, wenn eine beträchtliche Anzahl von Zeilen gelöscht wurde. In einem solchen Fall können Sie eine Reorganisation der Tabelle in Betracht ziehen, um die Tabelle mit dem Index in Übereinstimmung zu bringen und Speicherplatz wieder verfügbar zu machen. Sie können sowohl die Systemkatalogtabellen als auch Datenbanktabellen reorganisieren.

Anmerkung: Da die Reorganisation einer Tabelle in der Regel mehr Zeit als das Erfassen von Statistikdaten beansprucht, können Sie das Dienstprogramm RUNSTATS ausführen, um die Statistiken für Ihre Daten auf den aktuellen Stand zu bringen, und einen Rebind für Ihre Anwendungen durchzuführen. Wenn aktualisierte Statistiken keine Leistungsverbesserungen erzielen, kann eine Reorganisation eventuell helfen. Detaillierte Informationen zu den Optionen und der Funktionsweise des Dienstprogramms REORG TABLE finden Sie in der zugehörigen Befehlsreferenz.

Beachten Sie die folgenden Faktoren, die Hinweise darauf geben können, dass eine Tabelle reorganisiert werden sollte:

- Es ist ein hohes Aufkommen an INSERT-, UPDATE- und DELETE-Aktivitäten an Tabellen zu verzeichnen, auf die durch Abfragen zugegriffen wird.
- Es treten wesentliche Änderungen in der Leistung von Abfragen auf, die einen Index mit einem hohen Clusterverhältnis verwenden.
- Die Leistung wird durch die Ausführung von RUNSTATS zur Aktualisierung der Statistikdaten nicht besser.
- Der Befehl REORGCHK zeigt den Bedarf einer Reorganisation der Tabelle an.
- Die Kosten einer zunehmenden Verschlechterung der Abfrageleistung und die Kosten einer Reorganisation der Tabelle, zu denen die CPU-Zeit, die zur Ausführung erforderliche Zeit sowie der verringerte Grad des gemeinsamen Zugriffs aufgrund der Sperre, die das Dienstprogramm REORG für die Tabelle bis zum Ende der Reorganisation aktiviert, zu rechnen sind, müssen gegeneinander abgewogen werden.

Verringern der Erforderlichkeit der Reorganisation von Tabellen

Damit die Reorganisation einer Tabelle weniger häufig erforderlich wird, führen Sie nach dem Erstellen der Tabelle folgende Aufgaben aus:

- Ändern Sie die Tabelle (ALTER), um PCTFREE hinzuzufügen.
- Erstellen Sie einen Clusterindex mit PCTFREE für Index.
- Sortieren Sie die Daten.
- Laden Sie die Daten.

Wenn Sie diese Maßnahmen durchgeführt haben, tragen der Clusterindex der Tabelle und die Einstellung von PCTFREE für die Tabelle dazu bei, die ursprüngli-

che Sortierreihenfolge der Daten beizubehalten. Wenn genügend Speicherplatz auf den Tabellenseiten freigehalten wird, können neue Daten auf den richtigen Seiten eingefügt werden, um die Clusterbildung in Bezug auf den Index beizubehalten. Mit wachsender Menge eingefügter Daten und zunehmender Füllung der Seiten der Tabelle werden Datensätze an das Ende der Tabelle angehängt, so dass die Tabelle allmählich ihre Clusterbildung verliert.

Wenn Sie das Dienstprogramm REORG TABLE ausführen oder eine Sortierung und ein Laden von Daten nach der Erstellung eines Clusterindex durchführen, versucht der Index eine bestimmte Ordnung der Daten zu erhalten, wodurch sich die Werte für die Statistiken CLUSTERRATIO bzw. CLUSTERFACTOR verbessern, die durch das Dienstprogramm RUNSTATS erfasst werden.

Anmerkung: Die Erstellung von Tabellen mit mehrdimensionalem Clustering (MDC) kann die Erforderlichkeit der Reorganisation von Tabellen senken. Bei MDC-Tabellen bleibt die Clusterbildung für die Spalten erhalten, die Sie als Argumente der Klausel ORGANIZE BY DIMENSIONS in der Anweisung CREATE TABLE angeben. Allerdings kann das Dienstprogramm REORGCHK eine Reorganisation einer MDC-Tabelle empfehlen, wenn es ermittelt, dass zu viele ungenutzte Blöcke vorhanden sind oder dass Blöcke zusammengelegt werden sollten.

Zugehörige Konzepte:

- „Snapshot monitor“ in *Systemmonitor Handbuch und Referenz*
- „Überlegungen zu DMS-Einheiten“ in *Systemverwaltung: Konzept*
- „Vom Datenbankmanager verwaltete Tabellenbereiche“ in *Systemverwaltung: Konzept*
- „Indexreorganisation“ auf Seite 366
- „Vom Betriebssystem verwaltete Tabellenbereiche“ in *Systemverwaltung: Konzept*
- „Tabellen- und Indexverwaltung für MDC-Tabellen“ auf Seite 387
- „Tabellen- und Indexverwaltung für Standardtabellen“ auf Seite 384

Zugehörige Tasks:

- „Auswählen einer Methode zur Tabellenreorganisation“ auf Seite 372
- „Feststellen des Zeitpunkts zur Reorganisation von Tabellen“ auf Seite 369

Zugehörige Referenzen:

- „REORG INDEXES/TABLE command using the ADMIN_CMD procedure“ in *Administrative SQL Routines and Views*
- „REORG INDEXES/TABLE command“ in *Command Reference*

Indexreorganisation

Wenn Tabellen durch Löschungen und Einfügungen aktualisiert werden, verschlechtert sich die Indexleistung auf folgende Weisen:

- Aufteilung von Blattseiten
Wenn Blattseiten aufgeteilt (fragmentiert) sind, steigen die E/A-Aufwände, weil mehr Blattseiten gelesen werden müssen, um Tabellenseiten abzurufen.
- Die physische Indexseitenreihenfolge stimmt nicht mehr mit der Reihenfolge der Schlüssel auf diesen Seiten überein. Dies wird als Index mit *schlechter Clusterbildung* bezeichnet.

Wenn Blattseiten schlechte Clusterbildung aufweisen, ist ein sequenzieller Vorab-
lesezugriff nicht effizient, was zu längeren E/A-Wartezeiten führt.

- Der Index bildet mehr als die maximal effiziente Anzahl von Stufen.
In einem solchen Fall sollte der Index reorganisiert werden.

Wenn Sie den Parameter MINPCTUSED bei der Erstellung eines Index definieren, führt der Datenbankserver automatisch Indexblattseiten zusammen, wenn ein Schlüssel gelöscht wird und der freie Speicherplatz weniger als der angegebene Prozentsatz ist. Dieser Vorgang wird als *Online-Indexdefragmentierung* bezeichnet. Um die Indexclusterbildung wiederherzustellen, Speicherplatz freizugeben und die Anzahl von Blattseitenstufen zu verringern, können Sie eine der folgenden Methoden anwenden:

- Löschen und erneutes Erstellen des Index
- Reorganisieren von Indizes online mit Hilfe des Befehls REORG INDEXES
Sie können diese Methode in einer Produktionsumgebung wählen, da sie Benutzern ermöglicht, die Tabelle zu lesen und Daten in sie zu schreiben, während ihre Indizes neu erstellt werden.
- Verwenden des Befehls REORG TABLE mit Optionen, die eine Offline-reorganisation der Tabelle und ihrer Indizes ermöglichen

Online-Indexreorganisation

Wenn Sie den Befehl REORG INDEXES mit der Option ALLOW WRITE ACCESS verwenden, werden alle Indizes für die angegebene Tabelle erneut erstellt, während der Lese- und Schreibzugriff auf die Tabelle möglich ist. Alle Änderungen an der zu Grunde liegenden Tabelle, die während des Reorganisationsprozesses einen Einfluss auf die Indizes haben würden, werden in den DB2-Protokollen aufgezeichnet. Außerdem werden dieselben Änderungen in den internen Speicherpufferbereich gestellt, sofern ein solcher Speicherbereich zur Verfügung steht. Bei der Reorganisation werden die protokollierten Änderungen während des erneuten Erstellens der Indizes zwecks Abgleichs mit der aktuellen Schreibaktivität verarbeitet. Der interne Speicherpufferbereich ist ein ausgewiesener Speicherbereich, der bei Bedarf vom Dienstprogrammzwischenspeicher zugeordnet wird, um die Änderungen an den erstellten bzw. reorganisierten Indizes zu speichern. Durch Verwendung dieses Speicherpufferbereichs können die Änderungen bei der Indexreorganisation verarbeitet werden, indem sie zunächst direkt aus dem Speicher gelesen werden. Gegebenenfalls werden anschließend auch die Protokolle gelesen, jedoch erst zu einem viel späteren Zeitpunkt. Der zugeordnete Speicher wird nach Beendigung der Reorganisation wieder freigegeben. Nach Beendigung der Reorganisation hat der erneut erstellte Index möglicherweise keine perfekte Clusterbildung. Wenn PCTFREE für einen Index angegeben wird, wird der angegebene Prozentsatz an Speicherplatz auf jeder Seite bei der Reorganisation freigehalten.

Für partitionierte Tabellen wird die Online-Indexreorganisation und die Bereinigung einzelner Indizes unterstützt. Zur Reorganisation einzelner Indizes geben Sie den Indexnamen an: REORG INDEX *indexname* FOR TABLE *tabellename*

Die Online-Indexreorganisation im ALLOW WRITE-Modus wird für räumliche Indizes und MDC-Tabellen (MDC - Multi-Dimensional Clustering) nicht unterstützt.

Anmerkung: Die Option CLEANUP ONLY des Befehls REORG INDEXES führt keine vollständige Reorganisation der Indizes durch. Die Option CLEANUP ONLY ALL entfernt die Schlüssel, die als gelöscht mar-

kiert sind und deren Löschung bekanntermaßen festgeschrieben ist. Sie gibt außerdem Seiten frei, auf denen alle Schlüssel als gelöscht markiert sind und diese Löschung bekanntermaßen festgeschrieben wurde. Bei der Freigabe von Seiten werden benachbarte Blattseiten zusammengefügt, wenn dadurch mindestens der Wert von PCTFREE an freiem Speicherplatz auf der zusammengeführten Seite übrig behalten werden kann. Der Wert für PCTFREE ist der Prozentsatz an freiem Speicherbereich, der für den Index bei seiner Erstellung definiert wird. Die Option CLEANUP ONLY PAGES löscht nur Seiten, auf denen alle Schlüssel als gelöscht markiert sind und diese Löschung bekanntermaßen festgeschrieben ist.

Bei der Reorganisation von Indizes für partitionierte Tabellen mit der Option CLEANUP ONLY werden alle Zugriffsebenen unterstützt. Wenn die Option CLEANUP ONLY nicht angegeben wird, ist die Standardzugriffsebene ALLOW NO ACCESS die einzig unterstützte Zugriffsebene.

Für den Befehl REORG INDEXES gelten folgende Voraussetzungen:

- Die Berechtigung SYSADM, SYSMANT, SYSCTRL oder DBADM oder das Zugriffsrecht CONTROL für die Indizes und die Tabelle
- Eine Menge an freiem Speicherplatz in dem Tabellenbereich, in dem die Indizes gespeichert werden, die der aktuellen Größe des Index entspricht
Ziehen Sie in Betracht, die Indizes, die einer Reorganisation unterliegen, in einem großen Tabellenbereich anzulegen, wenn Sie die Anweisung CREATE TABLE ausführen.

- Zusätzlicher Protokollspeicherbereich

Der Befehl REORG INDEXES protokolliert seine Aktivitäten. Infolgedessen kann die Reorganisation fehlschlagen, insbesondere wenn das System ausgelastet ist und andere gleichzeitige Aktivitäten protokolliert werden.

Anmerkung: Wenn der Befehl REORG INDEXES ALL mit der Option ALLOW NO ACCESS fehlschlägt, werden die Indizes als beschädigt markiert, und die Operation wird nicht rückgängig gemacht. Wenn jedoch ein REORG-Befehl mit der Option ALLOW READ ACCESS oder ein REORG-Befehl mit der Option ALLOW WRITE ACCESS fehlschlägt, wird das ursprüngliche Indexobjekt wiederhergestellt.

Zugehörige Konzepte:

- „Verwenden von relationalen Indizes zur Leistungsverbesserung“ auf Seite 29
- „Tipps zur Planung von relationalen Indizes“ auf Seite 33
- „Online-Indexdefragmentierung“ auf Seite 392
- „Tipps zur Leistung von relationalen Indizes“ auf Seite 35
- „Indexbereinigung und Indexpflege“ auf Seite 390

Zugehörige Tasks:

- „Auswählen einer Methode zur Tabellenreorganisation“ auf Seite 372

Zugehörige Referenzen:

- „REORG INDEXES/TABLE command“ in *Command Reference*

Feststellen des Zeitpunkts zur Reorganisation von Tabellen

Normale Datenbankaktivitäten wie zum Beispiel wiederholte Einfüge-, Lösch- und Aktualisierungsoperationen können sich mit der Zeit auf die Organisation der Daten in Ihren Tabellen und Indizes auswirken und die Datenbankleistung beeinträchtigen. Das Dienstprogramm erfasst Statistiken zur Organisation der Daten in Ihren Datenbanktabellen, so dass Sie statistische Informationen über die Tabellengröße und die Datenverteilung, das Clusterverhältnis (Cluster ratio) von Indizes, die Anzahl von Blattseiten (Leaf pages) in Indizes, die Anzahl von Tabellenzeilen, die aus ihren ursprünglichen Seiten überlaufen, sowie die Anzahl gefüllter und leerer Seiten in einer Tabelle ermitteln können. Darüber hinaus können Sie mit Hilfe von RUNSTATS auch Informationen zur Effizienz des Vorablesezugriffs (Prefetching) sammeln.

Die von RUNSTATS erfassten Statistikdaten werden in den Systemkatalogtabellen gespeichert. Diese Informationen können Sie bei der Entscheidung unterstützen, wann Tabellen und Indizes zu reorganisieren sind und welcher Typ von Reorganisation erforderlich ist.

Neben der Verwendung von RUNSTATS zur Erfassung von Informationen zu Ihrer Datenbankorganisation zu einem bestimmten Zeitpunkt empfiehlt es sich auch, RUNSTATS regelmäßig auszuführen, um allgemeine Trends zu ermitteln, die mit Änderungen in der Datenbankleistung verbunden sein können.

Der Befehl REORGCHK liefert ebenfalls Informationen zur Datenorganisation und kann Sie beraten, ob bestimmte Tabellen reorganisiert werden müssen. Allerdings kann die Ausführung spezieller Abfragen auf die Katalogstatistiktabellen in regelmäßigen Intervallen ein Leistungsdatenprotokoll liefern, das es Ihnen ermöglicht, Trends auszumachen, die eine größere Tragweite für die Leistung haben.

Vorgehensweise:

Fragen Sie die Katalogstatistiken ab und überwachen Sie die folgenden Statistiken, um zu bestimmen, ob Sie Tabellen reorganisieren müssen:

1. Überlauf von Zeilen

Fragen Sie die Spalte OVERFLOW in der Tabelle SYSSTAT.TABLES ab, um den OVERFLOW-Wert zu überwachen. Die Werte in dieser Spalte stellen die Anzahl von Zeilen dar, die nicht auf ihre ursprünglichen Seiten passen. Zeilendaten können überlaufen, wenn VARCHAR-Spalten mit Werten aktualisiert werden, die länger als die ersten Werte sind. In diesen Fällen wird an der ursprünglichen Stelle der Zeile ein Zeiger hinterlegt, und der tatsächliche Wert an einer anderen, von dem Zeiger angegebenen Stelle gespeichert. Dadurch kann die Leistung beeinträchtigt werden, da der Datenbankmanager dem Zeiger folgen muss, um den Inhalt der Zeile zu finden. Durch diese beiden Schritte verlängert sich die Verarbeitungszeit und erhöht sich möglicherweise auch die Anzahl der erforderlichen E/A-Operationen.

Eine Reorganisation der Tabellendaten beseitigt die Zeilenüberläufe. Daher erhöht sich mit steigender Anzahl von Überlaufzeilen auch der potenzielle Nutzen einer Reorganisation der Tabellendaten.

2. Abrufstatistiken (Fetch)

Fragen Sie die drei folgenden Spalten in den Katalogstatistiktabellen SYSCAT.INDEXES und SYSSTAT.INDEXES ab, um die Effektivität der Vorablesefunktionen beim Zugriff auf die Tabellen in der Indexreihenfolge zu ermitteln.

Diese Statistiken vermitteln einen Eindruck von der durchschnittlichen Leistung der Vorablesefunktionen für die zugrunde liegende Tabelle.

- In der Spalte `AVERAGE_SEQUENCE_FETCH_PAGES` wird die durchschnittliche Anzahl von Seiten gespeichert, auf die in der Reihenfolge in der Tabelle zugegriffen werden kann. Die Seiten, auf die in der Reihenfolge zugegriffen werden kann, kommen für den Vorablesezugriff in Betracht. Ein kleiner Wert gibt an, dass die Vorablesefunktionen nicht so effizient sind, wie sie sein könnten, weil sie nicht die gesamte Anzahl von Seiten, die durch die Einstellung `PREFETCHSIZE` für den Tabellenbereich definiert sind, einlesen können. Ein hoher Wert zeigt an, dass die Vorablesefunktionen effektiv arbeiten. Für eine Tabelle mit Clusterindex sollte dieser Wert dem Wert für `NPAGES` nahe kommen, der die Anzahl von Seiten angibt, die Zeilen enthalten.
 - In der Spalte `AVERAGE_RANDOM_FETCH_PAGES` wird die durchschnittliche Anzahl von Tabellenseiten gespeichert, auf die beim Abrufen von Tabellenzeilen über den Index zwischen sequenziellen Seitenzugriffen ein wahlfreier Zugriff erfolgt. Die Vorablesefunktionen ignorieren kleine Anzahlen wahlfreier Seiten, wenn die Mehrzahl der Seiten in der Reihenfolge vorliegt, und setzen den Vorablesezugriff bis zur konfigurierten Vorablesegröße fort. Mit zunehmender Unordnung in der Reihenfolge der Tabelle steigt die Anzahl von Seiten, auf die ein wahlfreier Zugriff erfolgen muss. Eine solche zunehmende Unordnung wird in der Regel durch Einfügungen außerhalb der Reihenfolge, das heißt, entweder am Ende der Tabelle oder in Überlaufseiten, verursacht. Dies führt zu Abrufen, die die Abfrageleistung verlangsamen, wenn der Index für den Zugriff auf einen Bereich von Werten verwendet wird.
 - In der Spalte `AVERAGE_SEQUENCE_FETCH_GAP` wird die durchschnittliche Lücke zwischen Tabellenseiten in der Reihenfolge beim Abrufen über den Index gespeichert. Erkennt beim Durchsuchen von Indexseiten stellt jede Lücke die durchschnittliche Anzahl von Seiten dar, die jeweils zwischen Seiten in der Reihenfolge wahlfrei abgerufen werden müssen. Solche Lücken treten auf, wenn auf viele Seiten wahlfrei zugegriffen wird, wodurch die Vorablesefunktionen angehalten werden. Ein hoher Wert zeigt an, dass eine Tabelle nicht gut organisiert oder bezüglich des Index eine niedrige Clusterbildung aufweist.
3. Anzahl von Indexblattseiten, die als gelöscht markierte, jedoch noch nicht gelöschte Satz-IDs (RIDs) enthalten
- In Indizes des Typs 2 werden Satz-IDs in der Regel nicht physisch gelöscht, wenn die Satz-ID als gelöscht markiert wird. Dies bedeutet, dass nützlicher Speicherplatz von diesen logisch gelöschten Satz-IDs belegt sein könnte. Zum Abrufen der Anzahl von Blattseiten, auf denen jede Satz-ID als gelöscht markiert ist, fragen Sie die Spalte `NUM_EMPTY_LEAFS` der Statistiktabelle `SYSSTAT.INDEXES` ab. Für Blattseiten, in denen nicht alle Satz-IDs als gelöscht markiert sind, wird die Gesamtzahl logisch gelöschter Satz-IDs in der Spalte `NUMRIDS_DELETED` gespeichert.
- Verwenden Sie diese Informationen zur Abschätzung, wie viel Speicherplatz durch eine Ausführung des Befehls `REORG INDEXES` mit der Option `CLEANUP ALL` zurückgewonnen werden könnte. Um nur den Speicherplatz in Seiten wieder verfügbar zu machen, in denen alle Satz-IDs als gelöscht markiert sind, führen Sie den Befehl `REORG INDEXES` mit der Option `CLEANUP ONLY PAGES` aus.
4. Statistiken zum Clusterverhältnis und zum Clusterfaktor für Indizes
- In der Spalte `CLUSTERRATIO` der Katalogtabelle `SYSTCAT.INDEXES` wird ein Statistikwert für das Clusterverhältnis gespeichert. Dieser Wert (zwischen 0 und 100) stellt den Grad der Datenclusterbildung bezüglich des Index dar. Wenn Sie

detaillierte Indexstatistikdaten (DETAILED) erfassen, wird ein feinerer Statistikwert zwischen 0 und 1 für den Clusterfaktor in der Spalte CLUSTERFACTOR gespeichert und der Wert der Spalte CLUSTERRATIO ist -1. Nur einer dieser beiden Werte zur Clusterbildung wird im Katalog SYSCAT.INDEXES aufgezeichnet. Zum Vergleich der CLUSTERFACTOR-Werte mit den CLUSTERRATIO-Werten müssen Sie den CLUSTERFACTOR-Wert mit 100 multiplizieren, um einen Prozentwert zu erhalten.

Anmerkung: Im Allgemeinen kann nur einer der Indizes in einer Tabelle einen hohen Grad an Clusterbildung aufweisen.

Indexsuchen, die nicht mit einem reinen Indexzugriff durchgeführt werden, erzielen bei höheren Clusterverhältnissen wahrscheinlich eine bessere Leistung. Ein niedriges Clusterverhältnis führt zu vermehrten Ein-/Ausgabeoperationen für diese Art der Suche, da nach dem ersten Zugriff auf eine Datenseite die Wahrscheinlichkeit geringer ist, dass sich diese Seite immer noch im Pufferpool befindet, wenn der nächste Zugriff auf sie erfolgt. Die Leistung für einen Index ohne Clusterbildung kann möglicherweise durch Erhöhen der Puffergröße verbessert werden.

Wenn Tabellendaten anfangs bezüglich eines bestimmten Index Clusterbildung aufwiesen und die Statistikdaten zur Clusterbildung nun anzeigen, dass in Bezug auf denselben Index nur eine geringe Clusterbildung vorhanden ist, kann es sinnvoll sein, die Tabelle zu reorganisieren, um die Daten bezüglich dieses Index wieder in Clustern anzuordnen.

5. Anzahl der Blattseiten (Leaf pages)

Fragen Sie die Spalte NLEAF in der Tabelle SYSCAT.INDEXES ab, um die Anzahl von Blattseiten zu ermitteln, die von einem Index belegt werden. Diese Anzahl gibt Auskunft darüber, wie viele E/A-Operationen für Indexseiten für ein komplettes Durchsuchen eines Index erforderlich sind.

Ein Index sollte idealerweise möglichst wenig Speicherplatz belegen, um die E/A-Operationen für eine Indexsuche zu reduzieren. Wahlfreie Aktualisierungen können dazu führen, dass Seiten geteilt werden und sich der Index dadurch vergrößert. Wenn Indizes während der Reorganisation einer Tabelle neu erstellt werden, kann jeder Index mit dem minimal erforderlichen Speicherbereich erstellt werden.

Anmerkung: Bei der Erstellung von Indizes werden standardmäßig zehn Prozent des freien Speicherbereichs auf jeder Indexseite nicht belegt. Zur Erhöhung des Betrages an freiem Speicherbereich geben Sie den Parameter PCTFREE bei der Erstellung des Index an. Der Wert für PCTFREE wird bei jeder Reorganisation des Index verwendet. Ein freier Speicherbereich von mehr als zehn Prozent kann die Häufigkeit der Indexreorganisation verringern, weil in dem zusätzlichen Speicherbereich weitere Indexeinfügungen untergebracht werden können.

6. Vergleich von Dateiseiten

Zur Berechnung der Anzahl leerer Seiten in einer Tabelle fragen Sie die Spalten FPAGES und NPAGES der Tabelle SYSCAT.TABLES ab und subtrahieren den NPAGES-Wert vom FPAGES-Wert. In der Spalte FPAGES wird die Gesamtanzahl der verwendeten Seiten, in der Spalte NPAGES die Anzahl von Seiten gespeichert, die Zeilen enthalten. Leere Seiten können auftreten, wenn ganze Bereiche von Zeilen gelöscht werden.

Mit dem Ansteigen der Anzahl leerer Seiten wächst die Notwendigkeit einer Reorganisation für eine Tabelle. Bei der Reorganisation einer Tabelle werden die leeren Seiten aus dem Speicherbereich für eine Tabelle entfernt und der von der

Tabelle belegte Speicherplatz verringert. Da leere Seiten bei einer Tabellensuche auch in den Pufferpool gelesen werden, kann durch das Entfernen ungenutzter Seiten die Leistung einer Tabellensuche verbessert werden.

Wenn die Gesamtanzahl Seiten (FPAGES) in einer Tabelle kleiner oder gleich $\text{NPARTITIONS} * 1 \text{ EXTENTSIZE}$ ist, wird keine Tabellenreorganisation empfohlen. NPARTITIONS ist die Anzahl von Datenpartitionen, wenn es sich um eine partitionierte Tabelle handelt. Ansonsten hat NPARTITIONS den Wert 1. Wird in einer DPF-Datenbank die Anzahl Datenbankpartitionen in einer Datenbankpartitionsgruppe der Tabelle mit eingerechnet, ändert sich die Bedingung in $\text{FPAGES} \leq \text{Anzahl der Datenbankpartitionen in einer Datenbankpartitionsgruppe einer Tabelle} * \text{NPARTITIONS} * 1 \text{ EXTENTSIZE}$.

Zugehörige Konzepte:

- „Katalogstatistiktabellen“ auf Seite 330
- „Indexreorganisation“ auf Seite 366
- „Tabellenreorganisation“ auf Seite 365

Zugehörige Tasks:

- „Erfassen von Katalogstatistiken“ auf Seite 322

Auswählen einer Methode zur Tabellenreorganisation

DB2 stellt zwei Methoden zur Reorganisation von Tabellen bereit: die klassische und die "am Platz" (engl. "in place"). Im Allgemeinen ist die klassische Tabellenreorganisation schneller, sollte jedoch nur verwendet werden, wenn Ihre Anwendungen während der Reorganisation ohne Schreibzugriff auf Tabellen funktionieren. Wenn Ihre Umgebung diese Einschränkung nicht zulässt, kann eine Reorganisation am Platz durchgeführt werden, die zwar langsamer ist, jedoch im Hintergrund stattfinden kann, während der normale Datenzugriff fortgesetzt wird. Machen Sie sich mit den Merkmalen jeder Methode vertraut und entscheiden Sie, welche Methode für Ihre Umgebung geeigneter ist.

Vorgehensweise:

Beachten Sie bei der Auswahl einer Methode zur Tabellenreorganisation die Merkmale der folgenden Methoden:

- **Klassische Tabellenreorganisation**

Diese Methode bietet die schnellste Tabellenreorganisation, insbesondere wenn Sie keine LOB- oder LONG-Daten reorganisieren müssen. Darüber hinaus werden Indizes in perfekter Reihenfolge nach der Reorganisation der Tabelle neu erstellt. Reine Leseanwendungen können auf die Originalkopie der Tabelle außer während der letzten Phasen der Reorganisation zugreifen. Dies sind die Phasen, in denen die Spiegelkopie der Tabelle durch die permanente Tabelle ersetzt und die Indizes neu erstellt werden.

Auf der anderen Seite sind die folgenden möglichen Nachteile zu berücksichtigen:

- Große Speicherplatzanforderungen

Da die klassische Tabellenreorganisation eine Spiegelkopie der Tabelle erstellt, kann sie das Zweifache des Speicherplatzes der ursprünglichen Tabelle beanspruchen. Wenn die reorganisierte Tabelle größer als die ursprüngliche Tabelle ist, kann die Reorganisation mehr als das Doppelte des Platzes der ursprünglichen Tabelle benötigen.

Die Spiegelkopie kann in einem temporären Tabellenbereich erstellt werden, wenn der Tabellenbereich nicht groß genug ist, jedoch erreicht die Ersetzungsphase ihre beste Leistung, wenn sie im gleichen DMS-Tabellenbereich stattfindet. Tabellen in SMS-Tabellenbereichen müssen die Spiegelkopie immer in einem temporären Tabellenbereich speichern.

- Begrenzter Tabellenzugriff

Auch ein reiner Lesezugriff ist auf die ersten Phasen des Reorganisationsprozesses beschränkt.

- Alles-oder-Nichts-Prozess

Wenn die Reorganisation an einem Punkt fehlschlägt, muss sie auf den Knoten, auf denen sie fehlgeschlagen ist, völlig neu gestartet werden.

- Ausführung innerhalb des Controllers der Anwendung, die sie aufruft

Die Reorganisation kann nur durch diese Anwendung bzw. durch einen Benutzer gestoppt werden, der weiß, wie der Prozess zu stoppen ist und die Berechtigung zur Ausführung des Befehls FORCE (erzwungenes Abbrechen) für die Anwendung hat.

Empfehlung: Wählen Sie diese Methode, wenn Sie für die Reorganisation von Tabellen ein Wartungszeitfenster zur Verfügung haben.

- **Tabellenreorganisation am Platz**

Die Am-Platz-Methode arbeitet langsamer und gewährleistet keine perfekt angeordneten Daten, jedoch kann sie Anwendungen den Zugriff auf die Tabelle während der Reorganisation ermöglichen. Darüber hinaus kann eine Tabellenreorganisation am Platz von einem Benutzer angehalten und fortgesetzt werden, der die entsprechende Berechtigung hat, indem er den Schemanamen und den Tabellennamen verwendet.

Anmerkung: Die Tabellenreorganisation am Platz ist nur für Tabellen mit Indizes des Typs 2 und ohne erweiterte Indizes zulässig. Die Tabellenreorganisation am Platz ist für Tabellen mit räumlichen Indizes nicht zulässig.

Berücksichtigen Sie die folgenden Aspekte bei der Abwägung der Vor- und Nachteile:

- Keine perfekte Indexreorganisation

Sie müssen Indizes eventuell später reorganisieren, um die Indexfragmentierung zu verringern und Indexobjektspeicherplatz wieder verfügbar zu machen.

- Längere Ausführungszeit

Falls erforderlich, gibt die Reorganisation am Platz gleichzeitigen Anwendungen Vorrang. Dies bedeutet, dass lang laufende Anweisungen oder RR- und RS-Leseoperationen in lang laufenden Anwendungen den Reorganisationsprozess bremsen können. Eine Reorganisation am Platz läuft in einer OLTP-Umgebung eventuell schneller, in der viele kleine Transaktionen ausgeführt werden.

- Höhere Protokollspeicheranforderungen

Da eine Tabellenreorganisation am Platz ihre Aktivitäten protokolliert, um eine Recovery nach einem unerwarteten Fehler zu ermöglichen, erfordert sie mehr Protokollspeicherplatz als eine klassische Reorganisation.

Es ist möglich, dass eine Reorganisation am Platz einen Protokollspeicher in mehrfacher Größe der reorganisierten Tabelle benötigt. Die Größe des erforderlichen Speicherplatzes hängt von der Anzahl der Zeilen, die versetzt werden, sowie von der Anzahl und Größe der Indizes für die Tabelle ab.

Eine Tabellenreorganisation am Platz reorganisiert nur Tabellendatenobjekte und keine Indizes, Langfeld- oder LOB-Daten.

Empfehlung: Wählen Sie die Tabellenreorganisation am Platz für Umgebungen im 24x7-Betrieb mit minimalen Wartungsfenstern aus.

Detaillierte Informationen zur Ausführung dieser Methoden zur Tabellenreorganisation finden Sie in den Beschreibungen der Syntax des Befehls REORG TABLE.

Überwachen des Fortschritts der Tabellenreorganisation

Informationen über den aktuellen Fortschritt der Tabellenreorganisation werden in die Protokolldatei für Datenbankaktivitäten geschrieben. Die Protokolldatei enthält einen Datensatz für jedes Reorganisationsereignis. Zum Anzeigen dieser Datei führen Sie den Befehl `db2 list history` für die Datenbank aus, in der sich die Tabelle befindet, die reorganisiert wird.

Sie können den Fortschritt der Tabellenreorganisation auch mit Hilfe von Tabellenmomentaufnahmen überwachen. Überwachungsdaten zur Tabellenreorganisation werden unabhängig von der Einstellung des Datenbankmonitorschalters für Tabellen aufgezeichnet.

Wenn ein Fehler auftritt, wird ein SQLCA-Auszug in die Protokolldatei geschrieben. Bei einer Tabellenreorganisation am Platz wird der Status mit PAUSED (angehalten) aufgezeichnet.

Zugehörige Konzepte:

- „Indexreorganisation“ auf Seite 366
- „Tabellenreorganisation“ auf Seite 365

Zugehörige Tasks:

- „Feststellen des Zeitpunkts zur Reorganisation von Tabellen“ auf Seite 369

Zugehörige Referenzen:

- „REORG INDEXES/TABLE command“ in *Command Reference*

Aktivieren der automatischen Reorganisation von Tabellen und Indizes

Über gut organisierte Tabellendaten zu verfügen, ist von kritischer Bedeutung für einen effizienten Datenzugriff und eine optimale Auslastungsleistung. Nach zahlreichen Änderungen an Tabellendaten können sich logisch sequenzielle Daten auf physisch nicht sequenziellen Datenseiten befinden, so dass der Datenbankmanager zusätzliche Leseoperationen ausführen muss, um auf die Daten zuzugreifen. Darüber hinaus sind zusätzliche Leseoperationen auch erforderlich, wenn eine beträchtliche Anzahl von Zeilen gelöscht wurde. Verwenden Sie die automatische Reorganisation von Tabellen, um DB2 zu veranlassen, die Offline-Reorganisation von Tabellen und Indizes zu verwalten, so dass Sie sich nicht darum zu kümmern brauchen, wann und wie die Daten reorganisiert werden. Sie können DB2 sowohl Systemkatalogtabellen als auch Datenbanktabellen reorganisieren lassen.

Vorgehensweise:

Sie können diese Funktion entweder über die Tools der grafischen Benutzerschnittstelle oder über die Befehlszeilenschnittstelle aktivieren.

- Gehen Sie wie folgt vor, um Ihre Datenbank für die automatische Reorganisation über die Tools der grafischen Benutzerschnittstelle zu konfigurieren:
 1. Öffnen Sie den Assistenten **Automatische Verwaltung konfigurieren** entweder über die Steuerzentrale, indem Sie ein Datenbankobjekt mit der rechten Maustaste anklicken, oder über die Diagnosezentrale, indem Sie die Datenbankinstanz, die Sie zur automatischen Reorganisation konfigurieren wollen, mit der rechten Maustaste anklicken. Wählen Sie **Automatische Verwaltung konfigurieren** im Kontextmenü aus.
 2. In diesem Assistenten können Sie die automatische Reorganisation zur Defragmentierung von Daten aktivieren, die Tabellen angeben, die automatisch reorganisiert werden sollen und ein Verwaltungsfenster zur Ausführung des REORG-Dienstprogramms angeben.
- Zur Konfiguration der automatischen Reorganisation Ihrer Datenbank über die Befehlszeilenschnittstelle setzen Sie die folgenden Konfigurationsparameter auf den Wert ON:
 - AUTO_MAINT
 - AUTO_TBL_MAINT
 - AUTO_REORG

Zugehörige Konzepte:

- „Automatische Reorganisation“ in *Systemverwaltung*: Konzept

Verwenden von Snapshot Monitor-Daten zum Überwachen der Reorganisation einer partitionierten Tabelle

Es gibt keine separate Datengruppe, die den Gesamtstatus einer Tabellenreorganisation für eine partitionierte Tabelle angibt. Eine partitionierte Tabelle arbeitet mit einem Datenorganisationsschema, bei dem Tabellendaten auf mehrere Speicherobjekte, die als Datenpartitionen oder Datenbereiche (*RANGE*) bezeichnet werden, entsprechend den Werten einer oder mehrerer Spalten der Tabelle, die den Tabellenpartitionierungsschlüssel bilden, verteilt werden. Sie können jedoch den globalen Status einer Tabellenreorganisation aus den Werten von Elementen in der einzelnen Datenpartitionsdatengruppe ableiten, die gerade reorganisiert wird. Die folgenden Informationen beschreiben einige der nützlichsten Methoden zur Überwachung des globalen Status einer Tabellenreorganisation.

Ermitteln der Anzahl von Datenpartitionen, die reorganisiert werden:

Sie können die Gesamtanzahl der Datenpartitionen, die in einer Tabelle reorganisiert werden, ermitteln, indem Sie die Anzahl von Überwachungsdatenblöcken für Tabellendaten zählen, die denselben Tabellennamen und denselben Schemanamen haben. Dieser Wert gibt die Anzahl der Datenpartitionen an, in denen die Reorganisation gestartet wurde. Die Beispiele 1 und 2 zeigen, dass momentan drei Datenpartitionen reorganisiert werden.

Ermitteln der Datenpartition, die reorganisiert wird:

Sie können die aktuelle Datenpartition, die gegenwärtig reorganisiert wird, aus der Phasenstartzeit der Reorganisation ableiten. Während der Phase der Sortierung, Erstellung und Ersetzung (SORT/BUILD/REPLACE) zeigen die Überwachungsdaten, die der Datenpartition, die gerade reorganisiert wird, entsprechen, die jüngste Phasenstartzeit. Während der Phase der Indexneuerstellung (INDEX_RECREATE) ist die Phasenstartzeit für alle Datenpartitionen die gleiche. In den Bei-

spielen 1 und 2 wird die Phase der Indexneuerstellung (INDEX_RECREATE) angegeben, so dass die Startzeit für alle Datenpartitionen gleich ist.

Ermitteln des Bedarfs einer Indexneuerstellung:

Sie können feststellen, ob eine Indexneuerstellung erforderlich ist, indem Sie den Wert für das Element 'Max. Phase' (Maximum der Reorganisationsphase) ermitteln, das einer der Datenpartitionen entspricht, die reorganisiert werden. Wenn das Element 'Max. Phase' den Wert 3 oder 4 hat, ist eine Indexneuerstellung erforderlich. Die Beispiele 1 und 2 zeigen den Wert 3 für das Element 'Max. Phase', der angibt, dass eine Indexneuerstellung erforderlich ist.

Die folgende Beispielausgabe stammt von einem Server mit drei Knoten, der eine Tabelle mit drei Datenpartitionen enthält:

```
CREATE TABLE sales (c1 INT, c2 INT, c3 INT)
PARTITION BY RANGE (c1)
(PART P1 STARTING FROM (1) ENDING AT (10) IN parttbs,
PART P2 STARTING FROM (11) ENDING AT (20) IN parttbs,
PART P3 STARTING FROM (21) ENDING AT (30) IN parttbs)
DISTRIBUTE BY (c2)
```

Ausgeführte Anweisung:

```
REORG TABLE sales ALLOW NO ACCESS ON ALL DBPARTITIONNUMS
```

Beispiel 1:

```
GET SNAPSHOT FOR TABLES ON DPARTDB GLOBAL
```

Die Ausgabe wurde modifiziert, so dass sie Tabelleninformationen nur für die relevante Tabelle zeigt.

Momentaufnahme einer Tabelle

```
Zeitmarke für erste Datenbankverbindung      = 06/28/2005 13:46:43.061690
Zeitmarke für letzte Zurücksetzung           = 06/28/2005 13:46:47.440046
Zeitmarke für Momentaufnahme                 = 06/28/2005 13:46:50.964033
Datenbankname                               = DPARTDB
Datenbankpfad                               = /work/sales/NODE0000/SQL00001/
Aliasname der Eingabedatenbank               = DPARTDB
Anzahl Tabellen im Zugriff                   = 5
```

```
Tabellenverzeichnis
Tabellenschema                             = NEWTON
Tabellenname                               = SALES
Tabellenart                                 = Benutzer
Datenpartitions-ID                          = 0
Datenobjektseiten                           = 3
Gelesene Zeilen                             = 12
Geschriebene Zeilen                         = 1
Überläufe                                  = 0
Seitenreorganisationen                     = 0
Informationen zur Tabellenreorganisation:
Knotennummer                               = 0
Reorganisationstyp                          =
Wiederherstellen
Tabellenreorganisation
Keinen Zugriff zulassen
Erneute Clustererstellung über Tabellensuche
Nur Daten reorganisieren
Reorganisationsindex                       = 0
Tabellenbereich für Reorg.                  = 3
ID des temporären LOB-Speicherbereichs     = 3
```

Startzeit	= 06/28/2005 13:46:49.816883
Reorganisationsphase	= 3 - Index erneut erstellen
Max. Phase	= 3
Phasenstartzeit	= 06/28/2005 13:46:50.362918
Status	= Beendet
Aktueller Zähler	= 0
Max. Zähler	= 0
Beendigungsstatus	= 0
Endzeit	= 06/28/2005 13:46:50.821244

Informationen zur Tabellenreorganisation:

Knotennummer	= 1
Reorganisationstyp	=
Wiederherstellen	
Tabellenreorganisation	
Keinen Zugriff zulassen	
Erneute Clustererstellung über Tabellensuche	
Nur Daten reorganisieren	
Reorganisationsindex	= 0
Tabellenbereich für Reorg.	= 3
ID des temporären LOB-Speicherbereichs	= 3
Startzeit	= 06/28/2005 13:46:49.822701
Reorganisationsphase	= 3 - Index erneut erstellen
Max. Phase	= 3
Phasenstartzeit	= 06/28/2005 13:46:50.420741
Status	= Beendet
Aktueller Zähler	= 0
Max. Zähler	= 0
Beendigungsstatus	= 0
Endzeit	= 06/28/2005 13:46:50.899543

Informationen zur Tabellenreorganisation:

Knotennummer	= 2
Reorganisationstyp	=
Wiederherstellen	
Tabellenreorganisation	
Keinen Zugriff zulassen	
Erneute Clustererstellung über Tabellensuche	
Nur Daten reorganisieren	
Reorganisationsindex	= 0
Tabellenbereich für Reorg.	= 3
ID des temporären LOB-Speicherbereichs	= 3
Startzeit	= 06/28/2005 13:46:49.814813
Reorganisationsphase	= 3 - Index erneut erstellen
Max. Phase	= 3
Phasenstartzeit	= 06/28/2005 13:46:50.344277
Status	= Beendet
Aktueller Zähler	= 0
Max. Zähler	= 0
Beendigungsstatus	= 0
Endzeit	= 06/28/2005 13:46:50.803619

Tabellenschema	= NEWTON
Tabellenname	= SALES
Tabellenart	= Benutzer
Datenpartitions-ID	= 1
Datenobjektseiten	= 3
Gelesene Zeilen	= 8
Geschriebene Zeilen	= 1
Überläufe	= 0
Seitenreorganisationen	= 0
Informationen zur Tabellenreorganisation:	
Knotennummer	= 0
Reorganisationstyp	=
Wiederherstellen	


```

Tabellenreorganisation
Keinen Zugriff zulassen
Erneute Clustererstellung über Tabellensuche
Nur Daten reorganisieren
Reorganisationsindex           = 0
Tabellenbereich für Reorg.     = 3
ID des temporären LOB-Speicherbereichs = 3
Startzeit                      = 06/28/2005 13:46:50.014617
Reorganisationsphase          = 3 - Index erneut erstellen
Max. Phase                     = 3
Phasenstartzeit               = 06/28/2005 13:46:50.362918
Status                         = Beendet
Aktueller Zähler              = 0
Max. Zähler                   = 0
Beendigungsstatus             = 0
Endzeit                       = 06/28/2005 13:46:50.821244

```

Informationen zur Tabellenreorganisation:

```

Knotennummer                   = 1
Reorganisationstyp             =
    Wiederherstellen
    Tabellenreorganisation
    Keinen Zugriff zulassen
    Erneute Clustererstellung über Tabellensuche
    Nur Daten reorganisieren
Reorganisationsindex           = 0
Tabellenbereich für Reorg.     = 3
ID des temporären LOB-Speicherbereichs = 3
Startzeit                      = 06/28/2005 13:46:50.026278
Reorganisationsphase          = 3 - Index erneut erstellen
Max. Phase                     = 3
Phasenstartzeit               = 06/28/2005 13:46:50.420741
Status                         = Beendet
Aktueller Zähler              = 0
Max. Zähler                   = 0
Beendigungsstatus             = 0
Endzeit                       = 06/28/2005 13:46:50.899543

```

Informationen zur Tabellenreorganisation:

```

Knotennummer                   = 2
Reorganisationstyp             =
    Wiederherstellen
    Tabellenreorganisation
    Keinen Zugriff zulassen
    Erneute Clustererstellung über Tabellensuche
    Nur Daten reorganisieren
Reorganisationsindex           = 0
Tabellenbereich für Reorg.     = 3
ID des temporären LOB-Speicherbereichs = 3
Startzeit                      = 06/28/2005 13:46:50.006392
Reorganisationsphase          = 3 - Index erneut erstellen
Max. Phase                     = 3
Phasenstartzeit               = 06/28/2005 13:46:50.344277
Status                         = Beendet
Aktueller Zähler              = 0
Max. Zähler                   = 0
Beendigungsstatus             = 0
Endzeit                       = 06/28/2005 13:46:50.803619

```

```

Tabellenschema                 = NEWTON
Tabellenname                   = SALES
Tabellenart                    = Benutzer
Datenpartitions-ID             = 2
Datenobjektseiten              = 3
Gelesene Zeilen                = 4
Geschriebene Zeilen            = 1

```

```

Überläufe                                     = 0
Seitenreorganisationen                       = 0
Informationen zur Tabellenreorganisation:
  Knotennummer                               = 0
  Reorganisationstyp                         =
    Wiederherstellen
    Tabellenreorganisation
    Keinen Zugriff zulassen
    Erneute Clustererstellung über Tabellensuche
    Nur Daten reorganisieren
  Reorganisationsindex                       = 0
  Tabellenbereich für Reorg.                 = 3
ID des temporären LOB-Speicherbereichs       = 3
  Startzeit                                 = 06/28/2005 13:46:50.199971
  Reorganisationsphase                       = 3 - Index erneut erstellen
  Max. Phase                                = 3
  Phasenstartzeit                           = 06/28/2005 13:46:50.362918
  Status                                     = Beendet
  Aktueller Zähler                           = 0
  Max. Zähler                               = 0
  Beendigungsstatus                         = 0
  Endzeit                                   = 06/28/2005 13:46:50.821244

Informationen zur Tabellenreorganisation:
  Knotennummer                               = 1
  Reorganisationstyp                         =
    Wiederherstellen
    Tabellenreorganisation
    Keinen Zugriff zulassen
    Erneute Clustererstellung über Tabellensuche
    Nur Daten reorganisieren
  Reorganisationsindex                       = 0
  Tabellenbereich für Reorg.                 = 3
ID des temporären LOB-Speicherbereichs       = 3
  Startzeit                                 = 06/28/2005 13:46:50.223742
  Reorganisationsphase                       = 3 - Index erneut erstellen
  Max. Phase                                = 3
  Phasenstartzeit                           = 06/28/2005 13:46:50.420741
  Status                                     = Beendet
  Aktueller Zähler                           = 0
  Max. Zähler                               = 0
  Beendigungsstatus                         = 0
  Endzeit                                   = 06/28/2005 13:46:50.899543

Informationen zur Tabellenreorganisation:
  Knotennummer                               = 2
  Reorganisationstyp                         =
    Wiederherstellen
    Tabellenreorganisation
    Keinen Zugriff zulassen
    Erneute Clustererstellung über Tabellensuche
    Nur Daten reorganisieren
  Reorganisationsindex                       = 0
  Tabellenbereich für Reorg.                 = 3
ID des temporären LOB-Speicherbereichs       = 3
  Startzeit                                 = 06/28/2005 13:46:50.179922
  Reorganisationsphase                       = 3 - Index erneut erstellen
  Max. Phase                                = 3
  Phasenstartzeit                           = 06/28/2005 13:46:50.344277
  Status                                     = Beendet
  Aktueller Zähler                           = 0
  Max. Zähler                               = 0
  Beendigungsstatus                         = 0
  Endzeit                                   = 06/28/2005 13:46:50.803619

```

Beispiel 2:

GET SNAPSHOT FOR TABLES ON DPARTDB AT DBPARTITIONNUM 2

Die Ausgabe wurde modifiziert, so dass sie Tabelleninformationen nur für die relevante Tabelle zeigt.

Momentaufnahme einer Tabelle

Zeitmarke für erste Datenbankverbindung	= 06/28/2005 13:46:43.617833
Zeitmarke für letzte Zurücksetzung	=
Zeitmarke für Momentaufnahme	= 06/28/2005 13:46:51.016787
Datenbankname	= DPARTDB
Datenbankpfad	= /work/sales/NODE0000/SQL00001/
Aliasname der Eingabedatenbank	= DPARTDB
Anzahl Tabellen im Zugriff	= 3

Tabellenverzeichnis

Tabellenschema	= NEWTON
Tabellenname	= SALES
Tabellenart	= Benutzer
Datenpartitions-ID	= 0
Datenobjektseiten	= 1
Gelesene Zeilen	= 0
Geschriebene Zeilen	= 0
Überläufe	= 0
Seitenreorganisationen	= 0
Informationen zur Tabellenreorganisation:	
Knotennummer	= 2
Reorganisationstyp	=
Wiederherstellen	
Tabellenreorganisation	
Keinen Zugriff zulassen	
Erneute Clustererstellung über Tabellensuche	
Nur Daten reorganisieren	
Reorganisationsindex	= 0
Tabellenbereich für Reorg.	= 3
ID des temporären LOB-Speicherbereichs	= 3
Startzeit	= 06/28/2005 13:46:49.814813
Reorganisationsphase	= 3 - Index erneut erstellen
Max. Phase	= 3
Phasenstartzeit	= 06/28/2005 13:46:50.344277
Status	= Beendet
Aktueller Zähler	= 0
Max. Zähler	= 0
Beendigungsstatus	= 0
Endzeit	= 06/28/2005 13:46:50.803619

Tabellenschema	= NEWTON
Tabellenname	= SALES
Tabellenart	= Benutzer
Datenpartitions-ID	= 1
Datenobjektseiten	= 1
Gelesene Zeilen	= 0
Geschriebene Zeilen	= 0
Überläufe	= 0
Seitenreorganisationen	= 0
Informationen zur Tabellenreorganisation:	
Knotennummer	= 2
Reorganisationstyp	=
Wiederherstellen	
Tabellenreorganisation	
Keinen Zugriff zulassen	
Erneute Clustererstellung über Tabellensuche	
Nur Daten reorganisieren	
Reorganisationsindex	= 0

```

Tabellenbereich für Reorg.          = 3
ID des temporären LOB-Speicherbereichs = 3
Startzeit                          = 06/28/2005 13:46:50.006392
Reorganisationsphase              = 3 - Index erneut erstellen
Max. Phase                        = 3
Phasenstartzeit                   = 06/28/2005 13:46:50.344277
Status                            = Beendet
Aktueller Zähler                  = 0
Max. Zähler                       = 0
Beendigungsstatus                 = 0
Endzeit                           = 06/28/2005 13:46:50.803619

```

```

Tabellenschema                    = NEWTON
Tabellenname                      = SALES
Tabellenart                       = Benutzer
Datenpartitions-ID                = 2
Datenobjektseiten                 = 1
Gelesene Zeilen                   = 4
Geschriebene Zeilen               = 1
Überläufe                        = 0
Seitenreorganisationen            = 0
Informationen zur Tabellenreorganisation:
  Knotennummer                    = 2
  Reorganisationstyp              =
    Wiederherstellen
    Tabellenreorganisation
    Keinen Zugriff zulassen
    Erneute Clustererstellung über Tabellensuche
    Nur Daten reorganisieren
  Reorganisationsindex            = 0
  Tabellenbereich für Reorg.      = 3
ID des temporären LOB-Speicherbereichs = 3
Startzeit                          = 06/28/2005 13:46:50.179922
Reorganisationsphase              = 3 - Index erneut erstellen
Max. Phase                        = 3
Phasenstartzeit                   = 06/28/2005 13:46:50.344277
Status                            = Beendet
Aktueller Zähler                  = 0
Max. Zähler                       = 0
Beendigungsstatus                 = 0
Endzeit                           = 06/28/2005 13:46:50.803619

```

Beispiel 3:

```
SELECT * FROM SYSIBMADM.SNAPLOCK WHERE tabname = 'SALES';
```

Die Ausgabe wurde modifiziert, so dass sie nur einen Teil der Tabelleninformationen für die relevante Tabelle zeigt.

...	TBSP_NAME	TABNAME	LOCK_OBJECT_TYPE	LOCK_MODE	LOCK_STATUS	...
...	PARTTBS	SALES	ROW_LOCK	X	GRNT	...
...	-	SALES	TABLE_LOCK	IX	GRNT	...
...	PARTTBS	SALES	TABLE_PART_LOCK	IX	GRNT	...
...	PARTTBS	SALES	ROW_LOCK	X	GRNT	...
...	-	SALES	TABLE_LOCK	IX	GRNT	...
...	PARTTBS	SALES	TABLE_PART_LOCK	IX	GRNT	...
...	PARTTBS	SALES	ROW_LOCK	X	GRNT	...
...	-	SALES	TABLE_LOCK	IX	GRNT	...
...	PARTTBS	SALES	TABLE_PART_LOCK	IX	GRNT	...

9 Satz/Sätze ausgewählt.

Ausgabe aus dieser Abfrage (Fortsetzung).

...	LOCK_ESCALATION	LOCK_ATTRIBUTES	DATA_PARTITION_ID	DBPARTITIONNUM
...	0	INSERT	2	2
...	0	NONE	-	2
...	0	NONE	2	2
...	0	INSERT	0	0
...	0	NONE	-	0
...	0	NONE	0	0
...	0	INSERT	1	1
...	0	NONE	-	1
...	0	NONE	1	1

Beispiel 4:

```
SELECT * FROM SYSIBMADM.SNAPTAB WHERE tabname = 'SALES';
```

Die Ausgabe wurde modifiziert, so dass sie nur einen Teil der Tabelleninformationen für die relevante Tabelle zeigt.

...	TABSCHEMA	TABNAME	TAB_FILE_ID	TAB_TYPE	DATA_OBJECT_PAGES	ROWS_WRITTEN	...
...	NEWTON	SALES	2	USER_TABLE	1	1	...
...	NEWTON	SALES	4	USER_TABLE	1	1	...
...	NEWTON	SALES	3	USER_TABLE	1	1	...

3 Satz/Sätze ausgewählt.

Ausgabe aus dieser Abfrage (Fortsetzung).

...	OVERFLOW_ACCESSES	PAGE_REORGS	DBPARTITIONNUM	TBSP_ID	DATA_PARTITION_ID
...	0	0	0	3	0
...	0	0	2	3	2
...	0	0	1	3	1

Beispiel 5:

```
SELECT * FROM SYSIBMADM.SNAPTAB_REORG WHERE tabname = 'SALES';;
```

Die Ausgabe wurde modifiziert, so dass sie nur einen Teil der Tabelleninformationen für die relevante Tabelle zeigt.

REORG_PHASE	REORG_MAX_PHASE	REORG_TYPE	...
INDEX_RECREATE	3	RECLAIM+OFFLINE+ALLOW_NONE+TABLESCAN+DATAONLY	...
INDEX_RECREATE	3	RECLAIM+OFFLINE+ALLOW_NONE+TABLESCAN+DATAONLY	...
INDEX_RECREATE	3	RECLAIM+OFFLINE+ALLOW_NONE+TABLESCAN+DATAONLY	...
INDEX_RECREATE	3	RECLAIM+OFFLINE+ALLOW_NONE+TABLESCAN+DATAONLY	...
INDEX_RECREATE	3	RECLAIM+OFFLINE+ALLOW_NONE+TABLESCAN+DATAONLY	...
INDEX_RECREATE	3	RECLAIM+OFFLINE+ALLOW_NONE+TABLESCAN+DATAONLY	...
INDEX_RECREATE	3	RECLAIM+OFFLINE+ALLOW_NONE+TABLESCAN+DATAONLY	...
INDEX_RECREATE	3	RECLAIM+OFFLINE+ALLOW_NONE+TABLESCAN+DATAONLY	...

9 Satz/Sätze ausgewählt.

Ausgabe aus dieser Abfrage (Fortsetzung).

... REORG_STATUS	REORG_TBSPC_ID	DBPARTITIONNUM	DATA_PARTITION_ID
... COMPLETED	3	2	0
... COMPLETED	3	2	1
... COMPLETED	3	2	2
... COMPLETED	3	1	0
... COMPLETED	3	1	1
... COMPLETED	3	1	2
... COMPLETED	3	0	0
... COMPLETED	3	0	1
... COMPLETED	3	0	2

Zugehörige Konzepte:

- „Datenpartitionen“ in *Systemverwaltung: Konzept*
- „Partitionierte Tabellen“ in *Systemverwaltung: Konzept*
- „Snapshot monitor“ in *Systemmonitor Handbuch und Referenz*

Zugehörige Referenzen:

- „data_partition_id - Data Partition Identifier monitor element“ in *Systemmonitor Handbuch und Referenz*
- „GET SNAPSHOT command“ in *Command Reference*
- „reorg_max_phase - Maximum Reorganize Phase monitor element“ in *Systemmonitor Handbuch und Referenz*
- „reorg_phase_start - Reorganize Phase Start Time monitor element“ in *Systemmonitor Handbuch und Referenz*
- „Snapshot monitor interface mappings to logical data groups“ in *Systemmonitor Handbuch und Referenz*
- „Snapshot monitor logical data groups and monitor elements“ in *Systemmonitor Handbuch und Referenz*
- „Snapshot monitor sample output“ in *Systemmonitor Handbuch und Referenz*
- „SNAPTAB administrative view and SNAP_GET_TAB_V91 table function – Retrieve table logical data group snapshot information“ in *Administrative SQL Routines and Views*
- „SNAPTAB_REORG administrative view and SNAP_GET_TAB_REORG table function – Retrieve table reorganization snapshot information“ in *Administrative SQL Routines and Views*

Tabellen- und Indexverwaltung für Standardtabellen

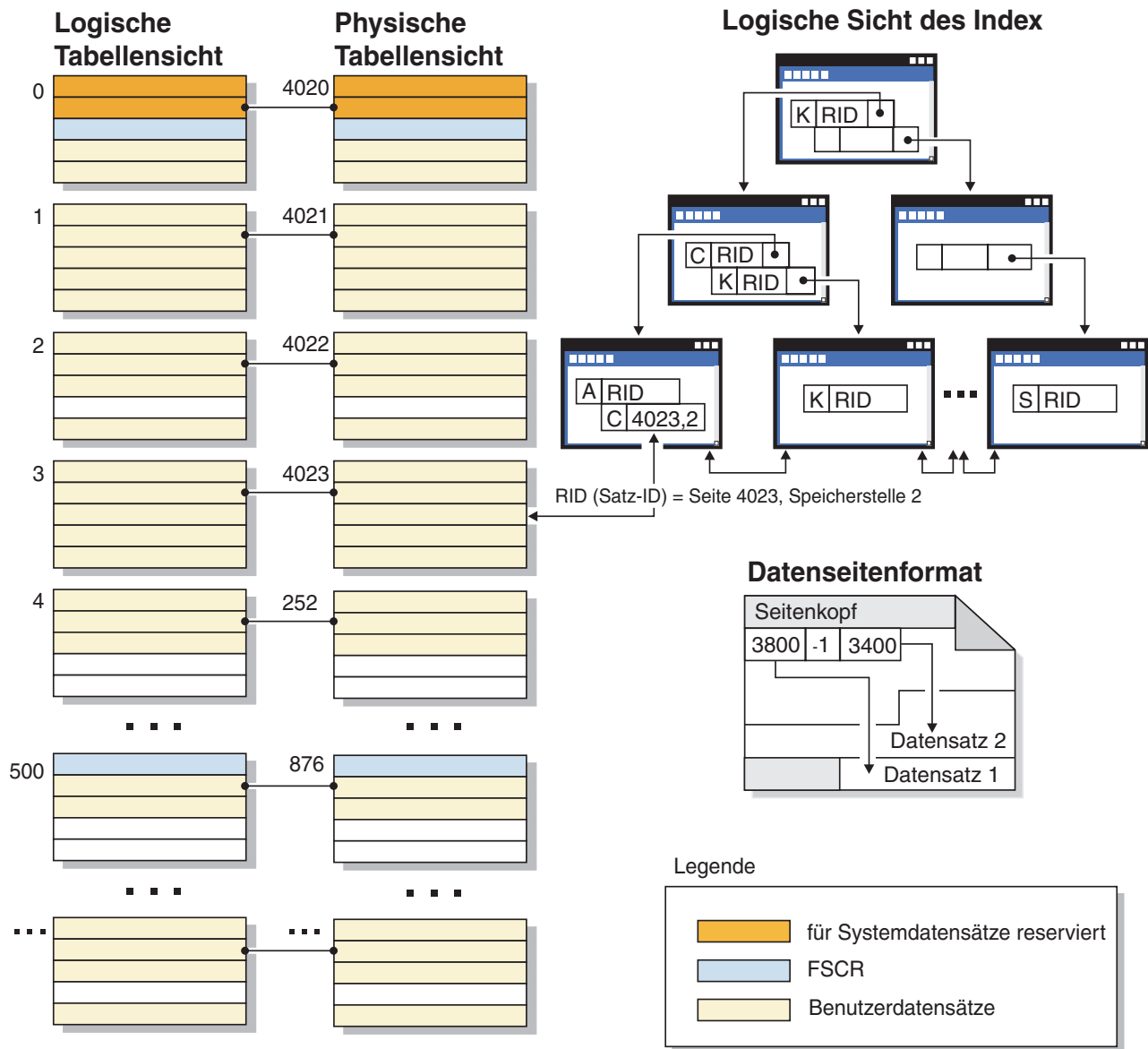


Abbildung 30. Logische Tabellen-, Datensatz und Indexstruktur für Standardtabellen

In Standardtabellen werden Tabellendaten logisch in Form einer Liste von Daten-seiten organisiert. Diese Daten-seiten werden logisch auf der Grundlage der EXTENTSIZE-Größe des Tabellenbereichs zu Gruppen zusammengefasst. Wenn die EXTENTSIZE-Größe beispielsweise vier beträgt, sind die Seiten null bis drei Teil des ersten EXTENTSIZE-Speicherbereichs, die Seiten vier bis sieben sind Teil des zweiten usw.

Die Zahl der Datensätze, die in den einzelnen Daten-seiten enthalten sind, kann je nach Größe der Daten-seite und Größe der Datensätze variieren. Maximal können 255 Datensätze auf einer Seite untergebracht werden. Die meisten Seiten enthalten nur Benutzerdatensätze. Eine kleine Zahl von Seiten enthält jedoch besondere interne Datensätze, die von DB2 zur Verwaltung der Tabelle verwendet werden. Auf jeder 500sten Seite befindet sich beispielsweise ein FSCR-Datensatz (Free Space

Control Record, Steuersatz für freien Speicherbereich). Diese Datensätze enthalten eine Zuordnung des freien Speicherbereichs für neue Datensätze auf jeder der folgenden 500 Datenseiten (bis zum nächsten FSCR-Datensatz). Dieser verfügbare freie Speicherbereich wird verwendet, wenn Datensätze in die Tabelle eingefügt werden.

Logisch werden Indexseiten als B-Baumstruktur organisiert, durch die Datensätze in der Tabelle, die einen angegebenen Schlüsselwert besitzen, auf effiziente Weise lokalisiert werden können. Die Zahl der Entitäten auf einer Indexseite ist nicht festgelegt, sondern hängt von der Größe des Schlüssels ab. Bei Tabellen in DMS-Tabellenbereichen verwenden RIDs (Record Identifiers, Satz-IDs) auf den Indexseiten Seitennummern, die nicht zum Objekt, sondern zum Tabellenbereich relativ sind. Dadurch kann bei einer **Indexsuche** direkt auf die Datenseiten zugegriffen werden, ohne dass eine Speicherbereichsmaske (EMP) für die Zuordnung erforderlich ist.

Jede Datenseite besitzt dasselbe Format. Alle Datenseiten beginnen mit einem Seitenkopf. Nach dem Seitenkopf folgt ein Speicherstellenverzeichnis. Jeder Eintrag im Speicherstellenverzeichnis entspricht einem anderen Datensatz auf der Seite. Der Eintrag selbst ist die relative Byteadresse auf der Datenseite, an der der Datensatz beginnt. Einträge mit minus eins (-1) entsprechen gelöschten Datensätzen.

Satz-IDs und Seiten

Bei Satz-IDs (Record Identifier, RID) handelt es sich um eine Seitennummer gefolgt von einer Slotnummer. Indexdatensätze des Typs 2 enthalten ein zusätzliches Feld mit der Bezeichnung ridFlag. Die ridFlag-Markierung speichert Informationen über den Status von Schlüssel im Index, zum Beispiel ob dieser Schlüssel als gelöscht markiert wurde. Wenn der Index zur Identifizierung einer Satz-ID verwendet wird, wird diese Satz-ID dazu verwendet, zur richtigen Datenseite und Speicherstellennummer auf dieser Seite zu gelangen. Wenn dem Datensatz eine Satz-ID zugeordnet ist, wird diese erst wieder bei einer Reorganisation der Tabelle geändert.

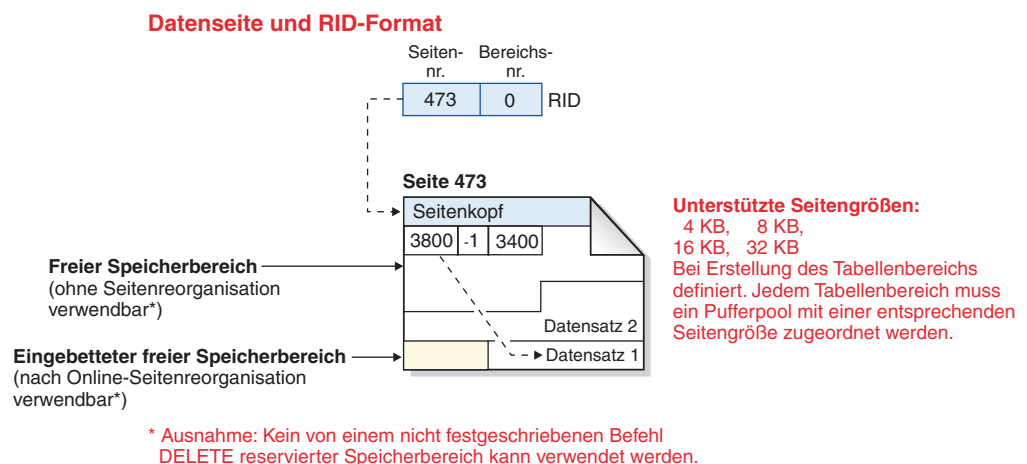


Abbildung 31. Format der Datenseite und der Satz-ID (RID)

Wenn eine Tabellenseite reorganisiert wird, wird eingebetteter freier Speicherbereich, der nach dem physischen Löschen eines Datensatzes auf der Seite verbleibt, in verwendbaren freien Speicherbereich umgewandelt. Satz-IDs werden durch Verschieben von Datensätzen auf einer Datenseite erneut definiert, damit der verwendbare freie Speicherbereich genutzt werden kann.

DB2 unterstützt verschiedene Seitengrößen. Verwenden Sie größere Seiten für Auslastungen, bei denen auf Zeilen normalerweise sequenziell zugegriffen wird. Sequenzieller Zugriff wird beispielsweise für Anwendungen zur Entscheidungshilfe oder in Fällen verwendet, in denen ein starker Gebrauch von temporären Tabellen gemacht wird. Verwenden Sie geringere Seitengrößen für Auslastungen, deren Zugriff normalerweise wahlfrei erfolgt. Wahlfreier Zugriff wird beispielsweise in OLTP-Umgebungen verwendet.

Indexverwaltung in Standardtabellen

DB2-Indizes verwenden eine optimierte B-Baumstrukturimplementierung auf der Basis einer effizienten Indexverwaltungsmethode mit einem hohen Grad des gemeinsamen Zugriffs und einer im Voraus schreibenden Protokollierung (Write Ahead Logging).

Die optimierte B-Baumstrukturimplementierung verfügt über bidirektionale Zeiger auf den Blattseiten, mit denen ein einzelner Index sowohl vorwärts als auch rückwärts gerichtete Suchoperationen unterstützen kann. Indexseiten werden normalerweise in der Mitte geteilt, wobei die HIGHKEY-Seite eine Ausnahme bildet, bei der eine 90/10-Teilung erfolgt. Das heißt, dass die hohen zehn Prozent der Indexschlüssel auf der neuen Seite Platz finden. Diese Art der Indexseitenteilung ist bei Auslastungen nützlich, bei denen oft INSERT-Anforderungen mit neuen HIGHKEY-Werten durchgeführt werden.

Seit Version 8.1 arbeitet DB2 mit Indizes des Typs 2. Wenn Sie eine Migration von früheren Versionen von DB2 durchführen, werden sowohl Indizes des Typs 1 als auch Indizes des Typs 2 verwendet, bis Sie die Indizes reorganisieren oder andere Aktionen durchführen, die Indizes des Typs 1 in Indizes des Typs 2 konvertieren. Der Indextyp bestimmt, wie gelöschte Schlüssel physisch von den Indexseiten entfernt werden.

- Für Indizes des Typs 1 werden Schlüssel von den Indexseiten beim Löschen der Schlüssel entfernt und Indexseiten freigegeben, wenn der letzte Indexschlüssel von der Seite entfernt ist.
- Für Indizes des Typs 2 werden Indexschlüssel beim Löschen des Schlüssels nur dann von der Seite entfernt, wenn eine exklusive Sperre (Modus X) für die Tabelle aktiviert ist. Wenn Schlüssel nicht sofort entfernt werden können, werden sie als gelöscht markiert und später entfernt. Weitere Informationen enthält der Abschnitt zu Indizes des Typs 2.

Wenn Sie die Online-Indexdefragmentierung aktiviert haben, indem Sie bei der Erstellung des Index die Klausel MINPCTUSED auf einen höheren Wert als null gesetzt haben, können Blattseiten online zusammengefügt werden. Der von Ihnen angegebene Wert ist der Schwellenwert für den minimalen Prozentsatz an Speicherplatz, der auf den Indexblattseiten genutzt wird. Wenn ein Schlüssel von einer Indexseite entfernt wurde und der Prozentsatz an Speicherplatz auf der Seite bei oder unter dem angegebenen Wert liegt, versucht der Datenbankmanager die verbleibenden Schlüssel mit denen einer benachbarten Seite zu einer Seite zusammenzufügen. Wenn genügend Platz vorhanden ist, wird die Operation zum Zusammenfügen ausgeführt und eine Indexseite gelöscht. Die Online-Indexdefragmentierung kann die Wiederverwendung von Speicherbereich verbessern. Wenn jedoch der Wert für MINPCTUSED zu hoch ist, wird mehr Zeit auf Versuche verwendet, Seiten zusammenzufügen, die jedoch mit geringerer Wahrscheinlichkeit erfolgreich sind. Der empfohlene Wert für diese Klausel ist 50 % oder weniger.

Anmerkung: Da die Onlinedefragmentierung nur stattfindet, wenn Schlüssel von einer Indexseite entfernt werden, findet sie in einem Index des Typs 2 nicht statt, wenn Schlüssel nur als gelöscht markiert, jedoch nicht physisch von der Seite entfernt wurden.

Die Klausel INCLUDE der Anweisung CREATE INDEX ermöglicht das Einfügen einer angegebenen Spalte bzw. von Spalten auf den Indexseiten zusätzlich zu den Schlüsselspalten. Dadurch kann die Zahl der Abfragen steigen, bei denen ein reiner Indexzugriff möglich ist. Gleichzeitig können jedoch die Anforderungen an Indexspeicherplatz und möglicherweise auch der Verwaltungsaufwand für den betreffenden Index steigen, wenn die eingefügten Spalten häufig aktualisiert werden. Der Verwaltungsaufwand zur Aktualisierung von INCLUDE-Spalten ist geringer als der zur Aktualisierung von Schlüsselspalten, jedoch höher als der zur Aktualisierung von Spalten, die nicht im Index vertreten sind. Das Ordnen der Index-Baumstruktur geschieht nur unter Verwendung der Schlüsselspalten, nicht der eingeschlossenen Spalten.

Zugehörige Konzepte:

- „Indexbereinigung und Indexpflege“ auf Seite 390
- „Verbessern der Einfügeleistung“ auf Seite 150
- „Tabellen- und Indexverwaltung für MDC-Tabellen“ auf Seite 387
- „Entwerfen von Tabellen mit mehrdimensionalem Clustering (MDC)“ in *Systemverwaltung: Konzept*
- „Überlegungen zur Auswahl von Tabellenbereichen für Tabellen“ in *Systemverwaltung: Konzept*
- „Erstellung, Platzierung und Verwendung von Tabellen mit mehrdimensionalem Clustering (MDC)“ in *Systemverwaltung: Konzept*
- „Speicherbedarf für Datenbankobjekte“ in *Systemverwaltung: Konzept*

Zugehörige Referenzen:

- „DB2 log records“ in *Administrative API Reference*

Tabellen- und Indexverwaltung für MDC-Tabellen

Die Tabellen- und Indexorganisation für Tabellen mit mehrdimensionalem Clustering (MDC) basiert auf den gleichen logischen Strukturen wie die Organisation von Standardtabellen. Ebenso wie Standardtabellen werden MDC-Tabellen in Seiten organisiert, die Datenzeilen enthalten, die wiederum in Spalten unterteilt sind. Die Zeilen jeder Seite werden durch Zeilen-IDs (RIDs) gekennzeichnet. Darüber hinaus werden die Seiten von MDC-Tabellen in EXTENTSIZE große Blöcke gruppiert. In der folgenden Abbildung, die eine Tabelle mit dem EXTENTSIZE-Wert 4 darstellt, bilden die ersten vier Seiten mit den Nummern 0 bis 3 den ersten Block in der Tabelle. Die nächste Gruppe von Seiten mit den Nummern 4 bis 7 bildet den zweiten Block der Tabelle.

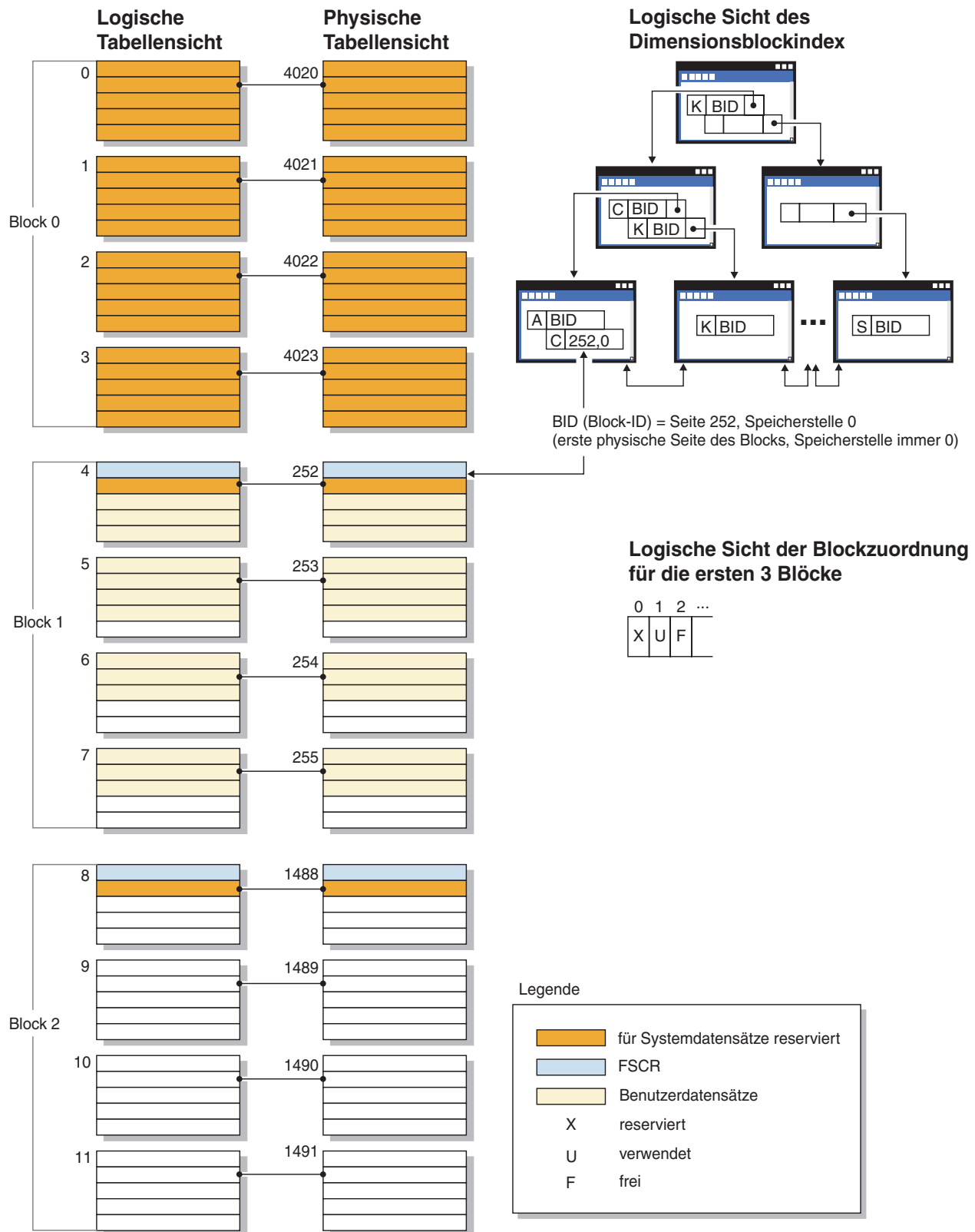


Abbildung 32. Logische Tabellen-, Datensatz- und Indexstruktur für MDC-Tabellen

Der Block enthält besondere interne Datensätze, die von DB2 zur Verwaltung der Tabelle verwendet werden. Dazu gehören auch die FSCR-Sätze, d. h., die Steuerdatensätze für freie Speicherbereiche (FSCR - Free-Space Control Record). In den

nachfolgenden Blöcken enthält jeweils die erste Seite die FSCR-Sätze. Ein FSCR-Satz ordnet den freien Speicher für neue Datensätze zu, der auf den jeweiligen Seiten im Block vorhanden ist. Dieser verfügbare freie Speicherbereich wird verwendet, wenn Datensätze in die Tabelle eingefügt werden.

Wie am Namen ersichtlich, ordnen MDC-Tabellen ihre Daten in mehr als einer Dimension in so genannten Clustern (d. h. in Datengruppen) an. Jede Dimension wird durch eine Spalte bzw. eine Gruppe von Spalten bestimmt, die Sie in der Klausel ORGANIZE BY DIMENSIONS der Anweisung CREATE TABLE angeben. Bei der Erstellung einer MDC-Tabelle werden die beiden folgenden Arten von Indizes automatisch erstellt:

- Ein Dimensionsblockindex, der Zeiger auf jeden belegten Block für eine einzelne Dimension enthält.
- Einen zusammengesetzten Blockindex, der alle Dimensionsschlüsselspalten enthält. Der zusammengesetzte Blockindex dient zur Erhaltung der Clusterbildung bei Einfüge- und Aktualisierungsaktivitäten.

Das Optimierungsprogramm zieht Zugriffspläne unter Nutzung von Dimensionsblockindizes in Betracht, wenn es den effizientesten Zugriffsplan für eine bestimmte Abfrage ermittelt. Wenn Abfragen Vergleichselemente für Dimensionswerte enthalten, kann das Optimierungsprogramm den Dimensionsblockindex verwenden, um die (EXTENTSIZE großen) Speicherbereiche, die diese Werte enthalten, zu ermitteln und aus diesen Daten abzurufen. Da diese Speicherbereiche physisch aufeinander folgende Seiten auf der Platte darstellen, führt dieses Verfahren zu einer effizienteren Leistung und minimiert den E/A-Aufwand.

Darüber hinaus können Sie auch spezielle Zeilen-ID-Indizes (RID-Indizes) erstellen, wenn die Analyse der Datenzugriffspläne darauf hinweist, dass solche Indizes die Abfrageleistung verbessern würden.

Neben den Dimensionsblockindizes und dem zusammengesetzten Blockindex pflegen MDC-Tabellen eine Blockzuordnung (engl. block map), die eine Bitmaske enthält, die den Verfügbarkeitsstatus jedes Blocks angibt. Die folgenden Attribute werden in der Liste der Bitmaske codiert:

- X (reserviert): Der erste Block enthält nur Systeminformationen für die Tabelle.
- U (in Gebrauch): Dieser Block wird verwendet und ist einem Dimensionsblockindex zugeordnet.
- L (geladen): In diesen Block wurden durch eine aktuelle Ladeoperation Daten geladen.
- C (Integritätsbedingung prüfen): Dieser Block wurde durch die Ladeoperation so eingestellt, dass er eine inkrementelle Prüfung auf Integritätsbedingungen während des Ladens angibt.
- T (Tabelle aktualisieren): Dieser Block wurde durch die Ladeoperation so eingestellt, dass er eine erforderliche Pflege von AST-Tabellen angibt.
- F (frei): Wenn kein anderes Attribut gesetzt ist, wird der Block als frei betrachtet.

Da jeder Block einen Eintrag in der Blockzuordnungsdatei hat, wächst die Datei, wenn die Tabelle größer wird. Diese Datei wird als separates Objekt gespeichert. In einem SMS-Tabellenbereich hat sie einen neuen Dateityp. In einem DMS-Tabellenbereich hat sie einen neuen Objektdeskriptor in der Objekttabelle.

Zugehörige Konzepte:

- „Entwerfen von Tabellen mit mehrdimensionalem Clustering (MDC)“ in *Systemverwaltung: Konzept*

- „Erstellung, Platzierung und Verwendung von Tabellen mit mehrdimensionalem Clustering (MDC)“ in *Systemverwaltung: Konzept*
- „Speicherbedarf für Datenbankobjekte“ in *Systemverwaltung: Konzept*
- „Verbessern der Einfügeleistung“ auf Seite 150

Indexbereinigung und Indexpflege

Wenn Sie Indizes erstellen, sinkt die Leistung, sofern Sie nicht dafür sorgen, dass die Indizes kompakt und organisiert bleiben. Berücksichtigen Sie die folgenden Empfehlungen, um Indizes so klein und effizient wie möglich zu halten:

- Aktivieren Sie die Online-Indexdefragmentierung.
Erstellen Sie Indizes mit der Klausel MINPCTUSED. Löschen Sie vorhandene Indizes und erstellen Sie sie erneut, falls erforderlich.
- Führen Sie häufige Commits durch oder aktivieren Sie X-Sperren für Tabellen, entweder explizit oder durch Sperreneskalation, falls häufige Commits nicht möglich sind.
Indexschlüssel, die als gelöscht markiert sind, können nach dem Commit aus der Tabelle entfernt werden. X-Sperren für Tabellen ermöglichen es, den gelöschten Schlüssel physisch zu entfernen, wenn er als gelöscht markiert ist, wie weiter unten erläutert wird.
- Verwenden Sie REORGCHK, um festzustellen, wann Indizes oder Tabellen (oder beides) zu reorganisieren sind und wann REORG INDEXES mit der Option CLEANUP ONLY zu verwenden ist.
Um einen Schreib- und Lesezugriff auf einen Index bei der Reorganisation zuzulassen, führen Sie den Befehl REORG INDEXES mit der Option ALLOW WRITE ACCESS aus.

Anmerkung: In DB2 Version 8.1 und späteren Versionen werden alle Indizes als Indizes des Typs 2 erstellt. Eine Ausnahme besteht dann, wenn Sie einer Tabelle einen Index hinzufügen, die bereits Indizes des Typs 1 hat. In diesem Fall wird auch der neue Index als Index des Typs 1 erstellt. Zur Ermittlung des Typs von Index, der für eine Tabelle vorhanden ist, führen Sie den Befehl INSPECT aus. Zur Umwandlung von Indizes des Typs 1 in Indizes des Typs 2 führen Sie den Befehl REORG INDEXES aus.

Die Hauptvorteile von Indizes des Typs 2 lassen sich folgendermaßen beschreiben:

- Ein Index kann für Spalten erstellt werden, deren Länge 255 Byte überschreitet.
- Das Sperren der nächsten Schlüssel wird auf ein Minimum reduziert, wodurch sich der gemeinsame Zugriff verbessert. Der größte Teil des Sperrens der nächsten Schlüssel wird vermieden, weil ein Schlüssel nur als gelöscht markiert, jedoch nicht physisch von der Indexseite entfernt wird. Informationen über das Sperren von Schlüsseln finden Sie in den Abschnitten über die Auswirkungen von Sperren auf die Leistung.

Indexschlüssel, die als gelöscht markiert sind, werden bei folgenden Aktionen bereinigt:

- Während nachfolgender Einfüge-, Aktualisierungs- oder Löschaktivitäten
Während des Einfügens von Schlüsseln werden Schlüssel, die als gelöscht markiert und bekanntermaßen festgeschrieben sind, bereinigt, wenn eine solche Bereinigung die Durchführung einer Seitentrennung vermeiden hilft und verhindert, dass der Index größer wird.

Während des Löschens von Schlüsseln wird, wenn sämtliche Schlüssel auf einer Seite als gelöscht markiert wurden, ein Versuch unternommen, eine andere Indexseite zu finden, auf der alle Schlüssel als gelöscht markiert sind und alle diese Löschungen mit COMMIT festgeschrieben wurden. Wenn eine solche Seite gefunden wird, wird sie aus der Indexstruktur gelöscht.

Wenn beim Löschen eines Schlüssels eine X-Sperre für die Tabelle aktiv ist, wird der Schlüssel physisch gelöscht und nicht nur als gelöscht markiert. Während dieser physischen Löschung werden alle gelöschten Schlüssel auf derselben Seite ebenfalls entfernt, wenn sie als gelöscht markiert sind und bekannt ist, dass diese Löschung festgeschrieben ist.

- Bei der Ausführung des Befehls REORG INDEXES mit CLEANUP-Optionen

Die Option CLEANUP ONLY PAGES sucht nach Indexseiten und gibt diese frei, wenn auf ihnen alle Schlüssel als gelöscht markiert sind und diese Löschung mit COMMIT festgeschrieben wurde.

Die Option CLEANUP ONLY ALL gibt nicht nur Indexseiten frei, auf denen alle Schlüssel als gelöscht markiert und festgeschrieben sind, sondern entfernt zudem auch Satz-IDs (RIDs), die als gelöscht markiert sind und deren Löschung festgeschrieben wurde, auf Seiten, die auch einige nicht gelöschte Satz-IDs enthalten.

Diese Option versucht außerdem, benachbarte Blattseiten zusammenzufügen, wenn dies zu einer zusammengeführten Blattseite führt, die mindestens den durch PCTFREE angegebenen freien Speicherbereich enthält. Der Wert für PCTFREE ist der Prozentsatz an freiem Speicherbereich, der für den Index bei seiner Erstellung definiert ist. Der Standardwert für PCTFREE ist 10 %. Wenn zwei Seiten zusammengefügt werden können, wird eine der Seiten freigegeben.

Bei partitionierten Tabellen sollten Sie RUNSTATS nach Abschluss einer asynchronen Indexbereinigung ausführen, um genaue Indexstatistiken mit vorhandenen Datenpartitionen, deren Zuordnung aufgehoben wurde, zu generieren. Um festzustellen, ob in der Tabelle Datenpartitionen vorhanden sind, deren Zuordnung aufgehoben wurde, können Sie das Statusfeld in der Tabelle SYSDATA-PARTITIONS überprüfen und nach dem Wert "I" (Indexbereinigung) oder "D" (Zuordnung aufgehoben zu abhängiger MQT) suchen.

- Bei einer Neuerstellung eines Index

Zu den Dienstprogrammen, die Indizes neu erstellen, gehören folgende:

- Der Befehl REORG INDEXES, wenn er nicht mit einer der CLEANUP-Optionen verwendet wird
- Der Befehl REORG TABLE, wenn er nicht mit der Option INPLACE verwendet wird
- Der Befehl IMPORT mit der Option REPLACE
- Der Befehl LOAD mit der Option INDEXING MODE REBUILD

Zugehörige Konzepte:

- „Verwenden von relationalen Indizes zur Leistungsverbesserung“ auf Seite 29
- „Tipps zur Planung von relationalen Indizes“ auf Seite 33
- „Indexreorganisation“ auf Seite 366
- „Indexstruktur“ auf Seite 31
- „Tabellenreorganisation“ auf Seite 365

Online-Indexdefragmentierung

Eine Onlinedefragmentierung wird durch den benutzerdefinierbaren Schwellenwert für die minimale Größe des verwendeten Speicherbereichs auf einer Indexseite (Blattseite) ermöglicht. Wenn ein Indexschlüssel aus einer Blattseite gelöscht und dabei der Schwellenwert überschritten wird, werden die benachbarten Indexseiten daraufhin überprüft, ob zwei Blattseiten zusammengefügt werden können. Wenn auf einer Seite ausreichend Platz zum Zusammenfügen zweier benachbarter Seiten vorhanden ist, erfolgt die Zusammenfügung unverzüglich im Hintergrund.

Eine Onlinedefragmentierung des Index ist nur möglich für Indizes, die in Version 6 und späteren Versionen erstellt wurden. Wenn vorhandene Indizes die Möglichkeit der Onlinezusammenfügung benötigen, müssen sie gelöscht und mit der Klausel `MINPCTUSED` erneut erstellt werden. Setzen Sie `MINPCTUSED` auf einen Wert kleiner als 100. Der empfohlene Wert für `MINPCTUSED` ist kleiner als 50, da der Zweck darin besteht, zwei benachbarte Indexblattseiten zusammenzufügen. Durch den Wert 0 für `MINPCTUSED`, der gleichzeitig der Standardwert ist, wird die Onlinedefragmentierung inaktiviert.

Seiten im Index werden freigegeben, wenn der letzte Indexschlüssel auf der betreffenden Seite entfernt wird. Eine Ausnahme zu dieser Regel ist der Fall, wenn Sie die Klausel `MINPCTUSED` in der Anweisung `CREATE INDEX` angeben. Mit der Klausel `MINPCTUSED` wird ein Prozentwert des Speicherplatzes auf einer Indexblattseite angegeben. Wenn ein Indexschlüssel gelöscht wird und der Prozentsatz des gefüllten Speicherplatzes auf der Seite bei oder unter dem angegebenen Wert liegt, versucht der Datenbankmanager die verbleibenden Schlüssel mit Schlüsseln einer benachbarten Seite auf eine Seite zusammenzufügen. Wenn genügend Platz auf einer benachbarten Seite vorhanden ist, wird die Zusammenfügung ausgeführt und eine Indexblattseite gelöscht.

Seiten von Indizes, die keine Blattseiten sind, werden bei einer Online-Indexdefragmentierung nicht zusammengefügt. Jedoch werden leere Nichtblattseiten gelöscht und zur Wiederverwendung durch andere Indizes für die gleiche Tabelle verfügbar gemacht. Um diese Nichtblattseiten für andere Objekte in einem DMS-Speichermodell freizugeben oder um Plattenspeicherplatz in einem SMS-Speichermodell freizugeben, führen Sie eine vollständige Reorganisation der Tabelle oder der Indizes aus. Eine vollständige Reorganisation der Tabelle und der Indizes kann den Index auf minimale Größe verkleinern. Nichtblattseiten von Indizes werden bei einer Online-Indexdefragmentierung nicht zusammengefügt, jedoch gelöscht und zur Wiederverwendung freigegeben, wenn sie leer werden. Die Anzahl der Stufen im Index und die Anzahl der Blatt- und Nichtblattseiten können sich verringern.

Für Indizes des Typs 2 werden Schlüssel von einer Seite beim Löschen von Schlüsseln nur dann entfernt, wenn eine exklusive Sperre (Modus X) für die Tabelle aktiviert ist. Bei einer solchen Operation ist eine Online-Indexdefragmentierung effektiv. Wenn beim Löschen von Schlüsseln hingegen keine X-Sperre für die Tabelle aktiviert ist, werden Schlüssel als gelöscht markiert, jedoch nicht physisch von der Indexseite entfernt. Infolgedessen wird auch kein Versuch einer Defragmentierung unternommen.

Zur Defragmentierung von Indizes des Typs 2, in denen Schlüssel zwar als gelöscht markiert werden, jedoch in der physischen Indexseite verbleiben, führen Sie den Befehl `REORG INDEXES` mit der Option `CLEANUP ONLY ALL` aus. Die Option `CLEANUP ONLY ALL` defragmentiert den Index ohne Rücksicht auf den Wert von `MINPCTUSED`. Wenn Sie den Befehl `REORG INDEXES` mit der Option

CLEANUP ONLY ALL ausführen, werden zwei benachbarte Blattseiten zusammengefügt, wenn eine solche Zusammenfügung mindestens den Wert von PCTFREE an freiem Speicherplatz auf der zusammengefügten Seite übrig lassen kann. PCTFREE wird bei der Erstellung eines Index angegeben und hat den Standardwert 10 Prozent.

Zugehörige Konzepte:

- „Verwenden von relationalen Indizes zur Leistungsverbesserung“ auf Seite 29
- „Indexreorganisation“ auf Seite 366
- „Indexstruktur“ auf Seite 31
- „Tipps zur Leistung von relationalen Indizes“ auf Seite 35

Kapitel 14. Umverteilung von Daten

Umverteilung von Daten

Zur Umverteilung von Tabellendaten auf die Datenbankpartitionen in einer Umgebung mit partitionierten Datenbanken verwenden Sie den Befehl REDISTRIBUTE DATABASE PARTITION GROUP.

Anmerkung: In früheren Versionen von DB2 arbeitete dieser Befehl mit dem Schlüsselwort NODEGROUP an Stelle der DATABASE PARTITION GROUP-Schlüsselwörter.

In einer partitionierten Datenbank kann die Umverteilung der Daten folgenden Zwecken dienen:

- Neuausgleich von Datenvolumen und verteilte Verarbeitung von Auslastungen in allen Datenbankpartitionen
Wenn der Datenzugriff auf mehrere Datenbankpartitionen verteilt werden kann, verbessert sich die Leistung.
- Ungleichmäßige (gewichtete) Verteilung der Daten auf die Datenbankpartitionen
Die Zugriffsleistung und der Durchsatz lassen sich möglicherweise erhöhen, wenn Sie die Daten in einer Tabelle, auf die häufig zugegriffen wird, so umverteilen, dass die Daten, auf die selten zugegriffen wird, in einer kleinen Anzahl von Datenbankpartitionen in der Datenbankpartitionsgruppe gespeichert werden, während die Daten, auf die häufig zugegriffen wird, auf eine größere Anzahl von Datenbankpartitionen verteilt werden. Dadurch würden die Zugriffsleistung und der Durchsatz für die am häufigsten ausgeführten Anwendungen verbessert.

Verwenden Sie zur Beibehaltung der Tabellenkollokation den Befehl REDISTRIBUTE DATABASE PARTITION GROUP, um Daten auf der Ebene der Datenbankpartitionsgruppe umzuverteilen. Alle Tabellen werden in einer einzigen Operation umverteilt. Zur Umsetzung der angegebenen Datenverteilung teilt der Befehl REDISTRIBUTE DATABASE PARTITION GROUP Tabellen beim Versetzen der Zeilen auf die Datenbankpartitionen auf. In Abhängigkeit von der angegebenen Option kann das Dienstprogramm entweder eine Zielverteilungszuordnung generieren oder eine vorhandene Verteilungszuordnung als Eingabe verwenden.

Funktionsweise der Umverteilung von Daten auf Datenbankpartitionen

Eine Datenumverteilung wird für eine Gruppe von Tabellen in der angegebenen Datenbankpartitionsgruppe einer Datenbank durchgeführt. Sie müssen eine Verbindung zu der Datenbank in der Katalogdatenbankpartition herstellen, bevor Sie den Befehl REDISTRIBUTE DATABASE PARTITION GROUP ausführen, um das Dienstprogramm zur Datenumverteilung (Data Redistribution) aufzurufen. Das Dienstprogramm verwendet sowohl die Quellenverteilungszuordnung als auch die Zielverteilungszuordnung, um zu ermitteln, welche Hashdatenbankpartitionen einer neuen Speicherposition, d. h. einer neuen Datenbankpartitionsnummer, zugeordnet wurden. Alle Zeilen, die einer Datenbankpartition mit einer neuen Speicherposition entsprechen, werden von der in der Quellenverteilungszuordnung angegebenen Datenbankpartition in die in der Zielverteilungszuordnung angegebenen Datenbankpartition versetzt.

Das Dienstprogramm zur Datenumverteilung führt die folgenden Schritte aus:

1. Es empfängt eine neue Verteilungszuordnungs-ID für die Zielverteilungszuordnung und fügt diese in die Katalogsicht SYSCAT.PARTITIONMAPS ein.
2. Es aktualisiert die Spalte REBALANCE_PMAP_ID in der Katalogsicht SYSCAT.DBPARTITIONGROUPS für die Datenbankpartitionsgruppe mit der neuen Verteilungszuordnungs-ID.
3. Es fügt die neuen Datenbankpartitionen der Katalogsicht SYSCAT.DBPARTITIONGROUPDEF hinzu.
4. Es setzt den Wert der Spalte IN_USE in der Katalogsicht SYSCAT.DBPARTITIONGROUPDEF auf 'D' (DROP) für jede Datenbankpartition, die zu löschen ist.
5. Es führt ein Commit für die Katalogaktualisierungen aus.
6. Es erstellt Datenbankdateien für alle neuen Datenbankpartitionen.
7. Es verteilt die Daten tabellenweise für jede Tabelle in der Datenbankpartitionsgruppe um.
 - a. Es sperrt die Zeile für die Tabelle in der Katalogtabelle SYSTABLES.
 - b. Es macht alle Pakete, die mit dieser Tabelle arbeiten, ungültig. Die der Tabelle zugeordnete Verteilungszuordnungs-ID wird geändert, da die Tabellenzeilen neu verteilt werden. Da die Pakete ungültig gemacht werden, muss der Compiler die neuen Datenbankpartitionierungsinformationen für die Tabelle abrufen und entsprechend Pakete generieren.
 - c. Es sperrt die Tabelle im Exklusivmodus (Exclusive).
 - d. Es verwendet DELETE- und INSERT-Anweisungen, um die Daten in der Tabelle umzuverteilen.
 - e. Wenn die Umverteilungsoperation erfolgreich abgeschlossen wird, setzt es eine COMMIT-Anweisung für die Tabelle ab und fährt mit der nächsten Tabelle in der Datenbankpartitionsgruppe fort. Falls die Operation fehlschlägt, bevor die Tabelle vollständig umverteilt ist, macht das Dienstprogramm die Aktualisierungen an der Tabelle rückgängig (ROLLBACK), beendet die gesamte Umverteilungsoperation und gibt eine Fehlermeldung zurück.
8. Es löscht Datenbankdateien und Einträge in der Katalogsicht SYSCAT.NODEGROUPDEF für Datenbankpartitionen, die zuvor als zu löschen ('D') markiert wurden.
9. Es aktualisiert den Datensatz für die Datenbankpartitionsgruppe in der Katalogsicht SYSCAT.NODEGROUPS, um den Wert von PMAP_ID auf den Wert von REBALANCE_PMAP_ID und den Wert von REBALANCE_PMAP_ID auf NULL zu setzen.
10. Es löscht die alte Verteilungszuordnung aus der Katalogsicht SYSCAT.PARTITIONMAPS.
11. Es führt ein Commit für alle Änderungen aus.

Zugehörige Konzepte:

- „Protokollspeicheranforderungen für die Datenumverteilung“ auf Seite 400
- „Fehlerbehebung nach einer Umverteilung“ auf Seite 403

Zugehörige Tasks:

- „Feststellen der Erforderlichkeit einer Datenumverteilung“ auf Seite 397
- „Umverteilen von Daten auf Datenbankpartitionen“ auf Seite 397

Feststellen der Erforderlichkeit einer Datenumverteilung

Vor der Entscheidung zur Umverteilung von Daten müssen Sie herausfinden, ob die Daten unter den Datenbankpartitionen ungleich verteilt sind. Wenn Sie über Informationen über die aktuelle Verteilung verfügen, können Sie mit Hilfe dieser Informationen eine eigene Umverteilungsdatei oder Verteilungszuordnung erstellen.

Vorgehensweise:

Gehen Sie wie folgt vor, um Informationen über die aktuellen Datenverteilungen für Datenbankpartitionen in einer Datenbankpartitionsgruppe abzurufen:

1. Ermitteln Sie, ob Datenbankpartitionen ungleiche Verteilungen von Zeilen aufweisen.

Verwenden Sie für die größte Tabelle eine geeignete Datenbankpartitionierungsspalte, und geben Sie eine Abfrage ähnlich der folgenden ein:

```
SELECT PARTITION(spaltenname), COUNT(*) FROM tabellenname
GROUP BY PARTITION(spaltenname)
ORDER BY PARTITION(spaltenname) DESC
FETCH FIRST 100 ROWS ONLY
```

Die SQL-Funktionen PARTITION und DBPARTITIONNUM ermitteln die aktuelle Datenverteilung über Hashpartitionen oder Datenbankpartitionen. Die Funktion PARTITION gibt den Verteilungszuordnungsindex für jede Zeile der Tabelle zurück. Die Funktion DBPARTITIONNUM gibt die Datenbankpartitionsnummer der Zeile zurück.

2. Führen Sie diese Abfrage für andere große Tabellen aus, die über die Datenbankpartitionsgruppe verteilt sind.
3. Verwenden Sie die Informationen, um eine Verteilungsdatei und eine Zielverteilungszuordnung zu erstellen.

Zugehörige Konzepte:

- „Umverteilung von Daten“ auf Seite 395
- „Protokollspeicheranforderungen für die Datenumverteilung“ auf Seite 400

Zugehörige Tasks:

- „Umverteilen von Daten auf Datenbankpartitionen“ auf Seite 397

Umverteilen von Daten auf Datenbankpartitionen

In einer Umgebung mit partitionierten Datenbanken können Sie Daten in folgenden Fällen auf die Datenbankpartitionen zum Ausgleichen des Datenzugriffs umverteilen:

- Wenn einige Datenbankpartitionen mehr Daten als andere enthalten.
- Wenn auf einige Datenbankpartitionen häufiger als auf andere zugegriffen wird.

Voraussetzungen:

Protokolldateigröße: Stellen Sie sicher, dass die Protokolldateien für die Operation der Datenumverteilung ausreichend groß sind. Die Protokolldatei der einzelnen betroffenen Datenbankpartitionen muss die dort durchgeführten INSERT- und DELETE-Operationen aufnehmen können.

Replizierte MQTs (MQT - Materialized Query Table, gespeicherte Abfrage-tabelle): Wenn die Daten in einer Datenbankpartitionsgruppe replizierte MQTs enthält, müssen Sie diese Tabellen löschen, bevor Sie die Daten umverteilen. Nach der Umverteilung der Daten können Sie die MQTs erneut erstellen.

Einschränkungen:

Die folgenden Operationen können an Objekten der Datenbankpartitionsgruppe ausgeführt werden, während das Dienstprogramm aktiv ist. Sie können jedoch nicht an der Tabelle ausgeführt werden, die momentan umverteilt wird. Mögliche Operationen:

- Erstellen von Indizes für andere Tabellen. Die Anweisung `CREATE INDEX` verwendet die Verteilungszuordnung der betroffenen Tabelle.
- Löschen anderer Tabellen. Die Anweisung `DROP TABLE` verwendet die Verteilungszuordnung der betroffenen Tabelle.
- Löschen von Indizes für andere Tabellen. Die Anweisung `DROP INDEX` verwendet die Verteilungszuordnung der betroffenen Tabelle.
- Abfragen anderer Tabellen
- Aktualisieren anderer Tabellen
- Erstellen neuer Tabellen in einem in der Datenbankpartitionsgruppe definierten Tabellenbereich. Die Anweisung `CREATE TABLE` verwendet die Zielverteilungszuordnung.
- Erstellen von Tabellenbereichen in der Datenbankpartitionsgruppe

Folgende Operationen stehen während der Ausführung des Dienstprogramms nicht zur Verfügung:

- Starten einer weiteren Umverteilungsoperation für die Datenbankpartitionsgruppe
- Ausführen einer Anweisung `ALTER TABLE` für eine Tabelle in der Datenbankpartitionsgruppe
- Löschen (`DROP`) der Datenbankpartitionsgruppe
- Ändern der Datenbankpartitionsgruppe

Mit dieser Prozedur können keine Daten umverteilt werden, nachdem eine Datenbankpartition einem Einzelpartitionssystem hinzugefügt wurde, sofern nicht alle betroffenen Tabellen über einen Verteilungsschlüssel verfügen. Der Befehl `REDISTRIBUTE DATABASE PARTITION GROUP` ist bei der Umverteilung von Daten von Verteilungsschlüsseln abhängig. Der Verteilungsschlüssel wird beim Erstellen einer Tabelle in einer Datenbankpartitionsgruppe mit mehreren Partitionen automatisch generiert oder kann mit Hilfe der SQL-Anweisungen `CREATE TABLE` oder `ALTER TABLE` explizit definiert werden. Wurden Ihre Tabellen in einer Partitionsgruppe mit nur einer Partition erstellt und Sie haben in der SQL-Anweisung `CREATE TABLE` nicht den Verteilungsschlüssel definiert, liegen keine definierten Verteilungsschlüssel vor. Vor dem Umverteilen der Daten müssen Sie für jede betroffene Tabelle mit Hilfe der SQL-Anweisung `ALTER TABLE` einen Verteilungsschlüssel erstellen.

Vorgehensweise:

Gehen Sie wie folgt vor, um Daten auf Datenbankpartitionen in einer Datenbankpartitionsgruppe umzuverteilen:

1. Stellen Sie eine Verbindung zu der Datenbankpartition her, in der die Systemkatalogtabellen enthalten sind.
2. Führen Sie die als Voraussetzungen aufgeführten Operationen durch, falls erforderlich.

3. Setzen Sie den Befehl REDISTRIBUTE DATABASE PARTITION GROUP ab.

Anmerkung: In früheren Versionen von DB2 arbeitete dieser Befehl mit dem Schlüsselwort NODEGROUP an Stelle der Schlüsselwörter DATABASE PARTITION GROUP.

Geben Sie die folgenden Argumente an:

Name der Datenbankpartitionsgruppe

Sie müssen die Datenbankpartitionsgruppe angeben, in der die Daten umzuverteilen sind.

UNIFORM

Wenn die Daten gleichmäßig verteilt sind und gleichmäßig verteilt bleiben sollen, geben Sie entweder das Argument UNIFORM an oder übergehen alle Argumente für den Verteilungstyp. UNIFORM ist der Standardwert.

USING DISTFILE verteilungsdateiname

Wenn Sie eine benutzerdefinierte Verteilung angeben wollen, die eine ungleichmäßige Datenverteilung korrigiert oder herbeiführt, geben Sie den Namen der Verteilungsdatei an. Das Dienstprogramm zur Datenumverteilung verwendet diese Datei zum Aufbau einer Zielverteilungs-zuordnung.

USING TARGETMAP zielzuordnungsname

Das Dienstprogramm zur Datenumverteilung verwendet die angegebene Zielzuordnung direkt.

Weitere Einzelheiten enthalten die Informationen zum Befehlszeilendienstprogramm REDISTRIBUTE DATABASE PARTITION GROUP.

4. Gehen Sie nach Abschluss der Umverteilung wie folgt vor:
- Erstellen Sie alle replizierten MQTs erneut, die Sie vor der Umverteilung gelöscht haben.
 - Führen Sie den Befehl RUNSTATS aus, um Statistikdaten über die Datenverteilung zu sammeln, die vom SQL-Compiler und Optimierungsprogramm bei der Auswahl von Datenzugriffsplänen für Abfragen verwendet werden können.

Anmerkung: Die EXPLAIN-Tabellen enthalten Informationen über die Verteilungszuordnung, die zur Umverteilung der Daten verwendet wird.

Zur Umverteilung von Daten in einer Datenbankpartitionsgruppe können Sie die API sqludrdt einer Clientanwendung aufrufen.

Zugehörige Konzepte:

- „Umverteilung von Daten“ auf Seite 395
- „Protokollspeicheranforderungen für die Datenumverteilung“ auf Seite 400
- „Fehlerbehebung nach einer Umverteilung“ auf Seite 403

Zugehörige Tasks:

- „Feststellen der Erforderlichkeit einer Datenumverteilung“ auf Seite 397

Zugehörige Referenzen:

- „Einschränkungen für den nativen XML-Datenspeicher“ in *XML-Handbuch*
- „sqludrdt API - Redistribute data across a database partition group“ in *Administrative API Reference*

Protokollspeicheranforderungen für die Datenumverteilung

Bevor Sie Daten auf Datenbankpartitionen umverteilen, müssen Sie die Protokollspeicheranforderungen in die Überlegungen einbeziehen.

Die Größe des Protokolls muss ausreichend sein, damit die INSERT- und DELETE-Operationen darin aufgenommen werden können, die in jeder von der Umverteilung betroffenen Datenbankpartition ausgeführt werden. Die größten Protokollanforderungen betreffen entweder die Datenbankpartition, die die meisten Daten abgibt, oder die Datenbankpartition, die die meisten Daten dazuerhält.

Wenn Sie einen Wechsel zu einer größeren Zahl von Datenbankpartitionen vornehmen, können Sie anhand des Verhältnisses der aktuellen Anzahl von Datenbankpartitionen zur neuen Anzahl von Datenbankpartitionen die Anzahl der INSERT- und DELETE-Operationen abschätzen. Betrachten Sie zum Beispiel eine Umverteilung von Daten, die vor der Umverteilung gleichmäßig verteilt sind. Wenn Sie die Anzahl der Datenbankpartitionen von vier auf fünf erhöhen, werden ca. 20 Prozent der Daten der vier ursprünglichen Datenbankpartitionen in die neue Datenbankpartition verschoben. Dies bedeutet, dass 20 Prozent der DELETE-Operationen in jeder der vier ursprünglichen Datenbankpartitionen und sämtliche INSERT-Operationen in der neuen Datenbankpartition stattfinden.

Betrachten Sie demgegenüber eine nicht gleichmäßige Verteilung von Daten, wie sie zum Beispiel vorliegt, wenn der Verteilungsschlüssel viele Nullwerte enthält. In diesem Fall würden alle Zeilen, die einen Nullwert im Verteilungsschlüssel enthalten, aus einer Datenbankpartition unter dem alten Verteilungsschema in eine andere Datenbankpartition unter dem neuen Verteilungsschema verschoben. Infolgedessen erhöht sich der erforderliche Protokollspeicherbereich in diesen beiden Datenbankpartitionen und kann möglicherweise den unter der Annahme einer gleichmäßigen Verteilung berechneten Wert deutlich übersteigen.

Die Umverteilung der Daten jeder einzelnen Tabelle stellt nur eine Transaktion dar. Aus diesem Grund müssen Sie bei der Abschätzung des Protokollspeicherbereichs den Prozentsatz an Änderungen, wie die 20 Prozent in obigem Beispiel, mit der Größe der größten Tabelle multiplizieren. Beachten Sie jedoch, dass die größte Tabelle vielleicht gleichmäßig verteilt ist, die zweitgrößte Tabelle hingegen zum Beispiel eine oder mehrere unverhältnismäßig große Datenbankpartitionen haben kann. In einem solchen Fall wäre die Verwendung der ungleichmäßig verteilten Tabelle anstelle der größten Tabelle in Betracht zu ziehen.

Anmerkung: Wenn Sie einen Schätzwert für die Höchstmenge an Daten berechnet haben, die in einer Datenbankpartition einzufügen und zu löschen sind, verdoppeln Sie diesen Schätzwert, um den Spitzenwert für die Größe der aktiven Protokolldatei zu ermitteln. Wenn dieser Wert die Begrenzung von 256 GB für die aktive Protokolldatei überschreitet, muss die Datenumverteilung schrittweise erfolgen. Verwenden Sie das Dienstprogramm „makepmap“, um eine Reihe von Zielverteilungszuordnungen, d. h. eine für jeden Schritt, zu generieren. Darüber hinaus könnten Sie auch den Datenbankkonfigurationsparameter *logsecond* auf den Wert -1 setzen, um die meisten Protokollspeicherprobleme zu vermeiden.

Zugehörige Konzepte:

- „Umverteilung von Daten“ auf Seite 395

Umverteilen von Daten unter Verwendung schrittweiser Umverteilungsprozeduren

Die gespeicherten Prozeduren für eine schrittweise durchgeführte Umverteilung können zur sicheren Umverteilung der Daten einer Datenbankpartitionsgruppe in einer Reihe von Schritten verwendet werden:

1. Analysieren der Datenbankpartitionsgruppe im Hinblick auf Protokollspeicher-
verfügbarkeit und Abweichung der Datenverteilung (unter Verwendung von
ANALYZE_LOG_SPACE procedure – Retrieve log space analysis information).
Die Funktion `analyze_log_space` gibt eine Ergebnismenge (einen geöffneten
Cursor) von Ergebnissen einer Speicherbereichsanalyse zurück, die Felder für
jede Datenbankpartition der angegebenen Datenbankpartitionsgruppe enthal-
ten.
2. Erstellen einer Datenverteilungsdatei für eine bestimmte Tabelle (unter Verwen-
dung von GENERATE_DISTFILE procedure – Generate a data distribution file).
Die Funktion `generate_distfile` generiert eine Datenverteilungsdatei für die
angegebene Tabelle und speichert sie unter dem angegebenen Dateinamen.
3. Erstellen und Dokumentieren des Inhalts eines schrittweise durchgeführten
Umverteilungsplans für die Datenbankpartitionsgruppe (unter Verwendung
von STEPWISE_REDISTRIBUTE_DBPG procedure – Redistribute part of data-
base partition group).
4. Erstellen einer Datenverteilungsdatei für eine bestimmte Tabelle (unter Verwen-
dung von GET_SWRD_SETTINGS procedure – Retrieve redistribute informa-
tion und SET_SWRD_SETTINGS procedure – Create or change redistribute
registry).
Die Funktion `get_swrdd_settings` liest die vorhandenen Umverteilungsregistrier-
datensätze für die angegebene Datenbankpartitionsgruppe.
Die Funktion `set_swrdd_settings` erstellt oder ändert die Umverteilungsregistrier-
datenbank. Wenn die Registrierdatenbank nicht vorhanden ist, erstellt die Funk-
tion sie und fügt ihr Datensätze hinzu. Wenn die Registrierdatenbank bereits
vorhanden ist, stellt die Funktion über das Feld `overwriteSpec` fest, welche
Feldwerte überschrieben werden müssen. Das Feld `overwriteSpec` gibt dieser
Funktion die Möglichkeit, NULL-Eingaben für Felder zu akzeptieren, die nicht
aktualisiert werden müssen.
5. Umverteilen der Daten der Datenbankpartitionsgruppe dem Plan entspre-
chende (unter Verwendung von STEPWISE_REDISTRIBUTE_DBPG procedure –
Redistribute part of database partition group).
Die Funktion `stepwise_redistribute_dbpg` führt die Umverteilung eines Teils der
Datenbankpartitionsgruppe entsprechend der Eingabe und der Einstellungs-
datei durch.

Verwendungsbeispiel

Das folgende Beispiel zeigt ein Script für den Befehlszeilenprozessor (CLP) unter
AIX:

```
# -----  
# Angabe der Datenbank für die zu erstellende Verbindung  
# -----  
dbName="SAMPLE"  
  
# -----  
# Angabe des Namens für die Zieldatenbankpartitionsgruppe  
# -----  
dbpgName="IBMDEFAULTGROUP"
```

```

# -----
# Angabe des Schemas und des Namens der Tabelle
# -----
tbSchema="$USER"
tbName="STAFF"

# -----
# Angabe des Namens der Datenverteilungsdatei
# -----
distFile="$HOME/sqllib/function/$dbName.IBMDEFAULTGROUP_swrData.dst"

export DB2INSTANCE=$USER
export DB2COMM=TCPIP

# -----
# Aufrufen der Aufrufanweisungen im Befehlszeilenprozessor (clp)
# -----
db2start
db2 -v "connect to $dbName"

# -----
# Analyse des Effekts einer hinzugefügten Datenbankpartition ohne Anwendung der Änderungen -
# Analyse eines hypothetischen Falls
#
# - Im folgenden Fall werden hypothetisch die Datenbankpartitionen '40,50,60' der
#   Datenbankpartitionsgruppe hinzugefügt; für die Datenbankpartitionen '10,20,30,40,50,60' werden
#   die Zielverhältnisse 1:2:1:2:1:2 verwendet.
#
# Hinweis: In diesem Beispiel sind nur die Partitionen 10, 20 und 30 tatsächlich in der
#   Datenbankpartitionsgruppe vorhanden.
# -----
db2 -v "call sysproc.analyze_log_space('$dbpgName', '$tbSchema', '$tbName', 2, ' ',
'A', '40,50,60', '10,20,30,40,50,60', '1,2,1,2,1,2')"

# -----
# Analyse des Effekts einer gelöschten Datenbankpartition ohne Anwendung der Änderungen
#
# - Im folgenden Fall wird hypothetisch die Datenbankpartition 30 aus der Datenbankpartitionsgruppe
#   gelöscht und die Daten werden auf die Datenbankpartitionen 10 und 20 unter Verwendung
#   eines Zielverhältnisses von 1 : 1 umverteilt.
#
# Hinweis: In diesem Beispiel sollten alle Datenbankpartitionen, 10, 20 und 30, in der
#   Datenbankpartitionsgruppe vorhanden sein.
# -----
db2 -v "call sysproc.analyze_log_space('$dbpgName', '$tbSchema', '$tbName', 2, ' ',
'D', '30', '10,20', '1,1')"

# -----
# Generieren einer Datenverteilungsdatei zur Verwendung durch den Umverteilungsprozess
# -----
db2 -v "call sysproc.generate_distfile('$tbSchema', '$tbName', '$distFile')"

# -----
# Schreiben eines schrittweisen Umverteilungsplans in eine Registrierdatenbank
#
# Der Wert 1 für den zehnten Parameter kann bewirken, dass die zurzeit ausgeführte
# gespeicherte Prozedur zur schrittweisen Umverteilung den aktuellen Schritt beendet und
# die Verarbeitung stoppt, bis dieser Parameter auf 0 zurückgesetzt wird und die
# gespeicherte Prozedur zur Umverteilung erneut aufgerufen wird.
# -----
db2 -v "call sysproc.set_swr_settings('$dbpgName', 255, 0, ' ', '$distFile', 1000,
12, 2, 1, 0, '10,20,30', '50,50,50')"

# -----
# Berichten des Inhalts des schrittweisen Umverteilungsplans für die angegebene
# Datenbankpartitionsgruppe

```

```
# -----
db2 -v "call sysproc.get_swrd_settings('$dbpgName', 255, ?, ?, ?, ?, ?, ?, ?, ?, ?)"

# -----
# Umverteilen der Daten der Datenbankpartitionsgruppe "dbpgName" entsprechend dem in
# der Registrierdatenbank durch set_swrd_settings gespeicherten Umverteilungsplan.
# Mit Start bei Schritt 3 werden die Daten umverteilt, bis zwei Schritte im
# Umverteilungsplan ausgeführt sind.
# -----
db2 -v "call sysproc.stepwise_redistribute_dbpg('$dbpgName', 3, 2)"
```

Zugehörige Konzepte:

- „Umverteilung von Daten“ auf Seite 395

Fehlerbehebung nach einer Umverteilung

Nach dem Starten der Umverteilungsoperation wird eine Datei in das Unterverzeichnis `redist` des Verzeichnisses `sqllib` geschrieben. In dieser Statusdatei werden alle Operationen, die in Datenbankpartitionen ausgeführt werden, die Namen der Tabellen, die umverteilt wurden, und der Beendigungsstatus der Operation aufgelistet. Wenn eine Tabelle nicht umverteilt werden kann, wird der Name der Tabelle und der betreffende SQLCODE in der Datei aufgeführt. Wenn die Umverteilungsoperation aufgrund eines falschen Eingabeparameters nicht beginnen kann, wird die Datei nicht geschrieben und ein SQLCODE zurückgegeben.

Für die Datei gilt folgende Namenskonvention:

Für UNIX-Plattformen:

Datenbankname.Datenbankpartitionsgruppenname.zeitmarke

Für andere Plattformen (nicht UNIX):

Datenbankname\Datenbankpartitionsgruppenname\datum\uhrzeit

Anmerkung: Auf Nicht-UNIX-Plattformen werden nur die ersten acht (8) Byte des Datenbankpartitionsgruppennamens verwendet.

Wenn die Operation zur Datenumverteilung fehlschlägt, sind die Daten einiger Tabellen eventuell bereits neu verteilt, während andere Tabellen noch nicht bearbeitet wurden. Dies hat seine Ursache darin, dass die Datenumverteilung jeweils für eine Tabelle gleichzeitig durchgeführt wird. Sie haben zwei Möglichkeiten der Fehlerbehandlung:

- Verwenden der Option `CONTINUE`, um die Operation zur Datenumverteilung für die übrigen Tabellen fortzusetzen.
- Verwenden der Option `ROLLBACK`, um die Datenumverteilung rückgängig zu machen und die umverteilten Tabellen in ihren ursprünglichen Zustand zurückzusetzen. Der Rollback kann dabei ebenso lange dauern wie die ursprüngliche Umverteilungsoperation.

Bevor Sie eine dieser Optionen verwenden können, muss zuvor eine Operation zur Datenumverteilung fehlgeschlagen sein, so dass die Spalte `REBALANCE_P MID` in der Tabelle `SYSCAT.DBPARTITIONGROUPS` auf einen Nicht-NULL-Wert gesetzt ist.

Falls Sie die Statusdatei versehentlich löschen, haben Sie immer noch die Möglichkeit, eine `CONTINUE`-Operation auszuführen.

Zugehörige Konzepte:

- „Umverteilung von Daten“ auf Seite 395

Zugehörige Tasks:

- „Umverteilen von Daten auf Datenbankpartitionen“ auf Seite 397

Kapitel 15. Informationen des Datenbanksystemmonitors

Der DB2-Datenbankmanager pflegt Daten über den Betrieb, die Leistung und die Anwendungen, die ihn verwenden. Diese Daten werden während des Betriebs des Datenbankmanagers verwaltet und können wichtige Informationen über die Leistung und zur Fehlerbehebung liefern. Zum Beispiel erhalten Sie Anhaltspunkte über:

- Die Anzahl von Anwendungen, die mit einer Datenbank verbunden sind, ihren Status und welche SQL- und XQuery-Anweisungen (falls überhaupt) von jeder Anwendung ausgeführt werden.
- Informationen, die zeigen, wie gut der Datenbankmanager und die Datenbank konfiguriert sind, und bei der Optimierung helfen.
- Aufgetretene Deadlocks für eine angegebene Datenbank, beteiligte Anwendungen und welche Sperren in der Konkurrenzsituation waren.
- Die Liste von Sperren, die von einer Anwendungen oder Datenbank aktiviert wurden. Wenn die Anwendung die Verarbeitung nicht fortsetzen kann, weil sie auf eine Sperre wartet, werden weitere Informationen zur Sperre, einschließlich der Anwendung, die sie aktiviert hat, bereitgestellt.

Da die Erfassung einiger dieser Daten mit Systemaufwand für den Betrieb von DB2 verbunden ist, sind **Monitorschalter** verfügbar, mit denen gesteuert werden kann, welche Informationen erfasst werden. Zum expliziten Einstellen der Monitorschalter wird der Befehl `UPDATE MONITOR SWITCHES` oder die API `sqlmon()` verwendet. (Sie benötigen hierzu die Berechtigung `SYSADM`, `SYSCTRL`, `SYSMAINT` oder `SYSMON`.)

Sie können auf die vom Datenbankmanager verwalteten Daten zugreifen, indem Sie entweder eine Momentaufnahme (Snapshot) erstellen oder einen Ereignismonitor verwenden.

Erstellen einer Momentaufnahme

Mit einer der drei folgenden Methoden können Sie eine Momentaufnahme erstellen:

- Verwenden des Befehls `GET SNAPSHOT` über die Befehlszeile
- Schreiben einer eigenen Anwendung mit dem API-Aufruf `db2GetSnapshot()`
- Verwenden der Tabellenfunktionen zur Erstellung von Momentaufnahmen, um Monitoraten über einen bestimmten Bereich des Datenbanksystems zurückzugeben

Verwenden eines Ereignismonitors

Ein Ereignismonitor (Event Monitor) erfasst Systemmonitordaten nach Eintreten bestimmter Ereignisse, wie zum Beispiel das Ende einer Transaktion, das Ende einer Anweisung oder die Erkennung eines Deadlocks. Diese Informationen können in Dateien oder eine benannte Pipe geschrieben werden.

Ein Ereignismonitor wird wie folgt verwendet:

1. Erstellen Sie die Definition des Ereignismonitors mit Hilfe der Steuerzentrale oder der SQL-Anweisung `CREATE EVENT MONITOR`. Diese Anweisung speichert die Definition in den Systemkatalogen der Datenbank.

2. Aktivieren Sie den Ereignismonitor über die Steuerzentrale oder mit der folgenden SQL-Anweisung:

```
SET EVENT MONITOR ergname STATE 1
```

Wenn die Ergebnisse in eine benannte Pipe geschrieben werden, starten Sie die Anwendung, die die Daten aus der benannten Pipe liest, bevor Sie den Ereignismonitor aktivieren. Sie können entweder eine eigene Anwendung zum Lesen schreiben oder den Befehl **db2evmon** verwenden. Sobald der Ereignismonitor aktiv ist und mit dem Schreiben von Ereignissen in die Pipe beginnt, liest **db2evmon** die Ereignisse, wie sie generiert werden, und schreibt Sie in die Standardausgabe.

3. Lesen Sie die Tracedaten. Wenn Sie einen Ereignismonitor verwenden, der in eine Datei schreibt, können Sie die binären Tracedaten, die er erstellt, mit einer der folgenden Methoden anzeigen:
 - Verwenden Sie das Tool **db2evmon** zum Formatieren der Tracedaten für die Standardausgabe.
 - Klicken Sie das Symbol für **Event Analyzer** in der Steuerzentrale unter einem Windows-basierten Betriebssystem an, um eine grafische Schnittstelle zum Anzeigen der Tracedaten, zum Suchen nach Schlüsselwörtern und zum Herausfiltern nicht erwünschter Daten aufzurufen.

Anmerkung: Wenn das Datenbanksystem, das Sie überwachen, nicht auf derselben Maschine wie die Steuerzentrale ausgeführt wird, müssen Sie die Ereignismonitordatei auf dieselbe Maschine wie die Steuerzentrale kopieren, damit die Tracedaten angezeigt werden können. Sie können die Datei alternativ auch in ein gemeinsam benutztes Dateisystem stellen, auf das beide Maschinen zugreifen können.

Leistungseinfluss bei der Verwendung eines Deadlock-Ereignismonitors

Wenn ein Deadlock-Ereignismonitor mit der Option HISTORY aktiv ist, wird die allgemeine Leistung eines DB2-Datenbanksystems wie folgt beeinflusst:

- Der Speicher, der im Paketcache für in den Cache gestellte dynamische SQL- und XQuery-Anweisungen verwendet wird, die im Anweisungsprotokoll aufgelistet werden, wird erst freigegeben, wenn das bestimmte Anweisungsprotokoll nicht mehr benötigt wird (d. h., die aktuelle UOW (Unit of Work, Arbeitseinheit) wird beendet). Dadurch kann die Größe des Paketcaches auf Grund der erhöhten Speicherbelegung im Cache, der nicht freigegeben werden kann, zunehmen.
- Das Kopieren von Anweisungsinformationen in die Liste des Anweisungsprotokolls hat einen geringfügigen Einfluss auf die Systemleistung.
- In jeder Datenbankpartition wird der DB2-Systemmonitorzwischenpeicher vermehrt genutzt, damit eine Protokoll-Liste zu den Anweisungen für jede aktive Anwendung in der Datenbankpartition aufbewahrt wird. Die Zunahme hängt von der Anzahl der Anweisungen ab, die in jeder UOW von jeder Anwendung ausgeführt werden. Im Folgenden sehen Sie eine Berechnung für den Monitorzwischenpeicher:

Wenn ein Ereignismonitor den Typ DEADLOCK aufweist und die Option WITH DETAILS HISTORY aktiviert ist, fügen Sie $X \times 100$ Byte mal die maximale Anzahl an gleichzeitig ablaufenden Anwendungen hinzu, die aktiv sein sollen; dabei ist X die erwartete maximale Anzahl an Anweisungen in der UOW Ihrer Anwendung.

Wenn der Ereignismonitor den Typ DEADLOCK aufweist und die Option WITH DETAILS HISTORY VALUES aktiviert ist, fügen Sie ebenfalls

X*Y Byte mal die maximale Anzahl an gleichzeitig ablaufenden Anwendungen hinzu, die aktiv sein sollen; dabei ist Y die Summe der erwarteten maximalen Größe der in Ihre SQL- und XQuery-Anweisungen gebundenen Parameterwerte.

Wenn ein Deadlock-Ereignismonitor mit der Option VALUES aktiv ist, wird die allgemeine Leistung eines DB2-Datenbanksystems wie folgt beeinflusst (samt der zuvor aufgelisteten Beeinflussungsmöglichkeiten für die Option HISTORY):

- Das Kopieren von Anweisungsinformationen in die Liste des Anweisungsprotokolls hat einen äußerst geringfügigen Einfluss auf die Systemleistung.
- In jeder Datenbankpartition wird der DB2-Systemmonitorzwischenpeicher vermehrt genutzt, damit eine Protokoll-Liste zu den Anweisungen für jede aktive Anwendung in der Datenbankpartition aufbewahrt wird. Die Zunahme hängt von der Anzahl der pro Anweisung verwendeten Datenwerte sowie der Anzahl der Anweisungen ab, die in jeder UOW von jeder Anwendung ausgeführt werden.
- Der Datenbankmanager verwaltet eine Zusatzkopie der Datenwerte, die, in Abhängigkeit von der Größe und Anzahl der Variablen, die Leistung beeinflussen können.

Die Speicherauswirkungen auf den DB2-Systemmonitorzwischenpeicher kann dann von nennenswerter Bedeutung sein, wenn die Optionen HISTORY und VALUES für den Deadlock-Ereignismonitor angegeben werden. Um die Auswirkungen so gering wie möglich zu halten, verwenden Sie diese Optionen nur, wenn sie wirklich erforderlich sind. Eine andere Möglichkeit, die Auswirkungen zu verringern, besteht darin, die konfigurierte Größe des DB2-Systemmonitorzwischenpeichers vor der Aktivierung des Ereignismonitors in allen Datenbankpartitionen zu vergrößern.

Wenn ein tatsächlicher Deadlock auftritt und ein Deadlock-Ereignismonitor aktiv ist, wird die Systemleistung durch die Generierung der Ereignismonitordatensätze beeinflusst. Der Grad und die Dauer der Beeinflussung hängt von der Anzahl der Anwendungen und Datenbankpartitionen, die am Deadlock beteiligt sind, sowie von der Anzahl der Anweisungen und Datenwerte in den relevanten Anweisungsprotokoll-Listen ab.

Zugehörige Konzepte:

- „Einstiegstipps für die Leistungsoptimierung“ auf Seite 7

Kapitel 16. Messen der Systemleistung mit Vergleichstests

Durchführen von Vergleichstests

Die Durchführung von Vergleichstests (Benchmarktests) ist ein natürlicher Bestandteil des Entwicklungszyklus für Anwendungen. Sie erfordert die Zusammenarbeit von Anwendungsentwicklern und Datenbankadministratoren (DBAs) und sollte für eine Anwendung stattfinden, um Daten über die aktuelle Leistung zu erhalten und Ansätze zur Leistungsoptimierung zu ermitteln. Wenn der Code einer Anwendung bereits mit größtmöglicher Effizienz arbeitet, können eventuell weitere Leistungsvorteile durch die Optimierung der Konfigurationsparameter der Datenbank und des Datenbankmanagers realisiert werden. Sie können außerdem Anwendungsparameter optimieren, um die Anforderungen einer Anwendung besser zu erfüllen.

Durch verschiedene Arten von Vergleichstests lassen sich bestimmte Arten von Informationen zu gewinnen:

- Ein Vergleichstest der Art *Transaktion pro Sekunde* liefert Anhaltspunkte über die Leistungskapazität des Datenbankmanagers unter bestimmten, eingeschränkten Laborbedingungen.
- Ein *Anwendungsvergleichstest* prüft die gleichen Durchsatzkapazitäten unter Bedingungen, die den Produktionsbedingungen näher kommen.

Die Optimierung von Konfigurationsparametern durch Vergleichstests legt diese „Realbedingungen“ zugrunde und erfordert ein wiederholtes Ausführen von SQL aus der Anwendung mit variierenden Parameterwerten, bis die Anwendung mit der höchstmöglichen Effizienz arbeitet.

Die hier beschriebenen Vergleichstestmethoden sind auf die Optimierung von Konfigurationsparametern ausgerichtet. Jedoch kann derselbe Grundansatz auch für die Optimierung anderer, die Leistung beeinflussender Faktoren herangezogen werden, wie zum Beispiel:

- SQL-Anweisungen
- Indizes
- Tabellenbereichskonfiguration
- Anwendungscode
- Hardwarekonfiguration

Vergleichstests geben Aufschluss darüber, wie der Datenbankmanager unter unterschiedlichen Bedingungen reagiert. Es können Szenarios entwickelt werden, um die Behandlung von Deadlocks, die Leistung von Dienstprogrammen, die verschiedenen Methoden zum Laden von Daten, die Transaktionsgeschwindigkeit bei wachsenden Benutzerzahlen und auch die Auswirkungen der Verwendung eines neuen Produktreleases auf die Anwendung zu testen.

Vergleichstestmethoden

Bei der Durchführung von Vergleichstests wird eine reproduzierbare Umgebung zugrunde gelegt, so dass der gleiche Test unter den gleichen Bedingungen Ergebnisse liefert, deren Vergleich legitim ist.

Sie können mit den Vergleichstests beginnen, indem Sie die Testanwendung in einer Normalumgebung ausführen. Wenn Sie ein Leistungsproblem orten, können

Sie spezielle Testszenarios entwickeln, die den Wirkungsbereich der getesteten Funktion begrenzen. Die speziellen Testszenarios brauchen nicht eine gesamte Anwendung zu emulieren, um wertvolle Informationen zu liefern. Es empfiehlt sich, mit einfachen Messungen zu beginnen und die Komplexität der Methoden nur dann zu erhöhen, wenn dies erforderlich ist.

Gute Vergleichstests oder Messungen besitzen folgende Merkmale:

- Die Tests sind wiederholbar.
- Jede Wiederholung eines Tests beginnt im gleichen Systemstatus.
- Es sind keine anderen Funktionen bzw. Anwendungen im System aktiv, sofern vom Testszenario nicht vorgesehen ist, dass bestimmte andere Systemaktivitäten gleichzeitig stattfinden.

Anmerkung: Gestartete Anwendungen belegen Speicher, selbst wenn Sie auf Symbolgröße verkleinert wurden oder momentan inaktiv sind. Dadurch erhöht sich die Wahrscheinlichkeit, dass Seitenauslagerungen zu einer Verzerrung der Vergleichstestergebnisse führen, wodurch die Wiederholbarkeitsregel verletzt wird.

- Die Hardware und die Software, die für die Vergleichstests verwendet werden, entsprechen der realen Produktionsumgebung.

Zur Durchführung von Vergleichstests erstellen Sie ein Szenario und führen die Anwendungen mehrere Male in diesem Szenario aus, indem Sie bei jeder Ausführung wichtige Informationen erfassen. Die Erfassung wichtiger Informationen nach jeder Ausführung ist von höchster Bedeutung bei der Ermittlung der Änderungen, die eine Leistungsverbesserung sowohl für die Anwendung als auch für die Datenbank bewirken können.

Zugehörige Konzepte:

- „Vorbereiten von Vergleichstests“ auf Seite 410
- „Vergleichstestanalyse - Beispiel“ auf Seite 426
- „Erstellung von Vergleichstests“ auf Seite 412
- „Ausführung von Vergleichstests“ auf Seite 424

Vorbereiten von Vergleichstests

Schließen Sie den logischen Entwurf der Datenbank ab, für die Sie die Anwendung ausführen, bevor Sie mit den Leistungsvergleichstests beginnen. Richten Sie Tabellen, Sichten und Indizes ein und füllen Sie sie mit Daten. Normalisieren Sie Tabellen, binden Sie Anwendungspakete und füllen Sie Tabellen mit realistischen Daten.

Das endgültige physische Modell der Datenbank sollte bereits ebenfalls feststehen. Platzieren Sie Datenbankmanagerobjekte an ihre endgültigen Datenträgerpositionen, definieren Sie die Größe von Protokolldateien, legen Sie die Position von Arbeitsdateien und Backups fest und testen Sie die Backup-Prozeduren. Prüfen Sie ferner Ihre Pakete, um sicherzustellen, dass Leistungsoptionen wie Zeilenblockung aktiviert werden, wenn dies möglich ist.

Sie sollten einen Stand der Programmier- und Testphasen für die Anwendung erreicht haben, der es Ihnen ermöglicht, die Vergleichstestprogramme zu erstellen. Während der Vergleichstests können die praktischen Grenzen einer Anwendungen zutage treten. Jedoch liegt das Ziel der hier beschriebenen Vergleichstests in der Messung der Leistung und nicht in der Feststellung von Fehlern oder abnormalen Beendigungsbedingungen.

Das Vergleichstestprogramm muss in einer möglichst wirklichkeitsnahen Nachbildung der tatsächlichen Produktionsumgebung ausgeführt werden. Im Idealfall sollte ein Server des gleichen Modells mit derselben Speicher- und Festplattenkonfiguration verwendet werden. Dies ist besonders dann von Bedeutung, wenn die Anwendung letztendlich eine große Anzahl von Benutzern und große Mengen von Daten verarbeiten soll. Das Betriebssystem und mögliche Einrichtungen der Datenübertragung oder des Dateiservice sollten ebenfalls bereits optimiert worden sein.

Stellen Sie sicher, dass Sie Vergleichstests mit einer Datenbank in Größe der Produktionsdatenbank durchführen. Eine einzelne SQL-Anweisung sollte die gleiche Menge an Daten liefern und den gleichen Aufwand für Sortiervorgänge erfordern wie in der Geschäftsumgebung. Diese Regel gewährleistet, dass die Anwendung repräsentative Speicheranforderungen testet.

Die zu testenden SQL-Anweisungen sollten entweder zur Kategorie *repräsentatives SQL* oder zur Kategorie *Extremfall-SQL* (Worst-Case) gehören, wie im Folgenden erläutert wird:

Repräsentatives SQL

Zu repräsentativem SQL werden solche Anweisungen gezählt, die während eines typischen Einsatzes der zu testenden Anwendung ausgeführt werden. Die Anweisungen, die zum Test ausgewählt werden, sind von der Art der Anwendung abhängig. Beispielsweise kann für eine Dateneingabeanwendung eine Anweisung INSERT getestet werden, während für eine Banktransaktion eine Anweisung FETCH, eine Anweisung UPDATE und mehrere Anweisungen INSERT getestet werden können. Die Häufigkeit, mit der die Anwendung ausgeführt wird, und der Umfang der von den ausgewählten Anweisungen verarbeiteten Daten sollte als durchschnittlich angesehen werden können. Wenn die aufgrund der Anweisungen zu verarbeitenden Datenmengen sehr umfangreich sind, sind diese Anweisungen unter der Kategorie *Extremfall-SQL* zu betrachten, selbst wenn es sich um typische SQL-Anweisungen handelt.

Extremfall-SQL

Zu dieser Kategorie gehören Anweisungen mit folgenden Merkmalen:

- Anweisungen, die häufig ausgeführt werden
- Anweisungen, durch die umfangreiche Datenmengen verarbeitet werden
- Anweisungen, die zeitkritisch sind

Ein Beispiel hierfür ist eine Anwendung, die ausgeführt wird, wenn ein Telefonanruf von einem Kunden eingeht, und deren Anweisungen ausgeführt werden müssen, um Daten des Kunden abzurufen oder zu aktualisieren, während der Kunde wartet.

- Anweisungen mit der größten Anzahl zu verknüpfender Tabellen, oder mit dem komplexesten SQL in der Anwendung

Ein Beispiel wäre eine Bankanwendung, die kombinierte Kontoauszüge über die monatlichen Kontobewegungen für sämtliche verschiedene Arten von Konten eines Kunden erstellt. Eine allgemeine Tabelle könnte vielleicht die Kundenadressen und die Kontonummern enthalten; jedoch müssen mehrere andere Tabellen verknüpft werden, um alle benötigten Daten über Kontotransaktionen verarbeiten und zusammenstellen zu können. Die Multiplikation des Aufwands, der für ein Konto erforderlich ist, mit den mehreren Tausend Konten, die im gleichen Zeitraum verarbeitet werden müssen, macht deutlich, dass jede potenzielle Zeiteinsparung die Leistungsanforderungen erhöht.

- Anweisungen, die einen ungünstigen Zugriffspfad verwenden, z. B. eine Anweisung, die nicht sehr oft ausgeführt wird und nicht von den Indizes unterstützt wird, die für die betroffenen Tabellen erstellt wurden
- Anweisungen, die über einen langen Zeitraum hinweg laufen
- Eine Anweisung, die nur bei der Initialisierung einer Anwendung ausgeführt wird, jedoch überproportionale Ressourcenanforderungen stellt

Ein Beispiel wäre eine Anwendung, die eine Liste der Arbeiten für Konten erstellt, die während des Arbeitstages auszuführen sind. Wenn die Anwendung gestartet wird, löst die erste größere SQL-Anweisung einen Join über zahlreiche Tabellen aus, um eine sehr umfangreiche Liste aller Konten zu erstellen, für die der Benutzer der Anwendung verantwortlich ist. Die Anweisung wird vielleicht nur wenige Male am Tag ausgeführt, aber ihre Ausführung nimmt einige Minuten in Anspruch, wenn sie nicht ausreichend optimiert wurde.

Zugehörige Konzepte:

- „Erstellung von Vergleichstests“ auf Seite 412
- „Durchführen von Vergleichstests“ auf Seite 409

Erstellung von Vergleichstests

Beim Entwurf und bei der Implementierung eines Vergleichstestprogramms sind eine Reihe von Faktoren zu beachten. Da der Hauptzweck des Programms darin besteht, eine Benutzeranwendung zu simulieren, ist die allgemeine Struktur des Programms unterschiedlich. Sie können die gesamte Anwendung zum Vergleichstest verwenden, so dass Sie nur die entsprechenden Mittel zur Erfassung der Zeiten für die zu analysierenden SQL-Anweisungen einfügen. Bei großen oder komplexen Anwendungen ist es eventuell praktischer, nur die Blöcke mit den wichtigen Anweisungen in das Vergleichstestprogramm aufzunehmen.

Zum Testen der Leistung bestimmter SQL-Anweisungen können Sie diese Anweisungen allein in das Vergleichstestprogramm aufnehmen und die benötigten Anweisungen CONNECT, PREPARE, OPEN und andere sowie Mechanismen zur Zeiterfassung hinzufügen.

Ein weiterer wichtiger Gesichtspunkt ist die Art des Vergleichs, die verwendet werden sollte. Eine Möglichkeit ist die, eine Gruppe von SQL-Anweisungen über ein Zeitintervall wiederholt auszuführen. Das Verhältnis der Anzahl der ausgeführten Anweisungen zu diesem Zeitintervall ergäbe einen Wert für den Durchsatz für die Anwendung. Eine andere Möglichkeit ist die, einfach die für die Ausführung einzelner SQL-Anweisungen erforderliche Zeit zu bestimmen.

Für alle Vergleichstests benötigen Sie ein effizientes Zeiterfassungssystem, um die während der Verarbeitung entweder einzelner SQL-Anweisungen oder der gesamten Anwendung abgelaufene Zeit zu berechnen. Bei der Simulation von Anwendungen, in denen einzelne SQL-Anweisungen isoliert ausgeführt werden, kann es wichtig sein, die Zeiten für die Anweisungen CONNECT, PREPARE und COMMIT zu verfolgen. Bei Programmen jedoch, die viele verschiedene Anweisungen verarbeiten, wird vielleicht nur eine einzige Anweisung CONNECT oder COMMIT benötigt, so dass die Erfassung der Ausführungszeit für eine einzelne Anweisung Priorität haben könnte.

Obwohl die Erfassung der benötigten Zeit für jede Abfrage einen wichtigen Faktor bei der Leistungsanalyse darstellt, werden durch sie nicht unbedingt potenzielle Leistungsengpässe offen gelegt. Zum Beispiel könnten Informationen über die CPU-Auslastung, über aktive Sperren und Pufferpool-ein-/ausgaben Hinweise darauf geben, dass eine Anweisung durch die Ein-/Ausgabeaktivitäten gebremst wird und nicht die volle Kapazität der CPU nutzt. Ein Vergleichstestprogramm sollte es ermöglichen, diese Art von Daten für eine detailliertere Analyse bei Bedarf zu erfassen.

Nicht alle Anwendungen senden die gesamte Menge der durch eine Abfrage abgerufenen Zeilen an eine Ausgabeeinheit. Zum Beispiel könnte die gesamte Antwortmenge als Eingabe für ein anderes Programm dienen, so dass keine Zeilen aus der ersten Anwendung als Ausgabe gesendet werden. Die Formatierung von Daten zur Ausgabe auf der Anzeige verursacht in der Regel einen hohen CPU-Aufwand und spiegelt den Benutzerbedarf nicht unbedingt wider. Um eine genaue Simulation zu erhalten, sollte ein Vergleichstestprogramm die Zeilenbehandlung der bestimmten Anwendung berücksichtigen. Wenn Zeilen an eine Ausgabeeinheit gesendet werden, könnte ineffizientes Formatieren den Hauptanteil der CPU-Verarbeitungszeit in Anspruch nehmen und so zu einer Verzerrung der tatsächlichen Leistungsdaten für die SQL-Anweisung als solche führen.

Das Vergleichstest-Tool db2batch: Es steht ein Vergleichstest-Tool (db2batch) im Unterverzeichnis bin des Verzeichnisses sql1ib Ihrer Instanz zur Verfügung. Dieses Tool setzt viele der Richtlinien zur Erstellung eines Vergleichstestprogramms um. Dieses Tool kann SQL-Anweisungen entweder aus einer unstrukturierten Datei oder von der Standardeingabeeinheit lesen, die Anweisungen dynamisch beschreiben und vorbereiten und eine Antwortmenge zurückliefern. Es gibt Ihnen außerdem die Möglichkeit, die Größe der Antwortmenge und die Anzahl der Zeilen, die aus dieser Antwortmenge an eine Ausgabeeinheit gesendet werden, zu steuern.

Sie können die Stufe der leistungsbezogenen Daten angeben, die geliefert werden sollen, einschließlich der benötigten Zeit, CPU- und Pufferpoolauslastung, Sperren sowie anderer statistischer Daten aus dem Datenbankmonitor. Bei der Zeitmessung für eine Gruppe von SQL-Anweisungen erstellt db2batch auch eine Ergebnisübersicht und berechnet arithmetische und geometrische Mittelwerte. Sie können Syntaxoptionen anzeigen, indem Sie den Befehl db2batch -h in eine Befehlszeile eingeben.

Beispiele für db2batch-Tests

Das folgende Beispiel zeigt, wie das Tool db2batch mit einer Eingabedatei db2batch.sql verwendet werden könnte:

```
-- db2batch.sql
-- -----
--#SET PERF_DETAIL 3
--#SET ROWS_OUT 5

-- This query lists employees, the name of their department
-- and the number of activities to which they are assigned for
-- employees who are assigned to more than one activity less than
-- full-time.
--#COMMENT Query 1
select lastname, firstnme,
deptname, count(*) as num_act
from employee, department, emp_act
where employee.workdept = department.deptno and
employee.empno = emp_act.empno and
emp_act.emptime < 1
group by lastname, firstnme, deptname
having count(*) > 2;
--#SET PERF_DETAIL 1
--#SET ROWS_OUT 5

--#COMMENT Query 2
select lastname, firstnme,
deptname, count(*) as num_act
from employee, department, emp_act
where employee.workdept = department.deptno and
employee.empno = emp_act.empno and
emp_act.emptime < 1
group by lastname, firstnme, deptname
having count(*) <= 2;
```

Abbildung 33. Beispieleingabedatei für das Vergleichstest-Tool: db2batch.sql

Geben Sie zum Beispiel den folgenden Aufruf für das Vergleichstest-Tool ein:

```
db2batch -d sample -f db2batch.sql
```

Dadurch wird die folgende Ausgabe erstellt:

```
* Zeitmarke: Thu Feb 02 2006 16:03:17 EST
* Kommentar: "Query 1"
-----

* SQL-Anweisung Nummer 1:

select lastname, firstnme,
deptname, count(*) as num_act
from employee, department, emp_act
where employee.workdept = department.deptno and
employee.empno = emp_act.empno and
emp_act.emptime < 1
group by lastname, firstnme, deptname
having count(*) > 2;
```

Abbildung 34. Beispielausgabe des Programms db2batch

LASTNAME	FIRSTNME	DEPTNAME	NUM_ACT
JEFFERSON	JAMES	ADMINISTRATION SYSTEMS	3
JOHNSON	SYBIL	ADMINISTRATION SYSTEMS	4
NICHOLLS	HEATHER	INFORMATION CENTER	4
PEREZ	MARIA	ADMINISTRATION SYSTEMS	4
SMITH	DANIEL	ADMINISTRATION SYSTEMS	7

* 5 Zeile(n) abgerufen, 5 Zeile(n) ausgegeben.

* Abgelaufene Zeit beträgt: 0,046319 Sekunden

Überwachungsdaten

Instanzname	= dyuping
Knotenname	=
Knotentyp	= Datenbankserver mit lokalen Clients
Zeitmarke für Momentaufnahme	= 02/02/2006 16:03:18.188050

Abbildung 35. Beispielausgabe des Programms db2batch

Momentaufnahme für Datenbankmanager

Anzahl Datenbankpartitionen in DB2-Instanz	= 1
Datenbankmanagerstatus	= Aktiv
Produktname	= DB2 v9.1.0.0
Servicestufe	= n060128
Zugeordneter privater Sortierspeicher	= 0
Obere Grenze für privaten Sortierspeicher	= 0
Sortiervorgang nach Schwellenwertüberschreitung	= 0
Angeforderte über Pipe geleitete Sortiervorgänge	= 1
Akzeptierte über Pipe geleitete Sortiervorgänge	= 1
Zeitmarke für Start des Datenbankmanagers	= 02/01/2006 19:07:35.301696
Zeitmarke für letzte Zurücksetzung	= 02/02/2006 16:03:18.138248
Ferne Verbindungen zum Datenbankmanager	= 4
Ferne im Datenbankmanager ausgeführte Verbindungen	= 0
Lokale Verbindungen	= 2
Lokale im Datenbankmanager ausgeführte Verbindungen	= 0
Aktive lokale Datenbanken	= 2
Obere Grenze für registrierte Agenten	= 9
Obere Grenze für Agenten, die auf Token warten	= 0
Registrierte Agenten	= 9
Agenten, die auf ein Token warten	= 0
Inaktive Agenten	= 0
Festgeschriebener privater Speicher (Byte)	= 1572864
Schalterliste für Datenbankpartitionsnummer 0	
Inform. zu Pufferpoolaktivitäten (BUFFERPOOL)	= ON 02/02/2006 16:03:18.071139
Informationen zu Sperren (LOCK)	= ON 02/02/2006 13:45:50.679836
Informationen zur Sortierung (SORT)	= ON 02/02/2006 16:03:18.071143
Informationen zu SQL-Anweisungen (STATEMENT)	= ON 02/02/2006 16:03:18.071136
Informationen zu Tabellenaktivitäten (TABLE)	= ON 02/02/2006 16:03:18.071138
Informationen zu Zeitmarke abrufen (TIMESTAMP)	= ON 02/01/2006 19:07:35.301696
UOW-Informationen (UOW)	= ON 02/02/2006 16:03:18.071133
Aus Pool zugeordnete Agenten	= 1578
Aus leerem Pool erstellte Agenten	= 10
Von anderer Anwendung neu zugeordnete Agenten	= 0
Obere Grenze für koordinierende Agenten	= 9
Max. Anzahl Versuche, MAXAGENTS zu überschreiten	= 0
Hash-Joins nach Überschreiten der Zwischenspeicherschwelle	= 0
Gesamtanzahl Gateway-Verbindungen	= 0
Aktuelle Anzahl Gateway-Verbindungen	= 0
Auf eine Host-Antwort wartende Gateway-Verbindungen	= 0
Auf eine Clientanforderung wartende Gateway-Verbindungen	= 0
Pool-Agenten der Gateway-Verbindung neu zugeordnet	= 0

Benötigte Speicherkapazität für Datenbankmanager:

Speicherpooltyp	= Zwischenspeicher für Datenbankmonitor
Aktuelle Größe (Byte)	= 655360
Obere Grenze (Byte)	= 655360
Konfigurierte Größe (Byte)	= 393216
Speicherpooltyp	= Sonstiger Speicher
Aktuelle Größe (Byte)	= 8323072
Obere Grenze (Byte)	= 8323072
Konfigurierte Größe (Byte)	= 45088768
Momentaufnahme einer Anwendung	
Anwendungskennzeichen	= 339
Anwendungsstatus	= UOW im Wartestatus
Statusänderungszeit	= 02/02/2006 16:03:18.185731
Codepage der Anwendung	= 819
Landescode der Anwendung	= 1
DUOW-Korrelationstoken	= *LOCAL.dyuping.060202210317
Anwendungsname	= db2batch
Anwendungs-IDD	= *LOCAL.dyuping.060202210317
Folgenummer	= 00001
Clientbenutzer-ID für TP-Monitor	=
Clientworkstationname für TP-Monitor	=
Clientanwendungsname für TP-Monitor	= DB2BATCH
Clientabrechnungszeichenfolge für TP-Monitor	=
CONNECT-Berechtigungs-ID	= DYUPING
Clientanmelde-ID	= dyuping
NNAME für die Konfiguration des Clients	= honcho
Datenbankmanagerprodukt-ID für Client	= SQL09010
Prozess-ID der Clientanwendung	= 2543638
Plattform der Clientanwendung	= AIX 64BIT
Kommunikationsprotokoll des Clients	= Lokaler Client
Datenbankname	= SAMPLE
Datenbankpfad	= /home/dyuping/dyuping/NODE0000/SQL00003/
Aliasname der Clientdatenbank	= SAMPLE
Aliasname der Eingabedatenbank	= SAMPLE
Höchste gewährte Berechtigungsstufe	=
Direkte Berechtigung DBADM	
Direkte Berechtigung CREATETAB	
Direkte Berechtigung BINDADD	
Direkte Berechtigung CONNECT	
Direkte Berechtigung CREATE_NOT_FENC	
Direkte Berechtigung LOAD	
Direkte Berechtigung IMPLICIT_SCHEMA	
Direkte Berechtigung CREATE_EXT_RT	
Direkte Berechtigung QUIESCE_CONN	
Indirekte Berechtigung SYSADM	
Indirekte Berechtigung CREATETAB	
Indirekte Berechtigung BINDADD	
Indirekte Berechtigung CONNECT	
Indirekte Berechtigung IMPLICIT_SCHEMA	
Nummer der koordinierenden Datenbankpartition	= 0
Koordinator-Agentenprozess oder Thread-ID	= 1626226
Der Anwendung zugeordnete Agenten	= 1
Zeitmarke für Start der Verbindungsanforderung	= 02/02/2006 16:03:17.581904
Zeitmarke für Beendigung der Verbindungsanforderung	= 02/02/2006 16:03:18.045845
Leerlaufzeit der Anwendung	= 0
Adresse für eingehende Kommunikation	= *LOCAL.dyuping
Zeitmarke für letzte Zurücksetzung	= 02/02/2006 16:03:18.138248
Neu zugeordnete Agenten	= 0
Bei Sperren wartende Agenten	= 0
Maximale Anzahl zugehöriger Agenten	= 1
Priorität für Anwendungsagenten	= 0
Prioritätsart	= Dynamisch
Sperren der Anwendung	= 3
Sperren seit Ausführung von Connect	= 0

Wartezeit der Anwendung bei Sperren (ms)	= 0
Erkannte Deadlocks	= 0
Sperreneskalationen	= 0
Exklusive Sperreneskalationen	= 0
Sperrzeitüberschreitungen seit Verbindung	= 0
Gesamtwartezeit für UOW bei Sperren (ms)	= 0
Gesamtanzahl Sortiervorgänge	= 1
Total sort time (ms)	= 0
Gesamtüberläufe bei Sortierung	= 0
In erweiterten Speicher kopierte Datenseiten	= 0
In erweiterten Speicher kopierte Indexseiten	= 0
Aus erweitertem Speicher kopierte Datenseiten	= 0
Aus erweitertem Speicher kopierte Indexseiten	= 0
Logische Lesevorgänge im Pufferpool	= 43
Physische Lesevorgänge im Pufferpool	= 10
Logische Lesevorgänge in temporären Daten des Pufferpools	= 0
Physische Lesevorgänge in temporären Daten des Pufferpools	= 0
Schreibvorgänge im Pufferpool	= 0
Logische Lesevorgänge im Pufferpoolindex	= 7
Physische Lesevorgänge im Pufferpoolindex	= 1
Logische Lesevorgänge im temporären Pufferpoolindex	= 0
Physische Lesevorgänge im temporären Pufferpoolindex	= 0
Schreibvorgänge im Pufferpoolindex	= 0
Logische XDA-Lesevorgänge im Pufferpool	= 0
Physische XDA-Lesevorgänge im Pufferpool	= 0
Logische Lesevorgänge in temporären XDA-Daten des Pufferpools	= 0
Physische Lesevorgänge in temporären XDA-Daten des Pufferpools	= 0
XDA-Schreibvorgänge im Pufferpool	= 0
Gesamtzeit der Lesevorgänge im Pufferpool (ms)	= 9
Gesamtzeit der Schreibvorgänge im Pufferpool (ms)	= 0
Wartezeit für Vorablesezugriff (ms)	= 0
Ungeliesene Seiten für Vorablesezugriff	= 0
Direkte Lesevorgänge	= 22
Direkte Schreibvorgänge	= 0
Direkte Leseanforderungen	= 5
Direkte Schreibanforderungen	= 0
Abgelaufene Zeit für direktes Lesen (ms)	= 7
Abgelaufene Zeit für direktes Schreiben (ms)	= 0
Anzahl SQL-Anforderungen seit letztem Commit	= 7
COMMIT-Anweisungen	= 0
ROLLBACK-Anweisungen	= 0
Versuchte dynamische SQL-Anweisungen	= 2
Versuchte statische SQL-Anweisungen	= 0
Fehlgeschlagene Anweisungsoperationen	= 0
Ausgeführte SQL-Anweisungen SELECT	= 1
Ausgeführte Xquery-Anweisungen	= 0
Ausgeführte Anweisungen UPDATE/INSERT/DELETE	= 0
Ausgeführte DDL-Anweisungen	= 0
Interne automatische Rebinds	= 0
Intern gelöschte Zeilen	= 0
Intern eingefügte Zeilen	= 0
Intern aktualisierte Zeilen	= 0
Interne Ausführungen von COMMIT	= 0
Interne Ausführungen von ROLLBACK	= 0
Interne ROLLBACKs wegen Deadlocks	= 0
Versuchte Binde-/Vorkompilieroperationen	= 0
Gelöschte Zeilen	= 0
Eingefügte Zeilen	= 0
Aktualisierte Zeilen	= 0
Ausgewählte Zeilen	= 5
Gelesene Zeilen	= 129
Geschriebene Zeilen	= 0
Protokollbereich für UOW (Byte)	= 0
Zeitmarke für Beendigung der vorigen UOW	= 02/02/2006 16:03:18.045845
Antwortzeit der letzten beendeten UOW (sek,ms)	= 0.000000
Zeitmarke für Start der UOW	= 02/02/2006 16:03:18.072281
Zeitmarke für Stopp der UOW	= 12/31/1969 19:00:00.000000
Beendigungsstatus für UOW	= 0
Offene ferne Cursor	= 0
Offene ferne Blockcursor	= 0
Zurückgewiesene Anforderungen 'Block Remote Cursor'	= 0

```

Akzeptierte Anforderungen 'Block Remote Cursor'      = 2
Offene lokale Cursor                                = 0
Offene lokale Blockcursor                            = 0
Gesamte Benutzer-CPU-Zeit für Agenten                = 0.020000
Gesamte System-CPU-Zeit für Agenten                  = 0.010000
Abgelaufene Verarbeitungszeit für Host                = 0.045308

Suchoperationen im Paketcache                        = 1
Einfügungen im Paketcache                            = 1
Suchoperationen im Anwendungsabschnitt               = 2
Einfügungen im Anwendungsabschnitt                  = 1
Suchoperationen im Katalogcache                      = 6
Einfügungen im Katalogcache                          = 3
Überläufe des Katalogcache                          = 0
Voller Katalogcache-Zwischenspeicher                 = 0
Obere Grenze für Katalogcache                       = 0

Informationen zum Arbeitsbereich
Gemeinsame obere Grenze                              = 0
Gesamte gemeinsame Überläufe                        = 0
Gesamtzahl Einfügeoperationen in gemeinsame Abschnitte = 0
Gesamtzahl Suchoperationen in gemeinsamen Abschnitten = 0
Private obere Grenze                                 = 25848
Gesamte private Überläufe                           = 0
Gesamtzahl Einfügeoperationen in private Abschnitte   = 1
Gesamtzahl Suchoperationen in privaten Abschnitten   = 1

Anzahl Hash-Joins                                   = 2
Anzahl Hash-Schleifen                               = 0
Anzahl Überläufe von Hash-Joins                     = 0
Anzahl kleiner Überläufe von Hash-Joins              = 0

Anweisungsart                                         = Dynamische SQL-Anweisung
Anweisung                                             = Anweisung CLOSE
Abschnittsnummer                                     = 4
Anwendungsersteller                                  = NULLID
Paketname                                             = SYSSN200
Konsistenztoken                                      = SYSLVL01
Paketversions-ID                                     = 
Cursorname                                           = SQL_CURSN200C4
Datenbankpartitionsnummer für Anweisung              = 0
Zeitmarke für Start der Anweisung                    = 02/02/2006 16:03:18.178967
Zeitmarke für Stopp der Anweisung                    = 02/02/2006 16:03:18.185705
Antwortzeit der letzten beendeten Anweisung (sek,ms) = 0.000124
Gesamte Benutzer-CPU-Zeit für die Anweisung          = 0.010000
Gesamte System-CPU-Zeit für die Anweisung            = 0.000000
SQL-Compiler-Aufwandsschätzung in Timeron           = 87
SQL-Compiler-Kardinalitätsschätzung                  = 13
Angeforderter Grad der Parallelität                  = 1
Anzahl für die Anweisung aktiver Agenten             = 0
Anz. der für Anweisung erstellten Subagenten         = 1
Sortiervorgänge für Anweisungen                     = 1
Gesamte Sortierzeit                                  = 0
Überläufe bei Sortierung                             = 0
Gelesene Zeilen                                      = 116
Geschriebene Zeilen                                  = 0
Gelöschte Zeilen                                     = 0
Aktualisierte Zeilen                                 = 0
Eingefügte Zeilen                                    = 0
Abgerufene Zeilen                                    = 5
Logische Lesevorgänge im Pufferpool                  = 13
Physische Lesevorgänge im Pufferpool                  = 1
Logische Lesevorgänge in temporären Daten des Pufferpools = 0
Physische Lesevorgänge in temporären Daten des Pufferpools = 0
Logische Lesevorgänge im Pufferpoolindex             = 0
Physische Lesevorgänge im Pufferpoolindex            = 0
Physische Lesevorgänge im temporären Pufferpoolindex = 0
Logische XDA-Lesevorgänge im Pufferpool              = 0
Physische XDA-Lesevorgänge im Pufferpool             = 0
Logische Lesevorgänge in temporären XDA-Daten des Pufferpools = 0
Physische Lesevorgänge in temporären XDA-Daten des Pufferpools = 0
Blockcursor                                          = JA
Text der dynamischen SQL-Anweisung: select lastname, firstnme,
deptname, count(*) as num_act from employee, department, emp_act
where employee.workdept = department.deptno and

```

```

employee.empno = emp_act.empno and
emp_act.emptime < 1
group by lastname, firstnme, deptname having count(*) > 2

```

Agentenprozess-/Thread-ID = 1626226

Benötigte Speicherkapazität für Agent:

```

Speicherpooltyp = Sonstiger Speicher
Aktuelle Größe (Byte) = 65536
Obere Grenze (Byte) = 65536
Konfigurierte Größe (Byte) = 4294967296

```

```

Speicherpooltyp = Anwendungszwischenspeicher
Aktuelle Größe (Byte) = 262144
Obere Grenze (Byte) = 327680
Konfigurierte Größe (Byte) = 1245184

```

```

Speicherpooltyp = Zwischenspeicher für Anwendungssteuerung
Aktuelle Größe (Byte) = 65536
Obere Grenze (Byte) = 65536
Konfigurierte Größe (Byte) = 655360

```

Momentaufnahme einer Datenbank

```

Datenbankname = SAMPLE
Datenbankpfad = /home/dyuping/dyuping/NODE0000/SQL00003/
Aliasname der Eingabedatenbank = SAMPLE
Datenbankstatus = Aktiv
Datenbankpartitionsnummer für Katalog = 0
Katalogname des Netzwerknotens =
Betriebssystem auf Datenbankserver = AIX 64BIT
Position der Datenbank = Lokal
Zeitmarke für erste Datenbankverbindung = 02/02/2006 16:03:17.581904
Zeitmarke für letzte Zurücksetzung = 02/02/2006 16:03:18.138248
Zeitmarke für letztes Backup = 12/31/1969 19:00:00.000000
Obere Grenze für Verbindungen = 4
Anwendungsverbindungen = 4
Gesamtanzahl sekundärer Verbindungen = 3
Aktuell verbundene Anwendungen = 4
Aktuell im DB-Manager ausgeführte Anwendungen = 0
Zu Anwendungen gehörende Agenten = 3
Max. Anzahl zu Anwendungen gehörender Agenten = 4
Max. Anzahl koordinierender Agenten = 4

```

```

Aktuelle Sperren = 3
Warten bei Sperren = 0
Wartezeit der Datenbank bei Sperren (ms) = 0
Verwendeter Speicher für Sperrenlisten (Byte) = 5472
Erkannte Deadlocks = 0
Sperreneskalationen = 0
Exklusive Sperreneskalationen = 0
Bei Sperren wartende Agenten = 0
Überschreitung der Sperrzeit = 0
Anzahl unbestätigter Transaktionen = 0

```

```

Gesamter zugeordneter privater Sortierspeicher = 0
Gesamter zugeordneter gemeinsamer Sortierspeicher = 0
Obere Grenze für gemeinsamen Sortierspeicher = 59
Gesamtanzahl Sortiervorgänge = 1
Gesamte Sortierzeit (ms) = 0
Überläufe bei Sortierung = 0
Aktive Sortiervorgänge = 0

```

```

Logische Lesevorgänge im Pufferpool = 43
Physische Lesevorgänge im Pufferpool = 10
Logische Lesevorgänge in temporären Daten des Pufferpools = 0
Physische Lesevorgänge in temporären Daten des Pufferpools = 0
Asynchrone Lesevorgänge in Poolseiten = 0
Schreibvorgänge im Pufferpool = 0
Asynchrone Schreibvorgänge in Poolseiten = 0
Logische Lesevorgänge im Pufferpoolindex = 7
Physische Lesevorgänge im Pufferpoolindex = 1
Logische Lesevorgänge im temporären Pufferpoolindex = 0

```


Physische Lesevorgänge im temporären Pufferpoolindex	= 0
Asynchrones Lesen der Poolindexseiten	= 0
Schreibvorgänge im Pufferpoolindex	= 0
Asynchrone Schreibvorgänge in Poolindexseiten	= 0
Logische XDA-Lesevorgänge im Pufferpool	= 0
Physische XDA-Lesevorgänge im Pufferpool	= 0
Logische Lesevorgänge in temporären XDA-Daten des Pufferpools	= 0
Physische Lesevorgänge in temporären XDA-Daten des Pufferpools	= 0
XDA-Schreibvorgänge im Pufferpool	= 0
Asynchrones Lesen der XDA-Poolseiten	= 0
Asynchrone Schreibvorgänge in XDA-Poolseiten	= 0
Gesamtzeit der Lesevorgänge im Pufferpool (ms)	= 9
Gesamtzeit der Schreibvorgänge im Pufferpool (ms)	= 0
Gesamtzeit für asynchrone Lesevorgänge	= 0
Gesamtzeit für asynchrone Schreibvorgänge	= 0
Asynchrone Leseanforderungen in Daten	= 0
Asynchrone Indexleseanforderungen	= 0
Asynchrone XDA-Leseanforderungen	= 0
Löschtrigger bei LSN-Lücke	= 0
Löschtrigger bei Mangel an leeren Seiten	= 0
Löschtrigger bei Höchstanzahl genutzter Seiten	= 0
Wartezeit für Vorablesezugriff (ms)	= 0
Ungelesene Seiten für Vorablesezugriff	= 0
Direkte Lesevorgänge	= 22
Direkte Schreibvorgänge	= 0
Direkte Leseanforderungen	= 5
Direkte Schreibanforderungen	= 0
Abgelaufene Zeit für direktes Lesen (ms)	= 7
Abgelaufene Zeit für direktes Schreiben (ms)	= 0
Geschlossene Datenbankdateien	= 0
In erweiterten Speicher kopierte Datenseiten	= 0
In erweiterten Speicher kopierte Indexseiten	= 0
Aus erweitertem Speicher kopierte Datenseiten	= 0
Aus erweitertem Speicher kopierte Indexseiten	= 0
Über einen Vektor definierte E/A-Operationen	= Nicht gesammelt
Seiten aus über einen Vektor definierten E/A-Operationen	= Nicht gesammelt
Block-E/A-Operationen	= Nicht gesammelt
Seiten aus Block-E/A-Operationen	= Nicht gesammelt
Zuordnungen physischer Seiten	= Nicht gesammelt
Abgelaufene Verarbeitungszeit für Host	= 0.045308
Versuchte COMMIT-Anweisungen	= 0
Versuchte ROLLBACK-Anweisungen	= 0
Versuchte dynamische Anweisungen	= 2
Versuchte statische Anweisungen	= 0
Fehlgeschlagene Anweisungsoperationen	= 0
Ausgeführte SQL-Anweisungen SELECT	= 1
Ausgeführte Xquery-Anweisungen	= 0
Ausgeführte Anweisungen UPDATE/INSERT/DELETE	= 0
Ausgeführte DDL-Anweisungen	= 0
Interne automatische Rebinds	= 0
Intern gelöschte Zeilen	= 0
Intern eingefügte Zeilen	= 0
Intern aktualisierte Zeilen	= 0
Interne Ausführungen von COMMIT	= 0
Interne Ausführungen von ROLLBACK	= 0
Interne ROLLBACKs wegen Deadlocks	= 0
Gelöschte Zeilen	= 0
Eingefügte Zeilen	= 0
Aktualisierte Zeilen	= 0
Ausgewählte Zeilen	= 5
Gelesene Zeilen	= 129
Versuchte Binde-/Vorkompilieroperationen	= 0
Für Datenbank verfügbarer Protokollplatz (Byte)=	20400000
Von Datenbank belegter Protokollplatz (Byte)	= 0
Max. Speicher für sekundäres Protokoll (Byte)	= 0
Maximaler Speicher für Gesamtprotokoll (Byte)	= 0
Aktuell zugeordnete sekundäre Protokolle	= 0
Gelesene Protokollseiten	= 0
Geschriebene Protokollseiten	= 0
Anwendungs-ID mit der ältesten Transaktion	= 339

Suchoperationen im Paketcache	= 1
Einfügungen im Paketcache	= 1
Überläufe des Paketcaches	= 0
Obere Grenze für Paketcache (Byte)	= 720896
Suchoperationen im Anwendungsabschnitt	= 2
Einfügungen im Anwendungsabschnitt	= 1
Suchoperationen im Katalogcache	= 6
Einfügungen im Katalogcache	= 3
Überläufe des Katalogcache	= 0
Voller Katalogcache-Zwischenspeicher	= 0
Obere Grenze für Katalogcache	= 65536
Informationen zum Arbeitsbereich	
Gemeinsame obere Grenze	= 0
Entsprechende gemeinsame Überläufe	= 0
Gesamtzahl Einfügeoperationen in gemeinsame Abschnitte	= 0
Gesamtzahl Suchoperationen in gemeinsamen Abschnitten	= 0
Private obere Grenze	= 25848
Entsprechende private Überläufe	= 0
Gesamtzahl Einfügeoperationen in private Abschnitte	= 1
Gesamtzahl Suchoperationen in privaten Abschnitten	= 1
Anzahl Hash-Joins	= 2
Anzahl Hash-Schleifen	= 0
Anzahl Überläufe von Hash-Joins	= 0
Anzahl kleiner Überläufe von Hash-Joins	= 0

Benötigte Speicherkapazität für Datenbank:

Speicherpooltyp	= Zwischenspeicher für Backup/Restore/Dienstprogramme
Aktuelle Größe (Byte)	= 65536
Obere Grenze (Byte)	= 65536
Konfigurierte Größe (Byte)	= 20512768
Speicherpooltyp	= Zwischenspeicher für Paketcache
Aktuelle Größe (Byte)	= 720896
Obere Grenze (Byte)	= 720896
Konfigurierte Größe (Byte)	= 4294967296
Speicherpooltyp	= Zwischenspeicher für Katalogcache
Aktuelle Größe (Byte)	= 65536
Obere Grenze (Byte)	= 65536
Konfigurierte Größe (Byte)	= 4294967296
Speicherpooltyp	= Pufferpoolzwischenspeicher
Aktuelle Größe (Byte)	= 4325376
Obere Grenze (Byte)	= 4325376
Konfigurierte Größe (Byte)	= 4294967296
Speicherpooltyp	= Pufferpoolzwischenspeicher
Aktuelle Größe (Byte)	= 589824
Obere Grenze (Byte)	= 589824
Konfigurierte Größe (Byte)	= 4294967296
Speicherpooltyp	= Pufferpoolzwischenspeicher
Aktuelle Größe (Byte)	= 327680
Obere Grenze (Byte)	= 327680
Konfigurierte Größe (Byte)	= 4294967296
Speicherpooltyp	= Pufferpoolzwischenspeicher
Aktuelle Größe (Byte)	= 196608
Obere Grenze (Byte)	= 196608
Konfigurierte Größe (Byte)	= 4294967296
Speicherpooltyp	= Pufferpoolzwischenspeicher
Aktuelle Größe (Byte)	= 131072
Obere Grenze (Byte)	= 131072
Konfigurierte Größe (Byte)	= 4294967296
Speicherpooltyp	= Gemeinsamer Sortierspeicher
Aktuelle Größe (Byte)	= 65536
Obere Grenze (Byte)	= 327680
Konfigurierte Größe (Byte)	= 20512768

```

Speicherpooltyp                = Zwischenspeicher für Sperrenmanager
  Aktuelle Größe (Byte)         = 655360
  Obere Grenze (Byte)           = 655360
  Konfigurierte Größe (Byte)    = 720896

Speicherpooltyp                = Datenbankzwischenpeicher
  Aktuelle Größe (Byte)         = 4653056
  Obere Grenze (Byte)           = 4653056
  Konfigurierte Größe (Byte)    = 8978432

Speicherpooltyp                = Sonstiger Speicher
  Aktuelle Größe (Byte)         = 196608
  Obere Grenze (Byte)           = 196608
  Konfigurierte Größe (Byte)    = 4294967296

```

4. Abbildung 4

Nehmen Sie folgende Änderung vor

```
--#SET PERF_DETAIL 1 ROWS_OUT 5
```

Query 2

Statement number: 2

```

select lastname, firstnme,
deptname, count(*) as num_act
from employee, department, emp_act
where employee.workdept = department.deptno and
employee.empno = emp_act.empno and
emp_act.emptime < 1
group by lastname, firstnme, deptname
having count(*) <= 2

```

LASTNAME	FIRSTNME	DEPTNAME	NUM_ACT
GEYER	JOHN	SUPPORT SERVICES	2
GOUNOT	JASON	SOFTWARE SUPPORT	2
HAAS	CHRISTINE	SPIFFY COMPUTER SERVICE DIV.	2
JONES	WILLIAM	MANUFACTURING SYSTEMS	2
KWAN	SALLY	INFORMATION CENTER	2

Number of rows retrieved is: 8

Number of rows sent to output is: 5

Elapsed Time is: 0.037 seconds

Summary of Results

=====

Statement #	Elapsed Time (s)	Agent CPU Time (s)	Rows Fetched	Rows Printed
1	0.074	0.020	5	5
2	0.037	Not Collected	8	5
Arith. mean	0.055			
Geom. mean	0.052			
to				

* Kommentar: "Abfrage 2"

* SQL-Anweisung Nummer 2:

```

select lastname, firstnme,
deptname, count(*) as num_act
from employee, department, emp_act
where employee.workdept = department.deptno and
employee.empno = emp_act.empno and
emp_act.emptime < 1
group by lastname, firstnme, deptname
having count(*) <= 2;

```

LASTNAME	FIRSTNME	DEPTNAME	NUM_ACT
GEYER	JOHN	SUPPORT SERVICES	2
GOUNOT	JASON	SOFTWARE SUPPORT	2
HAAS	CHRISTINE	SPIFFY COMPUTER SERVICE DIV.	2
JONES	WILLIAM	MANUFACTURING SYSTEMS	2
KWAN	SALLY	INFORMATION CENTER	2

* 8 Zeile(n) abgerufen, 5 Zeile(n) ausgegeben.

* Abgelaufene Zeit beträgt: 0,004534 Sekunden

* Übersichtstabelle:

Typ	Anzahl	Wiederholungen	Gesamtzeit (s)	Mindestzeit (s)	...
Anweisung	1	1	0.046319	0.046319	...
Anweisung	2	1	0.004534	0.004534	...
...	Maximalzeit (s)	Arithmetisches Mittel	Geometrisches Mittel	Abgerufene Zeilen	Zeilenausgabe
...	0.046319	0.046319	0.046319	5	5
...	0.004534	0.004534	0.004534	8	5

* Gesamtanzahl Einträge: 2
 * Gesamtzeit: 0,050853 Sekunden
 * Mindestzeit: 0,004534 Sekunden
 * Maximalzeit: 0,046319 Sekunden
 * Zeit (arithmetisches Mittel): 0,025426 Sekunden
 * Zeit (geometrisches Mittel): 0,014492 Sekunden

* Zeitmarke: Thu Feb 02 2006 16:03:18 EST

Abbildung 36. Beispielausgabe des Programms db2batch

Die gezeigte Beispielausgabe umfasst spezifische Datenelemente, die vom Datenbanksystemmonitor geliefert werden.

Im folgenden Beispiel (unter UNIX) wird lediglich die Übersichtstabelle erstellt.

```
db2batch -d sample -f db2batch.sql -r /dev/null,
```

Mit diesem Aufruf wird nur die Ergebnisübersicht angezeigt. Durch die Option -r wird die Ausgabedatei 1 (outfile1) durch /dev/null ersetzt und die Ausgabedatei 2 (outfile2), die nur die Ergebnistabelle enthält, bleibt leer, so dass die Ausgabe von db2batch an die Anzeige gesendet wird:

* Übersichtstabelle:

Typ	Anzahl	Wiederholungen	Gesamtzeit (s)	Mindestzeit (s)	...
Anweisung	1	1	0.052655	0.052655	...
Anweisung	2	1	0.004518	0.004518	...
...	Maximalzeit (s)	Arithmetisches Mittel	Geometrisches Mittel	Abgerufene Zeilen	Zeilenausgabe
...	0.052655	0.052655	0.052655	5	5
...	0.004518	0.004518	0.004518	8	5

* Gesamtanzahl Einträge: 2
 * Gesamtzeit: 0,057173 Sekunden
 * Mindestzeit: 0,004518 Sekunden
 * Maximalzeit: 0,052655 Sekunden
 * Zeit (arithmetisches Mittel): 0,028587 Sekunden
 * Zeit (geometrisches Mittel): 0,015424 Sekunden

Abbildung 37. Beispielausgabe für db2batch - Nur Übersichtstabelle

Zugehörige Konzepte:

- „Vergleichstestanalyse - Beispiel“ auf Seite 426
- „Erstellung von Vergleichstests“ auf Seite 412
- „Ausführung von Vergleichstests“ auf Seite 424

Ausführung von Vergleichstests

Für eine Art von Datenbankvergleichstest wählen Sie einen Konfigurationsparameter aus und führen den Test mit verschiedenen Werten für den gewählten Parameter aus, bis die maximale Leistungssteigerung erzielt ist. Ein einzelner Test sollte die mehrmalige Ausführung der Anwendung (z. B. zwanzig- oder dreißigmal) mit demselben Parameterwert beinhalten, um einen Durchschnittswert für die benötigte Zeit zu ermitteln, der die Auswirkungen einer Änderung des Parameterwertes klarer wiedergibt.

Bei der Ausführung des Vergleichstests sollte der erste Durchlauf, der so genannte Aufwärmdurchlauf (Warm-up Run) von den nachfolgenden Iterationen, d. h. den Normaldurchläufen (Normal Run), getrennt betrachtet werden. Da der Aufwärmdurchlauf einige Startaktivitäten, wie zum Beispiel die Initialisierung des Pufferpools, durchführt, dauert er etwas länger als Normaldurchläufe. Obwohl die Daten aus dem Aufwärmdurchlauf durchaus realistische Werte darstellen können, sind sie für die statistische Auswertung nicht relevant. Zur Berechnung des Durchschnittswerts für die benötigte Zeit oder die CPU-Auslastung für eine bestimmte Gruppe von Parameterwerten sollten Sie nur die Ergebnisse aus Normaldurchläufen verwenden.

Sie können auch in Betracht ziehen, mit dem *Konfigurationsadvisor* den Aufwärmdurchlauf des Vergleichstests zu erstellen. Die im *Konfigurationsadvisor* gestellten Fragen geben einen Einblick in die Gesichtspunkte, die bei der Anpassung der Konfiguration Ihrer Umgebung für Normaldurchläufe in der Vergleichstestphase zu beachten sind. Sie können den *Konfigurationsadvisor* über die Steuerzentrale oder durch Ausführen des Befehls `db2 autoconfigure` mit den entsprechenden Optionen starten.

Wenn Vergleichstests mit einzelnen Abfragen ausgeführt werden, müssen Sie sicherstellen, dass Sie die potenziellen Auswirkungen früherer Abfragen minimieren, indem Sie den Pufferpool reinigen. Zum Reinigen des Pufferpools füllen Sie ihn mit einer Anzahl von Seiten, die für die Abfrage irrelevant sind (Flushing).

Nach der Ausführung der Durchläufe für eine bestimmte Gruppe von Parameterwerten können Sie den Wert eines einzelnen Parameters ändern. Zwischen den einzelnen Durchläufen müssen Sie jedoch folgende Maßnahmen durchführen, um die Vergleichstestumgebung wieder in den Ausgangszustand zurückzusetzen:

- Wenn die Katalogstatistiken für den Test aktualisiert wurden, stellen Sie sicher, dass für jeden Durchlauf dieselben Werte für die Statistiken verwendet werden.
- Die Daten, die im Test verwendet werden, müssen konsistent sein, wenn sie durch die Tests aktualisiert werden. Dies kann folgendermaßen sichergestellt werden:
 - Durch Verwenden des Dienstprogramms `RESTORE`, um die gesamte Datenbank wiederherzustellen. Die Backup-Kopie der Datenbank enthält den früheren Zustand, der für den nächsten Test bereit ist.
 - Durch Verwenden des Dienstprogramms `IMPORT` oder `LOAD`, um eine exportierte Kopie der Daten wiederherzustellen. Diese Methode erlaubt den Restore nur der Daten, die vom Test betroffen waren. Die Dienstprogramme `REORG` und `RUNSTATS` sollten für die Tabellen und Indizes, die diese Daten enthalten, ausgeführt werden.
- Binden Sie die Anwendung erneut an die Datenbank, um sie in ihren Ausgangszustand zurückzusetzen.

Zusammenfassung: Führen Sie die folgenden Schritte oder Durchläufe aus, um Vergleichstests für eine Datenbankanwendung durchzuführen:

Schritt 1

Belassen Sie alle Parameter zur Optimierung der Datenbank und des Datenbankmanagers außer folgenden auf ihren **Standardwerten**:

- Die Parameter, die für die Belastung durch den Test und die Zielsetzung des Tests von Bedeutung sind. (Sie werden selten genügend Zeit haben, um Vergleichstests zur Optimierung aller Parameter durchzuführen, so dass es empfehlenswert ist, zu Beginn gute Schätzwerte für einige Parameter festzulegen und von diesen ausgehend die Optimierung vorzunehmen.)
- Protokolldateigrößen, die während der Einheiten- oder Systemtests für Ihre Anwendung festgelegt werden sollten.
- Alle die Parameter, die geändert werden müssen, damit die Anwendung ausgeführt werden kann (d. h., Änderungen zur Verhinderung negativer SQL-Rückkehrcodes aufgrund von Ereignissen wie Speicherknappheit für den Anweisungszwischenspeicher).

Führen Sie Ihre Anzahl von Durchläufen (Iterationen) für diesen Anfangszustand aus, und berechnen Sie den Durchschnittswert für die Zeitmesswerte oder die CPU.

Schritt 2

Wählen Sie einen und nur einen für die Optimierung zu testenden Parameter aus, und ändern Sie seinen Wert.

Schritt 3

Führen Sie eine weitere Anzahl von Durchläufen (Iterationen) aus, und berechnen Sie den Durchschnittswert für die Zeitmesswerte oder die CPU.

Schritt 4

Ergreifen Sie in Abhängigkeit von den Ergebnissen des Vergleichstests eine der folgenden Maßnahmen:

- Wenn die Leistung besser wird, ändern Sie den Wert desselben Parameters und kehren zu Schritt 3 zurück. Ändern Sie diesen Parameter so lange, bis der maximale Leistungswert gezeigt wird.
- Wenn die Leistung sinkt oder unverändert bleibt, setzen Sie den Parameter auf seinen vorigen Wert zurück, kehren zu Schritt 2 zurück und wählen einen anderen Parameter. Wiederholen Sie diese Prozedur, bis alle Parameter getestet sind.

Anmerkung: Wenn Sie die Leistungsergebnisse in grafischer Darstellung festhalten würden, müssten Sie nach Stellen suchen, an denen die Kurve einen Maximalwert erreicht und dort verbleibt bzw. wieder absinkt.

Sie können ein Treiberprogramm schreiben, das Sie bei der Durchführung der Vergleichstests unterstützt. Dieses Treiberprogramm könnte in einer Sprache wie REXX bzw. bei auf UNIX-basierte Plattformen mit Hilfe von Shell-Scripts erstellt werden.

Dieses Treiberprogramm könnte das Vergleichstestprogramm ausführen, ihm dabei die richtigen Parameter übergeben, den Test durch mehrere Durchläufe führen, die Umgebung in einen konsistenten Zustand zurückversetzen, den nächsten Test mit neuen Parameterwerten vorbereiten und die Testdaten sammeln bzw. konsolidieren.

Diese Treiberprogramme können so flexibel gestaltet werden, dass sie zur Ausführung einer ganzen Reihe von Vergleichstests, zur Analyse der Ergebnisse und zur Erstellung eines Berichts über die endgültigen und optimalen Parameterwerte für einen bestimmten Test verwendet werden könnten.

Zugehörige Konzepte:

- „Vorbereiten von Vergleichstests“ auf Seite 410
- „Durchführen von Vergleichstests“ auf Seite 409
- „Beispiele für db2batch-Tests“ auf Seite 414

Vergleichstestanalyse - Beispiel

Die Ausgabe des Vergleichstestprogramms sollte eine Kennung für jeden Test (Test Number), die Iteration (Durchlauf) der Programmausführung (Iteration Number), die Anweisungsnummer (Statement Number) und die für die Ausführung erfasste Zeit (Timing) enthalten.

Eine Übersicht über Vergleichstestergebnisse nach einer Reihe von Messungen könnte wie folgt aussehen:

Test Numbr	Iter. Numbr	Stmnt Numbr	Timing (hh:mm:ss.ss)	SQL Statement
002	05	01	00:00:01.34	CONNECT TO SAMPLE
002	05	10	00:02:08.15	OPEN cursor_01
002	05	15	00:00:00.24	FETCH cursor_01
002	05	15	00:00:00.23	FETCH cursor_01
002	05	15	00:00:00.28	FETCH cursor_01
002	05	15	00:00:00.21	FETCH cursor_01
002	05	15	00:00:00.20	FETCH cursor_01
002	05	15	00:00:00.22	FETCH cursor_01
002	05	15	00:00:00.22	FETCH cursor_01
002	05	20	00:00:00.84	CLOSE cursor_01
002	05	99	00:00:00.03	CONNECT RESET

Abbildung 38. Beispielergebnisse eines Vergleichstestprogramms

Anmerkung: Die Daten im gezeigten Bericht dienen nur der Illustration. Sie stellen **keine** durch Tests ermittelten Ergebnisse dar.

Eine Analyse zeigt, dass die Anweisung CONNECT (Anweisung 01) 1,34 Sekunden dauerte, die Anweisung OPEN CURSOR (Anweisung 10) 2 Minuten und 8,15 Sekunden, die Anweisungen FETCH (Anweisung 15) sieben Zeilen mit der längsten Verzögerung von 0,28 Sekunden lieferten, die Anweisung CLOSE CURSOR (Anweisung 20) 0,84 Sekunden benötigte und die Anweisung CONNECT RESET (Anweisung 99) 0,03 Sekunden in Anspruch nahm.

Wenn das Programm die Daten in einem DEL-Format (Delimited ASCII) ausgeben könnte, könnten diese später in eine Datenbanktabelle oder ein Arbeitsblatt zur weiteren statistischen Analyse importiert werden.

Eine Beispielausgabe für einen Vergleichstestbericht könnte folgendermaßen aussehen:

PARAMETER	VALUES FOR EACH BENCHMARK TEST				
TEST NUMBER	001	002	003	004	005
locklist	63	63	63	63	63
maxappls	8	8	8	8	8
applheapsz	48	48	48	48	48
dbheap	128	128	128	128	128
sortheap	256	256	256	256	256
maxlocks	22	22	22	22	22
stmthep	1024	1024	1024	1024	1024
SQL STMT	AVERAGE TIMINGS (seconds)				
01	01.34	01.34	01.35	01.35	01.36
10	02.15	02.00	01.55	01.24	01.00
15	00.22	00.22	00.22	00.22	00.22
20	00.84	00.84	00.84	00.84	00.84
99	00.03	00.03	00.03	00.03	00.03

Abbildung 39. Beispielbericht zu den vom Vergleichstest ermittelten Ausführungszeiten

Anmerkung: Die Daten im gezeigten Bericht dienen nur der Illustration. Sie stellen **keine** durch Tests ermittelten Ergebnisse dar.

Zugehörige Konzepte:

- „Erstellung von Vergleichstests“ auf Seite 412
- „Ausführung von Vergleichstests“ auf Seite 424
- „Durchführen von Vergleichstests“ auf Seite 409

Teil 4. Anhänge und Schlussteil

Anhang A. DB2-Konfigurationsparameter

Einzelheiten zu Parametern nach Funktion

Die folgenden Abschnitte enthalten zusätzliche Einzelinformationen, die Ihnen das Verständnis der Parameter erleichtern und Sie bei der Optimierung der verschiedenen Konfigurationsparameter unterstützen sollen. Die Beschreibungen der einzelnen Parameter sind nach Funktion bzw. Zweck der Parameter geordnet:

- „Kapazitätsverwaltung“ auf Seite 432
- „Protokollieren und Recovery“ auf Seite 496
- „Datenbankverwaltung“ auf Seite 534
- „Kommunikation“ auf Seite 550
- „Umgebung mit partitionierter Datenbank“ auf Seite 553
- „Instanzverwaltung“ auf Seite 561
- „DB2-Verwaltungsserver“ auf Seite 589

Die Beschreibung der Parameter enthält jeweils folgende Informationen:

Konfigurationstyp

Gibt an, welche Konfigurationsdatei die Einstellung für den Parameter enthält:

- *Datenbankmanager* betrifft eine Instanz des Datenbankmanagers und alle Datenbanken, die innerhalb dieser Instanz definiert sind.
- *Datenbank* betrifft eine bestimmte Datenbank.

Parametertyp

Gibt an, ob der Parameterwert geändert werden kann oder nicht und ob die Änderung online wirksam wird:

- *Konfigurierbar*

Eine Bandbreite von Werten ist möglich, so dass der Parameter nach dem Kenntnisstand des Datenbankadministrators über die Anwendungen oder mit entsprechender Testerfahrung optimiert werden muss.

- *Online konfigurierbar*

Eine Bandbreite von Werten ist möglich, so dass der Parameter nach dem Kenntnisstand des Datenbankadministrators über die Anwendungen und/oder mit entsprechender Testerfahrung optimiert werden muss. Änderungen können wirksam angewendet werden, während die Datenbank online ist, ohne dass der Datenbankmanager gestoppt und erneut gestartet oder die Datenbank erneut aktiviert werden muss.

- *Informativ*

Die Werte dieser Parameter werden nur durch den Datenbankmanager selbst geändert und enthalten Informationen, wie z. B. das Release von DB2, unter dem eine Datenbank erstellt wurde, oder die Angabe, dass ein erforderliches Backup ansteht.

Kapazitätsverwaltung

Es gibt eine Reihe von Konfigurationsparametern sowohl auf Datenbank- als auch auf Datenbankmanagerebene, die Auswirkungen auf die Leistung des Systems haben können. Diese Parameter können in folgende Kategorien eingeteilt werden:

- „Gemeinsam benutzter Datenbankspeicher“
- „Gemeinsamer Anwendungsspeicher“ auf Seite 448
- „Privater Agentenspeicher“ auf Seite 452
- „Agenten-/Anwendungskommunikationsspeicher“ auf Seite 461
- „Speicher der Datenbankmanagerinstanz“ auf Seite 466
- „Sperren“ auf Seite 471
- „Ein-/Ausgabe und Speicher“ auf Seite 475
- „Agenten“ auf Seite 482
- „Gespeicherte Prozeduren und benutzerdefinierte Funktionen“ auf Seite 493

Gemeinsam benutzter Datenbankspeicher

Die folgenden Parameter wirken sich auf den globalen Datenbankspeicher aus, der auf dem System zugeordnet wird:

- „catalogcache_sz - Katalogcachegröße “
- „database_memory - Größe des gemeinsam genutzten Datenbankspeichers “ auf Seite 434
- „dbheap - Zwischenspeicher für Datenbank “ auf Seite 437
- „db_mem_thresh - Schwellenwert für Datenbankspeicher “ auf Seite 436
- „locklist - Maximaler Speicher für Sperrenliste “ auf Seite 438
- „logbufsz - Protokollpuffergröße “ auf Seite 441
- „pckcachesz - Größe des Paketcache “ auf Seite 442
- „self_tuning_mem - Speicher mit automatischer Leistungsoptimierung “ auf Seite 444
- „sheapthres_shr - Schwellenwert für Sortierspeicher für gemeinsame Sortiervorgänge “ auf Seite 446
- „util_heap_sz - Zwischenspeichergröße für Dienstprogramme “ auf Seite 448

catalogcache_sz - Katalogcachegröße

Konfigurationstyp	Datenbank
Parametertyp	Online konfigurierbar
Weitergabeklasse	Sofort
Standardwert [Bereich]	-1 [8 – 524 288]
Maßeinheit	Seiten (4 KB)
Zuordnung	Wenn die Datenbank initialisiert wird
Freigabe	Wenn die Datenbank heruntergefahren wird

Dieser Parameter wird aus dem gemeinsam benutzten Datenbankspeicher zugeordnet und für das Caching (Zwischenspeichern) von Systemkataloginformationen verwendet. In einem partitionierten Datenbanksystem gibt es für jede Datenbankpartition einen Katalogcache.

Das Caching von Kataloginformationen in Einzeldatenbankpartitionen ermöglicht dem Datenbankmanager, den internen Systemaufwand zu verringern, da auf die Systemkataloge (oder auf den Katalogknoten in einer Umgebung mit partitionierten Datenbanken) zum Abrufen von Informationen, die bereits abgerufen wurden, nicht mehr zugegriffen werden muss. Der Katalogcache wird zum Speichern folgender Informationen verwendet:

- SYSTABLES-Informationen (einschließlich gepackter Deskriptoren)
- Berechtigungsinformationen, einschließlich SYSDBAUTH-Informationen und Zugriffsrechte für das Ausführen von Routinen
- SYSROUTINES-Informationen

Das Verwenden des Katalogcache kann helfen, die Gesamtleistung folgender Operationen und Anwendungen zu erhöhen:

- Binden von Paketen und Kompilieren von SQL- und XQuery-Anweisungen
- Operationen, die eine Überprüfung der Zugriffsrechte auf Datenbankebene umfassen
- Operationen, die eine Überprüfung der Zugriffsrechte für das Ausführen von Routinen umfassen
- Anwendungen, die mit Nicht-Katalogknoten in einer Umgebung mit partitionierten Datenbanken verbunden sind

Durch Definieren des Standardwerts (-1) in einer Serverumgebung oder in einer Umgebung mit partitionierten Datenbanken wird als Wert für die Berechnung der Seitenzuordnung das Vierfache des Werts für den Konfigurationsparameter *max-appls* genommen. Wenn das Vierfache von *maxappls* jedoch kleiner als 8 ist, gilt dies nicht. In diesem Fall setzt der Standardwert -1 den Parameter *catalogcache_sz* auf den Wert 8.

Empfehlung: Verwenden Sie zu Beginn den Standardwert, und optimieren Sie den Wert mit Hilfe des Datenbanksystemmonitors. Beim Optimieren dieses Parameters sollten Sie überlegen, ob der zusätzliche Speicher, der für den Katalogcache reserviert wird, möglicherweise besser für einen anderen Zweck zugeordnet werden sollte, z. B. für den Pufferpool oder den Paketcache.

Eine Optimierung dieses Parameters ist besonders wichtig, wenn die Auslastung für kurze Zeit viele SQL- oder XQuery-Kompilierungen umfasst und anschließend nur noch wenige oder gar keine Kompilierungen stattfinden. Wenn der Cache zu groß ist, kann Speicher für Kopien nicht mehr benötigter Informationen verschwendet werden.

Überlegen Sie, ob in einer Umgebung mit partitionierten Datenbanken für *catalogcache_sz* auf dem Katalogknoten ein größerer Wert festgelegt werden muss, da die auf Nicht-Katalogknoten benötigten Kataloginformationen immer zuerst auf dem Katalogknoten zwischengespeichert werden.

Mit Hilfe der Monitorelemente *cat_cache_lookups* (Suchfunktionen des Katalogcache), *cat_cache_inserts* (Einfügungen im Katalogcache), *cat_cache_overflows* (Überläufe des Katalogcache) und *cat_cache_size_top* (Obere Grenze des Katalogcache) können Sie ermitteln, ob dieser Konfigurationsparameter angepasst werden sollte.

Anmerkung: Der Katalogcache ist auf allen Knoten in einer Umgebung mit partitionierten Datenbanken vorhanden. Da für jeden Knoten eine lokale Datenbankkonfigurationsdatei vorhanden ist, definiert der Wert des Parameters *catalogcache_sz* für jeden Knoten die Größe des lokalen Katalogcache. Um ein effizientes Caching zu gewährleisten und Überlaufszenarios zu vermeiden, müssen Sie den Wert für *catalogcache_sz* auf jedem Knoten explizit festlegen und dabei die Möglichkeit in Betracht ziehen, den Wert für *catalogcache_sz* auf Nicht-Katalogknoten kleiner zu wählen als auf Katalogknoten. Bedenken Sie, dass die Informationen, die auf Nicht-Katalogknoten zwischengespeichert werden müssen, aus dem Cache des Katalogknotens abgerufen werden. Ein Katalogcache auf Nicht-Katalogknoten ist wie eine Unter- menge der Informationen des Katalogcache auf dem Katalogknoten.

Im Allgemeinen ist ein größerer Cachespeicher erforderlich, wenn eine UOW (Unit of Work, Arbeitseinheit) dynamische SQL- oder XQuery-Anweisungen enthält oder wenn Sie Pakete binden, die eine große Anzahl statischer SQL- oder XQuery-Anweisungen enthalten.

Zugehörige Referenzen:

- „maxappls - Maximale Anzahl aktiver Anwendungen “ auf Seite 487
- „cat_cache_size_top - Catalog Cache High Water Mark monitor element“ in *Systemmonitor Handbuch und Referenz*
- „cat_cache_inserts - Catalog Cache Inserts monitor element“ in *Systemmonitor Handbuch und Referenz*
- „cat_cache_lookups - Catalog Cache Lookups monitor element“ in *Systemmonitor Handbuch und Referenz*
- „cat_cache_overflows - Catalog Cache Overflows monitor element“ in *Systemmonitor Handbuch und Referenz*
- „Einschränkungen für den nativen XML-Datenspeicher“ in *XML-Handbuch*
- „GET DATABASE CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE CONFIGURATION command“ in *Command Reference*

database_memory - Größe des gemeinsam genutzten Datenbankspeichers

Konfigurationstyp

Datenbank

Gilt für

- Datenbankserver mit lokalen und fernen Clients
- Datenbankserver mit lokalen Clients
- Partitionierten Datenbankserver mit lokalen und fernen Clients

Parametertyp

Konfigurierbar

Standardwert [Bereich]

AIX und Windows

Automatic [Automatic, Computed, 0 - 4.294.967.295]

Linux, HP-UX, Solaris-Betriebssystem

Computed [Computed, 0 - 4.294.967.295]

Maßeinheit	Seiten (4 KB)
Zuordnung	Wenn die Datenbank aktiviert wird
Freigabe	Wenn die Datenbank inaktiviert wird

Dieser Parameter gibt die Menge des gemeinsam genutzten Speichers an, die für den gemeinsam genutzten Speicherbereich einer Datenbank reserviert ist. Ist diese Menge geringer als die aus den einzelnen Speicherparametern errechnete Menge (beispielsweise aus Sperrenlisten, Zwischenspeichern von Dienstprogrammen, Pufferpools usw.) wird die größere Menge verwendet.

Der Wert AUTOMATIC für diesen Parameter unter Windows und AIX aktiviert die automatische Optimierung. Wenn sie aktiviert ist, bestimmt die Speicheroptimierungsfunktion den Gesamtpeicherbedarf für die Datenbank und erhöht oder verringert auf der Basis des aktuellen Bedarfs der Datenbank die Speichergröße, die für den gemeinsam genutzten Datenbankspeicher zugeordnet ist. Wenn zum Beispiel der aktuelle Bedarf der Datenbank hoch ist und ausreichend freier Speicher auf dem System zur Verfügung steht, wird mehr Speicher für den gemeinsam genutzten Datenbankspeicher in Anspruch genommen. Wenn der Bedarf an Datenbankspeicher sinkt oder die Größe des freien Speichers auf dem System auf einen zu niedrigen Wert zurückgeht, wird ein Teil des gemeinsam genutzten Datenbankspeichers freigegeben.

Die Speicheroptimierungsfunktion lässt je nach dem berechneten Vorteil einer Bereitstellung zusätzlichen Speichers für die Instanz immer eine minimale Menge an Speicher frei. Wenn eine Bereitstellung von mehr Speicher für eine Instanz einen großen Vorteil erzielt, hält die Speicheroptimierungsfunktion eine kleinere Größe an Speicher frei. Wenn der Vorteil kleiner ist, wird mehr freier Speicher behalten. Dadurch können Datenbanken in der Verteilung von Systemspeicher kooperieren.

Da die Speicheroptimierungsfunktion Speicherressourcen auf verschiedene Speicherverbraucher verteilt, müssen mindestens zwei Speicherverbraucher für die automatische Optimierung aktiviert sein, damit eine automatische Optimierung erfolgen kann.

Um die Verwaltung dieses Parameters zu vereinfachen, wird DB2 durch die Einstellung COMPUTED angewiesen, die erforderliche Speicherkapazität zu berechnen und den Speicher bei der Aktivierung der Datenbank zuzuordnen. DB2 ordnet außerdem zusätzlichen Speicher zu, um dem Speicherbedarf in Zeiten hoher Auslastung für die Zwischenspeicher im gemeinsam genutzten Datenbankspeicherbereich gerecht zu werden, wenn ein Zwischenspeicher seine konfigurierte Größe überschreitet. Andere Operationen, wie zum Beispiel dynamische Aktualisierungen der Konfiguration, haben ebenfalls Zugriff auf diesen zusätzlichen Speicher. Der Befehl **db2pd** mit der Option **-memsets** kann zur Überwachung des im gemeinsam genutzten Datenbankspeicherbereich verbliebenen nicht genutzten Speichers verwendet werden. Unter DB2 für AIX (64 Bit) wächst die Nutzung des realen gemeinsam genutzten Speichers der Datenbank bis zur Kapazität des physischen Speichers im System dynamisch an, um den Anforderungen der Datenbank Rechnung zu tragen, sodass der Parameter *database_memory* nicht explizit gesteuert werden muss. Die Einstellung der Registrierdatenbankvariablen DB2_PINNED_BP oder DB2_LGPAGE_BP schränkt jedoch die Möglichkeit zur Vergrößerung des gemeinsamen Datenbankspeichers ein. Eine Beschreibung dieser Registrierdatenbankvariablen finden Sie unter „Leistungsvariablen“.

Anmerkung: In DB2 Version 9.1 ist die Einstellung COMPUTED äquivalent zur Einstellung AUTOMATIC in DB2 UDB Version 8. In DB2 Version 9.1 dient die Einstellung AUTOMATIC zur Aktivierung der automatischen Optimierung.

Empfehlung: Dieser Wert bleibt in der Regel auf AUTOMATIC (AIX und Windows) oder COMPUTED (Linux, HP-UX oder Solaris-Betriebsumgebung) gesetzt. Bei Plattformen, welche die Einstellung AUTOMATIC nicht unterstützen, kann dieser Parameter dazu verwendet werden, zusätzlichen Speicher für künftige Zuwächse zu reservieren, indem er auf einen größeren Wert als die Summe der im gemeinsam genutzten Datenbankspeicher enthaltenen Zwischenspeicher (Heaps) gesetzt wird. Zum Beispiel kann der zusätzliche Speicher zur Erstellung neuer Pufferpools oder zur Vergrößerung vorhandener Pufferpools verwendet werden.

Zugehörige Konzepte:

- „Speicher mit automatischer Leistungsoptimierung“ auf Seite 295
- „Neue Funktionen für Version 9.1: Änderung des Konfigurationsparameters DATABASE_MEMORY“ in *Neue Funktionen*

Zugehörige Referenzen:

- „db_mem_thresh - Schwellenwert für Datenbankspeicher “ auf Seite 436
- „self_tuning_mem - Speicher mit automatischer Leistungsoptimierung “ auf Seite 444
- „db2pd - Monitor and troubleshoot DB2 database command“ in *Command Reference*
- „GET DATABASE CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE CONFIGURATION command“ in *Command Reference*

db_mem_thresh - Schwellenwert für Datenbankspeicher

Konfigurationstyp	Datenbank
Parametertyp	Online konfigurierbar
Weitergabeklasse	Sofort
Standardwert [Bereich]	10 [0 - 100]
Maßeinheit	Prozent

Dieser Datenbankkonfigurationsparameter bezieht sich darauf, wie der Datenbankmanager überschüssigen gemeinsam genutzten Speicher der Datenbank behandelt. Wenn Seiten von Speicher durch einen Prozess beansprucht werden, werden sie in der Regel für den Prozess fest reserviert. Das heißt, eine Speicherseite wird vom Betriebssystem zugeordnet und belegt einen Bereich entweder im physischen Speicher oder in einer Auslagerungsdatei auf der Platte. Abhängig von der Auslastung der Datenbank kann es zu gewissen Tageszeiten zu Spitzenbelastungen der Datenbank mit höchstem Speicherbedarf kommen. Wenn das Betriebssystem einmal genügend Speicher für diese Spitzenlastzeiten reserviert hat, bleibt dieser Speicher reserviert, auch wenn der Speicherbedarf später wieder nachlässt. Dieser Datenbankkonfigurationsparameter gibt den maximalen Prozentsatz des reservierten, jedoch momentan ungenutzten gemeinsamen Datenbankspeichers an, der vom Datenbankmanager zugelassen wird, bevor damit begonnen wird, reservierte Speicherseiten freizugeben und an das Betriebssystem zurückzugeben. Gültige Werte sind ganze Zahlen aus dem Bereich von 0 (sofortige Freigabe jedes nicht genutzten gemeinsamen Datenbankspeichers) bis 100 (keine Freigabe von unge-

nutztem gemeinsamen Datenbankspeicher). Der Standardwert ist 10 (Freigabe von ungenutztem Speicher erst, wenn mehr als 10 % des gemeinsam genutzten Datenbankspeichers zurzeit nicht genutzt werden).

Der Standardwert sollte für die meisten Auslastungen geeignet sein. Dieser Parameter muss mit Vorsicht aktualisiert werden, da ein zu niedrig eingestellter Wert zur einer übermäßigen Speicherbelastung für das System führen kann (da ständig Speicherseiten reserviert und wieder freigegeben werden), während ein zu hoch eingestellter Wert möglicherweise verhindert, dass der Datenbankmanager überhaupt gemeinsam genutzten Datenbankspeicher an das Betriebssystem zur Verwendung durch andere Prozesse zurückgibt.

Dieser Konfigurationsparameter kann dynamisch aktualisiert werden.

Dieser Konfigurationsparameter wird ignoriert (d. h. ungenutzte Seiten des gemeinsam genutzten Datenbankspeichers bleiben reserviert), wenn der Speicherbereich des gemeinsam genutzten Datenbankspeichers durch die Registrierdatenbankvariable DB2_PINNED_BP fixiert ist, für große Seiten durch die Registrierdatenbankvariable DB2_LARGE_PAGE_MEM konfiguriert ist oder die Freigabe von Speicher durch die Registrierdatenbankvariable DB2MEMDISCLAIM explizit inaktiviert ist.

Einige Versionen von Linux unterstützen die Freigabe von Unterbereichen eines gemeinsam genutzten Speichersegments an das Betriebssystem nicht. Auf solchen Plattformen wird dieser Parameter ignoriert.

Zugehörige Referenzen:

- „database_memory - Größe des gemeinsam genutzten Datenbankspeichers “ auf Seite 434
- „GET DATABASE CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE CONFIGURATION command“ in *Command Reference*

dbheap - Zwischenspeicher für Datenbank

Konfigurationstyp	Datenbank
Parametertyp	Online konfigurierbar
Weitergabeklasse	Sofort
Standardwert [Bereich]	UNIX1200 [32 – 524 288] Windows-Datenbankserver mit lokalen und fern- nen Clients600 [32 – 524 288] Windows-64-Bit-Datenbankserver mit lokalen Cli- ents600 [32 – 524 288] Windows-32-Bit-Datenbankserver mit lokalen Cli- ents300 [32 – 524 288]
Maßeinheit	Seiten (4 KB)
Zuordnung	Wenn die Datenbank aktiviert ist
Freigabe	Wenn die Datenbank inaktiviert ist

Für jede Datenbank gibt es einen Datenbankzwischenspeicher, der vom Datenbankmanager für alle Anwendungen, die auf die Datenbank zugreifen, verwendet wird.

Er enthält Steuerblockdaten für Tabellen, Indizes, Tabellenbereiche und Pufferpools. Er enthält außerdem Speicherbereich für den Protokollpuffer (*logbufsz*) sowie temporären Speicher, der von Dienstprogrammen verwendet wird. Daher ist die Größe des Zwischenspeichers von vielen Variablen abhängig. Die Steuerblockdaten werden im Zwischenspeicher gehalten, bis alle Anwendungen die Verbindung zur Datenbank getrennt haben.

Der Mindestspeicherbereich, den der Datenbankmanager zu Beginn benötigt, wird beim Herstellen der ersten Verbindung zugeordnet. Der Datenbereich wird nach Bedarf erweitert, bis der gesamte nicht reservierte Speicherplatz im gemeinsamen Datenbankspeicherbereich belegt ist.

Mit Hilfe des Datenbanksystemmonitors können Sie unter Verwendung des Elements *db_heap_top* (Maximal zugeordneter Datenbankzwischenspeicher) die größte Speichermenge ermitteln, die für den Datenbankzwischenspeicher verwendet wurde.

Zugehörige Referenzen:

- „db_heap_top - Maximum Database Heap Allocated monitor element“ in *System-monitor Handbuch und Referenz*
- „logbufsz - Protokollpuffergröße “ auf Seite 441
- „GET DATABASE CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE CONFIGURATION command“ in *Command Reference*

locklist - Maximaler Speicher für Sperrenliste

Konfigurationstyp	Datenbank
Parametertyp	Online konfigurierbar
Weitergabeklasse	Sofort
Standardwert [Bereich]	UNIX Automatic [4 - 524.288] Windows-Datenbankserver mit lokalen und fern- nen Clients Automatic [4 - 524.288] Windows-64-Bit-Datenbankserver mit lokalen Cli- ents Automatic [4 - 524.288] Windows-32-Bit-Datenbankserver mit lokalen Cli- ents Automatic [4 - 524.288]
Maßeinheit	Seiten (4 KB)
Zuordnung	Wenn die erste Anwendung die Verbindung zur Datenbank herstellt
Freigabe	Wenn die letzte Anwendung die Verbindung zur Datenbank beendet

Dieser Parameter gibt die Speichermenge an, die für die Sperrenliste zugeordnet wird. Für jede Datenbank gibt es eine Sperrenliste, die die Sperren aller gleichzeitig mit der Datenbank verbundenen Anwendungen enthält. Das Sperren ist eine Funktion des Datenbankmanagers zur Steuerung des gleichzeitigen Zugriffs auf Daten

der Datenbank durch mehrere Anwendungen. Sowohl Zeilen als auch Tabellen können gesperrt werden. Der Datenbankmanager fordert eventuell auch Sperren zur internen Verwendung an.

Der Wert `AUTOMATIC` für diesen Parameter aktiviert ihn für die automatische Leistungsoptimierung. Dadurch ist die Speicheroptimierungsfunktion in der Lage, die durch diesen Parameter gesteuerte Größe des Hauptspeicherbereichs an geänderte Auslastungsanforderungen je nach Bedarf dynamisch anzupassen. Da die Speicheroptimierungsfunktion Speicherressourcen auf verschiedene Speicher- verbraucher verteilt, müssen mindestens zwei Speicherverbraucher für die automatische Optimierung aktiviert sein, damit eine automatische Optimierung erfolgen kann.

Der Wert des Parameters *locklist* wird zusammen mit dem Parameter *maxlocks* optimiert, so dass eine Inaktivierung der automatischen Optimierung des Parameters *locklist* automatisch auch eine Inaktivierung der automatischen Optimierung des Parameters *maxlocks* bewirkt. Die Aktivierung der automatischen Optimierung für den Parameter *locklist* führt automatisch zu einer Aktivierung der automatischen Optimierung des Parameters *maxlocks*.

Auf 32-Bit-Plattformen erfordert jede Sperre 48 oder 96 Byte der Sperrenliste, und zwar in Abhängigkeit davon, ob für das Objekt andere Sperren aktiv sind:

- 96 Byte sind erforderlich, um eine Sperre für ein Objekt zu aktivieren, für das keine anderen Sperren aktiv sind.
- 48 Byte sind erforderlich, um eine Sperre für ein Objekt einzutragen, für das bereits eine vorhandene Sperre aktiv ist.

Auf 64-Bit-Plattformen (außer HP-UX/PA-RISC) erfordert jede Sperre 64 oder 128 Byte der Sperrenliste in Abhängigkeit davon, ob für das Objekt andere Sperren aktiv sind:

- 128 Byte sind erforderlich, um eine Sperre für ein Objekt zu aktivieren, für das keine anderen Sperren aktiviert sind.
- 64 Byte sind erforderlich, um eine Sperre für ein Objekt einzutragen, für das bereits eine vorhandene Sperre aktiv ist.

Auf HP-UX/PA-RISC-Plattformen (64 Bit) erfordert jede Sperre 80 oder 160 Byte der Sperrenliste in Abhängigkeit davon, ob für das Objekt andere Sperren aktiv sind.

Wenn der Prozentsatz der Sperrenliste, der von einer Anwendung verwendet wird, den Wert des Parameters *maxlocks* erreicht, führt der Datenbankmanager eine Sperreneskalation von Zeilenebene auf Tabellenebene für die Sperren aus, die von dieser Anwendung aktiviert wurden. Obwohl der Eskalationsprozess an sich nicht viel Zeit in Anspruch nimmt, wird durch Sperren ganzer Tabellen (im Vergleich zum Sperren einzelner Zeilen) der gemeinsame Zugriff eingeschränkt und die allgemeine Datenbankleistung kann bei nachfolgenden Zugriffen auf die betroffenen Tabellen beeinträchtigt werden. Es wird empfohlen, die Größe der Sperrenliste folgendermaßen klein zu halten:

- Führen Sie häufig Commits aus, um Sperren freizugeben.
- Wenn viele Aktualisierungen ausgeführt werden sollen, sperren Sie vor den Aktualisierungen die ganze Tabelle (mit der SQL-Anweisung `LOCK TABLE`). Dadurch wird nur eine Sperre verwendet und der Zugriff anderer auf die zu aktualisierenden Daten verhindert. Der gemeinsame Zugriff auf die Daten wird jedoch eingeschränkt.

Sie können die Option LOCKSIZE der Anweisung ALTER TABLE auch verwenden, um das Sperren einer bestimmten Tabelle zu steuern.

Die Verwendung der Isolationsstufe RR (Wiederholtes Lesen) kann zu einer automatischen Tabellensperre führen.

- Verwenden Sie die Isolationsstufe der Cursorstabilität (CS), wenn möglich, um die Anzahl der Sperren für gemeinsamen Zugriff zu verringern. Wenn dadurch die durch die Anwendung festgelegten Integritätsanforderungen nicht beeinträchtigt werden, verwenden Sie anstatt der Cursorstabilität die Isolationsstufe UR (Nicht festgeschriebener Lesevorgang), um die Anzahl der Sperren weiter zu verringern.
- Setzen Sie *locklist* auf AUTOMATIC. Die Sperrenliste wird synchron vergrößert, um eine Sperreneskalation bzw. ein vollständiges Füllen der Sperrenliste zu vermeiden.

Wenn die Sperrenliste voll ist, kann die Leistung herabgesetzt werden, da die Sperreneskalation mehr Tabellensperren und weniger Zeilensperren erzeugt und auf diese Weise den gemeinsamen Zugriff auf gemeinsam benutzte Objekte in der Datenbank einschränkt. Zudem kann es mehr Deadlocks zwischen Anwendungen geben (da sie alle auf eine verringerte Anzahl von Tabellensperren warten), was dazu führt, dass Transaktionen mit ROLLBACK rückgängig gemacht werden. Ihre Anwendung empfängt einen SQLCODE-Wert -912, wenn die maximale Anzahl von Sperranforderungen für die Datenbank erreicht wurde.

Empfehlung: Wenn Sperreneskalationen zu Leistungseinbußen führen, müssen Sie eventuell den Wert dieses Parameters oder den Wert des Parameters *maxlocks* erhöhen. Mit dem Datenbanksystemmonitor können Sie feststellen, ob Sperreneskalationen auftreten. Weitere Informationen finden Sie in der Beschreibung des Monitorelements *lock_escals* (Sperreneskalationen).

Die folgenden Schritte können Ihnen bei der Ermittlung der Anzahl der Seiten, die für Ihre Sperrenliste erforderlich sind, vielleicht helfen:

1. Berechnen Sie eine Untergrenze für die Größe der Sperrenliste, und verwenden Sie dazu *eine* der folgenden Formeln, abhängig von Ihrer jeweiligen Umgebung:
 - a. $(512 * x * \text{maxappls}) / 4096$
 - b. Bei aktiviertem Konzentrator:
 $(512 * x * \text{max\_coordagents}) / 4096$
 - c. In einer partitionierten Datenbank mit aktiviertem Konzentrator:
 $(512 * x * \text{max\_coordagents} * \text{Anzahl Datenbankpartitionen}) / 4096$

Dabei ist 512 ein Schätzwert für die durchschnittliche Anzahl von Sperren pro Anwendung, und x ist die Anzahl der benötigten Byte für jede Sperre eines Objekts, für das bereits eine Sperre aktiv ist (40 Byte auf 32-Bit-Plattformen, 64 Byte auf 64-Bit-Plattformen).

2. Berechnen Sie eine Obergrenze für die Größe der Sperrenliste:
 $(512 * y * \text{maxappls}) / 4096$

In dieser Formel ist y die Anzahl der benötigten Byte für die erste Sperre eines Objekts (80 Byte auf 32-Bit-Plattformen, 128 Byte auf 64-Bit-Plattformen).

3. Schätzen Sie das Aufkommen an gemeinsamem Zugriff durch Anwendungen auf Ihre Daten, und wählen Sie nach Ihren Schätzungen einen Anfangswert für *locklist*, der zwischen der berechneten Ober- und Untergrenze liegt.
4. Mit dem Datenbanksystemmonitor können Sie, wie unten beschrieben, den Wert dieses Parameters optimieren.

Sie können mit dem Datenbanksystemmonitor die maximale Anzahl der von einer bestimmten Transaktion aktivierten Sperren feststellen. Weitere Informationen finden Sie in der Beschreibung des Monitorelements *locks_held_top* (*Maximale Anzahl aktiver Sperren*).

Anhand dieser Informationen können Sie die geschätzte Anzahl der Sperren pro Anwendung bestätigen oder anpassen. Um diese Auswertung durchzuführen, müssen Sie mehrere Probeanwendungen ausführen und dabei beachten, dass die Monitordaten auf einer Transaktionsebene und nicht auf einer Anwendungsebene geliefert werden.

Der Wert des Parameters *locklist* sollte eventuell auch dann heraufgesetzt werden, wenn der Parameter *maxappls* erhöht wird oder wenn die Anwendungen, die ausgeführt werden, nur relativ selten Commits ausführen.

Wenn Sie diesen Parameter geändert haben, sollten Sie für die Anwendungen eventuell einen Rebind durchführen (mit dem Befehl REBIND).

Zugehörige Konzepte:

- „Speicher mit automatischer Leistungsoptimierung“ auf Seite 295

Zugehörige Referenzen:

- „locks_held_top - Maximum Number of Locks Held monitor element“ in *Systemmonitor Handbuch und Referenz*
- „lock_escals - Number of Lock Escalations monitor element“ in *Systemmonitor Handbuch und Referenz*
- „REBIND command“ in *Command Reference*
- „maxappls - Maximale Anzahl aktiver Anwendungen “ auf Seite 487
- „maxlocks - Maximale Anzahl Sperren pro Anwendung “ auf Seite 474
- „self_tuning_mem - Speicher mit automatischer Leistungsoptimierung “ auf Seite 444
- „GET DATABASE CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE CONFIGURATION command“ in *Command Reference*

logbufsz - Protokollpuffergröße

Konfigurationstyp	Datenbank
Parametertyp	Konfigurierbar
Standardwert [Bereich]	

32-Bit-Plattformen

8 [4 — 4 096]

64-Bit-Plattformen

8 [4 — 65 535]

Maßeinheit	Seiten (4 KB)

Mit diesem Parameter kann die Menge an Datenbankzwischenspeicher (der durch den Parameter *dbheap* definiert wird) angegeben werden, die als Puffer für Protokollsätze verwendet werden soll, bevor diese Protokollsätze auf die Festplatte geschrieben werden. Die Protokollsätze werden bei Eintreten eines der folgenden Umstände auf die Festplatte geschrieben:

- Eine Transaktion oder eine Gruppe von Transaktionen wird entsprechend der Angaben für den Konfigurationsparameter *mincommit* festgeschrieben.
- Der Protokollpuffer ist voll.
- Ein anderes internes Ereignis im Datenbankmanager macht das Schreiben auf die Festplatte erforderlich.

Der Wert dieses Parameters muss außerdem kleiner oder gleich dem Wert des Parameters *dbheap* sein. Durch Puffern der Protokollsätze werden E/A-Operationen für die Protokolldateien effektiver, da weniger häufig Schreiboperationen ausgeführt und bei jeder Schreiboperation mehr Protokollsätze auf die Festplatte geschrieben werden.

Empfehlung: Erhöhen Sie den Wert für die Größe dieses Pufferbereichs, wenn umfangreiche Leseaktivitäten auf einer dedizierten Protokollplatte auftreten oder die Festplatte in erheblichem Maße beansprucht wird. Wenn Sie den Wert dieses Parameters erhöhen, sollten Sie auch den Parameter *dbheap* berücksichtigen, da der Protokollpuffer einen Speicherbereich verwendet, der mit dem Parameter *dbheap* gesteuert wird.

Sie können mit dem Datenbanksystemmonitor feststellen, wie viel Speicherbereich des Protokollpuffers für eine bestimmte Transaktion (oder UOW = Unit of Work, Arbeitseinheit) verwendet wird. Weitere Informationen finden Sie in der Beschreibung des Monitorelements *log_space_used* (verwendeter Speicherbereich für UOW).

Zugehörige Referenzen:

- „uow_log_space_used - Unit of Work Log Space Used monitor element“ in *Systemmonitor Handbuch und Referenz*
- „mincommit - Anzahl der Gruppencommits “ auf Seite 512
- „dbheap - Zwischenspeicher für Datenbank “ auf Seite 437
- „GET DATABASE CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE CONFIGURATION command“ in *Command Reference*

pckcachesz - Größe des Paketcache

Konfigurationstyp	Datenbank
Parametertyp	Online konfigurierbar
Weitergabeklasse	Sofort
Standardwert [Bereich]	32-Bit-Plattformen Automatic [-1, 32 - 128.000] 64-Bit-Plattformen Automatic [-1, 32 - 524.288]
Maßeinheit	Seiten (4 KB)
Zuordnung	Wenn die Datenbank initialisiert wird
Freigabe	Wenn die Datenbank heruntergefahren wird

Dieser Parameter wird aus dem gemeinsam benutzten Datenbankspeicher zugeordnet und für das Caching (Zwischenspeichern) von Abschnitten statischer und

dynamischer SQL- und XQuery-Anweisungen in einer Datenbank verwendet. In einem partitionierten Datenbanksystem gibt es für jede Datenbankpartition einen Paketcache.

Das Caching von Paketen ermöglicht dem Datenbankmanager, den internen Systemaufwand zu verringern, da beim erneuten Laden eines Pakets kein Zugriff auf die Systemkataloge bzw. bei dynamischen SQL- oder XQuery-Anweisungen keine Kompilierung mehr erforderlich ist. Die Abschnitte werden im Paketcache behalten, bis einer der folgenden Umstände eintritt:

- Die Datenbank wird heruntergefahren.
- Das Paket oder die dynamische SQL- oder XQuery-Anweisung wird ungültig gemacht.
- Im Cache ist nicht mehr genügend Platz.

Dieses Caching des Abschnitts für eine statische oder dynamische SQL- oder XQuery-Anweisung kann die Leistung besonders dann verbessern, wenn dieselbe Anweisung mehrere Male von Anwendungen verwendet wird, die mit einer Datenbank verbunden sind. Dies ist insbesondere für eine Anwendung wichtig, die Transaktionen verarbeitet.

Der Wert `AUTOMATIC` für diesen Parameter aktiviert ihn für die automatische Leistungsoptimierung. Wenn der Parameter `self_tuning_mem` auf `ON` gesetzt ist, passt die Speicheroptimierungsfunktion die durch den Parameter `pckcachesz` gesteuerte Größe des Hauptspeicherbereichs an geänderte Auslastungsanforderungen je nach Bedarf dynamisch an. Da die Speicheroptimierungsfunktion Speicherressourcen auf verschiedene Speicherverbraucher verteilt, müssen mindestens zwei Speicherverbraucher für die automatische Optimierung aktiviert sein, damit eine automatische Optimierung erfolgen kann.

Wenn dieser Parameter auf den Wert `-1` gesetzt ist, wird als Wert für die Berechnung der Seitenzuordnung das Achtfache des für den Konfigurationsparameter `maxappls` angegebenen Werts verwendet. Dies gilt jedoch nicht, wenn das Achtfache von `maxappls` kleiner als 32 ist. In diesem Fall entspricht der Standardwert `-1` für `pckcachesz` dem Wert 32.

Empfehlung: Beim Optimieren dieses Parameters sollten Sie überlegen, ob der zusätzliche Speicher, der für den Paketcache reserviert wird, nicht günstiger für einen anderen Zweck zugeordnet werden könnte, z. B. für den Pufferpool oder einen Katalogcache. Aus diesem Grund sollten Sie zur Optimierung dieses Parameters Vergleichstests (Benchmark-Tests) durchführen.

Die optimale Einstellung dieses Parameters ist von besonderer Bedeutung, wenn zu Beginn mehrere Abschnitte verwendet werden und nur wenige Abschnitte wiederholt ausgeführt werden. Wenn der Cache zu groß ist, wird Speicher zum Behalten von Kopien der Anfangsabschnitte verschwendet.

Mit Hilfe der folgenden Monitorelemente können Sie ermitteln, ob Sie den Wert dieses Konfigurationsparameters anpassen sollten:

- `pkg_cache_lookups` (Zugriffe auf Paketcache)
- `pkg_cache_inserts` (Einfügungen in Paketcache)
- `pkg_cache_size_top` (obere Paketcachegrenze)
- `pkg_cache_num_overflows` (Überläufe des Paketcache)

Anmerkung: Der Paketcache ist ein Arbeitscache. Deshalb kann dieser Parameter nicht auf den Wert 0 gesetzt werden. Diesem Cache muss ausreichend Speicherplatz für alle Abschnitte der SQL- oder XQuery-Anweisungen zugeordnet sein, die momentan ausgeführt werden. Wenn mehr als der momentan benötigte Speicherplatz zugeordnet ist, werden Abschnitte zwischengespeichert. Diese Abschnitte können einfach ausgeführt werden, wenn sie das nächste Mal benötigt werden, und müssen nicht erneut geladen oder kompiliert werden.

Der durch den Parameter *pkcachesz* angegebene Grenzwert ist ein veränderlicher Grenzwert. Dieser Grenzwert kann, falls erforderlich, überschritten werden, wenn in dem von den Datenbanken gemeinsam benutzten Speicher noch Speicherplatz verfügbar ist. Sie können mit dem Monitorelement *pkg_cache_size_top* den Höchstwert ermitteln, bis zu dem der Paketcache angewachsen ist. Mit dem Monitorelement *pkg_cache_num_overflows* können Sie ermitteln, wie häufig der durch den Parameter *pkcachesz* angegebene Grenzwert überschritten wurde.

Zugehörige Konzepte:

- „Speicher mit automatischer Leistungsoptimierung“ auf Seite 295

Zugehörige Referenzen:

- „pkg_cache_size_top - Package Cache High Water Mark monitor element“ in *Systemmonitor Handbuch und Referenz*
- „pkg_cache_inserts - Package Cache Inserts monitor element“ in *Systemmonitor Handbuch und Referenz*
- „pkg_cache_lookups - Package Cache Lookups monitor element“ in *Systemmonitor Handbuch und Referenz*
- „pkg_cache_num_overflows - Package Cache Overflows monitor element“ in *Systemmonitor Handbuch und Referenz*
- „maxappls - Maximale Anzahl aktiver Anwendungen “ auf Seite 487
- „self_tuning_mem - Speicher mit automatischer Leistungsoptimierung “ auf Seite 444
- „GET DATABASE CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE CONFIGURATION command“ in *Command Reference*

self_tuning_mem - Speicher mit automatischer Leistungsoptimierung

Konfigurationstyp	Datenbank
Parametertyp	Online konfigurierbar
Weitergabeklasse	Sofort
Standardwert [Bereich]	

Umgebungen mit Einzelpartitionsdatenbanken
ON [ON; OFF]

Umgebungen mit mehreren Datenbankpartitionen
OFF [ON; OFF]

Wenn dieser Parameter auf den Wert ON gesetzt wird, verteilt die Speicheroptimierungsfunktion verfügbare Speicherressourcen nach Bedarf unter den

Speicherkonsumenten, für die die automatische Optimierung aktiviert ist. Da der Speicher zwischen Speicherkonsumenten aufgeteilt wird, müssen mindestens zwei Speicherkonsumenten für die automatische Optimierung aktiviert sein, damit die Speicheroptimierungsfunktion aktiv werden kann. Wenn der Parameter *self_tuning_mem* auf den Wert ON gesetzt ist, jedoch weniger als zwei Speicherkonsumenten für die automatische Optimierung aktiviert sind, bleibt die Speicheroptimierungsfunktion inaktiv. (Eine Ausnahme bildet der Speicherbereich des Sortierzwischenspeichers, der unabhängig davon optimiert werden kann, ob andere Speicherkonsumenten für die automatische Optimierung aktiviert sind oder nicht.)

Dieser Parameter hat in Umgebungen mit einer Einzeldatenbankpartition standardmäßig den Wert ON. In Umgebungen mit mehreren Datenbankpartitionen ist er standardmäßig OFF.

Die folgenden Speicherkonsumenten können für die automatische Optimierung aktiviert werden:

- Pufferpools (Steuerung durch den Parameter SIZE der Anweisungen ALTER BUFFERPOOL und CREATE BUFFERPOOL)
- Paketcache (Steuerung durch den Konfigurationsparameter *pckcachesz*)
- Sperrenliste (Steuerung durch die Konfigurationsparameter *locklist* und *maxlocks*)
- Sortierspeicher (Steuerung durch die Konfigurationsparameter *sheapthres_shr* und *sortheap*)
- Gemeinsam genutzter Datenbankspeicher (Steuerung durch den Konfigurationsparameter *database_memory*)

Zum Anzeigen der aktuellen Einstellung dieses Parameters verwenden Sie den Befehl GET DATABASE CONFIGURATION mit dem Parameter SHOW DETAIL. Die folgenden Einstellungen, die für diesen Parameter zurückgegeben werden, sind möglich:

Speicher mit automatischer Leistungsoptimierung	(SELF_TUNING_MEM) = OFF
Speicher mit automatischer Leistungsoptimierung	(SELF_TUNING_MEM) = ON (Aktiv)
Speicher mit automatischer Leistungsoptimierung	(SELF_TUNING_MEM) = ON (Inaktiv)

Wenn der Parameter den Wert ON (Aktiv) hat, bedeutet dies, dass die Speicheroptimierung den Speicher im System aktiv optimiert. Wenn der Parameter den Wert ON (Inaktiv) zeigt, bedeutet dies, dass die Speicheroptimierung zwar aktiviert ist, jedoch nicht aktiv ausgeführt wird, da weniger als zwei Speicherkonsumenten für die automatische Optimierung aktiviert sind.

In Umgebungen mit partitionierten Datenbanken zeigt der Konfigurationsparameter *self_tuning_mem* den Wert ON (Aktiv) nur für die Datenbankpartition, in der die Optimierungsfunktion ausgeführt wird. Auf allen anderen Knoten hat *self_tuning_mem* den Wert ON (Inaktiv). Wenn Sie feststellen möchten, ob die Speicheroptimierung in einer partitionierten Datenbank aktiv ist, müssen daher Sie den Parameter *self_tuning_mem* in allen Datenbankpartitionen überprüfen.

Wenn Sie eine Migration von einer früheren Version von DB2 auf DB2 Version 9 durchgeführt haben und planen, die Funktion für die automatische Speicheroptimierung zu nutzen, sollten Sie die folgenden Diagnoseanzeiger konfigurieren, um die Überprüfung von Schwellenwerten oder Statusangaben zu inaktivieren:

- Auslastung des gemeinsamen Sortierspeichers - *db.sort_shrmem_util*
- Prozentsatz der Sortierüberläufe - *db.spilled_sorts*
- Langfristige Auslastung des gemeinsamen Sortierspeichers - *db.max_sort_shrmem_util*

- Auslastung der Sperrenliste - db.locklist_util
- Rate für Sperreneskalation - db.lock_escal_rate
- Trefferquote im Paketcache - db.pkgcache_hitratio

Eines der Ziele der automatischen Speicheroptimierungsfunktion besteht in der Vermeidung, dass Speicher einem Speicherkonsumenten zugeordnet wird, wenn er nicht unverzüglich gebraucht wird. Daher kann die Auslastung des Speichers, der einem Speicherkonsumenten zugeordnet ist, dem Wert 100 % sehr nahe kommen, bevor mehr Speicher zugeordnet wird. Durch die Inaktivierung dieser Diagnosean- zeiger vermeiden Sie unnötige Alerts, die durch die hohe Speicherauslastung für einen Speicherkonsumenten ausgelöst werden.

Bei Instanzen, die in DB2 Version 9 erstellt werden, sind diese Diagnoseanzeiger standardmäßig inaktiviert.

In einer Datenbank, die von einer früheren Version von DB2 migriert wurde, ist der Parameter *self_tuning_mem* auf den Wert OFF gesetzt.

Zugehörige Konzepte:

- „Speicher mit automatischer Leistungsoptimierung“ auf Seite 295

Zugehörige Tasks:

- „Aktivieren des Speichers mit automatischer Leistungsoptimierung“ auf Seite 296

Zugehörige Referenzen:

- „ALTER BUFFERPOOL statement“ in *SQL Reference, Volume 2*
- „database_memory - Größe des gemeinsam genutzten Datenbankspeichers “ auf Seite 434
- „locklist - Maximaler Speicher für Sperrenliste “ auf Seite 438
- „maxlocks - Maximale Anzahl Sperren pro Anwendung “ auf Seite 474
- „pckcachesz - Größe des Paketcache “ auf Seite 442
- „sheapthres - Schwellenwert für Sortierspeicher “ auf Seite 456
- „sheapthres_shr - Schwellenwert für Sortierspeicher für gemeinsame Sortiervor- gänge “ auf Seite 446
- „sortheap - Zwischenspeichergroße für Sortierlisten “ auf Seite 458

sheapthres_shr - Schwellenwert für Sortierspeicher für gemein- same Sortiervorgänge

Konfigurationstyp	Datenbank
Parametertyp	Online konfigurierbar
Weitergabeklasse	Sofort
Standardwert [Bereich]	

32-Bit-Plattformen

5000 [Automatic, 250 - 2.097.152]

64-Bit-Plattformen

5000 [Automatic, 250 - 2.147.483.647
]

Maßeinheit	Seiten (4 KB)

Dieser Parameter gibt einen veränderlichen Grenzwert für die Gesamtmenge des gemeinsam genutzten Datenbankspeichers an, die gleichzeitig von Sortierspeicherverbrauchern verwendet werden kann. Neben Sortiervorgängen nehmen auch andere Verbraucher Sortierspeicher in Anspruch, wie zum Beispiel Hash-Joins, logisches Verknüpfen von Indizes und Blockindizes über AND (Index ANDing), Mischjoins und Tabellen im Speicher. Wenn sich die Gesamtgröße des gemeinsam genutzten Speichers für gemeinsame Sortierspeicherverbraucher dem Grenzwert *sheapthres_shr* nähert, wird ein Drosselungsmechanismus aktiviert, so dass weiteren Anforderungen durch Verbraucher von gemeinsam genutztem Sortierspeicher eventuell weniger Speicher als angefordert zugeordnet wird. Jedoch wird in jedem Fall mehr als das Minimum zugeordnet, das zur Ausführung der Operation erforderlich ist. Wenn der Grenzwert *sheapthres_shr* überschritten wird, erhalten alle Anforderungen von gemeinsam genutztem Sortierspeicher durch Sortierspeicherverbraucher nur die minimale Größe an Speicher, die erforderlich ist, um die Operation zu beenden. Wenn die Gesamtmenge des gemeinsam genutzten Speichers für aktive gemeinsame Sortiervorgänge diesen Grenzwert erreicht, können nachfolgende Sortiervorgänge fehlschlagen (SQL0955C).

Wenn der Datenbankkonfigurationsparameter *sheapthres* den Wert 0 besitzt, verwenden alle Sortierspeicherverbraucher für die Datenbank den gemeinsam genutzten Datenbankspeicher mit *sheapthres_shr* an Stelle des privaten Sortierspeichers.

Der Wert AUTOMATIC aktiviert *sheapthres_shr* für die automatische Leistungsoptimierung. Dadurch ist die Speicheroptimierungsfunktion in der Lage, die durch diesen Parameter gesteuerte Größe des Hauptspeicherbereichs an geänderte Auslastungsanforderungen je nach Bedarf dynamisch anzupassen. Da die Speicheroptimierungsfunktion Speicherressourcen auf verschiedene Speicherverbraucher verteilt, müssen mindestens zwei Speicherverbraucher für die automatische Optimierung aktiviert sein, damit eine automatische Optimierung erfolgen kann.

Die automatische Optimierung von *sheapthres_shr* ist nur zulässig, wenn der Datenbankkonfigurationsparameter *sheapthres* auf den Wert 0 gesetzt ist.

Der Wert des Parameters *sortheap* wird zusammen mit dem Parameter *sheapthres_shr* optimiert, so dass eine Inaktivierung der automatischen Optimierung des Parameters *sortheap* automatisch auch eine Inaktivierung der automatischen Optimierung des Parameters *sheapthres_shr* bewirkt. Die Aktivierung der automatischen Optimierung für den Parameter *sheapthres_shr* führt automatisch zu einer Aktivierung der automatischen Optimierung des Parameters *sortheap*.

Wenn der Wert dieses Parameters online aktualisiert wird, wird der neue Wert nur von neuen Anforderungen für gemeinsam genutzten Sortierspeicher verwendet, die nach der Aktualisierung erfolgen. Es wird empfohlen, vor einer Verringerung des Werts von *sheapthres_shr* den Wert von *sortheap* herabzusetzen und vor einer Erhöhung des Werts von *sortheap* den Wert von *sheapthres_shr* heraufzusetzen.

Wenn der Wert des Datenbankkonfigurationsparameters *sheapthres* größer als 0 ist, ist der Parameter *sheapthres_shr* nur in zwei Fällen von Bedeutung:

- Wenn der Konfigurationsparameter *intra_parallel* des Datenbankmanagers auf *yes* gesetzt ist, da keine gemeinsamen Sortiervorgänge durchgeführt werden, wenn *intra_parallel* auf *no* gesetzt ist.
- Wenn der Konzentrator aktiviert ist (d. h. wenn *max_connections* größer als *max_coordagents* ist), da Sortiervorgängen, die einen mit der Option WITH HOLD deklarierten Cursor verwenden, Speicher aus dem gemeinsam genutzten Speicher zugeordnet wird.

Zugehörige Referenzen:

- „GET DATABASE CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE CONFIGURATION command“ in *Command Reference*
- „sheapthres - Schwellenwert für Sortierspeicher “ auf Seite 456

util_heap_sz - Zwischenspeichergröße für Dienstprogramme

Konfigurationstyp	Datenbank
Parametertyp	Online konfigurierbar
Weitergabeklasse	Sofort
Standardwert [Bereich]	5000 [16 – 524 288]
Maßeinheit	Seiten (4 KB)
Zuordnung	Wenn für die Dienstprogramme des Datenbankmanagers erforderlich
Freigabe	Wenn der Speicher nicht mehr vom Dienstprogramm benötigt wird

Dieser Parameter gibt die maximale Speichergröße an, die gleichzeitig von den Dienstprogrammen BACKUP, RESTORE und LOAD (einschließlich Recovery) verwendet werden kann.

Empfehlung: Verwenden Sie den Standardwert, bis Ihren Dienstprogrammen nicht mehr genügend Speicher zur Verfügung steht. Wenn der Speicher nicht ausreicht, müssen Sie den Wert erhöhen. Wenn der auf Ihrem System verfügbare Speicher eingeschränkt ist, können Sie den Wert für diesen Parameter herabsetzen, um den von den Dienstprogrammen der Datenbank verwendeten Speicher zu begrenzen. Wenn ein zu niedriger Parameterwert angegeben wird, können Sie Dienstprogramme eventuell nicht gleichzeitig ausführen. Sie sollten diesen Parameter nach Bedarf dynamisch aktualisieren. Für eine kleine Anzahl von Dienstprogrammen setzen Sie diesen Parameter auf einen kleinen Wert. Für eine große Anzahl von Dienstprogrammen oder für speicherintensive Dienstprogramme sollten Sie diesen Parameter auf einen großen Wert setzen.

Zugehörige Referenzen:

- „GET DATABASE CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE CONFIGURATION command“ in *Command Reference*

Gemeinsamer Anwendungsspeicher

Die folgenden Parameter geben den Arbeitsbereich an, der von allen Agenten (sowohl koordinierende als auch Subagenten) verwendet wird, die für eine Anwendung arbeiten:

- „app_ctl_heap_sz - Zwischenspeichergröße für Anwendungssteuerung “ auf Seite 449
- „appgroup_mem_sz - Maximale Speichergröße für Anwendungsgruppe “ auf Seite 450
- „groupheap_ratio - Für Anwendungsgruppenzwischenspeicher vorgesehener Speicher in Prozent “ auf Seite 451

app_ctl_heap_sz - Zwischenspeichergröße für Anwendungssteuerung

Konfigurationstyp

Datenbank

Parametertyp

Konfigurierbar

Standardwert [Bereich]

Datenbankserver mit lokalen und fernen Clients

128 [1 - 64.000], wenn INTRA-
_PARALLEL nicht aktiviert ist

512 [1 - 64.000], wenn INTRA-
_PARALLEL aktiviert ist

Datenbankserver mit lokalen Clients

64 [1 - 64.000] (Nicht-UNIX-Platt-
formen), wenn INTRA_PARALLEL
nicht aktiviert ist

512 [1 - 64.000] (Nicht-UNIX-Platt-
formen), wenn INTRA_PARALLEL
aktiviert ist

128 [1 - 64.000] (UNIX-basierte
Plattformen), wenn INTRA-
_PARALLEL nicht aktiviert ist

512 [1 - 64.000] (UNIX-basierte
Plattformen), wenn INTRA-
_PARALLEL aktiviert ist

Partitionierter Datenbankserver mit lokalen und fernen Clients

512 [1 - 64.000]

Maßeinheit

Seiten (4 KB)

Zuordnung

Beim Starten einer Anwendung

Freigabe

Beim Beenden einer Anwendung

Bei partitionierten und nicht partitionierten Datenbanken mit aktivierter partitions-interner Parallelität (intra_parallel=ON) gibt dieser Parameter die durchschnittliche Größe des gemeinsam benutzten Speicherbereichs an, der einer Anwendung zugeordnet ist. Bei nicht partitionierten Datenbanken, bei denen die partitions-interne Parallelität nicht aktiviert ist (intra_parallel=OFF), ist dies der maximale private Speicher, der dem Zwischenspeicher zugeordnet wird. Pro Verbindung einer Datenbankpartition gibt es einen Zwischenspeicher für Anwendungssteuerung.

Der Zwischenspeicher für Anwendungssteuerung wird in erster Linie für die gemeinsame Benutzung von Informationen durch Agenten benötigt, die für dieselbe Anforderung arbeiten. Die Auslastung dieses Zwischenspeichers ist bei nicht partitionierten Datenbanken minimal, wenn Abfragen mit einem Parallelitätsgrad gleich 1 ausgeführt werden.

Dieser Zwischenspeicher wird auch zum Speichern von Deskriptorinformationen für deklarierte temporäre Tabellen verwendet. Die Deskriptorinformationen für alle deklarierten temporären Tabellen, die nicht explizit gelöscht wurden, werden in diesem Zwischenspeicher aufbewahrt und können erst nach dem Löschen der deklarierten temporären Tabelle gelöscht werden.

Empfehlung: Verwenden Sie zunächst den Standardwert. Sie müssen den Wert eventuell erhöhen, wenn Sie komplexe Anwendungen ausführen oder ein System mit zahlreichen Datenbankpartitionen bzw. deklarierte temporäre Tabellen verwenden. Die erforderliche Speicherkapazität erhöht sich mit der Anzahl deklarerter temporärer Tabellen, die gleichzeitig aktiv sind. Der Tabellendeskriptor einer deklarierten temporären Tabelle mit vielen Spalten ist größer als jener einer Tabelle mit wenigen Spalten. Daher erhöht eine große Anzahl Spalten in den deklarierten temporären Tabellen einer Anwendung auch den Bedarf an Zwischenspeicher zur Anwendungssteuerung.

Zugehörige Referenzen:

- „intra_parallel - Partitionsinterne Parallelität aktivieren “ auf Seite 559
- „applheapsz - Zwischenspeichergröße für Anwendungen “ auf Seite 454
- „appgroup_mem_sz - Maximale Speichergröße für Anwendungsgruppe “ auf Seite 450
- „groupheap_ratio - Für Anwendungsgruppenzwischenspeicher vorgesehener Speicher in Prozent “ auf Seite 451
- „GET DATABASE CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE CONFIGURATION command“ in *Command Reference*

appgroup_mem_sz - Maximale Speichergröße für Anwendungsgruppe

Konfigurationstyp

Datenbank

Parametertyp

Konfigurierbar

Standardwert [Bereich]

UNIX-Datenbankserver mit lokalen Clients (nicht 32-Bit-HP-UX) 20 000 [1 – 1 000 000]

32-Bit-HP-UX

- Datenbankserver mit lokalen Clients
- Datenbankserver mit lokalen und fernen Clients
- Partitionierter Datenbankserver mit lokalen und fernen Clients

10 000 [1 – 1 000 000]

Windows-Datenbankserver mit lokalen Clients

10 000 [1 – 1 000 000]

Datenbankserver mit lokalen und fernen Clients (nicht 32-Bit-HP-UX)

30 000 [1 – 1 000 000]

Partitionierter Datenbankserver mit lokalen und fernen Clients (nicht 32-Bit-HP-UX)

40 000 [1 – 1 000 000]

Maßeinheit

Seiten (4 KB)

Dieser Parameter legt die Größe des von der Anwendungsgruppe gemeinsam benutzten Speichersegments fest. Informationen, die den für dieselbe Anwendung

arbeitenden Agenten gemeinsam zur Verfügung stehen müssen, werden im gemeinsam benutzten Speichersegment der Anwendungsgruppe gespeichert.

In einer partitionierten Datenbank oder in einer nicht partitionierten Datenbank mit aktivierter partitionsinterner Parallelität oder aktiviertem Konzentrator benutzen mehrere Anwendungen eine Anwendungsgruppe gemeinsam. Der Anwendungsgruppe wird ein gemeinsam benutztes Speichersegment zugeordnet. Innerhalb des gemeinsam benutzten Speichersegments der Anwendungsgruppe ist für jede Anwendung ein eigener Zwischenspeicher für die Anwendungssteuerung vorhanden, und alle Anwendungen der Anwendungsgruppe benutzen gemeinsam einen Zwischenspeicher.

Die Anzahl Anwendungen in einer Anwendungsgruppe wird wie folgt berechnet:
$$\text{appgroup\_mem\_sz} / \text{app\_ctl\_heap\_sz}$$

Die Größe des von der Anwendungsgruppe gemeinsam benutzten Zwischenspeichers wird wie folgt berechnet:
$$\text{appgroup\_mem\_sz} * \text{groupheap\_ratio} / 100$$

Die Größe des Zwischenspeichers für die Anwendungssteuerung wird für jede Anwendung wie folgt berechnet:
$$\text{app\_ctl\_heap\_sz} * (100 - \text{groupheap\_ratio}) / 100$$

Empfehlung: Behalten Sie den Standardwert für diesen Parameter bei, solange Sie keine Leistungsprobleme feststellen.

Zugehörige Referenzen:

- „app_ctl_heap_sz - Zwischenspeichergröße für Anwendungssteuerung “ auf Seite 449
- „groupheap_ratio - Für Anwendungsgruppenzwischenspeicher vorgesehener Speicher in Prozent “ auf Seite 451
- „GET DATABASE CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE CONFIGURATION command“ in *Command Reference*

groupheap_ratio - Für Anwendungsgruppenzwischenspeicher vorgesehener Speicher in Prozent

Konfigurationstyp	Datenbank
Parametertyp	Konfigurierbar
Standardwert [Bereich]	70 [1 – 99]
Maßeinheit	Prozent

Dieser Parameter gibt den Prozentsatz an Speicher im gemeinsam benutzten Speicher zur Anwendungssteuerung an, der für den von der Anwendungsgruppe gemeinsam benutzten Zwischenspeicher vorgesehen ist.

Dieser Parameter hat keine Auswirkungen auf eine nicht partitionierte Datenbank mit inaktiviertem Konzentrator und inaktivierter partitionsinterner Parallelität.

Empfehlung: Behalten Sie den Standardwert für diesen Parameter bei, solange Sie keine Leistungsprobleme feststellen.

Zugehörige Referenzen:

- „app_ctl_heap_sz - Zwischenspeichergröße für Anwendungssteuerung ” auf Seite 449
- „appgroup_mem_sz - Maximale Speichergröße für Anwendungsgruppe ” auf Seite 450
- „GET DATABASE CONFIGURATION command” in *Command Reference*
- „RESET DATABASE CONFIGURATION command” in *Command Reference*
- „UPDATE DATABASE CONFIGURATION command” in *Command Reference*

Privater Agentenspeicher

Die folgenden Parameter wirken sich auf die Menge an Speicher aus, die für jeden Datenbankagenten verwendet wird:

- „agent_stack_sz - Größe des Agentenstacks ”
- „applheapsz - Zwischenspeichergröße für Anwendungen ” auf Seite 454
- „query_heap_sz - Größe des Abfragezwischenspeichers ” auf Seite 455
- „sheapthres - Schwellenwert für Sortierspeicher ” auf Seite 456
- „sortheap - Zwischenspeichergröße für Sortierlisten ” auf Seite 458
- „stat_heap_sz - Größe des Statistikzwischenspeichers ” auf Seite 459
- „stmtheap - Größe des Anweisungszwischenspeichers ” auf Seite 460

agent_stack_sz - Größe des Agentenstacks

Konfigurationstyp

Datenbankmanager

Gilt für

- Datenbankserver mit lokalen und fernen Clients
- Datenbankserver mit lokalen Clients
- Partitionierten Datenbankserver mit lokalen und fernen Clients

Parametertyp

Konfigurierbar

Standardwert [Bereich]

16 [8 – 1000]

Maßeinheit

Seiten (4 KB)

Zuordnung

Wenn ein Agent zur Arbeit für eine Anwendung initialisiert wird

Freigabe

Wenn ein Agent die Arbeit für eine Anwendung beendet

Der Agentenstack ist der virtuelle Speicherbereich, den DB2 jedem Agenten zuordnet. Dieser Speicher wird festgeschrieben, wenn er zur Verarbeitung einer SQL- oder XQuery-Anweisung erforderlich ist. Sie können diesen Parameter zur Optimierung der Speicherauslastung des Servers für einen bestimmten Satz von Anwendungen verwenden. Komplexere Abfragen benötigen im Vergleich zu einfachen Abfragen mehr Stapelspeicherbereich.

Mit diesem Parameter wird die ursprüngliche festgeschriebene Stackgröße für jeden Agenten in einer Windows-Umgebung festgelegt. Jeder Agentenstack kann standardmäßig bis auf die Größe des Standardreservestacks anwachsen, d. h. bis auf 256 KB (64 4-KB-Seiten). Dieser Grenzwert ist für die meisten Datenbankoperationen ausreichend.

Wenn Sie jedoch eine umfangreiche SQL- oder XQuery-Anweisung vorbereiten, kann für den Agenten möglicherweise ein größerer Stackbereich erforderlich sein, und das System generiert eine Ausnahmebedingung aufgrund eines Stacküberlaufs (0xC000000D). Wenn dies eintritt, fährt der Server herunter, da der Fehler nicht behebbar ist.

Die Größe des Agentenstacks kann erhöht werden, indem Sie den Parameter *agent_stack_sz* auf einen höheren Wert als die Standardreservestackgröße von 64 Seiten setzen. Wenn der Wert des Parameters *agent_stack_sz* über der Standardreservestackgröße liegt, wird er vom Windows-Betriebssystem auf das nächste Vielfache von 1 MB gerundet. Wird die Agentenstackgröße beispielsweise auf 128 4-KB-Seiten erhöht, wird für jeden Agenten in Wirklichkeit ein Stack der Größe 1 MB reserviert. Wenn Sie für *agent_stack_sz* einen Wert festlegen, der unter der Standardreservestackgröße liegt, hat dies keine Auswirkungen auf die maximale Begrenzung, da der Agentenstack trotzdem bei Bedarf auf die Größe des Standardreservestacks anwächst. In diesem Fall ist der Wert für *agent_stack_sz* der ursprüngliche festgeschriebene Speicher für den Stack, wenn ein Agent erstellt wird.

Sie können die Standardreservestackgröße ändern, indem Sie mit dem Dienstprogramm db2hdr die Headerdaten für die Datei db2syscs.exe ändern. Wenn Sie die Standardreservestackgröße ändern, wirkt sich dies auf alle Threads aus, während sich eine Änderung des Werts für *agent_stack_sz* nur auf die Stackgröße für Agenten auswirkt. Das Ändern der Standardstackgröße mit Hilfe des Dienstprogramms db2hdr bietet den Vorteil einer besseren Granularität, wodurch die Stackgröße auf die minimal erforderliche Stackgröße festgelegt werden kann. Sie müssen jedoch DB2 stoppen und neu starten, damit die an db2syscs.exe vorgenommenen Änderungen wirksam werden.

Empfehlung: Wenn Sie beabsichtigen, mit umfangreichen oder komplexen XML-Daten in einer 32-Bit-Umgebung zu arbeiten, sollten Sie den Parameter *agent_stack_sz* auf mindestens 256 4-KB-Seiten aktualisieren. Bei sehr komplexen XML-Schemata kann es erforderlich sein, den Parameter *agent_stack_sz* wesentlich näher an den Grenzwert zu setzen, um Ausnahmebedingungen wegen Stacküberläufen beim Registrieren von Schemata oder Prüfen von XML-Dokumenten zu vermeiden.

Sie können die Stackgröße eventuell verringern, um anderen Clients mehr Adressraum zur Verfügung zu stellen. Dies ist möglich, wenn Ihre Umgebung folgende Kriterien erfüllt:

- Die Umgebung enthält nur einfache Anwendungen (z. B. einfache Online-Transaktionsprogramme, OLTP), in denen es keine komplexen Abfragen gibt.
- Für die Umgebung sind relativ viele gleichzeitig aktive Clients erforderlich z. B. über 100).

Die Größe des Agentenstapelspeichers und die Anzahl gleichzeitig ausgeführter Clients stehen in einem umgekehrten Verhältnis zueinander: Durch eine Vergrößerung des Stapelspeichers wird die mögliche Anzahl der Clients verringert, die gleichzeitig ausgeführt werden können. Die Ursache hierfür ist, dass der Adressraum auf Windows-Plattformen begrenzt ist.

Dieser Parameter gilt nicht für auf UNIX basierende Plattformen.

Zugehörige Referenzen:

- „GET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*

- „RESET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE MANAGER CONFIGURATION command“ in *Command Reference*

applheapsz - Zwischenspeichergröße für Anwendungen

Konfigurationstyp	Datenbank
Parametertyp	Konfigurierbar
Standardwert [Bereich]	32-Bit-Datenbankserver mit lokalen und fernen Clients 256 [16 – 60 000] 64-Bit-Datenbankserver mit lokalen und fernen Clients 256 [16 – 60 000] Partitionierter 32-Bit-Datenbankserver mit lokalen und fernen Clients 64 [16 – 60 000] Partitionierter 64-Bit-Datenbankserver mit lokalen und fernen Clients 128 [16 – 60 000]
Maßeinheit	Seiten (4 KB)
Zuordnung	Wenn ein Agent zur Arbeit für eine Anwendung initialisiert wird
Freigabe	Wenn ein Agent die Arbeit für eine Anwendung beendet

Mit diesem Parameter wird die Anzahl der Seiten an privatem Speicher definiert, die zur Verwendung durch den Datenbankmanager für einen bestimmten Agenten oder Subagenten zur Verfügung stehen.

Der Zwischenspeicher wird zugeordnet, wenn ein Agent oder Subagent für eine Anwendung initialisiert wird. Die zugeordnete Speichermenge ist die Mindestmenge, die zur Verarbeitung der an den Agenten oder Subagenten übergebenen Anforderung benötigt wird. Wenn der Agent oder Subagent mehr Zwischenspeicher zur Verarbeitung größerer SQL- oder XQuery-Anweisungen benötigt, ordnet der Datenbankmanager weiteren Speicher nach Bedarf bis zu der durch diesen Parameter angegebenen Obergrenze zu.

Anmerkungen:

1.

Anmerkung: In einer Umgebung mit partitionierten Datenbanken wird der Zwischenspeicher für die Anwendungssteuerung (*app_ctl_heap_sz*) verwendet, um Kopien der in Ausführung befindlichen Abschnitte von SQL-Anweisungen für Agenten und Subagenten zu speichern. SMP-Subagenten verwenden hingegen den Parameter *applheapsz*, wie es auch Agenten in allen anderen Umgebungen tun.

2.

Anmerkung: Umfangreiche XML-Schemata erfordern ausreichend Speicher im Anwendungszwischenspeicher. Es ist empfehlenswert, den Anwendungszwischenspeicher wie folgt zu vergrößern:

```
db2 update db cfg using applheapsz 1000
```

Empfehlung: Erhöhen Sie den Wert dieses Parameters, wenn Ihre Anwendungen einen Fehler empfangen, weil im Anwendungszwischenspeicher nicht genügend Speicher verfügbar ist.

Der Anwendungszwischenspeicher (*applheapsz*) wird aus dem privaten Agentenspeicher heraus zugeordnet.

Zugehörige Referenzen:

- „app_ctl_heap_sz - Zwischenspeichergröße für Anwendungssteuerung “ auf Seite 449
- „GET DATABASE CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE CONFIGURATION command“ in *Command Reference*

query_heap_sz - Größe des Abfragezwischenspeichers

Konfigurationstyp

Datenbankmanager

Gilt für

- Datenbankserver mit lokalen und fernen Clients
- Datenbankserver mit lokalen Clients
- Partitionierten Datenbankserver mit lokalen und fernen Clients

Parametertyp

Konfigurierbar

Standardwert [Bereich]

1000 [2 – 524 288]

Maßeinheit

Seiten (4 KB)

Zuordnung

Wenn eine Anwendung (lokal oder fern) eine Verbindung zur Datenbank herstellt

Freigabe

Wenn die Anwendung die Verbindung zur Datenbank oder zur Instanz trennt

Mit diesem Parameter wird die **maximale** Speichermenge angegeben, die für den Abfragezwischenspeicher zugeordnet werden kann. Ein Abfragezwischenspeicher wird zur Speicherung jeder Abfrage im privaten Speicher des Agenten verwendet. Die Informationen für jede Abfrage bestehen aus dem Eingabe- und Ausgabe-SQL-Deskriptorbereich, dem Text der Anweisung, dem SQL-Kommunikationsbereich, dem Paketnamen, dem Ersteller, der Abschnittsnummer und dem Konsistenz-Token. Mit diesem Parameter kann sichergestellt werden, dass eine Anwendung nicht unnötig große virtuelle Speicherbereiche innerhalb eines Agenten belegt.

Aus dem Abfragezwischenspeicher wird auch der Speicher für Blockcursor zugeordnet. Dieser Speicher besteht aus einem Cursorsteuerungsblock und einem vollständig formatierten Ausgabe-SQL-Deskriptorbereich.

Zu Anfang ist der zugeordnete Abfragezwischenspeicher ebenso groß wie der Zwischenspeicher für die Anwendungsunterstützungsebene, der durch den Parameter *aslheapsz* definiert ist. Der Abfragezwischenspeicher muss größer oder gleich 2 sowie größer oder gleich dem Wert des Parameters *aslheapsz* sein. Wenn dieser

Abfragezwischenspeicher zur Verarbeitung einer Anforderung nicht ausreicht, wird neuer Speicher in der für die Anforderung erforderlichen Größe (nicht über den Wert *query_heap_sz* hinaus) zugeordnet. Wenn dieser neue Abfragezwischenspeicher mehr als anderthalbmal so groß wie der Wert für *aslheapsz* ist, wird der Abfragezwischenspeicher in der Größe von *aslheapsz* neu zugeordnet, wenn die Abfrage beendet ist.

Empfehlung: In den meisten Fällen ist der Standardwert ausreichend. Als Minimalwert sollte für *query_heap_sz* ein Wert angegeben werden, der mindestens fünfmal so groß wie der Wert für *aslheapsz* ist. Dadurch können Abfragen größer als *aslheapsz* sein, und es wird zusätzlicher Speicher für drei oder vier Blockcursor bereitgestellt, die gleichzeitig geöffnet sein können.

Wenn Sie sehr umfangreiche LOBs (große Objekte) haben, müssen Sie möglicherweise den Wert dieses Parameters erhöhen, damit der Abfragezwischenspeicher groß genug für diese LOBs ist.

Zugehörige Referenzen:

- „aslheapsz - Zwischenspeichergröße für Anwendungsunterstützungsebene “ auf Seite 461
- „GET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE MANAGER CONFIGURATION command“ in *Command Reference*

sheapthres - Schwellenwert für Sortierspeicher

Konfigurationstyp	Datenbankmanager
Gilt für	• Datenbankserver mit lokalen und fernen Clients • Datenbankserver mit lokalen Clients • Partitionierten Datenbankserver mit lokalen und fernen Clients
Parametertyp	Online konfigurierbar
Weitergabeklasse	Sofort
Standardwert [Bereich]	UNIX-Plattformen (32-Bit) 0 [0 - 2.097.152] Windows-Plattformen (32-Bit) 0 [0 - 2.097.152] 64-Bit-Plattformen 0 [0 - 2.147.483.647]
Maßeinheit	Seiten (4 KB)

Dieser Parameter ein instanzweiter, veränderlicher Grenzwert für den maximalen Speicherbereich, der zu einem beliebigen Zeitpunkt von privaten Sortiervorgängen beansprucht werden kann. Wenn die Gesamtnutzung an privatem Sortierspeicher-

bereich für eine Instanz diesen Grenzwert erreicht, wird der für weitere eingehende private Sortieranforderungen zugeordnete Speicherbereich beträchtlich verkleinert.

Der Sortierspeicher wird beispielsweise bei folgenden Operationen verwendet: Sortierungen, Hash-Joins, dynamische Bitzuordnungen (die für logisches Verknüpfen von Indizes über AND (Index ANDing) und Sternjoins verwendet werden) und Operationen, bei denen sich die Tabelle im Speicher befindet.

Durch die explizite Angabe des Schwellenwerts wird verhindert, dass der Datenbankmanager übermäßig große Speichermengen für sehr viele Sortiervorgänge verwendet.

Es gibt keinen Grund, den Wert dieses Parameter beim Versetzen von einer Umgebung mit nicht partitionierten Datenbanken in eine Umgebung mit partitionierten Datenbanken zu erhöhen. Wenn Sie die Konfigurationsparameter der Datenbank und des Datenbankmanagers in einer Umgebung mit einer nicht partitionierten Datenbank optimiert haben, funktionieren diese Werte in einer Umgebung mit partitionierten Datenbanken zumeist ebenfalls einwandfrei. Sie können diesen Parameter in anderen Knoten oder Datenbankpartitionen nur auf unterschiedliche Werte setzen, indem Sie mehrere DB2-Instanzen erstellen. Dies erfordert das Verwalten verschiedener DB2-Datenbanken in verschiedenen Datenbankpartitionsgruppen. Ein solches Vorgehen macht viele Vorteile einer Umgebung mit partitionierten Datenbanken zunichte.

Wenn der Parameter *sheapthres* der Instanzebene auf den Wert 0 gesetzt ist, erfolgt die Verfolgung der Sortierspeichernutzung nur auf der Datenbankebene und die Speicherzuordnung für Sortiervorgänge wird durch den Wert des Konfigurationsparameters *sheapthres_shr* auf der Datenbankebene begrenzt.

Die automatische Optimierung von *sheapthres_shr* ist nur zulässig, wenn der Datenbankkonfigurationsparameter *sheapthres* auf den Wert 0 gesetzt ist.

Dieser Parameter ist nicht dynamisch aktualisierbar, wenn eine der folgenden Bedingungen zutrifft:

- Der Anfangswert für *sheapthres* ist 0, und der Zielwert ist ungleich 0.
- Der Anfangswert für *sheapthres* ist ungleich 0, und der Zielwert ist 0.

Empfehlung: Im Idealfall sollten Sie den Wert dieses Parameters möglichst auf ein angemessenes Vielfaches des Parameters *sortheap* mit dem größten Wert setzen, der in Ihrer Datenbankmanagerinstanz definiert ist. Der Wert dieses Parameters sollte **mindestens** zweimal so groß sein wie der größte Sortierspeicher (*sortheap*), der für eine Datenbank in der Instanz definiert ist.

Wenn Sie private Sortiervorgänge ausführen und Ihr System über genügend Speicher verfügt, kann der Idealwert für diesen Parameter auf folgende Weise berechnet werden:

1. Berechnen Sie die normale Sortierspeicherbelegung für jede Datenbank:
(normale Anzahl Agenten, die gleichzeitig für die Datenbank ausgeführt werden)
* (sortheap, wie für die Datenbank definiert)
2. Berechnen Sie die Summe der obigen Ergebnisse. Dies ergibt einen Wert für den gesamten Sortierspeicher, der unter normalen Umständen für alle Datenbanken innerhalb der Instanz verwendet werden kann.

Sie sollten Vergleichstests (Benchmark-Tests) ausführen, um die optimale Einstellung für diesen Parameter zu ermitteln, die Sortierleistung und Speicherbedarf gleichermaßen berücksichtigt.

Mit Hilfe des Datenbanksystemmonitors können Sie unter Verwendung des Monitorelements *post_threshold_sorts* (Sortierungen nach Erreichen des Schwellenwerts) die Sortieraktivitäten verfolgen.

Zugehörige Referenzen:

- „post_threshold_sorts - Post Threshold Sorts monitor element“ in *Systemmonitor Handbuch und Referenz*
- „sheaphres_shr - Schwellenwert für Sortierspeicher für gemeinsame Sortiervorgänge “ auf Seite 446
- „sortheap - Zwischenspeichergröße für Sortierlisten “ auf Seite 458
- „GET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE MANAGER CONFIGURATION command“ in *Command Reference*

sortheap - Zwischenspeichergröße für Sortierlisten

Konfigurationstyp	Datenbank
Parametertyp	Online konfigurierbar
Weitergabeklasse	Sofort
Standardwert [Bereich]	32-Bit-Plattformen Automatic [16 - 524.288] 64-Bit-Plattformen Automatic [16 - 4.194.303]
Maßeinheit	Seiten (4 KB)
Zuordnung	Wie zur Ausführung von Sortiervorgängen erforderlich
Freigabe	Wenn der Sortiervorgang abgeschlossen ist

Mit diesem Parameter wird die maximale Anzahl von Seiten des privaten Speichers definiert, die für private Sortiervorgänge verwendet werden soll, bzw. die maximale Anzahl von Seiten des gemeinsamen Speichers, die für gemeinsame Sortiervorgänge verwendet werden soll. Wenn es sich um einen privaten Sortiervorgang handelt, bezieht sich dieser Parameter auf den privaten Agentenspeicher. Handelt es sich um einen gemeinsamen Sortiervorgang, bezieht sich dieser Parameter auf den gemeinsamen Datenbankspeicher. Jeder Sortiervorgang verwendet einen getrennten Sortierspeicher, der bei Bedarf vom Datenbankmanager zugeordnet wird. Dieser Sortierspeicher ist der Bereich, in dem Daten sortiert werden. Bei Steuerung durch das Optimierungsprogramm wird anhand der vom Optimierungsprogramm bereitgestellten Informationen ein kleinerer als der durch diesen Parameter angegebene Sortierspeicher zugeordnet.

Der Wert AUTOMATIC für diesen Parameter aktiviert ihn für die automatische Leistungsoptimierung. Dadurch ist die Speicheroptimierungsfunktion in der Lage,

die durch diesen Parameter gesteuerte Größe des Hauptspeicherbereichs an geänderte Auslastungsanforderungen je nach Bedarf dynamisch anzupassen.

Der Wert des Parameters *sortheap* wird zusammen mit dem Parameter *sheapthres_shr* optimiert, so dass eine Inaktivierung der automatischen Optimierung des Parameters *sortheap* automatisch auch eine Inaktivierung der automatischen Optimierung des Parameters *sheapthres_shr* bewirkt. Die Aktivierung der automatischen Optimierung für den Parameter *sheapthres_shr* führt automatisch zu einer Aktivierung der automatischen Optimierung des Parameters *sortheap*. Der Parameter *sortheap* kann jedoch für die automatische Optimierung aktiviert werden, ohne dass der Parameter *sheapthres_shr* auf AUTOMATIC gesetzt ist.

Die automatische Optimierung von *sortheap* ist nur zulässig, wenn der Datenbankkonfigurationsparameter *sheapthres* auf den Wert 0 gesetzt ist.

Empfehlung: Bei der Arbeit mit dem Sortierspeicher ist Folgendes zu beachten:

- Geeignete Indizes können die Verwendung des Sortierspeichers minimieren.
- Hash-Join-Puffer, logisches Verknüpfen von Blockindizes über AND (Index ANDing), Mischjoins, Tabellen im Speicher und dynamische Bitzuordnungen (die für logisches Verknüpfen von Indizes über AND (Index ANDing) und Sternjoins verwendet werden) nutzen Sortierspeicher. Erhöhen Sie den Wert dieses Parameters, wenn diese Verfahren verwendet werden.
- Erhöhen Sie den Wert dieses Parameters, wenn häufig umfangreiche Sortiervorgänge ausgeführt werden müssen.
- Wenn Sie den Wert dieses Parameters erhöhen, sollten Sie überprüfen, ob auch die Werte der Parameter *sheapthres* und *sheapthres_shr* in der Konfigurationsdatei des Datenbankmanagers angepasst werden müssen.
- Die Größe des Zwischenspeichers für Sortierlisten wird vom Optimierungsprogramm zur Bestimmung der Zugriffspfade verwendet. Wenn Sie diesen Parameter geändert haben, sollten Sie für die Anwendungen eventuell einen Rebind durchführen (mit dem Befehl REBIND).

Zugehörige Konzepte:

- „Speicher mit automatischer Leistungsoptimierung“ auf Seite 295

Zugehörige Referenzen:

- „REBIND command“ in *Command Reference*
- „GET DATABASE CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE CONFIGURATION command“ in *Command Reference*
- „self_tuning_mem - Speicher mit automatischer Leistungsoptimierung “ auf Seite 444
- „sheapthres - Schwellenwert für Sortierspeicher “ auf Seite 456
- „sheapthres_shr - Schwellenwert für Sortierspeicher für gemeinsame Sortiervorgänge “ auf Seite 446

stat_heap_sz - Größe des Statistikzweischenspeichers

Konfigurationstyp	Datenbank
Parametertyp	Konfigurierbar
Standardwert [Bereich]	4384 [1096 – 524 288]
Maßeinheit	Seiten (4 KB)

Zuordnung	Wenn das Dienstprogramm RUNSTATS gestartet wird
Freigabe	Wenn das Dienstprogramm RUNSTATS beendet ist

Mit diesem Parameter wird die **maximale** Größe des Zwischenspeichers angegeben, der bei Erfassung statistischer Daten mit dem Befehl RUNSTATS verwendet wird.

Empfehlung: Der Standardwert ist geeignet, wenn keine Verteilungsstatistikdaten gesammelt werden oder wenn die Verteilungsstatistikdaten nur für relativ schmale Tabellen gesammelt werden. Der Minimalwert ist **nicht** zu empfehlen, wenn Verteilungsstatistikdaten gesammelt werden, da nur Tabellen mit einer oder zwei Spalten in den Zwischenspeicher passen.

Dieser Parameter sollte nach der Anzahl der Spalten, für die statistische Daten erfasst werden, angepasst werden. Schmale Tabellen mit relativ wenigen Spalten erfordern weniger Speicher für die zu sammelnden Verteilungsstatistikdaten. Breite Tabellen mit vielen Spalten machen erheblich mehr Speicher erforderlich. Wenn Sie Verteilungsstatistikdaten für sehr breite Tabellen sammeln und einen großen Statistikzwischenpeicher benötigen, sollten Sie die Statistikdaten in einer Phase geringer Systemaktivität sammeln, um die Speicheranforderungen anderer Benutzer nicht einzuschränken.

Zugehörige Referenzen:

- „num_freqvalues - Anzahl der häufigsten Werte “ auf Seite 544
- „num_quantiles - Anzahl der Quantile für Spalten “ auf Seite 546
- „RUNSTATS command“ in *Command Reference*
- „GET DATABASE CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE CONFIGURATION command“ in *Command Reference*

stmtheap - Größe des Anweisungszwischenspeichers

Konfigurationstyp	Datenbank
Parametertyp	Online konfigurierbar
Weitergabeklasse	Anweisungsgrenzwert
Standardwert [Bereich]	32-Bit-Plattformen 2048 [128 – 524 288] 64-Bit-Plattformen 4096 [128 – 524 288]
Maßeinheit	Seiten (4 KB)
Zuordnung	Für jede Anweisung während des Vorkompilierens oder des Bindens
Freigabe	Wenn das Vorkompilieren oder Binden der betreffenden Anweisung abgeschlossen ist

Der Anweisungszwischenspeicher wird als Arbeitsbereich für den SQL- oder XQuery-Compiler während des Kompilierens einer SQL- oder XQuery-Anweisung verwendet. Mit diesem Parameter wird die Größe dieses Arbeitsbereichs angegeben.

Dieser Bereich bleibt nicht permanent zugeordnet, sondern wird für die Verarbeitung jeder SQL- oder XQuery-Anweisung einzeln zugeordnet und anschließend wieder freigegeben. Beachten Sie, dass dieser Arbeitsbereich bei dynamischen SQL- oder XQuery-Anweisungen während der Ausführung Ihres Programms, bei statischen SQL- oder XQuery-Anweisungen hingegen während des Bindens, und nicht während der Ausführung des Programms, verwendet wird.

Empfehlung: In den meisten Fällen kann der Standardwert für diesen Parameter übernommen werden. Wenn Sie sehr große SQL- oder XQuery-Anweisungen verwenden und der Datenbankmanager bei dem Versuch, eine Anweisung zu optimieren, einen Fehler ausgibt (dass die Anweisung zu komplex ist), sollten Sie den Wert dieses Parameters in gleichmäßigen Schritten (z. B. 256 oder 1024) erhöhen, bis die Fehlersituation bereinigt ist.

Zugehörige Referenzen:

- „sortheap - Zwischenspeichergröße für Sortierlisten “ auf Seite 458
- „applheapsz - Zwischenspeichergröße für Anwendungen “ auf Seite 454
- „stat_heap_sz - Größe des Statistikzwischenspeichers “ auf Seite 459
- „GET DATABASE CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE CONFIGURATION command“ in *Command Reference*

Agenten-/Anwendungskommunikationsspeicher

Die folgenden Parameter wirken sich auf die Menge an Speicher aus, der zum Übergeben von Daten zwischen den Anwendungs- und Agentenprozessen zugeordnet wird:

- „aslheapsz - Zwischenspeichergröße für Anwendungsunterstützungsebene “
- „min_dec_div_3 - 3 Kommastellen bei Dezimaldivision “ auf Seite 463
- „rqrioblk - E/A-Blockgröße für Clients “ auf Seite 464

aslheapsz - Zwischenspeichergröße für Anwendungsunterstützungsebene

Konfigurationstyp	Datenbankmanager
Gilt für	• Datenbankserver mit lokalen und fernen Clients • Datenbankserver mit lokalen Clients • Partitionierten Datenbankserver mit lokalen und fernen Clients
Parametertyp	Konfigurierbar
Standardwert [Bereich]	15 [1 – 524 288]
Maßeinheit	Seiten (4 KB)
Zuordnung	Wenn der Agentenprozess des Datenbankmanagers für die lokale Anwendung gestartet wird
Freigabe	Wenn der Agentenprozess des Datenbankmanagers beendet wird

Der Zwischenspeicher für die Anwendungsunterstützungsebene ist ein Kommunikationspuffer zwischen der lokalen Anwendung und dem zugeordneten Agenten.

Dieser Puffer wird als gemeinsam benutzter Speicher von jedem Datenbankmanageragenten, der gestartet wird, zugeordnet.

Wenn die Anforderung an den Datenbankmanager oder die zugehörige Antwort nicht in den Puffer passt, wird sie in zwei oder mehr Sende-/Empfangspufferpaare aufgeteilt. Die Größe dieses Puffers sollte so festgelegt werden, dass die Mehrzahl der Anforderungen mit einem einzigen Sende-/Empfangspufferpaar verarbeitet werden kann. Die Größe der Anforderung hängt von der Speichermenge ab, die zur Speicherung folgender Daten erforderlich ist:

- Der Eingabe-SQL-Deskriptorbereich
- Alle zugeordneten Daten in den SQLVARs
- Der Ausgabe-SQL-Deskriptorbereich
- Andere Felder, die im Allgemeinen 250 Byte nicht überschreiten

Außer zur Steuerung dieses Kommunikationspuffers dient dieser Parameter auch den beiden folgenden Zwecken:

- Dieser Parameter wird verwendet, um die Größe des E/A-Blocks festzulegen, wenn ein Blockcursor geöffnet wird. Dieser Speicher für Blockcursor wird aus dem privaten Adressraum der Anwendung zugeordnet. Sie sollten daher die optimale Größe des privaten Speichers ermitteln, der jedem Anwendungsprogramm zugeordnet werden soll. Wenn der Datenbankclient keinen Bereich für einen Blockcursor aus dem privaten Speicher der Anwendung zuordnen kann, wird ein Cursor ohne Blockung geöffnet.
- Mit diesem Parameter wird die Kommunikationsgröße zwischen Agenten und db2fmp-Prozessen ermittelt. (Ein db2fmp-Prozess kann eine benutzerdefinierte Funktion oder eine abgeschirmte gespeicherte Prozedur sein.) Die Anzahl Byte wird jedem db2fmp-Prozess bzw. jedem im System aktiven Thread aus dem gemeinsam benutzten Speicher zugeordnet.

Die Daten, die von der lokalen Anwendung gesendet werden, werden vom Datenbankmanager in einem Bereich zusammenhängenden Speichers empfangen, der aus dem Abfragezwischenspeicher zugeordnet wird. Mit dem Parameter *aslheapsz* wird die Anfangsgröße des Abfragezwischenspeichers (für lokale und ferne Clients) festgelegt. Die maximale Größe des Abfragezwischenspeichers wird durch den Parameter *query_heap_sz* definiert.

Empfehlung: Wenn die Anforderungen Ihrer Anwendung im Allgemeinen klein sind und die Anwendung auf einem System mit eingeschränkter Speicherkapazität ausgeführt wird, ist es vielleicht sinnvoll, den Wert dieses Parameters zu verringern. Wenn Ihre Abfragen im Allgemeinen sehr groß sind, mehr als eine Sende- und Empfangsanforderung erfordern und die Speicherkapazität Ihres Systems nicht eingeschränkt ist, kann es sinnvoll sein, den Wert dieses Parameters zu erhöhen.

Berechnen Sie anhand der folgenden Formel die Mindestanzahl der Seiten für *aslheapsz*:

```
aslheapsz >= ( sizeof(Eingabe-SQL-Deskriptorbereich)
               + sizeof(jeder Eingabe-SQLVAR)
               + sizeof(Ausgabe-SQL-Deskriptorbereich)
               + 250 ) / 4096
```

Dabei ist `sizeof(x)` die Größe von `x` in Byte zur Berechnung der Seitenanzahl eines bestimmten Eingabe- oder Ausgabewerts.

Beachten Sie außerdem, welche Auswirkung dieser Parameter auf die Anzahl und die mögliche Größe von Blockcursoren hat. Große Zeilenblöcke können zu einer höheren Leistung führen, wenn die Anzahl oder die Größe der Zeilen, die übertragen werden, groß ist (z. B., wenn die Datenmenge größer als 4 096 Byte ist). Dies hat jedoch den Nachteil, dass größere Datensatzblöcke die Größe des für jede Verbindung benötigten Arbeitsspeichers erhöhen.

Größere Satzblöcke können außerdem mehr Abrufanforderungen verursachen, als tatsächlich für die Anwendung erforderlich wären. Sie können die Anzahl der Abrufanforderungen mit der Klausel OPTIMIZE FOR in der Anweisung SELECT in Ihrer Anwendung steuern.

Zugehörige Referenzen:

- „query_heap_sz - Größe des Abfragezwischenspeichers ” auf Seite 455
- „GET DATABASE MANAGER CONFIGURATION command” in *Command Reference*
- „RESET DATABASE MANAGER CONFIGURATION command” in *Command Reference*
- „UPDATE DATABASE MANAGER CONFIGURATION command” in *Command Reference*

min_dec_div_3 - 3 Kommastellen bei Dezimaldivision

Konfigurationstyp	Datenbank
Parametertyp	Konfigurierbar
Standardwert [Bereich]	No [Yes, No]

Der Datenbankkonfigurationsparameter *min_dec_div_3* bietet die Möglichkeit, die Berechnung der Kommastellen bei der Dezimaldivision in SQL schnell zu ändern. Mögliche Werte für *min_dec_div_3* sind "Yes" und "No". Der Standardwert für *min_dec_div_3* ist "No".

Der Datenbankkonfigurationsparameter *min_dec_div_3* ändert die Anzahl der Kommastellen im Ergebnis einer Dezimalrechenoperation, die eine Division umfasst. Wenn der Wert des Parameters "No" ist, wird die Anzahl der Kommastellen als 31-p+s-s' berechnet. Wenn der Wert "Yes" ist, wird die Anzahl der Kommastellen als MAX(3, 31-p+s-s') berechnet. Dies führt dazu, dass das Ergebnis einer Dezimaldivision immer mindestens 3 Kommastellen hat. Die Genauigkeit ist immer 31.

Eine Änderung dieses Datenbankkonfigurationsparameters kann Änderungen bei Anwendungen für vorhandene Datenbanken bedingen. Dazu kann es kommen, wenn die Anzahl der Kommastellen bei Ergebnissen von Dezimaldivisionen von der Änderung dieses Datenbankkonfigurationsparameters betroffen wäre. Nachfolgend werden einige mögliche Szenarios beschrieben, die Auswirkungen auf Anwendungen haben können. Betrachten Sie diese Szenarios, bevor Sie den Parameter *min_dec_div_3* auf einem Datenbankserver mit vorhandenen Datenbanken ändern.

- Wenn die Anzahl der Kommastellen im Ergebnis für eine der Spalten geändert wird, könnte eine Sicht, die in einer Umgebung mit einer Einstellung definiert ist, mit SQLCODE -344 fehlschlagen, wenn nach Änderung des Datenbankkonfigurationsparameters auf diese Sicht verwiesen wird. Die Nachricht SQL0344N verweist auf rekursive allgemeine Tabellenausdrücke, wenn der

Objektnamen (erstes Token) jedoch eine Sicht ist, müssen Sie die Sicht löschen und sie erneut erstellen, um diesen Fehler zu vermeiden.

- Das Verhalten eines statischen Pakets ändert sich erst, wenn das Paket implizit oder explizit erneut gebunden wird. Zum Beispiel werden nach einer Änderung des Werts von NO in YES die zusätzlichen Kommastellen in den Ergebnissen u. U. erst berücksichtigt, nachdem das Paket erneut gebunden wurde. Für jedes geänderte statische Paket kann mit einem expliziten REBIND-Befehl eine erneute Bindeoperation erzwungen werden.
- Eine Prüfung auf Integritätsbedingung schließt u. U. einige Werte aus, die zuvor akzeptiert wurden. Solche Zeilen verletzen jetzt die Integritätsbedingung, werden jedoch erst entdeckt, wenn eine der Spalten aktualisiert wird, die in der auf Integritätsbedingung geprüften Zeile enthalten ist, oder die Anweisung SET INTEGRITY mit der Option IMMEDIATE CHECKED verarbeitet wird. Sie können die Prüfung auf eine solche Integritätsbedingung erzwingen, indem Sie eine ALTER TABLE-Anweisung ausführen, um die Integritätsbedingung zu löschen und die Integritätsbedingung dann mit einer ALTER TABLE-Anweisung wieder hinzuzufügen.

Anmerkung: Für *min_dec_div_3* gelten außerdem die folgenden Einschränkungen:

1. Der Befehl GET DB CFG FOR DBNAME zeigt die Einstellung *min_dec_div_3* nicht an. Die aktuelle Einstellung lässt sich am besten dadurch ermitteln, dass Sie beobachten, welchen Nebeneffekt das Ergebnis einer Dezimaldivision hat. Betrachten Sie z. B. die folgende Anweisung:

```
VALUES (DEC(1,31,0)/DEC(1,31,5))
```

Wenn diese Anweisung den SQLCODE-Wert SQL0419N zurückgibt, unterstützt die Datenbank *min_dec_div_3* nicht oder der Parameter ist auf "No" gesetzt. Wenn die Anweisung 1,000 zurückgibt, ist *min_dec_div_3* auf "Yes" gesetzt.

2. *min_dec_div_3* erscheint nicht in der Liste der Konfigurationsschlüsselwörter, wenn Sie den folgenden Befehl ausführen: ?
UPDATE DB CFG

Zugehörige Referenzen:

- „GET DATABASE CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE CONFIGURATION command“ in *Command Reference*

rqrioblk - E/A-Blockgröße für Clients

Konfigurationstyp Datenbankmanager

Gilt für

- Datenbankserver mit lokalen und fernen Clients
- Client
- Datenbankserver mit lokalen Clients
- Partitionierten Datenbankserver mit lokalen und fernen Clients

Parametertyp Konfigurierbar

Standardwert [Bereich] 32 767 [4 096 – 65 535]

Maßeinheit Byte

Zuordnung

- Wenn eine ferne Client-Anwendung eine Verbindungsanforderung für eine Server-Datenbank absetzt
- Wenn ein Blockcursor geöffnet wird, werden zusätzliche Blöcke auf dem Client geöffnet

Freigabe

- Wenn die ferne Anwendung die Verbindung zur Server-Datenbank trennt
- Wenn der Blockcursor geschlossen wird

Mit diesem Parameter wird die Größe des Puffers für die Kommunikation zwischen fernen Anwendungen und ihren Datenbankagenten auf dem Datenbankserver festgelegt. Wenn ein Datenbankclient die Verbindung zu einer fernen Datenbank anfordert, wird dieser Kommunikationspuffer auf dem Client zugeordnet. Auf dem Datenbankserver wird zu Anfang ein Kommunikationspuffer von 32 767 Byte zugeordnet, bis eine Verbindung hergestellt ist und der Server den Wert des Parameters *rqrioblk* auf dem Client ermitteln kann. Wenn der Server diesen Wert ermittelt hat, wird der Kommunikationspuffer auf dem Server neu zugeordnet, sofern der Client-Puffer nicht 32 767 Byte groß ist.

Außer für diesen Kommunikationspuffer wird dieser Parameter auch dazu verwendet, die Größe des E/A-Blocks auf dem Datenbankclient festzulegen, wenn ein Blockcursor geöffnet wird. Dieser Speicher für Blockcursor wird aus dem privaten Adressraum der Anwendung zugeordnet. Sie sollten daher die optimale Größe des privaten Speichers ermitteln, der jedem Anwendungsprogramm zugeordnet werden soll. Wenn der Datenbankclient keinen Bereich für einen Blockcursor aus dem privaten Speicher der Anwendung zuordnen kann, wird ein Cursor ohne Blockung geöffnet.

Empfehlung: Bei Cursors ohne Blockung könnte ein Grund zur Erhöhung des Werts für diesen Parameter darin bestehen, dass die Daten (z. B. LOB-Daten), die durch eine einzige Abfrageanweisung übertragen werden sollen, so umfangreich sind, dass der Standardwert nicht ausreicht.

Beachten Sie außerdem, welche Auswirkung dieser Parameter auf die Anzahl und die mögliche Größe von Blockcursoren hat. Große Zeilenblöcke können zu einer höheren Leistung führen, wenn die Anzahl oder die Größe der Zeilen, die übertragen werden, groß ist (z. B., wenn die Datenmenge größer als 4 096 Byte ist). Dies hat jedoch den Nachteil, dass größere Datensatzblöcke die Größe des für jede Verbindung benötigten Arbeitsspeichers erhöhen.

Größere Satzblöcke können außerdem mehr Abrufanforderungen verursachen, als tatsächlich für die Anwendung erforderlich wären. Sie können die Anzahl der Abrufanforderungen mit der Klausel *OPTIMIZE FOR* in der Anweisung *SELECT* in Ihrer Anwendung steuern.

Zugehörige Referenzen:

- „GET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE MANAGER CONFIGURATION command“ in *Command Reference*

Speicher der Datenbankmanagerinstanz

Die folgenden Parameter wirken sich auf den Speicher aus, der auf Instanzebene zugeordnet und verwendet wird:

- „audit_buf_sz - Prüfpuffergröße “
- „dir_cache - Verzeichniscacheunterstützung “ auf Seite 467
- „instance_memory - Instanzspeicher “ auf Seite 468
- „java_heap_sz - Maximale Zwischenspeichergröße für Java-Interpreter “ auf Seite 469
- „mon_heap_sz - Zwischenspeichergröße für Datenbanksystemmonitor “ auf Seite 470

audit_buf_sz - Prüfpuffergröße

Konfigurationstyp Datenbankmanager

Gilt für

- Datenbankserver mit lokalen und fernen Clients
- Datenbankserver mit lokalen Clients
- Partitionierten Datenbankserver mit lokalen und fernen Clients

Parametertyp Konfigurierbar

Standardwert [Bereich] 0 [0 – 65 000]

Maßeinheit Seiten (4 KB)

Zuordnung Beim Starten von DB2

Freigabe Beim Stoppen von DB2

Mit diesem Parameter wird die Größe des Puffers für die Prüfung der Datenbank angegeben.

Der Standardwert für diesen Konfigurationsparameter ist Null (0). Wenn der Wert Null (0) ist, wird der Prüfpuffer nicht verwendet. Wenn der Wert größer als Null (0) ist, wird dem Prüfpuffer Speicherbereich zugeordnet, in den die von der Prüffunktion generierten Prüfsätze gestellt werden. Der Wert multipliziert mit 4-KB-Seiten ergibt die für den Prüfpuffer zugeordnete Speichermenge. Der Prüfpuffer kann nicht dynamisch zugeordnet werden; DB2 muss gestoppt und anschließend erneut gestartet werden, bevor der neue Wert für diesen Parameter in Kraft tritt.

Wenn Sie den Standardwert für diesen Parameter in einen Wert ändern, der größer als Null (0) ist, schreibt die Prüffunktion Datensätze asynchron im Vergleich zur Ausführung der Anweisungen, die die Prüfsätze generieren, auf Platte. Durch das Erhöhen des Standardparameterwerts Null (0) wird die Leistung von DB2 verbessert. Der Wert Null (0) bedeutet, dass die Prüffunktion Datensätze synchron zur (d. h. zur gleichen Zeit wie die) Ausführung der Anweisungen, die die Prüfsätze generieren, auf Platte schreibt. Die synchrone Verarbeitung während der Prüfung verringert die Leistung von unter DB2 ausgeführten Anwendungen.

Zugehörige Referenzen:

- „GET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*

- „UPDATE DATABASE MANAGER CONFIGURATION command“ in *Command Reference*

dir_cache - Verzeichniscacheunterstützung

Konfigurationstyp	Datenbankmanager
Gilt für	• Datenbankserver mit lokalen und fernen Clients • Client • Datenbankserver mit lokalen Clients • Partitionierten Datenbankserver mit lokalen und fernen Clients
Parametertyp	Konfigurierbar
Standardwert [Bereich]	Yes [Yes; No]
Zuordnung	• Wenn eine Anwendung die erste Verbindungsanforderung absetzt, wird der Anwendungsverzeichniscache zugeordnet. • Wenn eine Datenbankmanagerinstanz gestartet (db2start) wird, wird der Serververzeichniscache zugeordnet.
Freigabe	• Wenn ein Anwendungsprozess beendet wird, wird der Anwendungsverzeichniscache freigegeben. • Wenn eine Datenbankmanagerinstanz gestoppt (db2stop) wird, wird der Serververzeichniscache freigegeben.

Wenn der Parameter *dir_cache* auf "Yes" gesetzt wird, werden die Datenbank-, Knoten- und DCS-Verzeichnisdateien im Cache zwischengespeichert. Durch die Verwendung des Verzeichniscache wird der Aufwand für die Verbindung verringert, indem die E/A-Operationen für Verzeichnisdateien vermieden und die Suchoperationen in Verzeichnissen zum Abruf von Verzeichnisdaten minimiert werden. Es gibt zwei Arten von Verzeichniscaches:

- Ein Anwendungsverzeichniscache, der für jeden Anwendungsprozess auf dem System zugeordnet und verwendet wird, auf dem die Anwendung ausgeführt wird.
- Ein Serververzeichniscache, der für einige der internen Datenbankmanagerprozesse zugeordnet und verwendet wird.

Wenn eine Anwendung die erste Verbindungsanforderung absetzt, werden in einem Anwendungsverzeichniscache alle Verzeichnisdateien gelesen, und die Informationen werden im privaten Speicher dieser Anwendung zwischengespeichert. Der Cache wird vom Anwendungsprozess auch für nachfolgende Verbindungsanforderungen verwendet und während der Dauer des Anwendungsprozesses beibehalten. Wenn eine Datenbank nicht im Anwendungsverzeichniscache gefunden wird, werden die Verzeichnisdateien nach den Informationen durchsucht, aber der Cache wird nicht aktualisiert. Wenn die Anwendung einen Verzeichniseintrag ändert, wird durch die nächste Verbindungsanforderung innerhalb dieser Anwendung eine Aktualisierung des Caches für diese Anwendung bewirkt. Der Anwendungsverzeichniscache für andere Anwendungen wird nicht aktualisiert.

Wenn der Anwendungsprozess beendet ist, wird der Cache freigegeben. (Zur Aktualisierung des Verzeichniscache, der von einer Sitzung des Befehlszeilenprozessors verwendet wird, geben Sie den Befehl `db2 terminate` ein.)

Wenn eine Datenbankmanagerinstanz gestartet wird (`db2start`), werden in einem Serververzeichniscache alle Verzeichnisdateien gelesen, und die Informationen werden im Servercache zwischengespeichert. Dieser Cache wird beibehalten, bis die Instanz gestoppt (`db2stop`) wird. Wenn ein Verzeichniseintrag in diesem Cache nicht gefunden wird, werden die Verzeichnisdateien nach den Daten durchsucht. Dieser Serververzeichniscache wird während der Ausführung der Instanz zu keinem Zeitpunkt aktualisiert.

Empfehlung: Verwenden Sie einen Verzeichniscache, wenn Ihre Verzeichnisdateien nicht häufig geändert werden und die Leistung von entscheidender Bedeutung ist.

Auf fernen Clients kann darüber hinaus der Verzeichniscache von Vorteil sein, wenn Ihre Anwendungen mehrere verschiedene Verbindungsanforderungen absetzen. In diesem Fall verringert das Zwischenspeichern die Häufigkeit, mit der eine einzige Anwendung die Verzeichnisdateien lesen muss.

Der Verzeichniscache kann auch die Leistung bei der Erstellung von Momentaufnahmen des Datenbanksystemmonitors erhöhen. Außerdem sollten Sie beim Aufruf der Momentaufnahme den Datenbanknamen explizit angeben und nicht den Aliasnamen für die Datenbank verwenden.

Anmerkung: Bei der Erstellung von Momentaufnahmen können Fehler auftreten, wenn der Verzeichniscache aktiviert wurde und Datenbanken katalogisiert, entkatalogisiert, erstellt oder gelöscht werden, nachdem der Datenbankmanager gestartet wurde.

Zugehörige Referenzen:

- „GET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE MANAGER CONFIGURATION command“ in *Command Reference*

instance_memory - Instanzspeicher

Konfigurationstyp	Datenbankmanager
Gilt für	• Datenbankserver mit lokalen und fernen Clients • Datenbankserver mit lokalen Clients • Partitionierten Datenbankserver mit lokalen und fernen Clients
Parametertyp	Konfigurierbar
Standardwert [Bereich]	Automatic [8 - 4.294.967.295]
Maßeinheit	Seiten (4 KB)
Zuordnung	Wenn die Instanz gestartet wird
Freigabe	Wenn die Instanz gestoppt wird

Dieser Parameter gibt die Speicherkapazität an, die zur Instanzverwaltung reserviert werden soll. Dies umfasst Speicherbereiche, die die Datenbanken in der Instanz beschreiben.

Wenn Sie diesen Parameter auf **AUTOMATIC** setzen, berechnet DB2 die für die aktuelle Konfiguration benötigte Speicherkapazität für die Instanz. DB2 ordnet außerdem zusätzlichen, nicht reservierten Speicher zu, der von jedem Zwischenspeicher im gemeinsam genutzten Datenbankspeicherbereich verwendet werden kann, um Spitzenwerte des Speicherbedarfs aufzufangen. Andere Operationen, wie zum Beispiel dynamische Aktualisierungen der Konfiguration, haben ebenfalls Zugriff auf diesen nicht reservierten Speicher. Der Befehl **db2pd** mit der Option **-memsets** kann zur Überwachung des im gemeinsam genutzten Instanzspeicherbereich verbliebenen nicht genutzten Speichers verwendet werden.

Zugehörige Referenzen:

- „maxagents - Maximale Anzahl von Agenten “ auf Seite 487
- „numdb - Maximale Anzahl gleichzeitig aktiver Datenbanken einschließlich Host- und iSeries-Datenbanken “ auf Seite 570
- „db2pd - Monitor and troubleshoot DB2 database command“ in *Command Reference*
- „GET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE MANAGER CONFIGURATION command“ in *Command Reference*

java_heap_sz - Maximale Zwischenspeichergröße für Java-Interpreter

Konfigurationstyp	Datenbankmanager
Gilt für	• Datenbankserver mit lokalen und fernen Clients • Client • Datenbankserver mit lokalen Clients • Partitionierten Datenbankserver mit lokalen und fernen Clients
Parametertyp	Konfigurierbar
Standardwert [Bereich]	32-Bit-Plattformen 512 [0 - 524.288] 64-Bit-Plattformen 1024 [0 - 524.288]
Maßeinheit	Seiten (4 KB)
Zuordnung	Wenn eine gespeicherte Java-Prozedur oder eine benutzerdefinierte Funktion gestartet wird
Freigabe	Wenn der db2fmp-Prozess (abgeschirmt) oder der db2agent-Prozess (gesichert) beendet wird

Dieser Parameter legt die maximale Größe des Zwischenspeichers fest, der von dem zur Verarbeitung von gespeicherten Java-DB2-Prozeduren und benutzerdefinierten Funktionen gestarteten Java-Interpreter verwendet wird.

Es gibt einen Zwischenspeicher für jeden DB2-Prozess (einen für jeden Agenten oder Subagenten auf UNIX-basierten Plattformen und einen für jede Instanz auf anderen Plattformen). Es gibt einen Zwischenspeicher für jede abgeschirmte benutzerdefinierte Funktion und für jede abgeschirmte gespeicherte Prozedur. Es gibt einen Zwischenspeicher für jeden Agenten (ausschließlich der Subagenten) für gesicherte Routinen. Es gibt einen Zwischenspeicher für jeden db2fmp-Prozess, der eine gespeicherte Java-Prozedur ausführt. Für Multithread-db2fmp-Prozesse werden mehrere Anwendungen, die threadsichere abgeschirmte Routinen verwenden, von einem einzigen Zwischenspeicher bedient. Nur die Agenten oder Prozesse, die in Java geschriebene benutzerdefinierte Funktionen oder gespeicherte Prozeduren ausführen, ordnen diesen Speicher zu. Bei partitionierten Datenbanksystemen wird in jeder Datenbankpartition derselbe Wert verwendet.

Zugehörige Referenzen:

- „jdk_path - Installationspfad für das Software Developer's Kit für Java " auf Seite 569
- „GET DATABASE MANAGER CONFIGURATION command" in *Command Reference*
- „RESET DATABASE MANAGER CONFIGURATION command" in *Command Reference*
- „UPDATE DATABASE MANAGER CONFIGURATION command" in *Command Reference*

mon_heap_sz - Zwischenspeichergröße für Datenbanksystemmonitor

Konfigurationstyp	Datenbankmanager
Gilt für	• Datenbankserver mit lokalen und fernen Clients • Datenbankserver mit lokalen Clients • Partitionierten Datenbankserver mit lokalen und fernen Clients
Parametertyp	Konfigurierbar
Standardwert [Bereich]	UNIX 90 [0 – 60 000] Windows-Datenbankserver mit lokalen und fernen Clients 66 [0 – 60 000] Windows-Datenbankserver mit lokalen Clients 46 [0 – 60 000]
Maßeinheit	Seiten (4 KB)
Zuordnung	Wenn der Datenbankmanager mit dem Befehl <i>db2start</i> gestartet wird
Freigabe	Wenn der Datenbankmanager mit dem Befehl <i>db2stop</i> gestoppt wird

Mit diesem Parameter wird der Speicherbereich in Seiten festgelegt, der für Daten des Datenbanksystemmonitors zugeordnet werden soll. Der Speicher wird aus dem Monitorzwischenpeicher zugeordnet, wenn Sie eine Datenbankmonitorfunktion ausführen, wie z. B. Erstellen einer Momentaufnahme (Snapshot), Aktivieren eines Monitorschalters oder eines Ereignismonitors, Zurücksetzen eines Monitors o. ä.

Erhält der Parameter den Wert 0, kann der Datenbankmanager keine Daten des Datenbanksystemmonitors sammeln.

Empfehlung: Die für die Monitorfunktionen erforderliche Speicherkapazität ist von der Anzahl der Überwachungsanwendungen (Anwendungen, die Momentaufnahmen machen, oder Ereignismonitoren), den aktivierten Schaltern und der Datenbankaktivität abhängig.

Wenn die konfigurierte Speicherkapazität in diesem Zwischenpeicher erschöpft und kein weiterer nicht reservierter Speicher mehr verfügbar ist, wird eine der folgenden Aktionen ausgeführt:

- Wenn die erste Anwendung eine Verbindung zu der Datenbank herstellt, für die dieser Ereignismonitor definiert ist, wird eine Fehlernachricht in das Benachrichtigungsprotokoll für die Verwaltung geschrieben.
- Wenn ein Ereignismonitor, der mit Hilfe der Anweisung SET EVENT MONITOR dynamisch gestartet wird, fehlschlägt, wird ein Fehlercode an die Anwendung zurückgegeben.
- Wenn ein Monitorbefehl oder eine API-Unterroutine fehlschlägt, wird ein Fehlercode an die Anwendung zurückgegeben.

Zugehörige Konzepte:

- „Database system monitor memory requirements“ in *Systemmonitor Handbuch und Referenz*

Zugehörige Referenzen:

- „dft_monswitches - Monitorschalter des Standarddatenbanksystems “ auf Seite 565
- „GET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE MANAGER CONFIGURATION command“ in *Command Reference*

Sperren

Die folgenden Parameter beeinflussen die Sperrenverwaltung in Ihrer Umgebung:

- „dlchktime - Zeitintervall für Prüfung von Deadlocks “
- „locktimeout - Zeitlimit für Sperren “ auf Seite 472
- „maxlocks - Maximale Anzahl Sperren pro Anwendung “ auf Seite 474

Siehe auch „locklist - Maximaler Speicher für Sperrenliste “ auf Seite 438.

dlchktime - Zeitintervall für Prüfung von Deadlocks

Konfigurationstyp

Datenbank

Parametertyp

Online konfigurierbar

Weitergabeklasse	Sofort
Standardwert [Bereich]	10 000 (10 Sekunden) [1 000 – 600 000]
Maßeinheit	Millisekunden

Beim Auftreten von Deadlocks müssen zwei oder mehr Anwendungen, die auf dieselbe Datenbank zugreifen, für einen unbegrenzten Zeitraum auf eine Ressource warten. Die Anwendungen warten unendlich lange, da jede Anwendung eine Ressource gesperrt hat, die von der anderen Anwendung zur Fortsetzung der Verarbeitung benötigt wird.

Der Parameter für das Intervall zur Prüfung auf Deadlocks definiert die Häufigkeit, mit der vom Datenbankmanager alle mit der Datenbank verbundenen Anwendungen auf Deadlocks überprüft werden.

Anmerkungen:

1. In einer Umgebung mit partitionierten Datenbanken gilt dieser Parameter nur für den Katalogknoten.
2. In einer Umgebung mit partitionierten Datenbanken wird ein Deadlock erst nach der zweiten Iteration markiert.

Empfehlung: Durch Erhöhung des Werts für diesen Parameter wird die Häufigkeit der Prüfung auf Deadlocks verringert, wodurch die Zeit, die Anwendungsprogramme auf die Aufhebung von Deadlocks warten müssen, erhöht wird.

Durch Angabe eines niedrigeren Werts für diesen Parameter wird die Häufigkeit der Prüfung auf Deadlocks erhöht, wodurch die Zeit, die Anwendungsprogramme auf die Aufhebung von Deadlocks warten müssen, verkürzt, die Zeit, die der Datenbankmanager für die Prüfung auf Deadlocks verwendet, jedoch verlängert wird. Wenn das Prüfintervall zu klein ist, kann dies zu einer Verschlechterung der Laufzeitleistung führen, weil der Datenbankmanager häufig nach Deadlocks sucht. Wenn dieser Parameter mit einem niedrigeren Wert versehen wird, um den gemeinsamen Zugriff zu verbessern, sollten Sie darauf achten, dass die Parameter *maxlocks* und *locklist* entsprechend eingestellt sind, um unnötige Sperreneskaltungen zu vermeiden, die zu Zugriffskonflikten und weiteren Deadlocks führen können.

Zugehörige Referenzen:

- „locklist - Maximaler Speicher für Sperrenliste “ auf Seite 438
- „maxlocks - Maximale Anzahl Sperren pro Anwendung “ auf Seite 474
- „GET DATABASE CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE CONFIGURATION command“ in *Command Reference*

locktimeout - Zeitlimit für Sperren

Konfigurationstyp	Datenbank
Parametertyp	Konfigurierbar
Standardwert [Bereich]	-1 [-1; 0 – 32 767]
Maßeinheit	Sekunden

Mit diesem Parameter wird die Anzahl der Sekunden angegeben, die eine Anwendung auf eine Sperre wartet. Dies trägt zur Vermeidung globaler Deadlocks von Anwendungen bei.

Wenn dieser Parameter auf 0 gesetzt wird, wird nicht auf Sperren gewartet. In diesem Fall erhält die Anwendung, wenn zum Zeitpunkt der Anforderung keine Sperre verfügbar ist, sofort die Rückmeldung -911.

Wenn dieser Parameter auf den Wert -1 gesetzt wird, wird die Überwachung des Zeitlimits für Sperren inaktiviert. In diesem Fall wird auf eine Sperre gewartet (wenn zum Zeitpunkt der Anforderung keine verfügbar ist), bis eine der folgenden Situationen eintritt:

- Die Sperre wird erteilt
- Ein Deadlock tritt ein

Empfehlung: In einer Umgebung, in der Transaktionen verarbeitet werden (OLTP-Umgebung), können Sie einen Anfangswert von 30 Sekunden verwenden. In einer Umgebung, in der nur Abfragen durchgeführt werden, können Sie mit einem höheren Wert beginnen. In beiden Fällen sollten Sie diesen Parameter jedoch mit Hilfe von Vergleichstests (Benchmark-Tests) optimieren.

Wenn Sie bei der Arbeit mit Data Links File Manager (DLFM) im Benachrichtigungsprotokoll für die Verwaltung der Instanz von Data Links File Manager Überschreitungen der Sperrzeit feststellen, erhöhen Sie den Wert für den Parameter *locktimeout*. Erhöhen Sie gegebenenfalls auch den Wert für *locklist*.

Der Wert sollte so eingestellt werden, dass schnell erkannt wird, wenn Wartezeiten aufgrund außergewöhnlicher Situationen auftreten, wie z. B. eine blockierte Transaktion (weil ein Benutzer seine Workstation verlassen hat). Sie sollten den Wert so hoch festlegen, dass gültige Sperranforderungen das Zeitlimit nicht aufgrund von Spitzenbelastungen überschreiten, da bei hoher Systemauslastung länger auf Sperren gewartet werden muss.

Mit dem Datenbanksystemmonitor können Sie die Häufigkeit verfolgen, mit der eine Anwendung (eine Verbindung) das Zeitlimit für Sperren überschreitet, oder erkennen, dass eine Datenbank eine Zeitlimitüberschreitung für alle verbundenen Anwendungen festgestellt hat.

Hohe Werte für das Monitorelement *lock_timeout* (Anzahl Zeitlimitüberschreitungen für Sperren) können folgende Ursachen haben:

- Ein zu niedriger Wert für diesen Konfigurationsparameter
- Eine Anwendung (Transaktion), die Sperren über einen längeren Zeitraum aufrechterhält. Mit Hilfe des Datenbanksystemmonitors können Sie solche Anwendungen näher untersuchen.
- Ein Problem in Bezug auf den gleichzeitigen Zugriff, verursacht durch Sperreneskulationen (von Sperren auf Zeilenebene zu Sperren auf Tabellenebene).

Zugehörige Konzepte:

- „Wartestatus und Zeitlimitüberschreitungen für Sperren“ auf Seite 80

Zugehörige Referenzen:

- „lock_timeouts - Number of Lock Timeouts monitor element“ in *Systemmonitor Handbuch und Referenz*
- „GET DATABASE CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE CONFIGURATION command“ in *Command Reference*
- „locklist - Maximaler Speicher für Sperrenliste “ auf Seite 438

- „maxlocks - Maximale Anzahl Sperren pro Anwendung ” auf Seite 474

maxlocks - Maximale Anzahl Sperren pro Anwendung

Konfigurationstyp	Datenbank
Parametertyp	Online konfigurierbar
Weitergabeklasse	Sofort
Standardwert [Bereich]	
	UNIX Automatic [1 - 100]
	Windows Automatic [1 - 100]
Maßeinheit	Prozent

Eine Sperreneskalation ist der Prozess, durch den jeweils mehrere Zeilensperren für eine Tabelle durch eine Tabellensperre ersetzt werden, um die Anzahl der Sperren in der Liste zu verringern. Mit diesem Parameter wird definiert, wie viel Prozent der Sperrenliste eine Anwendung belegen muss, damit der Datenbankmanager eine Eskalation ausführt. Wenn die Anzahl der Sperren, die von einer Anwendung aktiviert wurden, diesen Prozentwert der Gesamtgröße der Sperrenliste erreicht, wird für die Sperren dieser Anwendung eine Sperreneskalation ausgeführt. Eine Sperreneskalation wird auch dann ausgeführt, wenn in der Sperrenliste kein weiterer Platz mehr verfügbar ist.

Der Datenbankmanager bestimmt die zu eskalierenden Sperren, indem er die Sperrenliste nach der Anwendung durchsucht und die Tabelle mit den meisten gesperrten Zeilen ermittelt. Wenn nach der Ersetzung dieser Sperren durch eine einzige Tabellensperre der Wert für *maxlocks* nicht mehr überschritten wird, wird die Sperreneskalation beendet. Andernfalls wird die Eskalation fortgesetzt, bis der von dieser Anwendung belegte Prozentsatz der Sperrenliste unter den Wert von *maxlocks* gesunken ist. Das Produkt aus dem Wert des Parameters *maxlocks* und dem Wert des Parameters *maxappls* darf nicht kleiner als 100 sein.

Der Wert AUTOMATIC für diesen Parameter aktiviert ihn für die automatische Leistungsoptimierung. Dadurch ist die Speicheroptimierungsfunktion in der Lage, die durch diesen Parameter gesteuerte Größe des Hauptspeicherbereichs an geänderte Auslastungsanforderungen je nach Bedarf dynamisch anzupassen. Da die Speicheroptimierungsfunktion Speicherressourcen auf verschiedene Speicher-verbraucher verteilt, müssen mindestens zwei Speicherverbraucher für die automatische Optimierung aktiviert sein, damit eine automatische Optimierung erfolgen kann.

Der Wert des Parameters *locklist* wird zusammen mit dem Parameter *maxlocks* optimiert, so dass eine Inaktivierung der automatischen Optimierung des Parameters *locklist* automatisch auch eine Inaktivierung der automatischen Optimierung des Parameters *maxlocks* bewirkt. Die Aktivierung der automatischen Optimierung für den Parameter *locklist* führt automatisch zu einer Aktivierung der automatischen Optimierung des Parameters *maxlocks*.

Empfehlung: Mit der folgenden Formel können Sie *maxlocks* so einstellen, dass eine Anwendung das Zweifache der durchschnittlichen Anzahl von Sperren aktivieren kann:

$$\text{maxlocks} = 2 * 100 / \text{maxappls}$$

Dabei wird 2 verwendet, um das Zweifache der durchschnittlichen Anzahl zu erzielen, und 100 ist der höchste zulässige Prozentwert. Wenn Sie nur wenige

Anwendungen haben, die gleichzeitig aktiv sind, können Sie alternativ zu der ersten Formel auch die folgende Formel verwenden:

$$\text{maxlocks} = 2 * 100 / (\text{durchschnittliche Anzahl der gleichzeitig aktiven Anwendungen})$$

Bei der Festlegung von *maxlocks* können Sie eine gleichzeitige Verwendung von *locklist* (Größe der Sperrenliste) in Erwägung ziehen. Die maximale Anzahl der Sperren, die eine Anwendung aktivieren kann, bevor eine Sperreneskulation erfolgt, wird wie folgt berechnet:

$$\text{maxlocks} * \text{locklist} * 4.096 / (100 * 48) \text{ auf einem 32-Bit-System}$$

$$\text{maxlocks} * \text{locklist} * 4.096 / (100 * 80) \text{ in einer HP-UX/PA-RISC-Umgebung (64-Bit-System)}$$

$$\text{maxlocks} * \text{locklist} * 4.096 / (100 * 64) \text{ auf anderen 64-Bit-Systemen}$$

Dabei ist 4.096 die Anzahl Byte auf einer Seite, 100 ist der höchste zulässige Prozentwert für *maxlocks*, 48 ist die Anzahl der Byte pro Sperre auf einem 32-Bit-System, 80 die Anzahl der Byte pro Sperre auf einem HP-UX/PA-RISC-System (64 Bit) und 64 die Anzahl der Byte pro Sperre auf anderen 64-Bit-Systemen. Wenn Sie wissen, dass eine Ihrer Anwendungen 1.000 Sperren benötigt, und Sie keine Sperreneskulation wünschen, sollten Sie die Werte für *maxlocks* und *locklist* in dieser Formel so wählen, dass das Ergebnis größer als 1.000 ist. (Wenn Sie für *maxlocks* 10 und für *locklist* 100 verwenden, ist das Ergebnis der Formel höher als die erforderlichen 1.000 Sperren.)

Wenn Sie für *maxlocks* einen zu niedrigen Wert wählen, tritt eine Sperreneskulation auf, obwohl für andere gleichzeitig ausgeführte Anwendungen noch genügend Speicher für Sperren verfügbar ist. Wenn ein zu hoher Wert für *maxlocks* gewählt wird, belegen einige wenige Anwendungen u. U. fast den gesamten Speicher für Sperren, während andere Anwendungen eine Sperreneskulation durchführen müssen. Die Notwendigkeit einer Sperreneskulation geht in diesem Fall zulasten des gleichzeitigen Zugriffs.

Mit Hilfe des Datenbanksystemmonitors können Sie diesen Konfigurationsparameter verfolgen und seinen Wert optimieren.

Zugehörige Konzepte:

- „Speicher mit automatischer Leistungsoptimierung“ auf Seite 295

Zugehörige Referenzen:

- „GET DATABASE CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE CONFIGURATION command“ in *Command Reference*
- „locklist - Maximaler Speicher für Sperrenliste “ auf Seite 438
- „maxappls - Maximale Anzahl aktiver Anwendungen “ auf Seite 487
- „self_tuning_mem - Speicher mit automatischer Leistungsoptimierung “ auf Seite 444

Ein-/Ausgabe und Speicher

Die folgenden Parameter beeinflussen den Ein-/Ausgabeaufwand und den Speicherbedarf, die mit dem Betrieb der Datenbank verbunden sind:

- „chnpggs_thresh - Schwellenwert für geänderte Seiten ”
- „dft_extent_sz - Standardwert für EXTENTSIZE bei Tabellenbereichen ” auf Seite 477
- „dft_prefetch_sz - Standardwert für PREFETCHSIZE ” auf Seite 477
- „num_iocleaners - Anzahl asynchroner Seitenlöschfunktionen ” auf Seite 478
- „num_ioservers - Anzahl von E/A-Servern ” auf Seite 480
- „numsegs - Standardanzahl von SMS-Containern ” auf Seite 481
- „seqdetect - Markierung für Sequenzerkennung ” auf Seite 481

chnpggs_thresh - Schwellenwert für geänderte Seiten

Konfigurationstyp	Datenbank
Parametertyp	Konfigurierbar
Standardwert [Bereich]	60 [5 – 99]
Maßeinheit	Prozent

Asynchrone Seitenlöschfunktionen schreiben geänderte Seiten aus dem Pufferpool (oder den Pufferpools) auf Platte, bevor der Bereich im Pufferpool von einem Datenbankagenten angefordert wird. Daher müssen Datenbankagenten in der Regel nicht die Auslagerung geänderter Seiten abwarten, bevor sie den Speicherbereich im Pufferpool nutzen können. Dadurch wird die Gesamtleistung der Datenbankanwendungen verbessert.

Mit diesem Parameter können Sie den Prozentsatz der geänderten Seiten angeben, bei dem die asynchronen Seitenlöschfunktionen gestartet werden, wenn sie zum gegebenen Zeitpunkt nicht aktiv sind. Wenn die Seitenlöschfunktionen gestartet werden, erstellen sie eine Liste der Seiten, die auf Platte zu schreiben sind. Wenn das Schreiben dieser Seiten auf Platte abgeschlossen ist, werden die Seitenlöschfunktionen wieder inaktiv und warten auf den nächsten Starttrigger.

In einer Umgebung mit reinem Lesezugriff (in der z. B. nur Abfragen ausgeführt werden) werden diese Seitenlöschfunktionen nicht verwendet.

Wenn die Registrierdatenbankvariable DB2_USE_ALTERNATE_PAGE_CLEANSING definiert ist (d. h., die alternative Methode der Seitenbereinigung verwendet wird), hat der Parameter *chnpggs_thresh* keine Wirkung und DB2 bestimmt automatisch, wie viele geänderte Seiten im Pufferpool zu behalten sind.

Empfehlung: Bei Datenbanken mit hohem Aufkommen an aktualisierenden Transaktionen können Sie allgemein sicherstellen, dass genügend freie Seiten im Pufferpool verfügbar sind, indem Sie diesen Parameter auf einen Wert setzen, der gleich groß wie oder kleiner als der Standardwert ist. Ein Prozentsatz über dem Standardwert kann sich positiv auf die Leistung auswirken, wenn in Ihrer Datenbank eine kleine Anzahl sehr großer Tabellen gespeichert ist.

Zugehörige Referenzen:

- „num_iocleaners - Anzahl asynchroner Seitenlöschfunktionen ” auf Seite 478
- „GET DATABASE CONFIGURATION command” in *Command Reference*
- „RESET DATABASE CONFIGURATION command” in *Command Reference*
- „UPDATE DATABASE CONFIGURATION command” in *Command Reference*

dft_extent_sz - Standardwert für EXTENTSIZE bei Tabellenbereichen

Konfigurationstyp	Datenbank
Parametertyp	Online konfigurierbar
Weitergabeklasse	Sofort
Standardwert [Bereich]	32 [2 – 256]
Maßeinheit	Seiten

Bei der Erstellung eines Tabellenbereichs kann wahlfrei EXTENTSIZE *n* angegeben werden, wobei *n* die Speicherbereichsgröße ist. Wenn Sie EXTENTSIZE *n* in der Anweisung CREATE TABLESPACE nicht angeben, verwendet der Datenbankmanager den Wert, der durch diesen Parameter definiert wird.

Empfehlung: In vielen Fällen geben Sie die Speicherbereichsgröße bei der Erstellung eines Tabellenbereichs explizit an. Bevor Sie einen Wert für diesen Parameter wählen, sollten Sie verstehen, wie die Speicherbereichsgröße in der Anweisung CREATE TABLESPACE explizit gewählt wird.

Zugehörige Konzepte:

- „Parameter EXTENTSIZE“ in *Systemverwaltung: Konzept*

Zugehörige Referenzen:

- „CREATE TABLESPACE statement“ in *SQL Reference, Volume 2*
- „dft_prefetch_sz - Standardwert für PREFETCHSIZE “ auf Seite 477
- „GET DATABASE CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE CONFIGURATION command“ in *Command Reference*

dft_prefetch_sz - Standardwert für PREFETCHSIZE

Konfigurationstyp	Datenbank
Parametertyp	Online konfigurierbar
Weitergabeklasse	Sofort
Standardwert [Bereich]	Automatic [0 — 32 767]
Maßeinheit	Seiten

Bei der Erstellung eines Tabellenbereichs kann optional PREFETCHSIZE *n* angegeben werden, wobei *n* die Anzahl der Seiten ist, die der Datenbankmanager liest, wenn ein Vorabesezugriff erfolgt. Wenn Sie in der Anweisung CREATE TABLESPACE keinen Wert für PREFETCHSIZE angeben, verwendet der Datenbankmanager den aktuellen Wert des Parameters *dft_prefetch_sz*.

Wenn ein Tabellenbereich mit der Angabe AUTOMATIC DFT_PREFETCH_SZ erstellt wird, wird die Größe des Vorabesezugriffs des Tabellenbereichs automatisch ermittelt. Dies bedeutet, dass DB2 den Wert für PREFETCHSIZE des Tabellenbereichs unter Verwendung der folgenden Gleichung entsprechend automatisch berechnet und aktualisiert:

$$\text{PREFETCHSIZE} = (\text{Anzahl Container}) * (\text{Anzahl physischer SpindeIn}) * \text{EXTENTSIZE}$$

Dabei gilt für die Anzahl der physischen Spindeln der Standardwert 1. Diese Anzahl kann über die DB2-Registrierdatenbankvariable DB2_PARALLEL_IO definiert werden. Diese Berechnung erfolgt zu folgenden Zeitpunkten:

- Beim Starten der Datenbank
- Bei der anfänglichen Erstellung eines Tabellenbereichs mit AUTOMATIC DFT_PREFETCH_SZ
- Bei einer Änderung der Anzahl von Containern für einen Tabellenbereich durch die Ausführung einer Anweisung ALTER TABLESPACE
- Bei einer Änderung des Werts für PREFETCHSIZE für einen Tabellenbereich in AUTOMATIC durch eine Anweisung ALTER TABLESPACE

Der Status AUTOMATIC für PREFETCHSIZE kann aktiviert oder inaktiviert werden, wenn der PREFETCHSIZE-Wert manuell durch eine Anweisung ALTER TABLESPACE aktualisiert wird.

Empfehlung: Mit Tools zur Systemüberwachung können Sie feststellen, ob Ihre CPU ausgelastet ist, während das System auf Ein-/Ausgaben wartet. Die Erhöhung dieses Parameterwerts kann nützlich sein, wenn für die Tabellenbereiche, die verwendet werden, kein Wert für PREFETCHSIZE definiert ist.

Dieser Parameter enthält den Standardwert für die gesamte Datenbank und ist eventuell nicht für alle Tabellenbereiche innerhalb der Datenbank geeignet. Beispielsweise kann der Wert 32 für einen Tabellenbereich mit dem Wert 32 Seiten für EXTENTSIZE geeignet sein, aber nicht für einen Tabellenbereich mit dem Wert 25 Seiten für EXTENTSIZE. Im Idealfall sollten Sie den Wert für PREFETCHSIZE für jeden Tabellenbereich explizit angeben.

Sie sollten als Wert für diesen Parameter einen Faktor oder ein ganzzahliges Vielfaches des Wertes des Parameters *dft_extent_sz* angeben, um die E/A-Operationen für Tabellenbereiche zu minimieren, die mit dem Standardwert für EXTENTSIZE (Parameter *dft_extent_sz*) definiert sind. Wenn z. B. für den Parameter *dft_extent_sz* der Wert 32 angegeben ist, könnten Sie den Wert von *dft_prefetch_sz* auf 16 (einen Bruchteil von 32) oder auf 64 (ein ganzzahliges Vielfaches von 32) setzen. Wenn für PREFETCHSIZE ein Vielfaches des Werts für EXTENTSIZE angegeben ist, kann der Datenbankmanager parallele E/A-Operationen ausführen, wenn die folgenden Bedingungen erfüllt sind:

- Die Bereiche, die vorab gelesen werden, befinden sich auf verschiedenen physischen Einheiten.
- Es sind mehrere E/A-Server konfiguriert (*num_ioservers*).

Zugehörige Referenzen:

- „ALTER TABLESPACE statement“ in *SQL Reference, Volume 2*
- „CREATE TABLESPACE statement“ in *SQL Reference, Volume 2*
- „GET DATABASE CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE CONFIGURATION command“ in *Command Reference*

num_iocleaners - Anzahl asynchroner Seitenlöschfunktionen

Konfigurationstyp	Datenbank
Parametertyp	Konfigurierbar
Standardwert [Bereich]	Automatic [0 - 255]
Maßeinheit	Zähler

Mit diesem Parameter können Sie die Anzahl asynchroner Seitenlöschfunktionen für eine Datenbank angeben. Diese Seitenlöschfunktionen schreiben geänderte Seiten aus dem Pufferpool auf Platte, bevor der Bereich im Pufferpool von einem Datenbankagenten angefordert wird. Daher müssen Datenbankagenten in der Regel nicht die Auslagerung geänderter Seiten abwarten, bevor sie den Speicherbereich im Pufferpool nutzen können. Dadurch wird die Gesamtleistung der Datenbankanwendungen verbessert.

Wenn der Parameter auf den Wert 0 gesetzt wird, werden keine Seitenlöschfunktionen gestartet, und infolgedessen schreiben die Agenten alle geänderten Seiten aus dem Pufferpool auf Platte. Dieser Parameter kann erhebliche Auswirkungen auf die Leistung einer Datenbank haben, die auf mehrere physische Speichereinheiten verteilt ist, weil in diesem Fall die Wahrscheinlichkeit größer ist, dass auf einer dieser Einheiten momentan keine Operationen ausgeführt werden. Wenn keine Seitenlöschfunktionen konfiguriert sind, könnten Ihre Anwendungen in regelmäßigen Abständen auf volle Protokolle stoßen.

Wenn dieser Parameter auf AUTOMATIC gesetzt wird, richtet sich die Anzahl der gestarteten Seitenlöschfunktionen nach der Anzahl der auf dem aktuellen System konfigurierten CPUs sowie nach der Anzahl lokaler logischer Datenbankpartitionen in einer Umgebung mit partitionierten Datenbanken. Es wird in jedem Fall mindestens eine Seitenlöschfunktion gestartet, wenn dieser Parameter auf AUTOMATIC gesetzt ist.

Die Anzahl der zu startenden Seitenlöschfunktionen, wenn dieser Parameter auf AUTOMATIC gesetzt ist, wird anhand der folgenden Formel berechnet:

Anzahl Seitenlöschfunktionen = $\max(\text{ceil}(\text{Anzahl CPUs} / \text{Anzahl lokaler logischer DBpartitionen}) - 1, 1)$

Diese Formel stellt sicher, dass die Anzahl der Seitenlöschfunktionen nahezu gleichmäßig auf Ihre logischen Datenbankpartitionen verteilt wird und dass nicht mehr Seitenlöschfunktionen als CPUs vorhanden sind.

Wenn die Anwendungen für eine Datenbank im Wesentlichen aus Transaktionen bestehen, mit denen die Daten aktualisiert werden, führt eine Erhöhung der Anzahl der Seitenlöschfunktionen zu einer Leistungsverbesserung. Durch die Erhöhung der Anzahl der Seitenlöschfunktionen wird außerdem die Zeit für Recoverys nach Systemausfällen, zum Beispiel aufgrund von Netzausfall, verringert, weil der Inhalt der Datenbank auf der Platte zu einem gegebenen Zeitpunkt aktueller ist.

Empfehlung: Bei der Einstellung dieses Parameters müssen folgende Faktoren beachtet werden:

- Anwendungsart
 - Wenn es sich um eine reine Abfragedatenbank handelt, die nicht aktualisiert wird, setzen Sie diesen Parameter auf den Wert Null (0). Eine Ausnahme hiervon wäre, wenn die Abfrageauslastung dazu führt, dass viele TEMP-Tabellen erstellt werden. (Verwenden Sie das Dienstprogramm EXPLAIN, um dies festzustellen.)
 - Wenn Transaktionen für die Datenbank ausgeführt werden, setzen Sie diesen Parameter auf einen Wert zwischen 1 und der Anzahl der physischen Speichereinheiten, die für diese Datenbank verwendet werden.

- Auslastung

Umgebungen mit hohem Aufkommen an aktualisierenden Transaktionen können eventuell die Konfiguration weiterer Seitenlöschfunktionen erforderlich machen.

- Pufferpoolgrößen
Umgebungen mit großen Pufferpools können eventuell auch die Konfiguration weiterer Seitenlöschfunktionen erforderlich machen.

Mit Hilfe des Datenbanksystemmonitors können Sie diesen Konfigurationsparameter optimieren, indem Sie die Informationen des Ereignismonitors zu Schreibaktivitäten aus einem Pufferpool heranziehen:

- Der Wert des Parameters kann verringert werden, wenn die beiden folgenden Bedingungen erfüllt sind:
 - *pool_data_writes* ist ungefähr gleich *pool_async_data_writes*.
 - *pool_index_writes* ist ungefähr gleich *pool_async_index_writes*.
- Der Wert des Parameters sollte erhöht werden, wenn eine der folgenden Bedingungen erfüllt ist:
 - *pool_data_writes* ist viel größer als *pool_async_data_writes*.
 - *pool_index_writes* ist viel größer als *pool_async_index_writes*.

Zugehörige Referenzen:

- „pool_async_data_writes - Buffer Pool Asynchronous Data Writes monitor element“ in *Systemmonitor Handbuch und Referenz*
- „pool_async_index_writes - Buffer Pool Asynchronous Index Writes monitor element“ in *Systemmonitor Handbuch und Referenz*
- „pool_data_writes - Buffer Pool Data Writes monitor element“ in *Systemmonitor Handbuch und Referenz*
- „pool_index_writes - Buffer Pool Index Writes monitor element“ in *Systemmonitor Handbuch und Referenz*
- „chngpgs_thresh - Schwellenwert für geänderte Seiten “ auf Seite 476
- „GET DATABASE CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE CONFIGURATION command“ in *Command Reference*

num_ioservers - Anzahl von E/A-Servern

Konfigurationstyp	Datenbank
Parametertyp	Konfigurierbar
Standardwert [Bereich]	Automatic [1 - 255]
Maßeinheit	Zähler
Zuordnung	Wenn eine Anwendung die Verbindung zu einer Datenbank herstellt
Freigabe	Wenn eine Anwendung die Verbindung zu einer Datenbank trennt

E/A-Server, die auch als Vorablesefunktionen bezeichnet werden, werden für Datenbankagenten verwendet, um Vorablese-E/A-Operationen und asynchrone E/A-Operationen durch Dienstprogramme wie BACKUP und RESTORE auszuführen. Mit diesem Parameter wird die Anzahl der E/A-Server für eine Datenbank definiert. Zu jedem beliebigen Zeitpunkt kann nur diese Anzahl von E/A-Servern zum Vorablesen und für Dienstprogramme für eine Datenbank aktiv sein. Ein E/A-Server wartet, während eine vom ihm eingeleitete E/A-Operation ausgeführt wird. Nicht vorabgelesene Ein-/Ausgaben werden direkt von den Datenbankagenten terminiert, so dass diese Ein-/Ausgaben nicht der Begrenzung durch den Parameter *num_ioservers* unterliegen.

Wenn dieser Parameter auf AUTOMATIC gesetzt wird, richtet sich die Anzahl der gestarteten Vorablesefunktionen nach den Einstellungen für die Parallelität der Tabellenbereiche in der aktuellen Datenbankpartition. (Die Einstellungen für die Parallelität werden mit Hilfe der Umgebungsvariablen DB2_PARALLEL_IO gesteuert.) Für jeden DMS-Tabellenbereich wird der Wert dieser Parallelitätseinstellung mit der maximalen Anzahl von Containern im Stripe-Set des Tabellenbereichs multipliziert. Für jeden SMS-Tabellenbereich wird der Wert dieser Parallelitätseinstellung mit der Anzahl von Containern im Tabellenbereich multipliziert. Das höchste Ergebnis aus allen Tabellenbereichen in der aktuellen Datenbankpartition wird als Anzahl der zu startenden Vorablesefunktionen verwendet. Es werden in jedem Fall mindestens drei Vorablesefunktionen gestartet, wenn dieser Parameter auf AUTOMATIC gesetzt ist.

Wenn dieser Parameter auf AUTOMATIC gesetzt ist, wird die Anzahl der bei der Aktivierung der Datenbank zu startenden Vorablesefunktionen anhand der folgenden Formel berechnet:

Anzahl Vorablesefunktionen = max(max über alle Tabellenbereiche(Parallelitätseinstellung * [SMS: Anz. Container; DMS: max. Anz. Container im Stripe-Set]), 3)

Empfehlung: Um alle E/A-Einheiten des Systems voll auszunutzen, empfiehlt sich im Allgemeinen ein Wert, der um 1 oder 2 höher ist als die Anzahl der physischen Einheiten, auf denen sich die Datenbank befindet. Es ist besser, zusätzliche E/A-Server zu konfigurieren, da mit jedem E/A-Server nur geringfügiger Systemaufwand verbunden ist und alle nicht benötigten E/A-Server inaktiv bleiben.

Zugehörige Referenzen:

- „GET DATABASE CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE CONFIGURATION command“ in *Command Reference*
- „dft_prefetch_sz - Standardwert für PREFETCHSIZE “ auf Seite 477
- „seqdetect - Markierung für Sequenzerkennung “ auf Seite 481

numsegs - Standardanzahl von SMS-Containern

Konfigurationstyp	Datenbank
Parametertyp	Informativ
Maßeinheit	Zähler

Dieser Parameter, der sich ausschließlich auf SMS-Tabellenbereiche bezieht, gibt die Anzahl der Container an, die innerhalb der Standardtabellenbereiche erstellt werden. Dieser Parameter zeigt den Wert an, der bei der Erstellung der Datenbank verwendet wurde, unabhängig davon, ob er in dem Befehl CREATE DATABASE explizit oder implizit angegeben wurde. Von der Anweisung CREATE TABLESPACE wird dieser Parameter **nicht** verwendet.

Zugehörige Referenzen:

- „GET DATABASE CONFIGURATION command“ in *Command Reference*

seqdetect - Markierung für Sequenzerkennung

Konfigurationstyp	Datenbank
Parametertyp	Online konfigurierbar
Weitergabeklasse	Sofort

Standardwert [Bereich]

Yes [Yes; No]

Der Datenbankmanager kann die E/A-Operationen überwachen und, wenn Seiten sequenziell gelesen werden, das E/A-Vorablesen (I/O Prefetching) aktivieren. Diese Art des sequenziellen Vorablesens ist die *Sequenzerkennung*. Mit dem Konfigurationsparameter *seqdetect* können Sie steuern, ob der Datenbankmanager die Sequenzerkennung ausführen soll.

Wenn für diesen Parameter der Wert "No" angegeben wird, findet das Vorablesen nur dann statt, wenn der Datenbankmanager erkennt, dass dies nützlich ist, z. B. bei Tabellensortierungen, Tabellensuchen oder einem Vorablesezugriff über Listen.

Empfehlung: In den meisten Fällen sollte der Standardwert für diesen Parameter verwendet werden. Inaktivieren Sie die Sequenzerkennung nur dann, wenn andere Optimierungsversuche bisher nicht zur Behebung schwerwiegender Leistungsprobleme bei Abfragen geführt haben.

Zugehörige Referenzen:

- „dft_prefetch_sz - Standardwert für PREFETCHSIZE ” auf Seite 477
- „GET DATABASE CONFIGURATION command” in *Command Reference*
- „RESET DATABASE CONFIGURATION command” in *Command Reference*
- „UPDATE DATABASE CONFIGURATION command” in *Command Reference*

Agenten

Die folgenden Parameter können die Anzahl der Anwendungen beeinflussen, die gleichzeitig ausgeführt werden können, um eine optimale Leistung zu erzielen:

- „agentpri - Agentenpriorität ”
- „avg_appls - Durchschnittliche Anzahl aktiver Anwendungen ” auf Seite 484
- „max_connections - Maximale Anzahl von Clientverbindungen ” auf Seite 485
- „max_coordagents - Maximale Anzahl koordinierender Agenten ” auf Seite 485
- „maxagents - Maximale Anzahl von Agenten ” auf Seite 487
- „maxappls - Maximale Anzahl aktiver Anwendungen ” auf Seite 487
- „maxcagents - Maximale Anzahl gleichzeitig aktiver Agenten ” auf Seite 489
- „maxfilop - Maximale Anzahl offener Datenbankdateien pro Anwendung ” auf Seite 490
- „maxtotfilop - Maximale Anzahl offener Dateien ” auf Seite 491
- „num_initagents - Anfangswert für die Anzahl Agenten im Pool ” auf Seite 491
- „num_poolagents - Agentenpoolgröße ” auf Seite 492

agentpri - Agentenpriorität

Konfigurationstyp

Datenbankmanager

Gilt für

- Datenbankserver mit lokalen und fernen Clients
- Datenbankserver mit lokalen Clients
- Partitionierten Datenbankserver mit lokalen und fernen Clients

Parametertyp

Konfigurierbar

Standardwert [Bereich]

AIX -1 (System) [41 - 125]

Andere UNIX-Plattformen

-1 (System) [41 - 128]

Windows

-1 (System) [0 - 6]

Mit diesem Parameter wird die Priorität gesteuert, die sowohl allen Agenten als auch anderen Prozessen und Threads der Datenbankmanagerinstanz vom Scheduler des Betriebssystems zugewiesen wird. In einer Umgebung mit partitionierten Datenbanken gehören dazu auch koordinierende Agenten und Subagenten, parallele Systemsteuerprogramme und die FCM-Dämonen. Durch diese Priorität wird festgelegt, wie den DB2-Prozessen, -Agenten und -Threads im Vergleich zu den anderen Prozessen und Threads, die auf dem System ausgeführt werden, CPU-Zeit zugewiesen wird. Wenn der Parameter auf den Wert -1 bzw. System gesetzt ist, wird keine besondere Aktion ausgeführt, und der Datenbankmanager erhält seine CPU-Zeit in der normalen Weise, in der das Betriebssystem allen Prozessen und Threads Prozessorzeit zuweist. Wenn der Parameter auf einen anderen Wert als -1 bzw. System gesetzt wird, erstellt der Datenbankmanager seine Prozesse und Threads mit einer statischen Priorität, die dem Wert des Parameters entspricht. Dadurch können Sie mit diesem Parameter die Priorität steuern, mit der die Prozesse und Threads des Datenbankmanagers auf Ihrem System ausgeführt werden.

Mit diesem Parameter kann der Durchsatz des Datenbankmanager erhöht werden. Die Werte für die Einstellung dieses Parameters sind von dem Betriebssystem abhängig, auf dem der Datenbankmanager ausgeführt wird. Beispielsweise ergeben in einer auf UNIX basierenden Umgebung niedrige numerische Werte hohe Prioritäten. Wenn der Parameter auf einen Wert zwischen 41 und 125 gesetzt wird, erstellt der Datenbankmanager seine Agenten mit einer statischen UNIX-Priorität, die dem Wert dieses Parameters entspricht. Dies ist in auf UNIX basierenden Umgebungen von Bedeutung, weil numerisch niedrige Werte hohe Prioritäten für den Datenbankmanager ergeben. Bei anderen Prozessen (einschließlich Anwendungen und Benutzern) können jedoch Verzögerungen auftreten, da sie nicht genügend CPU-Zeit erhalten. Sie sollten den Wert für diesen Parameter mit den anderen Aktivitäten, die Sie auf der Maschine erwarten, abstimmen.

Empfehlung: Zu Anfang sollte der Standardwert verwendet werden. Dieser Wert stellt einen guten Kompromiss zwischen den Antwortzeiten für andere Benutzer bzw. Anwendungen und dem Durchsatz des Datenbankmanagers dar.

Anmerkung: Auf dem Solaris-Betriebssystem sollten Sie den Standardwert (-1) nicht ändern. Durch das Ändern des Standardwerts wird die Priorität des DB2-Prozesses auf Echtzeit gesetzt, wodurch alle verfügbaren Ressourcen in dem System monopolisiert werden können.

Wenn die Datenbankleistung von Bedeutung ist, können Sie durch Vergleichstests (Benchmark-Tests) die optimale Einstellung für diesen Parameter bestimmen. Eine Erhöhung der Priorität des Datenbankmanagers sollte nur mit großer Vorsicht vorgenommen werden, da die Leistung anderer Benutzerprozesse erheblich beeinträchtigt werden kann, besonders dann, wenn die CPU-Auslastung sehr hoch ist. Durch Erhöhen der Priorität der Datenbankmanagerprozesse und -threads können bedeutende Leistungssteigerungen erzielt werden.

Anmerkung: Wenn Sie auf UNIX-Plattformen für diesen Parameter einen anderen als den Standardwert verwenden, können Sie Governor nicht verwenden, um Agentenprioritäten zu ändern.

Zugehörige Referenzen:

- „GET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE MANAGER CONFIGURATION command“ in *Command Reference*

avg_appls - Durchschnittliche Anzahl aktiver Anwendungen

Konfigurationstyp	Datenbank
Parametertyp	Online konfigurierbar
Weitergabeklasse	Anweisungsgrenzwert
Standardwert [Bereich]	Automatic [1 - maxappls]
Maßeinheit	Zähler

Mit Hilfe dieses Parameters versucht das Abfrageoptimierungsprogramm zu ermitteln, wie viel Pufferpool zur Laufzeit für den ausgewählten Zugriffsplan verfügbar ist.

Für Datenbanken, die in IBM DB2 Version 9.1 erstellt werden, wird dieser Parameter standardmäßig auf den Wert AUTOMATIC gesetzt. Wenn der Parameter *avg_appls* auf den Wert AUTOMATIC gesetzt ist, verwendet das Optimierungsprogramm den Wert 3, wenn die Registrierdatenbankvariable DB2_WORKLOAD=SAP definiert ist und der Wert von *max_appls* größer als 3 ist. Anderenfalls verwendet das Optimierungsprogramm den Wert 1, wenn *avg_appls* den Wert AUTOMATIC hat.

Empfehlung: Wenn DB2 in einer Mehrbenutzerumgebung ausgeführt wird, kann es vorteilhaft sein, wenn das Abfrageoptimierungsprogramm weiß, dass mehrere Abfragebenutzer das System verwenden (besonders bei komplexen Abfragen und einem großen Pufferpool), sodass das Optimierungsprogramm in seinen Annahmen zur Verfügbarkeit von Pufferpool weniger großzügig ist.

Bei der Einstellung dieses Parameters sollten Sie die Anzahl der Anwendungen mit komplexen Abfragen berechnen, die die Datenbank durchschnittlich verwenden. Aus dieser Berechnung sollten alle einfachen OLTP-Anwendungen (Online-Transaktionsprogramme) ausgeklammert werden. Wenn die Bestimmung dieser Anzahl schwierig ist, können Sie folgende Werte miteinander multiplizieren:

- Eine Durchschnittsanzahl aller Anwendungen, die für Ihre Datenbank ausgeführt werden. Der Datenbanksystemmonitor kann Informationen zur Anzahl der Anwendungen zu einem gegebenen Zeitpunkt liefern, so dass Sie anhand mehrerer Probewerte die Durchschnittsanzahl über einen gewissen Zeitraum hinweg berechnen können. Die Informationen des Datenbanksystemmonitors umfassen sowohl OLTP- als auch Nicht-OLTP-Anwendungen.
- Den von Ihnen geschätzten Prozentsatz an Anwendungen mit komplexen Abfragen.

Wie bei der Anpassung anderer Konfigurationsparameter, die das Optimierungsprogramm beeinflussen, sollten Sie auch den Wert dieses Parameters in kleinen Schritten ändern. Dadurch können Sie die Differenzen bei der Pfadauswahl minimieren.

Wenn Sie diesen Parameter geändert haben, sollten Sie für die Anwendungen eventuell einen Rebind durchführen (mit dem Befehl REBIND PACKAGE).

Zugehörige Referenzen:

- „maxappls - Maximale Anzahl aktiver Anwendungen “ auf Seite 487
- „GET DATABASE CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE CONFIGURATION command“ in *Command Reference*

max_connections - Maximale Anzahl von Clientverbindungen

Konfigurationstyp	Datenbankmanager
Parametertyp	Konfigurierbar
Standardwert [Bereich]	-1 (<i>max_coordagents</i>) [-1; 1 - 64.000]

Wenn der Konzentrador inaktiviert ist, gibt dieser Parameter die maximal zulässige Anzahl Clientverbindungen pro Datenbankpartition an. Der Konzentrador ist inaktiviert, wenn *max_connections* kleiner oder gleich *max_coordagents* ist. Der Konzentrador ist aktiviert, wenn *max_connections* größer als *max_coordagents* ist.

Dieser Parameter steuert die maximale Anzahl der Anwendungen, die mit der Instanz verbunden sein können. In der Regel wird jede Anwendung einem Koordinatoragenten zugeordnet. Ein Agent erleichtert die Operationen zwischen der Anwendung und der Datenbank. Die Konzentradorfunktion wird nicht aktiviert, wenn der Standardwert für diesen Parameter verwendet wird. Daher arbeitet jeder Agent mit seinem eigenen privaten Speicher und benutzt Ressourcen des Datenbankmanagers und globale Datenbankressourcen, wie z. B. den Pufferpool, gemeinsam mit anderen Agenten. Die Konzentradorfunktion wird hingegen aktiviert, wenn der Parameter auf einen Wert gesetzt wird, der den Standardwert übersteigt. Es ist die Aufgabe des Konzentrators, die Anzahl der Serverressourcen pro Clientanwendung so weit zu verringern, dass ein DB2 Connect-Gateway mehr als 10.000 Clientverbindungen verarbeiten kann.

Der Wert -1 gibt an, dass der Grenzwert gleich dem Wert des Parameters *max_coordagents* ist.

In früheren Versionen von DB2 hatte dieser Parameter den Namen *max_logicagents*.

Zugehörige Referenzen:

- „GET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE MANAGER CONFIGURATION command“ in *Command Reference*

max_coordagents - Maximale Anzahl koordinierender Agenten

Konfigurationstyp	Datenbankmanager

Gilt für

- Datenbankserver mit lokalen und fernen Clients
- Datenbankserver mit lokalen Clients

	• Partitionierten Datenbankserver mit lokalen und fernen Clients
Parametertyp	Konfigurierbar
Standardwert [Bereich]	-1 (<i>maxagents</i> – <i>num_initagents</i>) [-1, 0 – <i>maxagents</i>] In Umgebungen mit partitionierten Datenbanken und Umgebungen, in denen <i>intra_parallel</i> auf "Yes" gesetzt ist, ist der Standardwert <i>maxagents</i> minus <i>num_initagents</i>; andernfalls ist der Standardwert <i>maxagents</i>. Dadurch wird sichergestellt, dass in Datenbankumgebungen ohne partitionierte Datenbanken <i>max_coordagents</i> immer gleich <i>maxagents</i> ist, außer wenn das System für partitionsinterne Parallelität konfiguriert ist. Wenn Sie nicht mit einer Umgebung mit partitionierten Datenbanken arbeiten und den Parameter <i>intra_parallel</i> nicht aktiviert haben, muss <i>max_coordagents</i> gleich <i>maxagents</i> sein.

Wenn der Konzentrator nicht aktiviert ist, d. h., wenn *max_connections* gleich *max_coordagents* ist, legt dieser Parameter die maximale Anzahl koordinierender Agenten fest, die in einer partitionierten oder nicht partitionierten Datenbankumgebung gleichzeitig auf einem Server vorhanden sein können.

Für jede lokale oder ferne Anwendung, die eine Verbindung zu einer Datenbank oder einer Instanz herstellt, wird ein koordinierender Agent aufgerufen. Anforderungen, für die eine Instanzverbindung erforderlich ist, sind z. B. CREATE DATABASE, DROP DATABASE und Befehle des Datenbanksystemmonitors.

Wenn der Konzentrator aktiviert ist, d. h., wenn *max_connections* größer als *max_coordagents* ist, kann es mehr Verbindungen als Koordinatoragenten für die Verbindungen geben. Eine Anwendung ist nur dann aktiv, wenn sie von einem Koordinatoragenten bedient wird. Andernfalls ist die Anwendung inaktiv. Anforderungen von einer aktiven Anwendung werden vom Koordinatoragenten der Datenbank (und in SMP- oder MPP-Konfigurationen von Subagenten) bedient. Anforderungen von einer inaktiven Anwendung werden in eine Warteschlange gestellt, bis ein Datenbankkoordinatoragent zur Bedienung der Anwendung zugeordnet wird, wenn diese aktiv wird. Daher kann dieser Parameter zur Steuerung der Systembelastung verwendet werden.

Zugehörige Referenzen:

- „num_initagents - Anfangswert für die Anzahl Agenten im Pool “ auf Seite 491
- „num_poolagents - Agentenpoolgröße “ auf Seite 492
- „maxagents - Maximale Anzahl von Agenten “ auf Seite 487
- „intra_parallel - Partitionsinterne Parallelität aktivieren “ auf Seite 559
- „GET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE MANAGER CONFIGURATION command“ in *Command Reference*

maxagents - Maximale Anzahl von Agenten

Konfigurationstyp	Datenbankmanager
Gilt für	• Datenbankserver mit lokalen und fernen Clients • Datenbankserver mit lokalen Clients • Partitionierten Datenbankserver mit lokalen und fernen Clients
Parametertyp	Konfigurierbar
Standardwert [Bereich]	200 [1 – 64 000] 400 [1 – 64 000] auf partitioniertem Datenbankserver mit lokalen und fernen Clients
Maßeinheit	Zähler

Mit diesem Parameter wird die maximale Anzahl von Datenbankmanageragenten (Koordinatoragenten und Subagenten) angegeben, die zu einem gegebenen Zeitpunkt zur Verfügung stehen, um Anwendungsanforderungen zu empfangen. Wenn Sie die Anzahl koordinierender Agenten begrenzen möchten, verwenden Sie den Parameter *max_coordagents*.

Dieser Parameter kann in Umgebungen mit eingeschränkten Speicherkapazitäten nützlich sein, um den Gesamtspeicherbedarf des Datenbankmanagers zu begrenzen, da jeder zusätzliche Agent zusätzlichen Speicher benötigt.

Empfehlung: Der Wert des Parameters *maxagents* sollte mindestens der Summe der Werte für *maxappls* aller Datenbanken, auf die gleichzeitig zugegriffen werden kann, entsprechen. Wenn die Anzahl der Datenbanken größer als der Wert des Parameters *numdb* ist, dann besteht die sicherste Methode zur Angabe dieses Werts darin, das Produkt von *numdb* und dem größten Wert für *maxappls* zu bilden.

Für die Initialisierung und Verwaltung jedes weiteren Agenten ist zusätzlicher Speicher erforderlich, der beim Starten des Datenbankmanagers zugeordnet wird.

Zugehörige Referenzen:

- „maxappls - Maximale Anzahl aktiver Anwendungen “ auf Seite 487
- „maxcagents - Maximale Anzahl gleichzeitig aktiver Agenten “ auf Seite 489
- „max_coordagents - Maximale Anzahl koordinierender Agenten “ auf Seite 485
- „fenced_pool - Maximale Anzahl abgeschirmter Prozesse “ auf Seite 493
- „num_poolagents - Agentenpoolgröße “ auf Seite 492
- „GET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE MANAGER CONFIGURATION command“ in *Command Reference*

maxappls - Maximale Anzahl aktiver Anwendungen

Konfigurationstyp	Datenbank
Parametertyp	Online konfigurierbar
Weitergabeklasse	Sofort

Standardwert [Bereich]	Automatic [Automatic; 1 – 60 000]
Maßeinheit	Zähler

Mit diesem Parameter wird die maximale Anzahl der (sowohl lokalen als auch fern-) Anwendungen angegeben, die gleichzeitig auf eine Datenbank zugreifen können. Da jede Anwendung, die die Verbindung zu einer Datenbank herstellt, die Zuordnung privaten Speichers erforderlich macht, entsteht durch Erhöhung dieses Parameters möglicherweise mehr Speicherbedarf.

Wenn Sie den Parameter *maxappls* auf *automatic* setzen, ermöglichen Sie dadurch eine beliebige Anzahl verbundener Anwendungen. DB2 ordnet die zur Unterstützung neuer Anwendungen erforderlichen Ressourcen dynamisch zu.

Wenn Sie diesen Parameter nicht auf *automatic* setzen wollen, muss der Wert dieses Parameters größer oder gleich der Summe der verbundenen Anwendungen plus der Anzahl eben dieser Anwendungen sein, die sich gleichzeitig im Prozess eines zweiphasigen Commits oder eines Rollbacks befinden können. Addieren Sie zu dieser Summe die vorausgeschätzte Anzahl unbestätigter Transaktionen, die zu einem gegebenen Zeitpunkt vorhanden sein können.

Wenn eine Anwendung versucht, die Verbindung zu einer Datenbank herzustellen, jedoch der Wert des Parameters *maxappls* bereits erreicht wurde, wird an die Anwendung ein Fehler zurückgegeben, dass die maximale Anzahl von Anwendungen bereits mit der Datenbank verbunden ist.

Wenn mehr Anwendungen Data Links Manager verwenden, sollte der Wert für *maxappls* erhöht werden. Berechnen Sie den erforderlichen Wert anhand folgender Formel:

$$\langle \text{maxappls} \rangle = 5 * (\text{Anzahl Knoten}) + (\text{Höchstanzahl aktiver Anwendungen, die Data Links Manager verwenden})$$

Der unterstützte Maximalwert für Data Links Manager ist 2 000.

In einer Umgebung mit partitionierten Datenbanken ist dies die maximale Anzahl von Anwendungen, die gleichzeitig auf einer Datenbankpartition aktiv sein können. Dieser Parameter beschränkt die Anzahl aktiver Anwendungen für eine Datenbankpartition auf einem Datenbankpartitionsserver unabhängig davon, ob der Server der Koordinatorknoten für die Anwendung ist oder nicht. Für den Katalogknoten in einer Umgebung mit partitionierten Datenbanken ist ein höherer Wert für *maxappls* erforderlich als für andere Umgebungstypen, weil in der Umgebung mit partitionierten Datenbanken jede Anwendung eine Verbindung zum Katalogknoten benötigt.

Empfehlung: Wenn der Wert dieses Parameters erhöht wird, ohne dass der Wert des Parameters *maxlocks* verringert oder der Wert des Parameters *locklist* erhöht wird, könnte die maximale Anzahl Sperren (Parameter *locklist*) früher erreicht werden als die maximale Anzahl Anwendungen, was zu ständigen Problemen durch Sperreneskulation führen kann.

Bis zu einem gewissen Grad wird die maximale Anzahl von Anwendungen auch vom Parameter *maxagents* bestimmt. Eine Anwendung kann nur dann eine Verbindung zur Datenbank herstellen, wenn eine freie Verbindung (*maxappls*) sowie ein freier Agent (*maxagents*) verfügbar ist. Darüber hinaus wird die maximale Anzahl von Anwendungen auch vom Konfigurationsparameter *max_coordagents* gesteuert,

weil keine neuen Anwendungen (d. h. Koordinatoragenten) gestartet werden können, wenn der Wert von *max_coordagents* erreicht ist.

Zugehörige Tasks:

- „Manuelles Auflösen unbestätigter Transaktionen“ in *Systemverwaltung: Konzept*

Zugehörige Referenzen:

- „maxagents - Maximale Anzahl von Agenten “ auf Seite 487
- „max_coordagents - Maximale Anzahl koordinierender Agenten “ auf Seite 485
- „maxlocks - Maximale Anzahl Sperren pro Anwendung “ auf Seite 474
- „locklist - Maximaler Speicher für Sperrenliste “ auf Seite 438
- „avg_appls - Durchschnittliche Anzahl aktiver Anwendungen “ auf Seite 484
- „GET DATABASE CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE CONFIGURATION command“ in *Command Reference*

maxcagents - Maximale Anzahl gleichzeitig aktiver Agenten

Konfigurationstyp

Datenbankmanager

Gilt für

- Datenbankserver mit lokalen und fernen Clients
- Datenbankserver mit lokalen Clients
- Partitionierten Datenbankserver mit lokalen und fernen Clients

Parametertyp

Konfigurierbar

Standardwert [Bereich]

-1 (*max_coordagents*) [-1; 1 – *max_coordagents*]

Maßeinheit

Zähler

Mit diesem Parameter wird die maximale Anzahl von Datenbankmanageragenten festgelegt, die gleichzeitig eine Datenbankmanagertransaktion ausführen können. Dieser Parameter wird zur Steuerung der Systembelastung in Phasen hoher gleichzeitiger Anwendungsaktivität verwendet. Sie könnten beispielsweise ein System haben, das eine große Anzahl von Verbindungen anfordert, jedoch nur über eine begrenzte Speicherkapazität zur Verarbeitung dieser Verbindungen verfügt. In diesem Fall kann die Anpassung dieses Parameters sehr nützlich sein, da es hier aufgrund hoher gleichzeitiger Aktivität zu übermäßigem Seitenwechseln durch das Betriebssystem kommen kann.

Dieser Parameter schränkt nicht die Anzahl der Anwendungen ein, die mit einer Datenbank verbunden sein können. Er begrenzt nur die Anzahl der Datenbankmanageragenten, die gleichzeitig vom Datenbankmanager verarbeitet werden können, wodurch der Bedarf an Systemressourcen in Phasen sehr starker Belastung eingeschränkt wird.

Der Wert -1 gibt an, dass der Grenzwert gleich dem Wert des Parameters *max_coordagents* ist.

Empfehlung: In den meisten Fällen kann der Standardwert für diesen Parameter übernommen werden. In Fällen, wo es durch den umfassenden gleichzeitigen Zugriff von Anwendungen zu Problemen kommt, können Sie durch Vergleichstests (Benchmark-Tests) die beste Einstellung für diesen Parameter ermitteln, um die Leistung der Datenbank zu optimieren.

Zugehörige Referenzen:

- „maxappls - Maximale Anzahl aktiver Anwendungen “ auf Seite 487
- „maxagents - Maximale Anzahl von Agenten “ auf Seite 487
- „max_coordagents - Maximale Anzahl koordinierender Agenten “ auf Seite 485
- „GET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE MANAGER CONFIGURATION command“ in *Command Reference*

maxfilop - Maximale Anzahl offener Datenbankdateien pro Anwendung

Konfigurationstyp	Datenbank
Parametertyp	Online konfigurierbar
Weitergabeklasse	Transaktionsgrenzwert
Standardwert [Bereich]	
	UNIX 64 [2 – 1950]
	Windows 64 [2 – 32 768]
Maßeinheit	Zähler

Mit diesem Parameter wird die maximale Anzahl der Dateikennungen angegeben, die für jeden einzelnen Datenbankagenten geöffnet sein können. Wenn durch das Öffnen einer Datei dieser Wert überschritten wird, werden einige von diesem Agenten verwendete Dateien geschlossen. Wenn der Wert für den Parameter *maxfilop* zu klein ist, wächst der Systemaufwand für das Öffnen und Schließen von Dateien, um diesen Grenzwert nicht zu überschreiten, übermäßig an und kann die Leistung beeinträchtigen.

Sowohl SMS-Tabellenbereiche als auch DMS-Tabellenbereichsdateicontainer werden bei der Interaktion des Datenbankmanagers mit dem Betriebssystem als Dateien behandelt, so dass Dateikennungen für sie erforderlich sind. In der Regel werden von SMS-Tabellenbereichen vergleichsweise mehr Dateien verwendet, als von DMS-Dateitabellenbereichen Container verwendet werden. Daher wird für diesen Parameter ein höherer Wert benötigt, wenn SMS-Tabellenbereiche verwendet werden, als für DMS-Dateitabellenbereiche erforderlich wäre.

Mit Hilfe dieses Parameters kann außerdem sichergestellt werden, dass die Gesamtzahl der Dateikennungen, die vom Datenbankmanager verwendet werden, den Grenzwert des Betriebssystems nicht überschreitet, indem die Anzahl der Dateikennungen pro Agent auf einen bestimmten Wert gesetzt wird. Die tatsächliche Anzahl ist je nach Anzahl der gleichzeitig aktiven Agenten unterschiedlich.

Zugehörige Referenzen:

- „maxtofilop - Maximale Anzahl offener Dateien “ auf Seite 491
- „maxappls - Maximale Anzahl aktiver Anwendungen “ auf Seite 487
- „GET DATABASE CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE CONFIGURATION command“ in *Command Reference*

maxtotfilop - Maximale Anzahl offener Dateien

Konfigurationstyp

Datenbankmanager

Gilt für

- Datenbankserver mit lokalen und fernen Clients
- Datenbankserver mit lokalen Clients
- Partitionierten Datenbankserver mit lokalen und fernen Clients

Parametertyp

Konfigurierbar

Standardwert [Bereich]

16 000 [100 – 32 768]

Maßeinheit

Zähler

Mit diesem Parameter wird die maximale Anzahl von Dateien definiert, die von allen Agenten und anderen Threads, die in einer Datenbankmanagerinstanz ausgeführt werden, geöffnet werden können. Wenn durch das Öffnen einer Datei der Wert dieses Parameters überschritten wird, wird ein Fehler an die Anwendung zurückgegeben.

Anmerkung: Dieser Parameter gilt nicht für auf UNIX basierende Plattformen.

Empfehlung: Bei der Einstellung dieses Parameters sollte die Anzahl der Dateikennungen beachtet werden, die für jede Datenbank in der Datenbankmanagerinstanz verwendet werden könnten. Ermitteln Sie auf folgende Weise einen oberen Grenzwert für diesen Parameter:

1. Berechnen Sie anhand der folgenden Formel die maximale Anzahl von Dateikennungen, die für jede Datenbank in der Instanz geöffnet werden könnten:
$$\text{maxappls} * \text{maxfilop}$$
2. Berechnen Sie die Summe der obigen Ergebnisse, und stellen Sie sicher, dass sie den Maximalwert des Parameters nicht überschreitet.

Wenn eine neue Datenbank erstellt wird, sollten Sie den Wert für diesen Parameter neu bestimmen.

Zugehörige Referenzen:

- „maxfilop - Maximale Anzahl offener Datenbankdateien pro Anwendung “ auf Seite 490
- „GET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE MANAGER CONFIGURATION command“ in *Command Reference*

num_initagents - Anfangswert für die Anzahl Agenten im Pool

Konfigurationstyp

Datenbankmanager

Gilt für

- Datenbankserver mit lokalen und fernen Clients
- Datenbankserver mit lokalen Clients
- Partitionierten Datenbankserver mit lokalen und fernen Clients

Parametertyp	Konfigurierbar
Standardwert [Bereich]	0 [0 — <i>num_poolagents</i>]

Dieser Parameter gibt an, wie viele inaktive Agenten beim Ausführen von DB2START im Agentenpool erstellt werden.

Zugehörige Referenzen:

- „maxagents - Maximale Anzahl von Agenten “ auf Seite 487
- „num_poolagents - Agentenpoolgröße “ auf Seite 492
- „max_coordagents - Maximale Anzahl koordinierender Agenten “ auf Seite 485
- „GET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE MANAGER CONFIGURATION command“ in *Command Reference*

num_poolagents - Agentenpoolgröße

Konfigurationstyp	Datenbankmanager

Gilt für

- Datenbankserver mit lokalen und fernen Clients
- Datenbankserver mit lokalen Clients
- Partitionierten Datenbankserver mit lokalen und fernen Clients

Parametertyp	Konfigurierbar
Standardwert [Bereich]	-1 (<i>maxagents</i> / 2) [-1; 0 - <i>maxagents</i>]

Wenn die Konzentratorkfunktion nicht aktiviert ist, d. h., wenn *max_connections* kleiner oder gleich *max_coordagents* ist, legt dieser Parameter die maximale Größe des Pools inaktiver Agenten fest. Inaktive Agenten können als parallele Subagenten oder als Koordinatoragenten verwendet werden. Wurden mehr Agenten erstellt, als der Wert dieses Parameters angibt, werden diese nach Ausführung ihrer aktuellen Anforderung nicht in den Pool zurückgestellt sondern beendet.

Wenn die Konzentratorkfunktion aktiviert ist, d. h., wenn *max_connections* größer als *max_coordagents* ist, werden Agenten unabhängig vom Wert dieses Parameters immer an den Pool zurückgegeben. Basierend auf der Systembelastung und der Zeit, die die Agenten inaktiv im Pool bleiben, können sich Agenten bei Bedarf selbst beenden, um die Größe des Pools inaktiver Agenten auf den konfigurierten Parameterwert zu reduzieren.

Außer bei aktivierter Konzentratorkfunktion werden, wenn dieser Parameter den Wert 0 hat, Agenten nach Bedarf erstellt und beendet, wenn sie die Ausführung ihrer aktuellen Anforderung abgeschlossen haben.

Empfehlung: Wenn Sie in einer Entscheidungshilfeumgebung arbeiten, in der nur wenige Anwendungen gleichzeitig mit der Datenbank verbunden sind, setzen Sie den Parameter *num_poolagents* auf einen kleinen Wert, um zu verhindern, dass ein Agentenpool mit inaktiven Agenten gefüllt wird.

Wenn Sie eine Transaktionsverarbeitungsumgebung betreiben, in der viele Anwendungen gleichzeitig mit der Datenbank verbunden sind, erhöhen Sie den Wert für *num_poolagents*, um den Aufwand für die häufige Erstellung und Beendigung von Agenten zu vermeiden.

Zugehörige Referenzen:

- „num_initagents - Anfangswert für die Anzahl Agenten im Pool “ auf Seite 491
- „maxagents - Maximale Anzahl von Agenten “ auf Seite 487
- „max_querydegree - Maximaler Grad der Parallelität bei Abfragen “ auf Seite 559
- „max_coordagents - Maximale Anzahl koordinierender Agenten “ auf Seite 485
- „GET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE MANAGER CONFIGURATION command“ in *Command Reference*

Gespeicherte Prozeduren und benutzerdefinierte Funktionen

Die folgenden Parameter können sich auf die Leistung abgeschirmter gespeicherter Prozeduren und benutzerdefinierter Funktionen auswirken:

- „fenced_pool - Maximale Anzahl abgeschirmter Prozesse “
- „keepfenced - Abgeschirmten Prozess beibehalten “ auf Seite 494
- „num_initfenced - Anfangswert für die Anzahl abgeschirmter Prozesse “ auf Seite 496

fenced_pool - Maximale Anzahl abgeschirmter Prozesse

Konfigurationstyp

Datenbankmanager

Gilt für

- Datenbankserver mit lokalen und fernen Clients
- Datenbankserver mit lokalen Clients
- Partitionierten Datenbankserver mit lokalen und fernen Clients

Parametertyp

Konfigurierbar

Standardwert [Bereich]

-1 (0 – *max_coordagents*)

Maßeinheit

Zähler

Für Thread-db2fmp-Prozesse (Prozesse, die threadsichere gespeicherte Prozeduren und benutzerdefinierte Funktionen bedienen) stellt dieser Parameter die Anzahl Threads dar, die in jedem db2fmp-Prozess zwischengespeichert werden. Für db2fmp-Prozesse, die keine Threadprozesse sind, stellt dieser Parameter die Anzahl zwischengespeicherter Prozesse dar.

Empfehlung: Wenn Ihre Umgebung abgeschirmte gespeicherte Prozeduren oder benutzerdefinierte Funktionen verwendet, kann über diesen Parameter sichergestellt werden, dass zur Verarbeitung der maximalen Anzahl gleichzeitig auf der Instanz ablaufender gespeicherter Prozeduren und benutzerdefinierter Funktionen eine ausreichende Anzahl db2fmp-Prozesse verfügbar ist. Dadurch wird sicherge-

stellt, dass beim Ausführen von gespeicherten Prozeduren und benutzerdefinierten Funktionen keine neuen Prozesse im abgeschirmten Modus erstellt werden müssen.

Wenn der Wert dieses Parameters auf –1 gesetzt wird, ist die maximale Anzahl von Prozessen im abgeschirmten Modus gleich dem für den Parameter *max_coordagents* festgelegten Wert.

Wenn sich herausstellt, dass der Standardwert für Ihre Umgebung nicht geeignet ist, weil eine unangemessen große Menge Systemressourcen für db2fmp-Prozesse verwendet wird und es deshalb zu Leistungseinbußen des Datenbankmanagers kommt, kann anhand der folgenden Informationen eine erste Optimierung für diesen Parameter vorgenommen werden:

fenced_pool = Anzahl Anwendungen, die gleichzeitig Aufrufe gespeicherter Prozeduren und UDF-Aufrufe absetzen dürfen

Wenn der Parameter *keepfenced* auf den Wert *yes* gesetzt ist, bleibt jeder im Cache-pool erstellte db2fmp-Prozess bestehen und verbraucht auch dann noch Systemressourcen, wenn der Aufruf der Routine im abgeschirmten Modus verarbeitet und an den Agenten zurückgegeben wurde.

Wenn *keepfenced* auf *no* gesetzt ist, werden db2fmp-Prozesse, die keine Threadprozesse sind, nach Beendigung der Ausführung beendet, und es gibt keinen Cache-pool. Multithread-db2fmp-Prozesse sind weiter vorhanden, es werden in diesen Prozessen jedoch keine Threads in den Pool gestellt. Dies bedeutet, dass Sie auf Ihrem System einen C-Thread-db2fmp-Prozess und einen Java-Thread-db2fmp-Prozess haben können, auch wenn *keepfenced* auf *no* gesetzt ist.

In früheren Versionen von DB2 hatte dieser Parameter den Namen *maxdari*.

Zugehörige Referenzen:

- „maxagents - Maximale Anzahl von Agenten “ auf Seite 487
- „keepfenced - Abgeschirmten Prozess beibehalten “ auf Seite 494
- „num_initfenced - Anfangswert für die Anzahl abgeschirmter Prozesse “ auf Seite 496
- „max_coordagents - Maximale Anzahl koordinierender Agenten “ auf Seite 485
- „GET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE MANAGER CONFIGURATION command“ in *Command Reference*

keepfenced - Abgeschirmten Prozess beibehalten

Konfigurationstyp Datenbankmanager

Gilt für

- Datenbankserver mit lokalen und fernen Clients
- Datenbankserver mit lokalen Clients
- Partitionierten Datenbankserver mit lokalen und fernen Clients

Parametertyp Konfigurierbar

Standardwert [Bereich] Yes [Yes; No]

Dieser Parameter gibt an, ob ein Prozess im abgeschirmten Modus beibehalten wird, nachdem der Aufruf einer Routine im abgeschirmten Modus beendet ist. Prozesse im abgeschirmten Modus werden als getrennte Systementitäten erstellt, um vom Benutzer geschriebenen Code im abgeschirmten Modus vom Agentenprozess des Datenbankmanagers zu isolieren. Dieser Parameter gilt nur für Datenbankserver.

Wenn *keepfenced* auf *no* gesetzt ist und die ausgeführte Routine nicht threadsicher ist, wird für jeden Aufruf im abgeschirmten Modus ein neuer Prozess im abgeschirmten Modus erstellt und zerstört. Wenn *keepfenced* auf *no* gesetzt ist und die ausgeführte Routine threadsicher ist, wird der Prozess im abgeschirmten Modus beibehalten, der für den Aufruf erstellte Thread wird jedoch beendet. Wenn *keepfenced* auf *yes* gesetzt ist, wird ein Thread oder Prozess im abgeschirmten Modus für nachfolgende Aufrufe im abgeschirmten Modus wiederverwendet. Wird der Datenbankmanager gestoppt, werden alle noch anstehenden Threads und Prozesse im abgeschirmten Modus beendet.

Wenn dieser Parameter auf *yes* gesetzt wird, werden vom Datenbankmanager zusätzliche Systemressourcen für jeden Prozess im abgeschirmten Modus in Anspruch genommen, der aktiviert wird, solange die durch den Parameter *fenced_pool* definierte Anzahl noch nicht erreicht ist. Ein neuer Prozess wird nur dann erstellt, wenn zur Verarbeitung eines nachfolgenden Aufrufs einer Routine im abgeschirmten Modus kein Prozess im abgeschirmten Modus verfügbar ist. Dieser Parameter wird ignoriert, wenn *fenced_pool* auf 0 gesetzt ist.

Empfehlung: In einer Umgebung, in der die Anzahl der Anforderungen im abgeschirmten Modus im Vergleich zu den übrigen Anforderungen groß ist und Systemressourcen nicht begrenzt sind, kann dieser Parameter auf *yes* gesetzt werden. Dadurch wird die Leistung des Prozesses im abgeschirmten Modus erhöht, weil der zusätzliche Systemaufwand zur Ersterstellung eines Prozesses im abgeschirmten Modus vermieden wird, da ein bereits vorhandener Prozess im abgeschirmten Modus zur Verarbeitung des Aufrufs verwendet wird. Dies erspart vor allem bei Java-Routinen den Aufwand, die JVM (Java Virtual Machine) zu starten, was eine erhebliche Leistungssteigerung bedeutet.

Zum Beispiel könnte bei einer OLTP-Bankanwendung (OLTP - Online-Transaktionsprogramm) für Soll- und Haben-Transaktionen der Code, mit dem jede Transaktion ausgeführt wird, in einer gespeicherten Prozedur ausgeführt werden, die in einem Prozess im abgeschirmten Modus ausgeführt wird. In dieser Anwendung wird die Hauptauslastung aus Prozessen im abgeschirmten Modus heraus ausgeführt. Wenn dieser Parameter auf *no* gesetzt wird, entsteht für jede Transaktion der Systemaufwand zum Erstellen eines neuen Prozesses im abgeschirmten Modus, wodurch die Leistung erheblich beeinträchtigt wird. Wenn dieser Parameter jedoch auf *yes* gesetzt wird, versucht jede Transaktion, einen vorhandenen Prozess im abgeschirmten Modus zu verwenden, wodurch der zusätzliche Systemaufwand vermieden wird.

In früheren Versionen von DB2 hatte dieser Parameter den Namen *keepdari*.

Zugehörige Referenzen:

- „fenced_pool - Maximale Anzahl abgeschirmter Prozesse “ auf Seite 493
- „GET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*

- „UPDATE DATABASE MANAGER CONFIGURATION command“ in *Command Reference*

num_initfenced - Anfangswert für die Anzahl abgeschirmter Prozesse

Konfigurationstyp	Datenbankmanager
Gilt für	• Datenbankserver mit lokalen und fernen Clients • Datenbankserver mit lokalen Clients • Partitionierten Datenbankserver mit lokalen und fernen Clients
Parametertyp	Konfigurierbar
Standardwert [Bereich]	0 [0 — $\max\_connections + (\max\_agents - \max\_coordinants)$]

Dieser Parameter gibt die Anfangszahl der inaktiven db2fmp-Prozesse an, die keine Threadprozesse sind und die zum Startzeitpunkt der Datenbank (DB2START) im db2fmp-Pool erstellt werden. Wenn Sie diesen Parameter festlegen, wird die einleitende Startzeit für das Ausführen von nicht threadsicheren C- und COBOL-Routinen verringert. Dieser Parameter wird ignoriert, wenn *keepfenced* nicht angegeben wird.

Es ist viel wichtiger, den Parameter *fenced_pool* auf eine für Ihr System geeignete Größe festzulegen, als zum Startzeitpunkt der Datenbank (DB2START) eine Reihe von db2fmp-Prozessen zu starten.

In früheren Versionen von DB2 hatte dieser Parameter den Namen *num_initdaris*.

Zugehörige Referenzen:

- „fenced_pool - Maximale Anzahl abgeschirmter Prozesse “ auf Seite 493
- „keepfenced - Abgeschirmten Prozess beibehalten “ auf Seite 494
- „GET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE MANAGER CONFIGURATION command“ in *Command Reference*

Protokollieren und Recovery

Die Recovery der Umgebung kann eine wichtige Maßnahme sein, um den Verlust kritischer Daten zu verhindern. Es steht eine Reihe von Parametern zur Verfügung, die Ihnen bei der Verwaltung der Umgebung helfen, um sicherstellen zu können, dass eine geeignete Recovery der Daten bzw. Transaktionen durchgeführt werden kann. Diese Parameter werden in folgende Kategorien unterteilt:

- „Datenbankprotokolldateien“ auf Seite 497
- „Datenbankprotokollierung“ auf Seite 507
- „Recovery“ auf Seite 517
- „DUOW-Recovery“ auf Seite 529

Datenbankprotokolldateien

Die folgenden Parameter enthalten Informationen zu Anzahl, Größe und Status der Dateien, die zur Datenbankprotokollierung verwendet werden:

- „logfilsiz - Protokolldateigröße “
- „loghead - Erste aktive Protokolldatei “ auf Seite 498
- „logpath - Pfad zu Protokolldateien “ auf Seite 498
- „logprimary - Anzahl primärer Protokolldateien “ auf Seite 499
- „logsecond - Anzahl sekundärer Protokolldateien “ auf Seite 501
- „max_log - Maximale Protokollgröße pro Transaktion “ auf Seite 502
- „mirrorlogpath - Pfad für Protokollspiegelung “ auf Seite 502
- „newlogpath - Datenbankprotokollpfad ändern “ auf Seite 504
- „num_log_span - Anzahl verwendeter Protokolldateien “ auf Seite 505
- „overflowlogpath - Überlaufprotokollpfad “ auf Seite 506

logfilsiz - Protokolldateigröße

Konfigurationstyp	Datenbank
Parametertyp	Konfigurierbar
Standardwert [Bereich]	

UNIX 1000 [4 — 262 144]

Windows 1000 [4 — 262 144]

Maßeinheit Seiten (4 KB)

Mit diesem Parameter wird die Größe jeder primären und sekundären Protokolldatei definiert. Die Größe dieser Protokolldateien beschränkt die Anzahl der Protokollsätze, die in die Dateien geschrieben werden können, bevor sie voll sind und eine neue Protokolldatei erforderlich wird.

Die Verwendung primärer und sekundärer Protokolldateien sowie die Maßnahmen, die ausgeführt werden, wenn eine Protokolldatei voll ist, sind von der Art der ausgeführten Protokollierung abhängig:

- Umlaufprotokollierung

Eine primäre Protokolldatei kann wieder verwendet werden, wenn die in ihr aufgezeichneten Änderungen mit COMMIT festgeschrieben wurden. Wenn die Größe der Protokolldatei beschränkt ist und Anwendungen eine große Anzahl von Änderungen an der Datenbank vorgenommen haben, ohne die Änderungen festzuschreiben, kann eine primäre Protokolldatei schnell voll werden. Wenn alle primären Protokolldateien voll sind, ordnet der Datenbankmanager sekundäre Protokolldateien für die neuen Protokollsätze zu.

- Protokollierung mit Protokollspeicherung

Wenn eine primäre Protokolldatei voll ist, wird das Protokoll archiviert und eine neue primäre Protokolldatei zugeordnet.

Empfehlung: Die Größe der Protokolldateien muss mit der Anzahl der primären Protokolldateien abgestimmt werden:

- Der Wert des Parameters *logfilsiz* sollte erhöht werden, wenn an der Datenbank eine große Anzahl von Aktualisierungs-, Lösch- oder Einfügetransaktionen ausgeführt wird, durch die die Protokolldatei sehr schnell voll wird.

Anmerkung: Der obere Grenzwert für die Größe der Protokolldatei zusammen mit dem oberen Grenzwert für die Anzahl Protokolldateien (*logprimary* + *logsecond*) ergibt für den aktiven Protokollspeicherbereich einen oberen Grenzwert von 256 GB.

Eine zu kleine Protokolldatei kann sich auf die Systemleistung auswirken, da das Archivieren alter Protokolldateien, das Zuordnen neuer Protokolldateien sowie das Warten auf verwendbare Protokolldateien einen erhöhten Systemaufwand mit sich bringen.

- Der Wert des Parameters *logfilesiz* sollte verringert werden, wenn Platten-speicherplatz knapp ist, da primäre Protokolldateien im Voraus mit dieser Größe zugeordnet werden.

Eine zu große Protokolldatei kann Ihre Flexibilität bei der Verwaltung archivierter Protokolldateien bzw. beim Kopieren der Protokolldateien einschränken, da auf einigen Platten vielleicht keine vollständige Protokolldatei gespeichert werden kann.

Wenn Sie die Protokollspeicherung verwenden, wird die aktuelle aktive Protokolldatei geschlossen und abgeschnitten, wenn die letzte Anwendung die Verbindung zu einer Datenbank trennt. Für die nächste zur Datenbank hergestellte Verbindung wird die nächste Protokolldatei verwendet. Wenn Sie also die Protokollanforderungen Ihrer gleichzeitig ausgeführten Anwendungen kennen, können Sie vielleicht eine Protokolldateigröße festlegen, durch die nicht zu viel Speicher umsonst zugeordnet wird.

Zugehörige Referenzen:

- „logprimary - Anzahl primärer Protokolldateien “ auf Seite 499
- „logsecond - Anzahl sekundärer Protokolldateien “ auf Seite 501
- „softmax - Vor bedingtem Prüfpunkt zu schreibende Protokollsätze “ auf Seite 514
- „GET DATABASE CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE CONFIGURATION command“ in *Command Reference*

loghead - Erste aktive Protokolldatei

Konfigurationstyp Datenbank

Parametertyp Informativ

Dieser Parameter gibt den Namen der zurzeit aktiven Protokolldatei an.

Zugehörige Referenzen:

- „GET DATABASE CONFIGURATION command“ in *Command Reference*

logpath - Pfad zu Protokolldateien

Konfigurationstyp Datenbank

Parametertyp Informativ

Dieser Parameter gibt den aktuellen Pfad an, der für die Protokolldateien verwendet wird. Dieser Parameter kann nicht direkt geändert werden, da der Wert vom Datenbankmanager festgelegt wird, wenn eine Änderung am Parameter *newlogpath* wirksam wird.

Bei der Erstellung einer Datenbank wird die zugehörige Datei des Recoveryprotokolls in einem Unterverzeichnis des Verzeichnisses erstellt, in dem sich die Datenbank befindet. Standardmäßig wird dieses Unterverzeichnis mit dem Namen SQLOGDIR in dem Verzeichnis angelegt, das für die Datenbank erstellt wurde.

Zugehörige Referenzen:

- „newlogpath - Datenbankprotokollpfad ändern “ auf Seite 504
- „GET DATABASE CONFIGURATION command“ in *Command Reference*

logprimary - Anzahl primärer Protokolldateien

Konfigurationstyp	Datenbank
Parametertyp	Konfigurierbar
Standardwert [Bereich]	3 [2 – 256]
Maßeinheit	Zähler
Zuordnung	• Wenn die Datenbank erstellt wird. • Wenn ein Protokoll an eine andere Speicherposition versetzt wird (dies geschieht, wenn der Parameter <i>logpath</i> aktualisiert wird). • Wenn nach der Erhöhung des Werts für den Parameter <i>logprimary</i> die nächste Verbindung zur Datenbank hergestellt wird, nachdem alle Benutzer die Verbindung getrennt hatten. • Wenn nach der Archivierung einer Protokolldatei eine neue Protokolldatei zugeordnet wird. (Der Parameter <i>logretain</i> oder <i>userexit</i> muss aktiviert sein.) • Wenn der Parameter <i>logfilsiz</i> geändert wurde, wird für die aktiven Protokolldateien die neue Größe aktiv, wenn die nächste Verbindung zur Datenbank hergestellt wird, nachdem alle Benutzer die Verbindung zur Datenbank getrennt hatten.
Freigabe	Eine Freigabe erfolgt, wenn der Wert für diesen Parameter herabgesetzt wird. Wenn der Wert herabgesetzt wird, werden nicht mehr benötigte Protokolldateien bei der nächsten Verbindung zur Datenbank gelöscht.

Die primären Protokolldateien reservieren eine feste Menge Speicher, der den Recoveryprotokollen zugeordnet wird. Mit diesem Parameter kann die Anzahl der im Voraus zugeordneten primären Protokolldateien festgelegt werden.

Bei der Umlaufprotokollierung werden die primären Protokolldateien nacheinander wiederholt verwendet. Das heißt, wenn ein Protokoll voll ist, wird die nächste primäre Protokolldatei in der Reihenfolge verwendet, wenn sie verfügbar ist. Ein Protokoll wird als verfügbar angesehen, wenn alle UOWs (UOW = Unit of Work, Arbeitseinheit) mit Protokollsätzen in diesem Protokoll mit COMMIT festgeschrieben oder mit ROLLBACK rückgängig gemacht wurden. Wenn die nächste Protokolldatei in der Reihenfolge nicht verfügbar ist, wird eine sekundäre Protokolldatei zugeordnet und verwendet. Es werden solange weitere sekundäre

Protokolldateien zugeordnet und verwendet, bis die nächste primäre Protokolldatei in der Reihenfolge verfügbar wird oder der durch den Parameter *logsecond* festgelegte Grenzwert erreicht wird. Sobald diese sekundären Protokolldateien vom Datenbankmanager nicht mehr benötigt werden, wird der für sie verwendete Speicher wieder freigegeben.

Für die Anzahl der primären und sekundären Protokolldateien muss Folgendes gelten:

- Wenn *logsecond* den Wert -1 besitzt, ist $\logprimary \leq 256$.
- Wenn *logsecond* nicht den Wert -1 besitzt, ist $(\logprimary + \logsecond) \leq 256$.

Empfehlung: Der Wert, der für diesen Parameter gewählt wird, ist von einer Reihe von Faktoren abhängig, zu denen auch die Art der Protokollierung, die Größe der Protokolldateien und die Art der Verarbeitungsumgebung (z.B. die Länge der Transaktionen und die Häufigkeit der Commits) gehören.

Durch die Erhöhung dieses Werts wird mehr Plattenspeicher für die Protokolle erforderlich, weil die primären Protokolldateien bereits bei der ersten Verbindung zur Datenbank im Voraus zugeordnet werden.

Wenn sich herausstellt, dass häufig sekundäre Protokolldateien zugeordnet werden, kann die Systemleistung möglicherweise durch eine Vergrößerung der Protokolldateien (*logfilsiz*) oder durch eine Erhöhung der Anzahl primärer Protokolldateien verbessert werden.

Bei Datenbanken, auf die nicht oft zugegriffen wird, sollte zur Einsparung von Plattenspeicherplatz dieser Parameter auf den Wert 2 gesetzt werden. Bei Datenbanken, für die die aktualisierende Recovery aktiviert ist, sollte dieser Parameter auf einen größeren Wert gesetzt werden, um den erhöhten Systemaufwand aufgrund der beinahe sofort erforderlichen Zuordnung neuer Protokolle zu vermeiden.

Sie können den Datenbanksystemmonitor zur Ermittlung der geeigneten Größe für die primären Protokolldateien verwenden. Die Beobachtung der folgenden Monitorwerte über einen gewissen Zeitraum hinweg kann sich für die geeignete Einstellung der Parameter als nützlich erweisen, da Durchschnittswerte Ihre tatsächlichen Anforderungen wahrscheinlich besser widerspiegeln.

- *sec_log_used_top* (Max. verwendeter sekundärer Protokollbereich)
- *tot_log_used_top* (Max. verwendeter Gesamtbereich für Protokolle)
- *sec_logs_allocated* (Momentan zugeordnete sekundäre Protokolle)

Zugehörige Referenzen:

- „*sec_log_used_top* - Maximum Secondary Log Space Used monitor element“ in *Systemmonitor Handbuch und Referenz*
- „*tot_log_used_top* - Maximum Total Log Space Used monitor element“ in *Systemmonitor Handbuch und Referenz*
- „*sec_logs_allocated* - Secondary Logs Allocated Currently monitor element“ in *Systemmonitor Handbuch und Referenz*
- „*logfilsiz* - Protokolldateigröße “ auf Seite 497
- „*logsecond* - Anzahl sekundärer Protokolldateien “ auf Seite 501
- „*logretain* - Beibehalten von Protokollen aktivieren “ auf Seite 511
- „*userexit* - Benutzerexit aktivieren “ auf Seite 516
- „GET DATABASE CONFIGURATION command“ in *Command Reference*

- „RESET DATABASE CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE CONFIGURATION command“ in *Command Reference*

logsecond - Anzahl sekundärer Protokolldateien

Konfigurationstyp	Datenbank
Parametertyp	Online konfigurierbar
Weitergabeklasse	Sofort
Standardwert [Bereich]	2 [-1; 0 – 254]
Maßeinheit	Zähler
Zuordnung	Nach Bedarf, wenn <i>logprimary</i> nicht ausreichend ist (siehe unten).
Freigabe	Wenn der Datenbankmanager feststellt, dass sie nicht mehr erforderlich sind.

Dieser Parameter gibt die Anzahl der sekundären Protokolldateien an, die (nach Bedarf) erstellt und für Recoveryprotokolle verwendet werden. Wenn die primären Protokolldateien voll sind, werden die sekundären Protokolldateien (in der Größe *logfilesiz*) nacheinander so zugeordnet, wie sie benötigt werden, und zwar höchstens so viele, wie durch diesen Parameter festgelegt wird. Wenn mehr sekundäre Protokolldateien erforderlich sind, als von diesem Parameter zugelassen werden, wird ein Fehlercode an die Anwendung zurückgegeben, und die Datenbank wird beendet.

Wenn Sie *logsecond* auf -1 setzen, wird die Datenbank mit unbegrenztem Speicherbereich für aktive Protokolle konfiguriert. Für die Größe oder Anzahl der unvollständigen Transaktionen, die in der Datenbank ausgeführt werden, gibt es keine Begrenzung. Wenn Sie *logsecond* auf den Wert -1 setzen, verwenden Sie weiter die Konfigurationsparameter *logprimary* und *logfilesiz*, um die Anzahl der Protokolldateien anzugeben, die DB2 im Pfad für aktive Protokolldateien behalten soll. Wenn DB2 Protokolldaten aus einer Protokolldatei lesen muss, die sich nicht im Pfad für aktive Protokolldateien befindet, ruft DB2 das Benutzerexitprogramm auf, um die Protokolldatei aus dem Archiv in den Pfad für aktive Protokolldateien abzurufen. (DB2 ruft die Dateien in den Überlaufprotokollpfad ab, falls Sie einen solchen Pfad konfiguriert haben.) Nach dem Abrufen der Protokolldatei wird diese Datei von DB2 im Pfad für aktive Protokolldateien zwischengespeichert, so dass das Lesen weiterer Protokolldaten aus dieser Datei schneller erfolgen kann. DB2 verwaltet das Abrufen, Zwischenspeichern und Entfernen dieser Protokolldateien nach Bedarf.

Wenn Ihr Protokollpfad eine Roheinheit ist, müssen Sie den Konfigurationsparameter *overflowlogpath* konfigurieren, um *logsecond* auf den Wert -1 zu setzen.

Wenn Sie *logsecond* auf den Wert -1 setzen, besteht für die Größe der UOW (Unit of Work, Arbeitseinheit) oder die Anzahl gleichzeitig ablaufender UOWs keine Begrenzung. Allerdings können Rollbacks (sowohl auf Sicherungspunktebene als auch auf UOW-Ebene) viel Zeit in Anspruch nehmen, da Protokolldateien aus dem Archiv abgerufen werden müssen. Aus demselben Grund kann die Recovery nach einem Systemabsturz ebenfalls sehr zeitintensiv sein. DB2 schreibt eine Nachricht in das Protokoll mit Benachrichtigungen für die Systemverwaltung, um Sie darauf hinzuweisen, wenn die aktuelle Menge aktiver UOWs die primären Protokoll-

dateien überschreitet. Dies weist zugleich darauf hin, dass Rollback- oder Recoveryoperationen nach einem Systemabsturz viel Zeit in Anspruch nehmen können.

Damit der Parameter *logsecond* auf den Wert -1 gesetzt werden kann, muss der Konfigurationsparameter *logarchmeth1* auf einen anderen Wert als OFF oder LOGRETAIN gesetzt sein.

Empfehlung: Verwenden Sie sekundäre Protokolldateien für Datenbanken, die in periodischen Zeitabständen immer wieder große Mengen an Protokollspeicher benötigen. Sekundäre Protokolldateien sollten beispielsweise eingesetzt werden, wenn eine Anwendung, die einmal im Monat ausgeführt wird, mehr Protokollspeicher benötigt, als durch die primären Protokolldateien zur Verfügung gestellt wird. Da sekundäre Protokolldateien keinen permanenten Dateibereich erfordern, sind sie für solche Zwecke gut geeignet.

Zugehörige Referenzen:

- „logfilesiz - Protokolldateigröße “ auf Seite 497
- „logprimary - Anzahl primärer Protokolldateien “ auf Seite 499
- „logretain - Beibehalten von Protokollen aktivieren “ auf Seite 511
- „userexit - Benutzerexit aktivieren “ auf Seite 516
- „overflowlogpath - Überlaufprotokollpfad “ auf Seite 506
- „GET DATABASE CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE CONFIGURATION command“ in *Command Reference*

max_log - Maximuale Protokollgröße pro Transaktion

Konfigurationstyp	Datenbank
Parametertyp	Online konfigurierbar
Weitergabeklasse	Sofort
Standardwert [Bereich]	0 [0 — 100]
Maßeinheit	Prozent

Wenn der Wert nicht 0 ist, zeigt dieser Parameter den Prozentsatz des Speicherbereichs für primäre Protokolle an, der von einer Transaktion verbraucht werden kann.

Wird der Wert auf 0 gesetzt, besteht keine Begrenzung des Prozentsatzes des gesamten aktiven Protokollbereichs, den eine Transaktion verbrauchen kann. Dies war die Funktionsweise von Transaktionen vor Version 8.

Zugehörige Referenzen:

- „num_log_span - Anzahl verwendeter Protokolldateien “ auf Seite 505
- „GET DATABASE CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE CONFIGURATION command“ in *Command Reference*

mirrorlogpath - Pfad für Protokollspiegelung

Konfigurationstyp	Datenbank
Parametertyp	Konfigurierbar

Standardwert [Bereich]

Null [gültiger Pfad oder gültige Einheit]

Mit diesem Parameter können Sie eine Zeichenfolge von bis zu 242 Byte für den Pfad für die Protokollspiegelung angeben. Diese Zeichenfolge muss auf einen Pfadnamen verweisen, der ein vollständig qualifizierter Pfadname und kein relativer Pfadname ist.

Anmerkung: In einer DB2 ESE-Umgebung mit einer einzelnen oder mehreren Partitionen wird die Knotennummer automatisch an den Pfad angefügt. Dadurch wird die Eindeutigkeit des Pfads in Konfigurationen mit mehreren logischen Knoten sichergestellt.

Wenn *mirrorlogpath* konfiguriert ist, erstellt DB2 aktive Protokolldateien sowohl im Protokollpfad als auch im Pfad für die Protokollspiegelung. Alle Protokolldaten werden in beide Pfade geschrieben. Der Pfad für die Protokollspiegelung enthält Kopien der aktiven Protokolldateien, so dass die Datenbank im Fall eines Plattenfehlers oder eines Benutzerfehlers, durch den aktive Protokolldateien in einem der Pfade zerstört werden, weiter funktioniert.

Wenn der Pfad für die Protokollspiegelung geändert wird, können sich im alten Pfad für die Protokollspiegelung immer noch Protokolldateien befinden. Diese Protokolldateien wurden eventuell nicht archiviert, so dass Sie sie möglicherweise manuell archivieren müssen. Wenn Sie zudem für diese Datenbank Replikation ausführen, benötigt Replikation eventuell weiterhin die vor der Protokollpfadänderung vorhandenen Protokolldateien. Wenn der Datenbankkonfigurationsparameter für Benutzerexit (*userexit*) für die Datenbank auf "Yes" gesetzt ist und wenn alle Protokolldateien entweder automatisch durch DB2 oder von Ihnen selbst manuell archiviert wurden, kann DB2 die Protokolldateien zum Beenden des Replikationsprozesses abrufen. Andernfalls können Sie die Dateien aus dem alten Pfad für die Protokollspiegelung in den neuen Pfad für die Protokollspiegelung kopieren.

Wenn *logpath* oder *newlogpath* eine unformatierte Einheit als Position zum Speichern der Protokolldateien angibt, ist die Protokollspiegelung, die durch *mirrorlogpath* angegeben wird, nicht zulässig. Wenn *logpath* oder *newlogpath* den Dateipfad als Position zum Speichern von Protokolldateien angegeben wird, ist die Protokollspiegelung zulässig, und *mirrorlogpath* muss ebenfalls einen Dateipfad angeben.

Empfehlung: Wie die Protokolldateien sollten sich auch die Spiegelprotokolldateien auf einem physischen Laufwerk mit nur geringem E/A befinden.

Es wird dringend empfohlen, dass dieser Pfad auf einer anderen Einheit eingerichtet wird, als der primäre Protokollpfad.

Mit dem Datenbanksystemmonitor können Sie die Anzahl der E/A-Operationen für die Datenbankprotokollierung verfolgen.

Die folgenden Datenelemente geben das Volumen der E/A-Aktivitäten für die Datenbankprotokollierung zurück. Sie können ein Tool zur Betriebssystemüberwachung verwenden, um Daten zu anderen Platten-E/A-Aktivitäten zu sammeln. Anschließend können Sie beide Arten von E/A-Aktivitäten vergleichen.

- *log_reads* (Anzahl gelesener Protokollseiten)
- *log_writes* (Anzahl geschriebener Protokollseiten)

Zugehörige Referenzen:

- „newlogpath - Datenbankprotokollpfad ändern “ auf Seite 504
- „logpath - Pfad zu Protokolldateien “ auf Seite 498
- „log_reads - Number of Log Pages Read monitor element“ in *Systemmonitor Handbuch und Referenz*
- „log_writes - Number of Log Pages Written monitor element“ in *Systemmonitor Handbuch und Referenz*
- „overflowlogpath - Überlaufprotokollpfad “ auf Seite 506
- „GET DATABASE CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE CONFIGURATION command“ in *Command Reference*

newlogpath - Datenbankprotokollpfad ändern

Konfigurationstyp	Datenbank
Parametertyp	Konfigurierbar
Standardwert [Bereich]	Null [gültiger Pfad oder gültige Einheit]

Mit diesem Parameter können Sie eine Zeichenfolge mit bis zu 242 Byte angeben, um die Speicherposition der Protokolldateien zu ändern. Die Zeichenfolge kann auf einen Pfadnamen oder auf eine Roheinheit verweisen. Beachten Sie, dass ab DB2 Version 9 die Verwendung von Roheinheiten für die Datenbankprotokollierung veraltet ist. Alternativ zur Protokollierung auf Roheinheiten können Sie die Funktionalität zur direkten Ein-/Ausgabe (Direct IO, DIO) oder zur gleichzeitigen Ein-/Ausgabe (Concurrent IO, CIO) verwenden.

Verweist die Zeichenfolge auf einen Pfadnamen, muss es sich um einen vollständig qualifizierten Pfad handeln und nicht um einen relativen Pfad.

In einer DB2 ESE-Umgebung mit einer einzelnen oder mehreren Partitionen wird die Knotennummer automatisch an den Pfad angefügt. Dadurch wird die Eindeutigkeit des Pfads in Konfigurationen mit mehreren logischen Knoten sichergestellt.

Wenn Sie mit Replikation arbeiten wollen und Ihr Protokollpfad eine Roheinheit ist, muss der Konfigurationsparameter *overflowlogpath* konfiguriert werden.

Geben Sie zum Angeben einer Einheit eine Zeichenfolge an, die vom Betriebssystem als eine Einheit erkannt wird. Zum Beispiel:

- Unter Windows: \\.\d: oder \\.\PhysicalDisk5

Anmerkung: Damit Protokolle auf eine Einheit geschrieben werden können, muss Windows Version 4.0 mit Service Pack 3 oder höher installiert sein.

- Auf UNIX-Plattformen /dev/rdblog8

Anmerkung: Eine Einheit können Sie nur auf AIX-, Windows 2000-, Windows-, Solaris-Betriebsumgebung, HP-UX- und Linux-Plattformen angeben.

Diese Einstellung wird nur dann zum Wert des Parameters *logpath* , wenn die beiden folgenden Bedingungen zutreffen:

- Die Datenbank ist in einem konsistenten Zustand, wie durch den Parameter *database_consistent* angegeben.
- Alle Benutzer sind von der Datenbank getrennt.

Wenn die erste neue Verbindung zur Datenbank hergestellt wird, versetzt der Datenbankmanager die Protokolle an die neue, von *logpath* angegebene Speicherposition.

Im alten Protokollpfad befinden sich eventuell noch Protokolldateien. Diese Protokolldateien wurden eventuell nicht archiviert. Sie müssen sie möglicherweise manuell archivieren. Wenn Sie zudem für diese Datenbank eine Replikation ausführen, benötigt die Replikation eventuell weiterhin die vor der Protokollpfadänderung vorhandenen Protokolldateien. Wenn der Datenbankkonfigurationsparameter für Benutzerexit (*userexit*) für die Datenbank auf "Yes" gesetzt ist und wenn alle Protokolldateien entweder automatisch durch DB2 oder von Ihnen selbst manuell archiviert wurden, dann kann DB2 die Protokolldateien zum Beenden des Replikationsprozesses abrufen. Andernfalls können Sie die Dateien aus dem alten Protokollpfad in den neuen Protokollpfad kopieren.

Wenn *logpath* oder *newlogpath* eine Roheinheit als Position zum Speichern der Protokolldateien angibt, ist die Protokollspiegelung, die durch *mirrorlogpath* angegeben wird, nicht zulässig. Wenn *logpath* oder *newlogpath* den Dateipfad als Position zum Speichern von Protokolldateien angegeben wird, ist die Protokollspiegelung zulässig, und *mirrorlogpath* muss ebenfalls einen Dateipfad angeben.

Empfehlung: Es empfiehlt sich, die Protokolldateien auf einer physischen Platte zu speichern, auf der **kein** großes Volumen an E/A-Operationen auftritt. Sie sollten beispielsweise die Protokolldateien nicht auf derselben Platte wie das Betriebssystem oder umfangreiche Datenbanken speichern. Dadurch werden die Protokollierungsvorgänge effizient und verursachen nur ein Minimum an Systemaufwand, wie z. B. Warten auf E/A-Operationen.

Mit dem Datenbanksystemmonitor können Sie die Anzahl der E/A-Operationen für die Datenbankprotokollierung verfolgen.

Die Monitorelemente *log_reads* (Anzahl gelesener Protokollseiten) und *log_writes* (Anzahl geschriebener Protokollseiten) geben den auf die Datenbankprotokollierung bezogenen Umfang an E/A-Aktivität zurück. Sie können ein Tool zur Betriebssystemüberwachung verwenden, um Daten zu anderen Platten-E/A-Aktivitäten zu sammeln. Anschließend können Sie beide Arten von E/A-Aktivitäten vergleichen.

Zugehörige Referenzen:

- „log_reads - Number of Log Pages Read monitor element“ in *Systemmonitor Handbuch und Referenz*
- „log_writes - Number of Log Pages Written monitor element“ in *Systemmonitor Handbuch und Referenz*
- „logpath - Pfad zu Protokolldateien “ auf Seite 498
- „database_consistent - Datenbank ist konsistent “ auf Seite 538
- „GET DATABASE CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE CONFIGURATION command“ in *Command Reference*

num_log_span - Anzahl verwendeter Protokolldateien

Konfigurationstyp	Datenbank
Parametertyp	Online konfigurierbar
Weitergabeklasse	Sofort

Standardwert [Bereich]	0 [0 — 65 535]
Maßeinheit	Zähler

Wenn der Wert nicht 0 ist, zeigt dieser Parameter die Anzahl aktiver Protokolldateien an, über die sich eine aktive Transaktion erstrecken kann.

Wird der Wert auf 0 gesetzt, besteht keine Begrenzung für die Anzahl der Protokolldateien, die eine einzelne Transaktion umfassen kann. Dies war die Funktionsweise von Transaktionen vor Version 8.

Zugehörige Referenzen:

- „max_log - Maximale Protokollgröße pro Transaktion “ auf Seite 502
- „GET DATABASE CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE CONFIGURATION command“ in *Command Reference*

overflowlogpath - Überlaufprotokollpfad

Konfigurationstyp	Datenbank
Parametertyp	Online konfigurierbar
Weitergabeklasse	Sofort
Standardwert [Bereich]	NULL [beliebiger gültiger Pfad]

Je nach Ihren Protokollierungsanforderungen kann dieser Parameter für verschiedene Funktionen verwendet werden.

- Mit diesem Parameter können Sie eine Speicherposition angeben, an der DB2 nach Protokolldateien suchen soll, die für eine ROLLFORWARD-Operation benötigt werden. Dies bietet eine Alternative zur Verwendung der Option OVERFLOW LOG PATH mit dem ROLLFORWARD-Befehl. Anstatt jedes Mal die Option OVERFLOW LOG PATH mit dem ROLLFORWARD-Befehl anzugeben, können Sie einmal diesen Konfigurationsparameter festlegen. Wenn allerdings beides verwendet wird, überschreibt die Option OVERFLOW LOG PATH den Konfigurationsparameter *overflowlogpath* für diese bestimmte ROLLFORWARD-Operation.
- Wenn *logsecond* auf -1 gesetzt ist, können Sie über den Parameter *overflowlogpath* ein Verzeichnis angeben, in dem DB2 aus dem Archiv abgerufene aktive Protokolldateien speichert. (Aktive Protokolldateien müssen für Rollbacks abgerufen werden, wenn sie sich nicht mehr im Pfad für aktive Protokolldateien befinden). Wird für *overflowlogpath* kein Wert angegeben, ruft DB2 die Protokolldateien in den Pfad für aktive Protokolldateien ab. Mit dem Parameter *overflowlogpath* können Sie DB2 zusätzliche Ressourcen zum Speichern von abgerufenen Protokolldateien bereitstellen. Dies bietet den Vorteil, den Ein-/Ausgabearbeit auf verschiedene Datenträger zu verteilen und mehr Protokolldateien im Pfad für aktive Protokolldateien speichern zu können.
- Wenn Sie beispielsweise zur Replikation die API db2ReadLog (vor DB2 Version 8 hatte db2ReadLog den Namen sqlurlog) benutzen müssen, können Sie beispielsweise mit dem Parameter *overflowlogpath* eine Position angeben, an der DB2 nach Protokolldateien sucht, die für diese API benötigt werden. Wenn die Protokolldatei nicht gefunden wird (weder im Pfad für aktive Protokolldateien noch im Überlaufprotokollpfad) und die Datenbank mit dem aktivierten Parameter *userexit* konfiguriert ist, ruft DB2 die Protokolldatei ab. Über den Parameter *overflowlogpath* können Sie auch ein Verzeichnis angeben, in dem DB2 die abgerufenen Protokolldateien speichert. Dadurch wird der Ein-/Ausgabearbeit für

den Pfad für aktive Protokolldateien reduziert, und es können mehr Protokolldateien im Pfad für aktive Protokolldateien gespeichert werden.

- Wenn Sie als Pfad für aktive Protokolldateien eine unformatierte Einheit (Roh-einheit) konfiguriert haben, muss *overflowlogpath* konfiguriert werden, wenn Sie *logsecond* auf -1 setzen oder die API *db2ReadLog* verwenden wollen.

Wenn Sie *overflowlogpath* festlegen wollen, geben Sie eine Zeichenfolge von maximal 242 Byte an. Diese Zeichenfolge muss auf einen Pfadnamen verweisen, der ein vollständig qualifizierter Pfadname und kein relativer Pfadname ist. Der Pfadname muss ein Verzeichnis sein, keine Roheinheit.

Anmerkung: In einer DB2 ESE-Umgebung mit einer einzelnen oder mehreren Partitionen wird die Knotennummer automatisch an den Pfad angefügt. Dadurch wird die Eindeutigkeit des Pfads in Konfigurationen mit mehreren logischen Knoten sichergestellt.

Zugehörige Referenzen:

- „logsecond - Anzahl sekundärer Protokolldateien “ auf Seite 501
- „ROLLFORWARD DATABASE command“ in *Command Reference*
- „db2ReadLog API - Extracts log records“ in *Administrative API Reference*
- „GET DATABASE CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE CONFIGURATION command“ in *Command Reference*

Datenbankprotokollierung

Die folgenden Parameter können den Typ und die Leistung der Datenbankprotokollierung beeinflussen:

- „archretrydelay - Wiederholungsintervall für Archivierung bei Fehler “
- „blk_log_dsk_ful - Bei voller Protokollplatte blockieren “ auf Seite 508
- „failarchpath - Protokollarchivpfad für Funktionsübernahme “ auf Seite 509
- „logarchmeth1 - Primäre Protokollarchivierungsmethode “ auf Seite 509
- „logarchmeth2 - Sekundäre Protokollarchivierungsmethode “ auf Seite 510
- „logarchopt1 - Optionen für primäres Protokollarchiv “ auf Seite 510
- „logarchopt2 - Optionen für sekundäres Protokollarchiv “ auf Seite 511
- „logindexbuild - Erstellte Indexseiten protokollieren “ auf Seite 511
- „logretain - Beibehalten von Protokollen aktivieren “ auf Seite 511
- „mincommit - Anzahl der Gruppencommits “ auf Seite 512
- „numarchretry - Anzahl der Wiederholungen bei Fehler “ auf Seite 514
- „softmax - Vor bedingtem Prüfpunkt zu schreibende Protokollsätze “ auf Seite 514
- „userexit - Benutzerexit aktivieren “ auf Seite 516
- „vendoropt - Lieferantenoptionen “ auf Seite 517

archretrydelay - Wiederholungsintervall für Archivierung bei Fehler

Konfigurationstyp

Datenbank

Gilt für

- Datenbankserver mit lokalen und fernen Clients
- Client

- Datenbankserver mit lokalen Clients
- Partitionierten Datenbankserver mit lokalen und fernen Clients

Parametertyp	Online konfigurierbar
Standardwert [Bereich]	20 [0 – 65 535]

Mit diesem Parameter wird die Anzahl von Sekunden angegeben, die nach einem fehlgeschlagenen Archivierungsversuch abzuwarten ist, bevor die Archivierung der Protokolldatei erneut versucht wird. Nachfolgende Wiederholungen finden nur statt, wenn der Datenbankkonfigurationsparameter *numarchretry* mindestens auf den Wert 1 gesetzt ist.

Zugehörige Referenzen:

- „GET DATABASE CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE CONFIGURATION command“ in *Command Reference*

blk_log_dsk_ful - Bei voller Protokollplatte blockieren

Konfigurationstyp	Datenbank
Parametertyp	Online konfigurierbar
Weitergabeklasse	Sofort
Standardwert [Bereich]	No [Yes; No]

Dieser Konfigurationsparameter kann so festgelegt werden, dass keine Fehler aufgrund erschöpfter Festplattenkapazität generiert werden, wenn DB2 keine neue Protokolldatei im aktiven Protokollpfad erstellen kann. Stattdessen wird DB2 alle fünf Minuten versuchen, die Protokolldatei zu erstellen, bis diese Datei erfolgreich erstellt wurde. Nach jedem Versuch schreibt DB2 eine Nachricht in das Protokoll mit Benachrichtigungen für die Systemverwaltung. Das Überwachen dieses Protokolls mit Benachrichtigung für die Systemverwaltung stellt die einzige Möglichkeit dar, zu bestätigen, dass eine Anwendung blockiert ist. Solange die Protokolldatei noch nicht erstellt werden konnte, kann keine der Benutzeranwendungen, die versucht, Tabellendaten zu aktualisieren, Transaktionen mit COMMIT festschreiben. Auf Lesezugriff beschränkte Abfragen sind u. U. nicht direkt betroffen. Wenn jedoch eine Abfrage auf Daten zugreifen muss, die aufgrund einer Aktualisierungsanforderung gesperrt sind, oder auf eine Datenseite, die im Pufferpool von der aktualisierenden Anwendung korrigiert wird, erscheinen auch auf Lesezugriff beschränkte Abfragen blockiert.

Wenn *blk_log_dsk_ful* auf *yes* gesetzt wird, werden Anwendungen blockiert, sobald DB2 einen Fehler aufgrund erschöpfter Festplattenkapazität feststellt. Dadurch können Sie den Fehler beheben und die Transaktion anschließend beenden. Bei erschöpfter Festplattenkapazität können Sie beispielsweise alte Protokolldateien auf ein anderes Dateisystem versetzen oder das Dateisystem vergrößern, so dass blockierte Anwendungen beendet werden können.

Wenn *blk_log_dsk_ful* auf *no* gesetzt ist und eine Transaktion einen Fehler aufgrund erschöpfter Festplattenkapazität feststellt, schlägt die Transaktion fehl und wird mit ROLLBACK rückgängig gemacht. In einigen Situationen kann die Datenbank abstürzen, wenn eine Transaktion einen Fehler aufgrund erschöpfter Festplattenkapazität verursacht.

Zugehörige Referenzen:

- „GET DATABASE CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE CONFIGURATION command“ in *Command Reference*

failarchpath - Protokollarchivpfad für Funktionsübernahme

Konfigurationstyp	Datenbank
Gilt für	• Datenbankserver mit lokalen und fernen Clients • Client • Datenbankserver mit lokalen Clients • Partitionierten Datenbankserver mit lokalen und fernen Clients
Parametertyp	Online konfigurierbar
Standardwert [Bereich]	Null []

Mit diesem Parameter wird ein Pfad angegeben, in dem DB2 versucht, Protokolldateien zu archivieren, wenn die Protokolldateien weder am primären noch am sekundären (falls definiert) Archivierungsziel archiviert werden können, weil diese Ziele von einem Datenträgerproblem betroffen sind. Der angegebene Pfad muss auf eine Platte verweisen.

Wenn sich Protokolldateien in dem durch den aktuellen Wert von *failarchpath* angegebenen Pfad befinden, werden sämtliche Aktualisierungen für *failarchpath* nicht sofort wirksam. Die Aktualisierung wird dann wirksam, wenn die Verbindung zu allen Anwendungen getrennt ist.

Zugehörige Referenzen:

- „GET DATABASE CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE CONFIGURATION command“ in *Command Reference*

logarchmeth1 - Primäre Protokollarchivierungsmethode

Konfigurationstyp	Datenbank
Gilt für	• Datenbankserver mit lokalen und fernen Clients • Client • Datenbankserver mit lokalen Clients • Partitionierten Datenbankserver mit lokalen und fernen Clients
Parametertyp	Online konfigurierbar
Standardwert [Bereich]	Off []

Mit diesem Parameter wird der Datenträgertyp des primären Ziels für archivierte Protokolle angegeben.

Zugehörige Konzepte:

- „Knotenübergreifende Fehlerbehebung mit dem Befehl 'db2adutl' und den Datenbankkonfigurationsparametern 'logarchopt1' und 'vendoropt'“ in *Daten-recovery und hohe Verfügbarkeit Handbuch und Referenz*

Zugehörige Referenzen:

- „GET DATABASE CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE CONFIGURATION command“ in *Command Reference*

logarchmeth2 - Sekundäre Protokollarchivierungsmethode**Konfigurationstyp**

Datenbank

Gilt für

- Datenbankserver mit lokalen und fernen Clients
- Client
- Datenbankserver mit lokalen Clients
- Partitionierten Datenbankserver mit lokalen und fernen Clients

Parametertyp

Online konfigurierbar

Standardwert [Bereich]

Off []

Mit diesem Parameter wird der Datenträgertyp des sekundären Ziels für archivierte Protokolle angegeben. Wenn dieser Pfad angegeben wird, werden Protokoll-dateien sowohl an diesem Ziel als auch an dem Ziel archiviert, das durch den Datenbankkonfigurationsparameter *logarchmeth1* angegeben wird.

Zugehörige Referenzen:

- „GET DATABASE CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE CONFIGURATION command“ in *Command Reference*

logarchopt1 - Optionen für primäres Protokollarchiv**Konfigurationstyp**

Datenbank

Gilt für

- Datenbankserver mit lokalen und fernen Clients
- Client
- Datenbankserver mit lokalen Clients
- Partitionierten Datenbankserver mit lokalen und fernen Clients

Parametertyp

Online konfigurierbar

Standardwert [Bereich]

Null []

Mit diesem Parameter wird das Feld der Optionen für das primäre Ziel für archivierte Protokolle (falls erforderlich) angegeben.

Zugehörige Referenzen:

- „GET DATABASE CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE CONFIGURATION command“ in *Command Reference*

logarchopt2 - Optionen für sekundäres Protokollarchiv

Konfigurationstyp	Datenbank
Gilt für	• Datenbankserver mit lokalen und fernen Clients • Client • Datenbankserver mit lokalen Clients • Partitionierten Datenbankserver mit lokalen und fernen Clients
Parametertyp	Online konfigurierbar
Standardwert [Bereich]	Null []

Mit diesem Parameter wird das Feld der Optionen für das sekundäre Ziel für archivierte Protokolle (falls erforderlich) angegeben.

Zugehörige Referenzen:

- „GET DATABASE CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE CONFIGURATION command“ in *Command Reference*

logindexbuild - Erstellte Indexseiten protokollieren

Konfigurationstyp	Datenbank
Gilt für	• Datenbankserver mit lokalen und fernen Clients • Client • Datenbankserver mit lokalen Clients • Partitionierten Datenbankserver mit lokalen und fernen Clients
Parametertyp	Online konfigurierbar
Standardwert [Bereich]	Off [On; Off]

Mit diesem Parameter wird angegeben, ob Operationen zur Erstellung, erneuten Erstellung oder Reorganisation von Indizes protokolliert werden sollen, sodass Indizes bei DB2-Operationen zur aktualisierenden Recovery oder bei Prozeduren zum Nachvollziehen von HADR-Protokollen (High Availability Disaster Recovery) wiederhergestellt werden können.

Zugehörige Referenzen:

- „GET DATABASE CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE CONFIGURATION command“ in *Command Reference*

logretain - Beibehalten von Protokollen aktivieren

Konfigurationstyp	Datenbank
Parametertyp	Konfigurierbar
Standardwert [Bereich]	No [Recovery; No]

Es gibt folgende Werte:

- "No" gibt an, dass die Protokolle nicht beibehalten werden.
- "Recovery" gibt an, dass die Protokolle beibehalten werden und zur aktualisierenden Recovery verwendet werden können.

Ist *logretain* auf "Recovery" oder *userexit* auf "Yes" gesetzt, werden die aktiven Protokolldateien gespeichert und als Online-Archivprotokolldateien in einer aktualisierenden Recovery eingesetzt. Dies wird Protokollierung mit Protokollspeicherung genannt.

Nach dem Setzen von *logretain* auf "Recovery" und/oder von *userexit* auf "Yes" müssen Sie ein Gesamtbackup der Datenbank vornehmen. Dieser Status wird durch den Markierungsparameter *backup_pending* angezeigt.

Ist *logretain* auf "No" und *userexit* auf "No" gesetzt, ist die aktualisierende Recovery für die Datenbank nicht verfügbar, da die Protokolle nicht beibehalten werden. In diesem Fall löscht der Datenbankmanager alle Protokolldateien im Verzeichnis *log-path* (einschließlich der Online-Archivprotokolldateien), ordnet er neue aktive Protokolldateien zu und aktiviert er erneut Umlaufprotokollierung.

Zugehörige Referenzen:

- „userexit - Benutzerexit aktivieren “ auf Seite 516
- „log_retain_status - Statusanzeiger für Beibehalten der Protokolle “ auf Seite 539
- „backup_pending - Backup anstehend “ auf Seite 538
- „GET DATABASE CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE CONFIGURATION command“ in *Command Reference*

mincommit - Anzahl der Gruppencommits

Konfigurationstyp	Datenbank
Parametertyp	Online konfigurierbar
Weitergabeklasse	Sofort
Standardwert [Bereich]	1 [1 – 25]
Maßeinheit	Zähler

Mit diesem Parameter können Sie das Schreiben von Protokollsätzen auf die Festplatte verzögern, bis eine Mindestanzahl von Commits ausgeführt wurde. Diese Verzögerung kann dazu beitragen, den Aufwand des Datenbankmanagers für das Schreiben von Protokollsätzen zu reduzieren. Dies führt zu einer Leistungssteigerung, wenn mehrere Anwendungen für eine Datenbank ausgeführt und von den Anwendungen viele Commits innerhalb kurzer Zeit angefordert werden.

Diese Gruppierung von Commits wird nur dann ausgeführt, wenn der Wert dieses Parameters größer als eins und die Anzahl der Anwendungen, die mit der Datenbank verbunden sind, größer oder gleich dem Wert dieses Parameters ist. Wenn die Gruppierung von Commits ausgeführt wird, können Commitanforderungen von Anwendungen so lange zurückgehalten werden, bis entweder eine Sekunde vergangen oder die Anzahl der Commitanforderungen gleich dem Wert dieses Parameters ist.

Dieser Parameter sollte nur in kleinen Schritten, zum Beispiel um den Wert 1, erhöht werden. Darüber hinaus sollten Sie mit Hilfe von Tests mit mehreren Benutzern sicherstellen, dass die Erhöhung des Werts dieses Parameters die erwarteten Ergebnisse liefert.

Am Wert dieses Parameters vorgenommene Änderungen werden sofort wirksam. Sie brauchen nicht abzuwarten, bis alle Anwendungen von der Datenbank getrennt wurden.

Empfehlung: Erhöhen Sie den Wert dieses Parameters über seinen Standardwert hinaus, wenn mehrere Schreib-/Leseanwendungen regelmäßig gleichzeitige Datenbankcommits anfordern. Dadurch wird eine höhere Effizienz bei E/A-Operationen für Protokolldateien erzielt, da diese Operationen weniger häufig auftreten und bei jeder Operation mehr Protokollsätze geschrieben werden.

Sie könnten auch die Anzahl der Transaktionen pro Sekunde ermitteln und diesen Parameter so anpassen, dass die Höchstanzahl von Transaktionen pro Sekunde (oder ein großer Prozentsatz davon) von diesem Parameter berücksichtigt wird. Durch die Berücksichtigung der Spitzenaktivitäten würde der Systemaufwand für das Schreiben von Protokollsätzen in Phasen mit hohem Transaktionsaufkommen minimiert.

Wenn der Wert des Parameters *mincommit* erhöht wird, kann es zudem notwendig werden, den Wert des Parameters *logbufsz* zu erhöhen, um zu vermeiden, dass ein voller Protokollpuffer eine Schreiboperation während dieser Phasen mit hohem Transaktionsaufkommen erzwingt. In diesem Fall sollte der Wert für den Parameter *logbufsz* nach folgender Formel festgelegt werden:

$\text{mincommit} * (\text{durchschnittlicher Protokollbereich für eine Transaktion})$

Sie können den Datenbanksystemmonitor folgendermaßen zur Optimierung des Werts dieses Parameters verwenden:

- Berechnen der Höchstanzahl von Transaktionen pro Sekunde:

Durch das Erstellen von Monitorstichproben über einen typischen Arbeitstag hinweg können Sie die Phasen mit hohem Transaktionsaufkommen ermitteln. Sie können die Gesamtanzahl der Transaktionen durch Addieren der Werte für folgende Monitorelemente berechnen:

- *commit_sql_stmts* (versuchte COMMIT-Anweisungen)
- *rollback_sql_stmts* (versuchte ROLLBACK-Anweisungen)

Anhand dieser Stichproben und der verfügbaren Zeitmarken können Sie die Anzahl der Transaktionen pro Sekunde berechnen.

- Berechnen des pro Transaktion verwendeten Protokollspeicherbereichs:

Mit Hilfe von Stichproben über einen gewissen Zeitraum hinweg und für eine Anzahl von Transaktionen können Sie den durchschnittlich verwendeten Protokollspeicherbereich mit dem folgenden Monitorelement berechnen:

- *log_space_used* (Protokollbereich für UOW)

Zugehörige Referenzen:

- „commit_sql_stmts - Commit Statements Attempted monitor element“ in *Systemmonitor Handbuch und Referenz*
- „rollback_sql_stmts - Rollback Statements Attempted monitor element“ in *Systemmonitor Handbuch und Referenz*
- „uow_log_space_used - Unit of Work Log Space Used monitor element“ in *Systemmonitor Handbuch und Referenz*

- „GET DATABASE CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE CONFIGURATION command“ in *Command Reference*

numarchretry - Anzahl der Wiederholungen bei Fehler

Konfigurationstyp

Datenbank

Gilt für

- Datenbankserver mit lokalen und fernen Clients
- Client
- Datenbankserver mit lokalen Clients
- Partitionierten Datenbankserver mit lokalen und fernen Clients

Parametertyp

Online konfigurierbar

Standardwert [Bereich]

5 [0 – 65 535]

Mit diesem Parameter wird die Anzahl der Versuche angegeben, die DB2 zur Archivierung einer Protokolldatei im primären oder sekundären Archivverzeichnis unternehmen soll, bevor es versucht, Protokolldateien im Verzeichnis für die Funktionsübernahme zu speichern. Dieser Parameter wird nur verwendet, wenn der Datenbankkonfigurationsparameter *failarchpath* definiert ist. Wenn *numarchretry* nicht definiert wird, setzt DB2 die Versuche zur Archivierung im primären oder sekundären Protokollpfad kontinuierlich fort.

Zugehörige Referenzen:

- „GET DATABASE CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE CONFIGURATION command“ in *Command Reference*

softmax - Vor bedingtem Prüfpunkt zu schreibende Protokollsätze

Konfigurationstyp

Datenbank

Parametertyp

Konfigurierbar

Standardwert [Bereich]

100 [1 – 100 * *logprimary*]

Maßeinheit

Prozentsatz der Größe einer primären Protokolldatei

Dieser Parameter hat folgende Funktionen:

- Beeinflussen der Anzahl von Protokolldateien, die für die Recovery nach einem Systemabsturz (z. B. nach einem Stromausfall) erforderlich sind. Wenn beispielsweise der Standardwert verwendet wird, versucht der Datenbankmanager, die Anzahl der wiederherzustellenden Protokolldateien auf 1 zu halten. Wenn Sie 300 als Wert für diesen Parameter angeben, versucht der Datenbankmanager, die Anzahl der wiederherzustellenden Protokolldateien auf 3 zu halten.

Beim Steuern der Anzahl der Protokolldateien, die für die Recovery nach einem Systemabsturz erforderlich sind, verwendet der Datenbankmanager diesen Parameter zum Starten der Seitenlöschfunktionen. Dadurch wird sichergestellt, dass Seiten, die älter sind als das angegebene Recoveryfenster, bereits auf Platte geschrieben sind.

- Festlegen der Frequenz der bedingten Prüfpunkte.

Zum Zeitpunkt einer Datenbankstörung, die zum Beispiel durch einen Stromausfall verursacht wird, kann für in der Datenbank ausgeführte Änderungen Folgendes gelten:

- Die Änderungen wurden nicht mit COMMIT festgeschrieben, jedoch wurden die Daten im Pufferpool aktualisiert.
- Die Änderungen wurden mit COMMIT festgeschrieben, jedoch noch nicht vom Pufferpool auf die Festplatte geschrieben.
- Die Änderungen wurden mit COMMIT festgeschrieben und vom Pufferpool auf die Festplatte geschrieben.

Wenn eine Datenbank erneut gestartet wird, werden die Protokolldateien verwendet, um eine Recovery der Datenbank nach dem Systemabsturz auszuführen, die sicherstellt, dass die Datenbank in einem konsistenten Zustand verbleibt (d. h., alle mit COMMIT festgeschriebenen Transaktionen werden in der Datenbank nachvollzogen, und keine der nicht festgeschriebenen Transaktionen werden in der Datenbank nachvollzogen).

Der Datenbankmanager verwendet eine Protokollsteuerdatei, um festzustellen, welche Datensätze aus der Protokolldatei in der Datenbank nachvollzogen werden müssen. Diese Protokollsteuerdatei wird in regelmäßigen Abständen auf die Festplatte geschrieben, und der Datenbankmanager kann abhängig von der Frequenz dieses Ereignisses Protokollsätze festgeschriebener Transaktionen oder Protokollsätze zu Änderungen, die bereits aus dem Pufferpool auf Platte geschrieben wurden, nachvollziehen. Diese Protokollsätze haben keine Auswirkung auf die Datenbank, das Nachvollziehen dieser Protokollsätze führt jedoch zu einem gewissen erhöhten Systemaufwand während des Neustarts der Datenbank.

Die Protokollsteuerdatei wird immer dann auf die Festplatte geschrieben, wenn eine Protokolldatei voll ist, und außerdem bei bedingten Prüfpunkten. Sie können diesen Konfigurationsparameter dazu verwenden, zusätzliche bedingte Prüfpunkte auszulösen.

Die Ablaufsteuerung für bedingte Prüfpunkte wird mit Hilfe der Differenz zwischen dem „aktuellen Stand“ und dem „aufgezeichneten Stand“ festgelegt. Diese Differenz wird als Prozentsatz vom Wert des Parameters *logfilesiz* angegeben. Der „aufgezeichnete Stand“ wird anhand des ältesten gültigen Protokollsatzes ermittelt, der von der Protokollsteuerdatei auf der Festplatte angegeben wird, während der „aktuelle Stand“ anhand der Protokollsteuerinformationen im Hauptspeicher ermittelt wird. (Der älteste gültige Protokollsatz ist der erste Protokollsatz, der bei einem Recoveryprozess gelesen wurde.) Der bedingte Prüfpunkt wird ausgelöst, wenn der nach der folgenden Formel berechnete Wert größer oder gleich dem Wert dieses Parameters ist:

$$(\text{Bereich zw. aufgezeichnetem u. aktuellem Stand}) / \text{logfilesiz}) * 100$$

Empfehlung: Sie können den Wert dieses Parameters erhöhen oder verringern, je nachdem, ob Ihr akzeptables Recoveryfenster größer oder kleiner als eine Protokolldatei ist. Wenn Sie den Wert dieses Parameters herabsetzen, wird der Datenbankmanager veranlasst, die Seitenlöschfunktionen häufiger auszulösen und häufiger bedingte Prüfpunkte anzusetzen. Diese Maßnahmen können die Anzahl der Protokollsätze, die verarbeitet werden müssen, und die Anzahl der überschüssigen Protokollsätze, die während der Recovery verarbeitet werden, verringern.

Beachten Sie jedoch, dass mehr Trigger von Seitenlöschfunktionen und häufigere bedingte Prüfpunkte den Systemaufwand erhöhen, der mit der Protokollierung der Datenbank verbunden ist, was sich negativ auf die Leistung des Datenbank-

managers auswirken kann. Daneben können auch folgende Umstände dazu führen, dass häufigere bedingte Prüfpunkte die für den Neustart einer Datenbank benötigte Zeit nicht verkürzen:

- Es werden sehr lange Transaktionen mit wenigen Commitpunkten ausgeführt.
- Der Pufferpool ist sehr groß, und die Seiten mit den festgeschriebenen Transaktionen werden nicht sehr oft auf die Platte zurückgeschrieben. (Beachten Sie, dass durch das Verwenden asynchroner Seitenlöschfunktionen solche Situationen vermieden werden können.)

In beiden Fällen ändern sich die Protokollsteuerdaten im Hauptspeicher nur selten, und es ist nur dann sinnvoll, die Protokollsteuerdaten auf die Festplatte zu schreiben, wenn sie sich geändert haben.

Zugehörige Referenzen:

- „logfilesiz - Protokolldateigröße “ auf Seite 497
- „logprimary - Anzahl primärer Protokolldateien “ auf Seite 499
- „GET DATABASE CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE CONFIGURATION command“ in *Command Reference*

userexit - Benutzerexit aktivieren

Konfigurationstyp	Datenbank
Parametertyp	Konfigurierbar
Standardwert [Bereich]	Off [On; Off]

Wenn dieser Parameter aktiviert ist, wird unabhängig von der Einstellung des Parameters *logretain* eine Protokollierung mit Protokollspeicherung ausgeführt. Mit diesem Parameter wird außerdem angegeben, dass ein Benutzerexitprogramm verwendet werden soll, um Protokolldateien zu archivieren und wieder abzurufen. Protokolldateien werden archiviert, wenn die Protokolldatei voll ist. Die Protokolldateien werden wieder abgerufen, wenn das Dienstprogramm ROLLFORWARD sie für den Restore einer Datenbank benötigt.

Nach dem Aktivieren des Parameters *logretain* und/oder *userexit* muss ein Gesamtbackup der Datenbank ausgeführt werden. Dieser Status wird durch den Markierungsparameter *backup_pending* angezeigt.

Wenn beide Parameter inaktiviert werden, ist die aktualisierende Recovery der Datenbank nicht mehr möglich, da keine Protokolldateien mehr gespeichert werden. In diesem Fall löscht der Datenbankmanager alle Protokolldateien im Verzeichnis *logpath* (einschließlich der Online-Archivprotokolldateien), ordnet neue aktive Protokolldateien zu und reaktiviert die Umlaufprotokollierung.

Zugehörige Referenzen:

- „ROLLFORWARD DATABASE command“ in *Command Reference*
- „Benutzerexit zur Datenbankrecovery“ in *Datenrecovery und hohe Verfügbarkeit Handbuch und Referenz*
- „logretain - Beibehalten von Protokollen aktivieren “ auf Seite 511
- „user_exit_status - Statusanzeiger für Benutzerexit “ auf Seite 540
- „backup_pending - Backup anstehend “ auf Seite 538
- „GET DATABASE CONFIGURATION command“ in *Command Reference*

- „RESET DATABASE CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE CONFIGURATION command“ in *Command Reference*

vendoropt - Lieferantenooptionen

Konfigurationstyp

Datenbank

Gilt für

- Datenbankserver mit lokalen und fernen Clients
- Client
- Datenbankserver mit lokalen Clients
- Partitionierten Datenbankserver mit lokalen und fernen Clients

Parametertyp

Online konfigurierbar

Standardwert [Bereich]

Null []

Mit diesem Parameter werden zusätzliche Parameter angegeben, die DB2 möglicherweise zur Kommunikation mit Speichersystemen bei Backup-, Restore- oder Ladekopieoperationen benötigt.

Zugehörige Konzepte:

- „Knotenübergreifende Fehlerbehebung mit dem Befehl 'db2adutl' und den Datenbankkonfigurationsparametern 'logarchopt1' und 'vendoropt'“ in *Datenrecovery und hohe Verfügbarkeit Handbuch und Referenz*

Zugehörige Referenzen:

- „GET DATABASE CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE CONFIGURATION command“ in *Command Reference*

Recovery

Die folgenden Parameter wirken sich auf die verschiedenen Aspekte der Datenbankrecovery aus:

- „autorestart - Automatischer Neustart aktiviert “ auf Seite 518
- „dft_loadrec_ses - Standardanzahl von Sitzungen für Recovery “ auf Seite 518
- „indexrec - Zeitpunkt für Indexneuerstellung “ auf Seite 523
- „num_db_backups - Anzahl der Datenbank-Backups “ auf Seite 526
- „rec_his_retentn - Aufbewahrungszeitraum für Recoveryprotokoll “ auf Seite 526
- „trackmod - Geänderte Seiten protokollieren “ auf Seite 527

Siehe auch „DUOW-Recovery“ auf Seite 529.

Die folgenden Parameter werden beim Einsatz von Tivoli Storage Manager (TSM) verwendet:

- „tsm_mgmtclass - Tivoli Storage Manager-Verwaltungs-klasse “ auf Seite 527
- „tsm_nodename - TSM-Knotenname “ auf Seite 528
- „tsm_owner - TSM-Eignername “ auf Seite 528
- „tsm_password - TSM-Kennwort “ auf Seite 529

Die folgenden Parameter beziehen sich auf HADR (High Availability Disaster Recovery):

- „hadr_db_role - Rolle der HADR-Datenbank “ auf Seite 519
- „hadr_local_host - Name des lokalen Hosts für HADR “ auf Seite 520
- „hadr_local_svc - Lokaler HADR-Servicename “ auf Seite 520
- „hadr_remote_host - Name des fernen Hosts für HADR “ auf Seite 521
- „hadr_remote_inst - HADR-Instanzname des fernen Servers “ auf Seite 521
- „hadr_remote_svc - Ferner HADR-Servicename “ auf Seite 522
- „hadr_syncmode - HADR-Synchronisationsmodus für Protokollierung im Peer-status “ auf Seite 522
- „hadr_timeout - HADR-Zeitlimitwert “ auf Seite 523

autorestart - Automatischer Neustart aktiviert

Konfigurationstyp	Datenbank
Parametertyp	Online konfigurierbar
Weitergabeklasse	Sofort
Standardwert [Bereich]	On [On; Off]

Wenn dieser Parameter auf ON gesetzt wird, ruft der Datenbankmanager bei Bedarf automatisch das Dienstprogramm zum Neustarten der Datenbank auf, wenn eine Anwendung die Verbindung zur Datenbank herstellt. Das Programm zum Neustarten der Datenbank wird verwendet, um eine *Recovery nach Systemabsturz* auszuführen. Es wird ausgeführt, wenn die Datenbank abnormal beendet wurde, während Anwendungen mit ihr verbunden waren. Eine abnormale Beendigung der Datenbank könnte z. B. durch einen Stromausfall oder einen Fehler der Systemsoftware verursacht werden. Bei der Recovery werden festgeschriebene Transaktionen, die sich im Pufferpool befanden, jedoch zum Zeitpunkt des Fehlers nicht auf die Festplatte geschrieben waren, in der Datenbank nachvollzogen. Außerdem werden alle nicht festgeschriebenen Transaktionen, die eventuell auf Platte geschrieben wurden, wieder zurückgesetzt.

Wenn der Parameter *autorestart* nicht aktiviert ist, empfängt eine Anwendung den Fehler SQL1015N, wenn sie versucht, die Verbindung zu einer Datenbank herzustellen, für die eine Recovery nach einem Systemabsturz ausgeführt werden muss (für die die Datenbank neu gestartet werden muss). In diesem Fall kann die Anwendung das Dienstprogramm zum Neustarten der Datenbank aufrufen, oder Sie können die Datenbank neu starten, indem Sie die Funktion zum Neustarten im Recoveryprogramm auswählen.

Zugehörige Konzepte:

- „Recovery nach einem Systemabsturz“ in *Datenrecovery und hohe Verfügbarkeit Handbuch und Referenz*

Zugehörige Referenzen:

- „RESTART DATABASE command“ in *Command Reference*
- „GET DATABASE CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE CONFIGURATION command“ in *Command Reference*

dft_loadrec_ses - Standardanzahl von Sitzungen für Recovery

Konfigurationstyp	Datenbank
Parametertyp	Online konfigurierbar

Weitergabeklasse	Sofort
Standardwert [Bereich]	1 [1 – 30 000]
Maßeinheit	Zähler

Mit diesem Parameter wird die Standardanzahl von Sitzungen angegeben, die während des Wiederabrufens einer Tabellenladekopie verwendet werden. Der Wert sollte auf eine optimale Anzahl von E/A-Sitzungen gesetzt werden, die zum Wiederabrufen einer Ladekopie verwendet werden sollen. Das Wiederabrufen einer Ladekopie ist eine ähnliche Operation wie ein Restore. Sie können diesen Parameter durch Einträge überschreiben, die sich in der Datei mit den Angaben zur Speicherposition der exportierten Daten befinden. Diese Datei wird von der Umgebungsvariablen DB2LOADREC angegeben.

Die Standardanzahl der Puffer, die für den Abruf der Ladekopien verwendet werden, ist um zwei größer als der Wert dieses Parameters. Die Anzahl der Puffer kann ebenfalls in der Datei mit den Angaben zur Speicherposition der exportierten Daten überschrieben werden.

Dieser Parameter ist nur gültig, wenn die aktualisierende Recovery aktiviert ist.

Zugehörige Konzepte:

- „LOAD - Übersicht“ in *Dienstprogramme für das Versetzen von Daten Handbuch und Referenz*

Zugehörige Referenzen:

- „GET DATABASE CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE CONFIGURATION command“ in *Command Reference*

hadr_db_role - Rolle der HADR-Datenbank

Konfigurationstyp	Datenbank

Gilt für

- Datenbankserver mit lokalen und fernen Clients
- Client
- Datenbankserver mit lokalen Clients
- Partitionierten Datenbankserver mit lokalen und fernen Clients

Parametertyp	Informativ

Dieser Parameter gibt die aktuelle Rolle einer Datenbank an, unabhängig davon, ob sie online oder offline ist. Gültige Werte: STANDARD, PRIMARY oder STANDBY.

Anmerkung: Der Befehl GET SNAPSHOT FOR DATABASE liefert zwar den HADR-Status (High Availability Disaster Recovery), jedoch nur, wenn die Datenbank online ist.

Zugehörige Referenzen:

- „GET DATABASE CONFIGURATION command“ in *Command Reference*

hadr_local_host - Name des lokalen Hosts für HADR

Konfigurationstyp

Datenbank

Gilt für

- Datenbankserver mit lokalen und fernen Clients
- Client
- Datenbankserver mit lokalen Clients
- Partitionierten Datenbankserver mit lokalen und fernen Clients

Parametertyp

Konfigurierbar

Standardwert

Null

Mit diesem Parameter wird der lokale Host für die HADR-TCP-Kommunikation (High Availability Disaster Recovery) angegeben. Es kann entweder ein Hostname oder eine IP-Adresse verwendet werden. Wenn ein Hostname angegeben wird und mehreren IP-Adressen zugeordnet ist, wird ein Fehler zurückgegeben und HADR wird nicht gestartet. Wenn der Hostname mehreren IP-Adressen zugeordnet ist (selbst wenn Sie den gleichen Hostnamen auf dem Primärsystem und dem Bereitschaftssystem angeben), besteht die Möglichkeit, dass das Primärsystem und das Bereitschaftssystem diesen Hostnamen verschiedenen IP-Adressen zuordnen, weil einige DNS-Server IP-Adresslisten in nicht deterministischer Reihenfolge zurückgeben.

Ein Hostname besitzt das Format: meinserver.ibm.com. Eine IP-Adresse besitzt das Format: "12.34.56.78".

Zugehörige Referenzen:

- „GET DATABASE CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE CONFIGURATION command“ in *Command Reference*

hadr_local_svc - Lokaler HADR-Servicename

Konfigurationstyp

Datenbank

Gilt für

- Datenbankserver mit lokalen und fernen Clients
- Client
- Datenbankserver mit lokalen Clients
- Partitionierten Datenbankserver mit lokalen und fernen Clients

Parametertyp

Konfigurierbar

Standardwert

Null

Mit diesem Parameter wird der TCP-Servicename oder die Portnummer angegeben, für die der HADR-Prozess (High Availability Disaster Recovery) Verbindungen akzeptiert.

Der Wert für *hadr_local_svc* auf dem primären Datenbanksystem oder dem Bereitschaftsdatenbanksystem darf nicht mit dem Wert für *svcname* bzw. mit *svcname +1* auf ihren jeweiligen Knoten übereinstimmen.

Zugehörige Referenzen:

- „GET DATABASE CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE CONFIGURATION command“ in *Command Reference*

hadr_remote_host - Name des fernen Hosts für HADR

Konfigurationstyp Datenbank

Gilt für

- Datenbankserver mit lokalen und fernen Clients
- Client
- Datenbankserver mit lokalen Clients
- Partitionierten Datenbankserver mit lokalen und fernen Clients

Parametertyp Konfigurierbar

Standardwert Null

Mit diesem Parameter wird der TCP/IP-Hostname oder die IP-Adresse des fernen HADR-Knotens (High Availability Disaster Recovery) angegeben. Ähnlich wie der Parameter *hadr_local_host* darf dieser Parameter nur einer IP-Adresse zugeordnet werden.

Zugehörige Referenzen:

- „GET DATABASE CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE CONFIGURATION command“ in *Command Reference*

hadr_remote_inst - HADR-Instanzname des fernen Servers

Konfigurationstyp Datenbank

Gilt für

- Datenbankserver mit lokalen und fernen Clients
- Client
- Datenbankserver mit lokalen Clients
- Partitionierten Datenbankserver mit lokalen und fernen Clients

Parametertyp Konfigurierbar

Standardwert Null

Mit diesem Parameter wird der Instanzname des fernen Servers angegeben. Verwaltungstools wie die DB2-Steuerzentrale verwenden diesen Parameter, um eine Verbindung zu dem fernen Server herzustellen. Ebenso prüft HADR (High Availability Disaster Recovery), ob eine ferne Datenbank, die eine Verbindung anfordert, zu der deklarierten fernen Instanz gehört.

Zugehörige Referenzen:

- „GET DATABASE CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE CONFIGURATION command“ in *Command Reference*

hadr_remote_svc - Ferner HADR-Servicename

Konfigurationstyp

Datenbank

Gilt für

- Datenbankserver mit lokalen und fernen Clients
- Client
- Datenbankserver mit lokalen Clients
- Partitionierten Datenbankserver mit lokalen und fernen Clients

Parametertyp

Konfigurierbar

Standardwert

Null

Mit diesem Parameter wird der TCP-Servicename oder die Portnummer angegeben, die vom fernen HADR-Knoten (High Availability Disaster Recovery) verwendet wird.

Zugehörige Referenzen:

- „GET DATABASE CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE CONFIGURATION command“ in *Command Reference*

hadr_syncmode - HADR-Synchronisationsmodus für Protokollierung im Peerstatus

Konfigurationstyp

Datenbank

Gilt für

- Datenbankserver mit lokalen und fernen Clients
- Client
- Datenbankserver mit lokalen Clients
- Partitionierten Datenbankserver mit lokalen und fernen Clients

Parametertyp

Konfigurierbar

Standardwert [Bereich]

NEARSYNC [ASYNC; SYNC]

Mit diesem Parameter wird der Synchronisationsmodus angegeben, der bestimmt, wie die Protokollschreiboperationen der Primärdatenbank mit der Bereitschaftsdatenbank synchronisiert werden, wenn sich die Systeme im Peerstatus befinden. Gültige Werte:

SYNC

Dieser Modus bietet den größten Schutz gegen Transaktionsverlust, jedoch auf Kosten einer längeren Transaktionsantwortzeit.

NEARSYNC

Dieser Modus bietet einen etwas geringeren Schutz gegen Transaktionsverlust, jedoch im Gegenzug auch eine kürzere Transaktionsantwortzeit als der Modus SYNC.

ASYNC

Dieser Modus hat das größte Risiko eines Transaktionsverlusts im Fall eines Ausfalls der Primärdatenbank, bietet allerdings auch die kürzeste Transaktionsantwortzeit unter den drei Modi.

Zugehörige Referenzen:

- „GET DATABASE CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE CONFIGURATION command“ in *Command Reference*

hadr_timeout - HADR-Zeitlimitwert**Konfigurationstyp**

Datenbank

Gilt für

- Datenbankserver mit lokalen und fernen Clients
- Client
- Datenbankserver mit lokalen Clients
- Partitionierten Datenbankserver mit lokalen und fernen Clients

Parametertyp

Konfigurierbar

Standardwert [Bereich]

120 [1 – 4 294 967 295]

Dieser Parameter gibt die Zeit (in Sekunden) an, die der HADR-Prozess (High Availability Disaster Recovery) wartet, bevor ein Kommunikationsversuch als fehlgeschlagen eingestuft wird.

Zugehörige Referenzen:

- „GET DATABASE CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE CONFIGURATION command“ in *Command Reference*

indexrec - Zeitpunkt für Indexneuerstellung**Konfigurationstyp**

Datenbank und Datenbankmanager

Gilt für

- Datenbankserver mit lokalen und fernen Clients
- Datenbankserver mit lokalen Clients
- Partitionierten Datenbankserver mit lokalen und fernen Clients

Parametertyp

Online konfigurierbar

Weitergabeklasse

Sofort

Standardwert [Bereich]**UNIX Datenbankmanager**

```
restart [ restart; restart_no_redo;
access; access_no_redo ]
```

Windows- Datenbankmanager

```
restart [ restart; restart_no_redo;
access; access_no_redo ]
```

Datenbank

```
Systemeinstellung verwenden: [
system; restart; restart_no_redo;
access; access_no_redo ]
```

Dieser Parameter gibt an, wann der Datenbankmanager versucht, ungültige Indizes neu zu erstellen und ob eine Indexerstellung während einer aktualisierenden DB2-

Recovery oder einer Wiedergabe des HADR-Protokolls in der Bereitschaftsdatenbank wiederholt wird. Für diesen Parameter sind fünf Einstellungen möglich:

SYSTEM *Verwenden Sie die Systemeinstellung, die in der Konfigurationsdatei des Datenbankmanagers angegeben wurde, um festzulegen, wann ungültige Indizes neu erstellt werden und ob Protokollsätze zur Indexerstellung während einer aktualisierenden DB2-Recovery oder während einer Wiedergabe des HADR-Protokolls wiederholt werden müssen. (Anmerkung: Diese Einstellung ist nur für Datenbankkonfigurationen gültig.)*

ACCESS Ungültige Indizes werden neu erstellt, wenn zum ersten Mal auf den Index zugegriffen wird. Alle vollständig protokollierten Indexerstellung werden während einer aktualisierenden DB2-Recovery oder während einer Wiedergabe des HADR-Protokolls wiederholt. Wenn HADR gestartet wird und eine HADR-Übernahme stattfindet, werden alle ungültigen Indizes nach der Übernahme, wenn zum ersten Mal auf die zu Grunde liegende Tabelle zugegriffen wird, neu erstellt.

ACCESS_NO_REDO

Ungültige Indizes werden erneut erstellt, wenn zum ersten Mal auf die zu Grunde liegende Tabelle zugegriffen wird. Keine der vollständig protokollierten Indexerstellung wird während einer aktualisierenden DB2-Recovery oder während einer Wiedergabe des HADR-Protokolls wiederholt; diese Indizes bleiben ungültig. Wenn HADR gestartet wird und eine HADR-Übernahme stattfindet, werden alle ungültigen Indizes nach der Übernahme, wenn zum ersten Mal auf die zu Grunde liegende Tabelle zugegriffen wird, neu erstellt.

RESTART Die Standardeinstellung für *indexrec*. Ungültige Indizes werden neu erstellt, wenn der Befehl RESTART DATABASE explizit oder implizit abgesetzt wird. Alle vollständig protokollierten Indexerstellung werden während einer aktualisierenden DB2-Recovery oder während einer Wiedergabe des HADR-Protokolls wiederholt. Wenn HADR gestartet wird und eine HADR-Übernahme stattfindet, werden alle ungültigen Indizes nach der Übernahme neu erstellt.

Beachten Sie, dass der Befehl RESTART DATABASE implizit abgesetzt wird, wenn der Parameter *autorestart* aktiviert ist.

RESTART_NO_REDO

Ungültige Indizes werden neu erstellt, wenn der Befehl RESTART DATABASE explizit oder implizit abgesetzt wird. (Der Befehl RESTART DATABASE wird implizit abgesetzt, wenn der Parameter *autorestart* aktiviert ist.) Keine der vollständig protokollierten Indexerstellung wird während einer aktualisierenden DB2-Recovery oder während einer Wiedergabe des HADR-Protokolls wiederholt; stattdessen werden diese Indizes neu erstellt, wenn die aktualisierende Recovery beendet wird oder wenn die HADR-Übernahme stattfindet.

Indizes können ungültig werden, wenn nicht behebbare Plattenfehler auftreten. Wenn dabei die Daten selbst beschädigt werden, können die Daten verloren gehen. Wird jedoch ein Index beschädigt, kann der Index durch Neuerstellung wiederhergestellt werden. Wird ein Index neu erstellt, während Benutzer mit der Datenbank verbunden sind, können zwei Probleme auftreten:

- Während der Erstellung der Indexdatei könnte es zu einer unerwarteten Verschlechterung der Antwortzeit kommen. Benutzer, die auf die Tabelle zugreifen und diesen bestimmten Index verwenden, müssen auf die Neuerstellung des Index warten.
- Nach der Indexneuerstellung könnten unerwartete, aktive Sperren beibehalten werden, besonders dann, wenn die Benutzertransaktion, die die Indexneuerstellung ausgelöst hat, keine Commit- oder Rollbackoperation ausgeführt hat.

Empfehlung: Die beste Option für diesen Parameter auf einem Server mit vielen Benutzern, wenn die Zeit für den Neustart keine kritische Rolle spielt, ist die Indexneuerstellung beim Neustart der Datenbank (DATABASE RESTART) als Teil des Vorgangs, mit dem die Datenbank nach einem Systemabsturz wiederhergestellt und online verfügbar gemacht wird.

Wenn dieser Parameter auf „ACCESS“ oder auf „ACCESS_NO_REDO“ gesetzt wird, verschlechtert sich während der Indexneuerstellung die Leistung des Datenbankmanagers. Jeder Benutzer, der auf den Index oder die Tabelle zugreift, der bzw. die gerade neu erstellt wird, muss zunächst auf die Beendigung der Indexneuerstellung warten.

Wenn dieser Parameter auf „RESTART“ gesetzt wird, dauert der Neustart der Datenbank wegen der Indexneuerstellung länger, jedoch wird die normale Verarbeitung nicht mehr beeinträchtigt, sobald die Datenbank wieder online verfügbar ist.

Anmerkung: Bei der Datenbankrecovery werden alle ausführbaren Dateien von SQL-Prozeduren im Dateisystem entfernt, die zu der gerade wiederhergestellten Datenbank gehören. Wenn der Parameter *indexrec* auf RESTART gesetzt wird, werden alle ausführbaren Dateien von SQL-Prozeduren aus dem Datenbankkatalog extrahiert und in das Dateisystem zurückgestellt, wenn die nächste Verbindung zur Datenbank hergestellt wird. Wenn der Parameter *indexrec* nicht auf RESTART gesetzt ist, wird die ausführbare Datei einer SQL-Prozedur erst in das Dateisystem extrahiert, wenn diese SQL-Prozedur zum ersten Mal ausgeführt wird.

Der Unterschied zwischen den Werten RESTART und RESTART_NO_REDO bzw. zwischen den Werten ACCESS und ACCESS_NO_REDO ist nur von Bedeutung, wenn die vollständige Protokollierung für Indexerstellungsoptionen, wie z. B. die Operationen CREATE INDEX und REORG INDEX, oder für eine Indexneuerstellung aktiviert ist. Sie können die Protokollierung aktivieren, indem Sie den Datenbankkonfigurationsparameter *logindexbuild* oder LOG INDEX BUILD beim Ändern einer Tabelle aktivieren. Wenn Sie *indexrec* entweder auf RESTART oder auf ACCESS setzen, kann für Operationen, die eine protokollierte Indexerstellung einbeziehen, eine aktualisierende Recovery durchgeführt werden, ohne dass das Indexobjekt einen ungültigen Status aufweist; dies hätte zur Folge, dass der Index später neu erstellt werden müsste.

Zugehörige Tasks:

- „Backing up and restoring SQL procedures created prior to DB2 8.2“ in *Developing SQL and External Routines*

Zugehörige Referenzen:

- „RESTART DATABASE command“ in *Command Reference*
- „GET DATABASE CONFIGURATION command“ in *Command Reference*

- „RESET DATABASE CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE CONFIGURATION command“ in *Command Reference*
- „GET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „autorestart - Automatischer Neustart aktiviert “ auf Seite 518
- „logindexbuild - Erstellte Indexseiten protokollieren “ auf Seite 511
- „ALTER TABLE statement“ in *SQL Reference, Volume 2*

num_db_backups - Anzahl der Datenbank-Backups

Konfigurationstyp	Datenbank
Parametertyp	Online konfigurierbar
Weitergabeklasse	Transaktionsgrenzwert
Standardwert [Bereich]	12 [1 — 32 767]

Dieser Parameter gibt die Anzahl der Datenbank-Backups an, die für eine Datenbank beibehalten werden sollen. Wird die angegebene Anzahl von Backups erreicht, werden alte Backups in der Datei des Recoveryprotokolls als abgelaufen markiert. Die Einträge in der Datei des Recoveryprotokolls für Tabellenbereichs-Backups und Ladekopie-Backups, die mit dem abgelaufenen Datenbank-Backup verbunden sind, werden ebenfalls als abgelaufen markiert. Wird ein Backup als abgelaufen markiert, können die physischen Backups von ihrem Speicherort (z. B. Platte, Band, TSM) entfernt werden. Das nächste Datenbank-Backup entfernt die abgelaufenen Einträge aus der Datei des Recoveryprotokolls.

Wird ein Datenbank-Backup in der Protokolldatei als abgelaufen markiert, werden entsprechende über einen DB2 Data Links Manager verbundene Datei-Backups aus dem Archivierungsserver gelöscht.

Der Konfigurationsparameter *rec_his_retentn* sollte auf einen Wert gesetzt werden, der mit dem Wert von *num_db_backups* kompatibel ist. Wenn *num_db_backup* z. B. auf einen hohen Wert gesetzt ist, muss der Wert für *rec_his_retentn* hoch genug sein, um diese Anzahl der Backups unterstützen zu können.

Zugehörige Referenzen:

- „rec_his_retentn - Aufbewahrungszeitraum für Recoveryprotokoll “ auf Seite 526
- „UPDATE DATABASE CONFIGURATION command“ in *Command Reference*
- „GET DATABASE CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE CONFIGURATION command“ in *Command Reference*

rec_his_retentn - Aufbewahrungszeitraum für Recoveryprotokoll

Konfigurationstyp	Datenbank
Parametertyp	Konfigurierbar
Standardwert [Bereich]	366 [-1; 0 — 30 000]
Maßeinheit	Tage

Mit diesem Parameter wird angegeben, wie viele Tage die Protokolldaten zu Backups aufbewahrt werden sollen. Wenn die Datei des Recoveryprotokolls nicht benötigt wird, um Backups, Restores und Ladevorgänge festzuhalten, kann dieser Parameter auf einen kleineren Wert gesetzt werden.

Wenn der Wert dieses Parameters -1 lautet, entspricht die Anzahl der Einträge, die Datenbankgesamtbackups anzeigen (sowie alle Tabellenbereichs-Backups, die dem Datenbank-Backup zugeordnet sind) dem Wert, der durch den Parameter *num_db_backups* angegeben ist. Weitere Einträge in der Datei des Recoveryprotokolls können nur explizit mit Hilfe der verfügbaren Befehle oder APIs entfernt werden.

Unabhängig davon, wie kurz der Aufbewahrungszeitraum ist, werden das aktuelle Datenbank-Backup und die zugehörige Restoregruppe immer zurückbehalten, sofern Sie nicht das Dienstprogramm PRUNE mit der Angabe WITH FORCE OPTION verwenden.

Zugehörige Referenzen:

- „PRUNE HISTORY/LOGFILE command“ in *Command Reference*
- „num_db_backups - Anzahl der Datenbank-Backups “ auf Seite 526
- „GET DATABASE CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE CONFIGURATION command“ in *Command Reference*

trackmod - Geänderte Seiten protokollieren

Konfigurationstyp	Datenbank
Parametertyp	Konfigurierbar
Standardwert [Bereich]	No [Yes, No]

Wenn dieser Parameter auf "Yes" gesetzt wird, verfolgt der Datenbankmanager Datenbankänderungen, damit das Backup-Dienstprogramm feststellen kann, welche Untergruppen der Datenbankseiten bei einem inkrementellen Backup zu berücksichtigen und eventuell in das Backup-Image aufzunehmen sind. Wenn Sie diesen Parameter auf "Yes" setzen, müssen Sie zunächst ein vollständiges Backup der Datenbank durchführen, damit Sie über eine Ausgangsbasis für inkrementelle Backups verfügen. Wenn dieser Parameter aktiviert ist und ein Tabellenbereich erstellt wird, müssen Sie außerdem ein Backup erstellen, das den betreffenden Tabellenbereich umfasst. Dabei kann es sich entweder um ein Datenbank-Backup oder ein Tabellenbereichs-Backup handeln. Nach erfolgtem Backup dürfen inkrementelle Backups diesen Tabellenbereich umfassen.

Zugehörige Referenzen:

- „GET DATABASE CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE CONFIGURATION command“ in *Command Reference*

tsm_mgmtclass - Tivoli Storage Manager-Verwaltungs-klasse

Konfigurationstyp	Datenbank
Parametertyp	Konfigurierbar
Standardwert [Bereich]	NULL [beliebige Zeichenfolge]

Die Tivoli Storage Manager-Verwaltungsklasse (TSM) legt fest, wie der TSM-Server die Backup-Versionen der Objekte verwalten soll, die gesichert werden.

Standardmäßig ist für DB2 keine Verwaltungsklasse angegeben.

Beim Ausführen eines TSM-Backups versucht TSM vor Verwendung der im Datenbankkonfigurationsparameter angegebenen Verwaltungsklasse zunächst, das Backup-Objekt an die Verwaltungsklasse zu binden, die in der Liste INCLUDE-EXCLUDE angegeben ist, die sich in der Datei mit den TSM-Clientoptionen befindet. Wird keine Übereinstimmung gefunden, wird die standardmäßige TSM-Verwaltungsklasse verwendet, die auf dem TSM-Server angegeben ist. Anschließend bindet TSM das Backup-Objekt erneut an die mit dem Datenbankkonfigurationsparameter angegebene Verwaltungsklasse.

Daher muss sowohl die Standardverwaltungsklasse als auch die mit dem Datenbankkonfigurationsparameter angegebene Verwaltungsklasse eine Backup-Kopiengruppe enthalten, da die Backup-Operation sonst fehlschlägt.

Zugehörige Referenzen:

- „Tivoli Storage Manager“ in *Datenrecovery und hohe Verfügbarkeit Handbuch und Referenz*
- „GET DATABASE CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE CONFIGURATION command“ in *Command Reference*

tsm_nodename - TSM-Knotenname

Konfigurationstyp	Datenbank
Parametertyp	Online konfigurierbar
Weitergabeklasse	Anweisungsgrenzwert
Standardwert [Bereich]	NULL [beliebige Zeichenfolge]

Mit diesem Parameter kann die Standardeinstellung für den Knotennamen, der dem Produkt Tivoli Storage Manager (TSM) zugeordnet ist, überschrieben werden. Der Knotenname ist erforderlich, um eine Datenbank wiederherstellen zu können, die von einem anderen Knoten aus in TSM gesichert wurde.

Standardmäßig können Sie eine Datenbank in TSM nur auf dem Knoten wiederherstellen, von dem aus das Backup ausgeführt wurde. Es ist möglich, dass der Wert für den Parameter *tsm_nodename* bei einem Backup mit DB2 (z. B. mit dem Befehl BACKUP DATABASE) überschrieben wird.

Zugehörige Referenzen:

- „GET DATABASE CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE CONFIGURATION command“ in *Command Reference*

tsm_owner - TSM-Eignername

Konfigurationstyp	Datenbank
Parametertyp	Online konfigurierbar
Weitergabeklasse	Anweisungsgrenzwert
Standardwert [Bereich]	NULL [beliebige Zeichenfolge]

Mit diesem Parameter kann die Standardeinstellung für den Eigner, der dem Produkt Tivoli Storage Manager (TSM) zugeordnet ist, überschrieben werden. Der Eignername ist erforderlich, um eine Datenbank wiederherstellen zu können, die von einem anderen Knoten aus in TSM gesichert wurde. Es ist möglich, dass der Wert für den Parameter *tsm_owner* bei einem Backup mit DB2 (z. B. mit dem Befehl BACKUP DATABASE) überschrieben wird.

Anmerkung: Beim Eigernamen muss die Groß-/Kleinschreibung beachtet werden.

Standardmäßig können Sie eine Datenbank in TSM nur auf dem Knoten wiederherstellen, von dem aus das Backup ausgeführt wurde.

Zugehörige Referenzen:

- „GET DATABASE CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE CONFIGURATION command“ in *Command Reference*

tsm_password - TSM-Kennwort

Konfigurationstyp	Datenbank
Parametertyp	Online konfigurierbar
Weitergabeklasse	Anweisungsgrenzwert
Standardwert [Bereich]	NULL [beliebige Zeichenfolge]

Mit diesem Parameter kann die Standardeinstellung für das Kennwort, das dem Produkt Tivoli Storage Manager (TSM) zugeordnet ist, überschrieben werden. Dieses Kennwort ist erforderlich, um eine Datenbank wiederherstellen zu können, die in TSM von einem anderen Knoten aus gesichert wurde.

Anmerkung: Wenn der Parameter *tsm_nodename* bei einem Backup mit DB2 (z. B. mit dem Befehl BACKUP DATABASE) überschrieben wird, muss möglicherweise auch der Parameter *tsm_password* festgelegt werden.

Standardmäßig können Sie eine Datenbank in TSM nur auf dem Knoten wiederherstellen, von dem aus das Backup ausgeführt wurde. Es ist möglich, dass der Wert für den Parameter *tsm_nodename* bei einem Backup mit DB2 überschrieben wird.

Zugehörige Referenzen:

- „GET DATABASE CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE CONFIGURATION command“ in *Command Reference*

DUOW-Recovery

Die folgenden Parameter wirken sich auf die Recovery von DUOW-Transaktionen (DUOW = Distributed Unit of Work, verteilte Arbeitseinheit) aus:

- „resync_interval - Intervall für Transaktionsresynchronisation “ auf Seite 530
- „spm_log_file_sz - Protokolldateigröße für SPM “ auf Seite 530
- „spm_log_path - Protokolldateipfad für SPM “ auf Seite 531
- „spm_max_resync - Maximale Anzahl von SPM-Resynchronisationsagenten “ auf Seite 532

- „spm_name - Name des Synchronisationspunktmanagers “ auf Seite 532
- „tm_database - Name für Transaktionsmanagerdatenbank “ auf Seite 533

resync_interval - Intervall für Transaktionsresynchronisation

Konfigurationstyp Datenbankmanager

Gilt für

- Datenbankserver mit lokalen und fernen Clients
- Datenbankserver mit lokalen Clients
- Partitionierten Datenbankserver mit lokalen und fernen Clients

Parametertyp Konfigurierbar

Standardwert [Bereich] 180 [1 – 60 000]

Maßeinheit Sekunden

Mit diesem Parameter wird das Zeitintervall in Sekunden angegeben, während dessen ein Transaktionsmanager (TM), ein Ressourcenmanager (RM) oder ein Synchronisationspunktmanager (SPM) versuchen soll, jede ausstehende unbestätigte Transaktion, die in dem TM, RM oder SPM gefunden wird, wiederherzustellen. Dieser Parameter ist gültig, wenn Sie Transaktionen in einer DUOW-Umgebung (DUOW = Distributed Unit of Work, verteilte Arbeitseinheit) ausführen. Dieser Parameter gilt ebenfalls für die Recovery von Systemen mit föderierten Datenbanken.

Empfehlung: Wenn in Ihrer Umgebung unbestätigte Transaktionen keine Störungen anderer Transaktionen für Ihre Datenbank verursachen, können Sie den Wert dieses Parameters auch erhöhen. Wenn Sie ein DB2 Connect-Gateway zum Zugriff auf DRDA2-Anwendungsserver verwenden, sollten Sie die Auswirkungen bedenken, die unbestätigte Transaktionen auf Anwendungsservern haben könnten, auch wenn lokal keine Störungen beim Datenzugriff zu erwarten sind. Gibt es keine unbestätigten Transaktionen, sind die Auswirkungen auf die Leistung minimal.

Zugehörige Referenzen:

- „GET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE MANAGER CONFIGURATION command“ in *Command Reference*

spm_log_file_sz - Protokolldateigröße für SPM

Konfigurationstyp Datenbankmanager

Gilt für

- Datenbankserver mit lokalen und fernen Clients
- Datenbankserver mit lokalen Clients
- Partitionierten Datenbankserver mit lokalen und fernen Clients

Parametertyp Konfigurierbar

Standardwert [Bereich] 256 [4 — 1 000]

Maßeinheit Seiten (4 KB)

Mit diesem Parameter wird die Größe der Protokolldatei für den Synchronisationspunktmanager (SPM) in 4-KB-Seiten angegeben. Die Protokolldatei befindet sich im Unterverzeichnis `spmlog` im Verzeichnis `sqllib` und wird erstellt, wenn der Synchronisationspunktmanager zum ersten Mal gestartet wird.

Empfehlung: Die Protokolldatei des Synchronisationspunktmanagers sollte einerseits groß genug sein, um die Leistung zu gewährleisten, andererseits aber auch klein genug, um die Verschwendung von Speicherbereich zu vermeiden. Die erforderliche Größe hängt von der Anzahl der Transaktionen ab, die geschützte Dialoge verwenden, und von der Häufigkeit, mit der Commits oder Rollbacks ausgeführt werden.

Gehen Sie wie folgt vor, um die Größe der SPM-Protokolldatei zu ändern:

1. Verwenden Sie den Befehl `LIST DRDA INDOUBT TRANSACTIONS`, um festzustellen, ob es unbestätigte Transaktionen gibt.
2. Wenn es keine unbestätigten Transaktionen gibt, stoppen Sie den Datenbankmanager.
3. Aktualisieren Sie die Konfiguration des Datenbankmanagers mit dem neuen Wert für die Größe der SPM-Protokolldatei.
4. Wechseln Sie in das Verzeichnis `$HOME/sqllib`, und löschen Sie die aktuelle SPM-Protokolldatei mit dem Befehl `rm -fr spmlog`. (Anmerkung: Dies ist der AIX-Befehl. Auf anderen Systemen sind wahrscheinlich andere Löschbefehle erforderlich.)
5. Starten Sie den Datenbankmanager. Beim Start des Datenbankmanagers wird eine neue SPM-Protokolldatei in der angegebenen Größe erstellt.

Zugehörige Referenzen:

- „GET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE MANAGER CONFIGURATION command“ in *Command Reference*

spm_log_path - Protokolldateipfad für SPM

Konfigurationstyp

Datenbankmanager

Gilt für

- Datenbankserver mit lokalen und fernen Clients
- Datenbankserver mit lokalen Clients
- Partitionierten Datenbankserver mit lokalen und fernen Clients

Parametertyp

Konfigurierbar

Standardwert [Bereich]

`sqllib/spmlog` [ein beliebiger gültiger Pfad oder eine beliebige gültige Einheit]

Dieser Parameter gibt an, in welches Verzeichnis die SPM-Protokolle geschrieben werden. Standardmäßig werden die Protokolle in das Verzeichnis `sqllib/spmlog` geschrieben; dies kann in Umgebungen mit hohem Transaktionsaufkommen zu E/A-Engpässen führen. Verwenden Sie diesen Parameter, damit SPM-Protokoll-

dateien auf eine Platte mit kürzeren Zugriffszeiten als das aktuelle Verzeichnis `sql/lib/spmlog` geschrieben werden. Dadurch wird der gemeinsame Zugriff der SPM-Agenten optimiert.

Zugehörige Referenzen:

- „GET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE MANAGER CONFIGURATION command“ in *Command Reference*

spm_max_resync - Maximale Anzahl von SPM-Resynchronisationsagenten

Konfigurationstyp	Datenbankmanager
Gilt für	• Datenbankserver mit lokalen und fernen Clients • Datenbankserver mit lokalen Clients • Partitionierten Datenbankserver mit lokalen und fernen Clients
Parametertyp	Konfigurierbar
Standardwert [Bereich]	20 [10 — 256]

Mit diesem Parameter wird die Anzahl der Agenten angegeben, die gleichzeitig Resynchronisationsoperationen ausführen können.

Zugehörige Referenzen:

- „GET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE MANAGER CONFIGURATION command“ in *Command Reference*

spm_name - Name des Synchronisationspunktmanagers

Konfigurationstyp	Datenbankmanager
Gilt für	• Datenbankserver mit lokalen und fernen Clients • Datenbankserver mit lokalen Clients • Partitionierten Datenbankserver mit lokalen und fernen Clients
Parametertyp	Konfigurierbar
Standardwert	Vom TCP/IP-Hostnamen abgeleitet

Mit diesem Parameter wird der Name der Instanz des Synchronisationspunktmanagers (SPM) gegenüber dem Datenbankmanager identifiziert.

Zugehörige Referenzen:

- „GET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE MANAGER CONFIGURATION command“ in *Command Reference*

tm_database - Name für Transaktionsmanagerdatenbank

Konfigurationstyp

Datenbankmanager

Gilt für

- Datenbankserver mit lokalen und fernen Clients
- Client
- Datenbankserver mit lokalen Clients
- Partitionierten Datenbankserver mit lokalen und fernen Clients

Parametertyp

Konfigurierbar

Standardwert [Bereich]

1ST_CONN [beliebiger gültiger Datenbankname]

Mit diesem Parameter wird der Name der Transaktionsmanagerdatenbank (TMD) für jede DB2-Instanz angegeben. Eine Transaktionsmanagerdatenbank kann eine der folgenden Datenbanken sein:

- Eine lokale DB2-Datenbank
- Eine ferne DB2-Datenbank, die sich nicht auf einem Host oder System AS/400 befindet
- Eine Datenbank von DB2 für OS/390 Version 5, wenn auf sie über TCP/IP zugegriffen und der Synchronisationspunktmanager (SPM) nicht verwendet wird

Es handelt sich dabei um eine Datenbank, die zu Protokoll- und Koordinierungsfunktionen verwendet wird und die zur Recovery unbestätigter Transaktionen dient.

Dieser Parameter kann auf den Wert **1ST_CONN** gesetzt werden, wodurch festgelegt wird, dass die Transaktionsmanagerdatenbank die erste Datenbank ist, zu der ein Benutzer eine Verbindung herstellt.

Empfehlung: Zur Vereinfachung der Verwaltung und des Betriebs können Sie einige Datenbanken in verschiedenen Instanzen erstellen und diese Datenbanken ausschließlich als Transaktionsmanagerdatenbanken verwenden.

Zugehörige Referenzen:

- „GET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE MANAGER CONFIGURATION command“ in *Command Reference*

Datenbankverwaltung

Eine Reihe von Parametern stellt Informationen zur Datenbank bereit oder beeinflusst die Verwaltung Ihrer Datenbank. Diese werden in folgende Kategorien unterteilt:

- „Query Enabler“
- „Attribute“
- „Status“ auf Seite 538
- „Compilereinstellungen“ auf Seite 540
- „Automatische Verwaltung“ auf Seite 547

Query Enabler

Der folgende Parameter stellt Informationen für die Steuerung von Query Enabler bereit:

- „dyn_query_mgmt - Dynamische SQL- und XQuery-Abfrageverwaltung“

dyn_query_mgmt - Dynamische SQL- und XQuery-Abfrageverwaltung

Konfigurationstyp	Datenbank
Parametertyp	Online konfigurierbar
Standardwert [Bereich]	0 (DISABLE) [1(ENABLE), 0 (DISABLE)]

Dieser Parameter ist dort relevant, wo DB2 Query Patroller installiert ist. Wenn dieser Parameter auf den Wert „ENABLE“ gesetzt wird, erfasst Query Patroller Informationen über die Abfrage, wie zum Beispiel die ID des übergebenden Benutzers und den geschätzten Ausführungsaufwand, der durch das Optimierungsprogramm berechnet wurde. Anhand dieser Werte wird unter Beachtung der Schwellenwerte auf Benutzer- und Systemebene bestimmt, ob die Abfrage durch Query Patroller verwaltet werden sollte.

Wenn dieser Parameter auf den Wert „DISABLE“ gesetzt wird, erfasst Query Patroller keine Informationen zu übergebenen Abfragen, und es findet keine Abfrageverwaltung statt.

Zugehörige Referenzen:

- „GET DATABASE CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE CONFIGURATION command“ in *Command Reference*

Attribute

Die folgenden Parameter stellen allgemeine Informationen zur Datenbank bereit:

- „alt_collate - Alternative Sortierfolge“ auf Seite 535
- „codepage - Codepage für die Datenbank“ auf Seite 535
- „codeset - Codierter Zeichensatz für die Datenbank“ auf Seite 535
- „collate_info - Informationen zur Sortierfolge“ auf Seite 536
- „country/region - Datenbankgebietscode“ auf Seite 537
- „database_level - Releasestand der Datenbank“ auf Seite 537
- „pagesize - Standardgröße der Datenbankseite“ auf Seite 537

- „release - Releasestand der Datenbankkonfiguration ” auf Seite 537
- „territory - Datenbankgebiet ” auf Seite 538

Mit Ausnahme des Parameters *alt_collate* dienen diese Parameter lediglich der Information.

alt_collate - Alternative Sortierfolge

Konfigurationstyp	Datenbank
Gilt für	• Datenbankserver mit lokalen und fernen Clients • Client • Datenbankserver mit lokalen Clients • Partitionierten Datenbankserver mit lokalen und fernen Clients
Parametertyp	Konfigurierbar
Standardwert [Bereich]	Null [IDENTITY_16BIT]

Mit diesem Parameter wird die Sortierfolge angegeben, die für Unicode-Tabellen in einer Nicht-Unicode-Datenbank zu verwenden ist. Unicode-Tabellen und -Routinen können erst in einer Nicht-Unicode-Datenbank erstellt werden, wenn dieser Parameter definiert wurde. Wenn dieser Parameter einmal definiert ist, kann er nicht mehr geändert oder zurückgesetzt werden.

Dieser Parameter kann nicht für Unicode-Datenbank definiert werden.

Zugehörige Referenzen:

- „GET DATABASE CONFIGURATION command” in *Command Reference*
- „RESET DATABASE CONFIGURATION command” in *Command Reference*
- „UPDATE DATABASE CONFIGURATION command” in *Command Reference*

codepage - Codepage für die Datenbank

Konfigurationstyp	Datenbank
Parametertyp	Informativ

Dieser Parameter zeigt die Codepage an, die zur Erstellung der Datenbank verwendet wurde. Der Wert des Parameters *codepage* wird auf der Basis des Werts für den Parameter *codeset* abgeleitet.

Zugehörige Referenzen:

- „codeset - Codierter Zeichensatz für die Datenbank ” auf Seite 535
- „GET DATABASE CONFIGURATION command” in *Command Reference*

codeset - Codierter Zeichensatz für die Datenbank

Konfigurationstyp	Datenbank
Parametertyp	Informativ

Dieser Parameter zeigt den codierten Zeichensatz an, der zur Erstellung der Datenbank verwendet wurde. Der codierte Zeichensatz wird vom Datenbankmanager zur Bestimmung des Parameters *codepage* verwendet.

Zugehörige Referenzen:

- „codepage - Codepage für die Datenbank ” auf Seite 535
- „GET DATABASE CONFIGURATION command” in *Command Reference*

collate_info - Informationen zur Sortierfolge

Dieser Parameter kann nur mit der API db2CfgGet angezeigt werden. Er kann **nicht** mit Hilfe des Befehlszeilenprozessors oder der Steuerzentrale angezeigt werden.

Konfigurationstyp Datenbank

Parametertyp Informativ

Dieser Parameter enthält 260 Byte mit Informationen zur Sortierfolge der Datenbank. Die ersten 256 Byte geben die Sortierfolge der Datenbank an, wobei Byte „n” die Sortierwertigkeit des Codepunkts enthält, dessen zugrundeliegende dezimale Darstellung „n” in der Codepage der Datenbank ist.

Die letzten 4 Byte enthalten interne Informationen zur Art der Sortierfolge. Bei den letzten vier Byte des Parameters handelt es sich um eine ganze Zahl. Die ganze Zahl ist abhängig von der Endian-Folge der Plattform. Die folgenden Werte sind möglich:

- **0** – Die Sortierfolge enthält nicht eindeutige Wertigkeiten.
- **1** – Die Sortierfolge enthält ausschließlich eindeutige Wertigkeiten.
- **2** – Die Sortierfolge ist die Identitätssortierfolge, nach der Zeichenfolgen Byte für Byte verglichen werden.
- **3** – Die Sortierfolge lautet NLSCHAR und wird für das Sortieren von Zeichen in der Datenbank für Thaiändisch TIS620-1 (Codepage 874) verwendet.
- **4** – Die Sortierfolge lautet IDENTITY_16BIT und implementiert den Algorithmus „CESU-8 Compatibility Encoding Scheme for UTF-16: 8-Bit”, wie er im Unicode Technical Report 26 angegeben wird, der auf der Website des Unicode Technical Consortium unter der Adresse <http://www.unicode.org> zur Verfügung steht.
- **X'8001'** – Die Sortierfolge lautet UCA400_NO und implementiert den UCA-Algorithmus (Unicode Collation Algorithm) auf der Grundlage von Unicode Standard Version 4.00 mit implizit aktivierter Normalisierung.
- **X'8002'** – Die Sortierfolge lautet UCA400_LTH und implementiert den UCA-Algorithmus (Unicode Collation Algorithm) auf der Grundlage von Unicode Standard Version 4.00. Sie sortiert alle Zeichen der thailändischen Sprache in der Reihenfolge des Royal Thai Dictionary.
- **X'8003'** – Die Sortierfolge lautet UCA400_LSK und implementiert den UCA-Algorithmus (Unicode Collation Algorithm) auf der Grundlage von Unicode Standard Version 4.00. Sie sortiert alle Zeichen der slowakischen Sprache in der richtigen Reihenfolge.

Wenn Sie diese internen Informationen zur Art der Sortierfolge verwenden, müssen Sie eine Bytefolgeumkehrung in Betracht ziehen, wenn Informationen zu einer Datenbank auf einer anderen Plattform abgerufen werden.

Sie können die Sortierfolge bei der Erstellung der Datenbank angeben.

Zugehörige Referenzen:

- „GET DATABASE CONFIGURATION command” in *Command Reference*

country/region - Datenbankgebietscode

Konfigurationstyp	Datenbank
Parametertyp	Informativ

Dieser Parameter zeigt den Gebietscode an, der beim Erstellen der Datenbank verwendet wurde.

Zugehörige Referenzen:

- „territory - Datenbankgebiet “ auf Seite 538
- „GET DATABASE CONFIGURATION command“ in *Command Reference*

database_level - Releasestand der Datenbank

Konfigurationstyp	Datenbank
Parametertyp	Informativ

Dieser Parameter gibt den Releasestand des Datenbankmanagers an, der die Datenbank verwenden kann. Im Fall einer nicht abgeschlossenen oder fehlgeschlagenen Migration (Umstellung) spiegelt dieser Parameter den Releasestand der nicht umgestellten Datenbank wider und kann daher vom Parameter *release* (Releasestand der Datenbankkonfiguration) abweichen. Ansonsten ist der Wert des Parameters *database_level* mit dem Wert des Parameters *release* identisch.

Zugehörige Referenzen:

- „release - Releasestand der Datenbankkonfiguration “ auf Seite 537
- „GET DATABASE CONFIGURATION command“ in *Command Reference*

pagesize - Standardgröße der Datenbankseite

Konfigurationstyp	Datenbank
Parametertyp	Informativ

Dieser Parameter enthält den Wert, der als Standardseitengröße bei der Erstellung der Datenbank verwendet wurde. Gültige Werte sind: 4.096, 8.192, 16.384 und 32.768. Wenn in dieser Datenbank ein Pufferpool oder ein Tabellenbereich erstellt wird, wird dieselbe Standardseitengröße verwendet.

Zugehörige Referenzen:

- „CREATE DATABASE command“ in *Command Reference*

release - Releasestand der Datenbankkonfiguration

Konfigurationstyp	Datenbankmanager, Datenbank

Gilt für

- Datenbankserver mit lokalen und fernen Clients
- Datenbankserver mit lokalen Clients
- Partitionierten Datenbankserver mit lokalen und fernen Clients

Parametertyp	Informativ

Dieser Parameter gibt den Releasestand der Konfigurationsdatei an.

Zugehörige Referenzen:

- „database_level - Releasestand der Datenbank ” auf Seite 537
- „GET DATABASE CONFIGURATION command” in *Command Reference*
- „GET DATABASE MANAGER CONFIGURATION command” in *Command Reference*

territory - Datenbankgebiet

Konfigurationstyp Datenbank

Parametertyp Informativ

Dieser Parameter zeigt das Gebiet an, mit dem die Datenbank erstellt wurde. Der Parameter *territory* wird vom Datenbankmanager bei der Verarbeitung von Daten verwendet, für die das Gebiet von Bedeutung ist.

Zugehörige Referenzen:

- „country/region - Datenbankgebietscode ” auf Seite 537
- „GET DATABASE CONFIGURATION command” in *Command Reference*

Status

Die folgenden Parameter stellen Informationen zum Status der Datenbank bereit:

- „backup_pending - Backup anstehend ”
- „database_consistent - Datenbank ist konsistent ”
- „log_retain_status - Statusanzeiger für Beibehalten der Protokolle ” auf Seite 539
- „multipage_alloc - Zuordnung aus mehreren Seiten bestehender Datei aktiv ” auf Seite 539
- „restore_pending - Restore anstehend ” auf Seite 540
- „rollfwd_pending - Aktualisierende Recovery anstehend ” auf Seite 540
- „user_exit_status - Statusanzeiger für Benutzerexit ” auf Seite 540

backup_pending - Backup anstehend

Konfigurationstyp Datenbank

Parametertyp Informativ

Wenn dieser Parameter aktiviert ist, bedeutet dies, dass Sie ein Gesamtbackup der Datenbank ausführen müssen, bevor Sie auf sie zugreifen. Dieser Parameter ist nur aktiviert, wenn die Datenbankkonfiguration geändert wurde, so dass die Datenbank jetzt nicht mehr nicht wiederherstellbar sondern wiederherstellbar ist (das bedeutet, die Parameter *logretain* und *userexit* waren zunächst auf NO gesetzt, später wird jedoch einer der Parameter (oder beide) auf YES gesetzt und die Aktualisierung der Datenbankkonfiguration wird akzeptiert).

Zugehörige Referenzen:

- „GET DATABASE CONFIGURATION command” in *Command Reference*

database_consistent - Datenbank ist konsistent

Konfigurationstyp Datenbank

Parametertyp Informativ

Dieser Parameter gibt an, ob die Datenbank in einem konsistenten Zustand ist.

YES gibt an, dass alle Transaktionen mit COMMIT festgeschrieben oder mit ROLLBACK rückgängig gemacht wurden, so dass die Daten konsistent sind. Wenn es zu einem Systemabsturz kommt, während die Datenbank konsistent ist, sind keinerlei Maßnahmen erforderlich, um die Datenbank wieder verwendbar zu machen.

NO gibt an, dass noch eine Transaktion ansteht oder eine andere Funktion in der Datenbank noch nicht abgeschlossen wurde, so dass die Daten zu diesem Zeitpunkt nicht konsistent sind. Wenn es zu einem Systemabsturz kommt, während die Datenbank nicht konsistent ist, müssen Sie die Datenbank mit dem Befehl RESTART DATABASE erneut starten, um sie wieder verwendbar zu machen.

Zugehörige Referenzen:

- „GET DATABASE CONFIGURATION command“ in *Command Reference*

log_retain_status - Statusanzeiger für Beibehalten der Protokolle

Konfigurationstyp	Datenbank
Parametertyp	Informativ

Wenn dieser Parameter gesetzt ist, zeigt er an, dass Protokolldateien für die aktualisierende Recovery gespeichert werden.

Dieser Parameter wird gesetzt, wenn der Parameter *logretain* auf Recovery gesetzt ist.

Zugehörige Referenzen:

- „logretain - Beibehalten von Protokollen aktivieren “ auf Seite 511
- „GET DATABASE CONFIGURATION command“ in *Command Reference*

multipage_alloc - Zuordnung aus mehreren Seiten bestehender Datei aktiv

Konfigurationstyp	Datenbank
Parametertyp	Informativ

Die Zuordnung aus mehreren Seiten bestehender Dateien erhöht die Leistung beim Einfügen. Sie gilt nur für SMS-Tabellenbereiche. Wenn dieser Parameter aktiviert ist, sind alle SMS-Tabellenbereiche davon betroffen. Es ist keine Auswahl für einzelne SMS-Tabellenbereiche möglich.

Der Standardwert für den Parameter ist "Yes", d. h. die Zuordnung aus mehreren Seiten bestehender Dateien ist aktiviert.

Im Anschluss an eine Datenbankerstellung kann dieser Parameter nicht auf "No" gesetzt werden. Die Zuordnung mehrseitiger Dateien kann nach der Aktivierung nicht mehr inaktiviert werden. Wenn die Zuordnung mehrseitiger Dateien nicht erwünscht ist, muss die DB2-Registrierdatenbankvariable DB2_NO_MPFA_FOR_NEW_DB entsprechend definiert werden, bevor die Datenbank erstellt wird. Das Tool **db2empfa** kann zur Aktivierung der Zuordnung mehrseitiger Dateien für eine Datenbank verwendet werden, für die dieses Merkmal inaktiviert ist.

Zugehörige Referenzen:

- „GET DATABASE CONFIGURATION command“ in *Command Reference*

restore_pending - Restore anstehend

Konfigurationstyp	Datenbank
Parametertyp	Informativ

Dieser Parameter gibt an, ob sich die Datenbank im Status *Restore anstehend* befindet.

Zugehörige Referenzen:

- „userexit - Benutzerexit aktivieren ” auf Seite 516
- „GET DATABASE CONFIGURATION command” in *Command Reference*

rollfwd_pending - Aktualisierende Recovery anstehend

Konfigurationstyp	Datenbank
Parametertyp	Informativ

Dieser Parameter kann einen der folgenden Status anzeigen:

- **DATABASE**, d. h. eine aktualisierende Recovery ist für diese Datenbank erforderlich
- **TABLESPACE**, d. h. für mindestens einen Tabellenbereich muss eine aktualisierende Recovery durchgeführt werden
- **NO**, d. h. die Datenbank ist verwendbar und keine aktualisierende Recovery erforderlich

Die Recovery (mit ROLLFORWARD DATABASE) muss erfolgreich beendet sein, bevor auf die Datenbank bzw. den Tabellenbereich zugegriffen werden kann.

Zugehörige Referenzen:

- „GET DATABASE CONFIGURATION command” in *Command Reference*

user_exit_status - Statusanzeiger für Benutzerexit

Konfigurationstyp	Datenbank
Parametertyp	Informativ

Wenn der Wert dieses Parameters "On" ist, heißt dies, dass der Datenbankmanager für die aktualisierende Recovery aktiviert ist und dass das Benutzerexitprogramm verwendet wird, um Protokolldateien zu archivieren und wieder abzurufen, wenn es vom Datenbankmanager aufgerufen wird.

Zugehörige Referenzen:

- „userexit - Benutzerexit aktivieren ” auf Seite 516
- „GET DATABASE CONFIGURATION command” in *Command Reference*

Compilereinstellungen

Die folgenden Parameter stellen Informationen zur Verfügung, die den Compiler beeinflussen:

- „dft_degree - Standardgrad der Parallelität ” auf Seite 541
- „dft_mttb_types - Standardtypen von verwalteten Tabellen zur Optimierung ” auf Seite 541
- „dft_queryopt - Standardabfrageoptimierungsklasse ” auf Seite 542
- „dft_refresh_age - Standardaktualisierungsalter ” auf Seite 542

- „dft_sqlmathwarn - Bei arithmetischen Ausnahmebedingungen fortsetzen “ auf Seite 543
- „num_freqvalues - Anzahl der häufigsten Werte “ auf Seite 544
- „num_quantiles - Anzahl der Quantile für Spalten “ auf Seite 546

dft_degree - Standardgrad der Parallelität

Konfigurationstyp	Datenbank
Parametertyp	Online konfigurierbar
Weitergabeklasse	Sofort
Standardwert [Bereich]	1 [-1, 1 – 32 767]

Mit diesem Parameter wird der Standardwert für das Sonderregister CURRENT DEGREE und die Bindeoption DEGREE angegeben.

Der Standardwert ist 1.

Bei Angabe des Werts 1 wird keine partitionsinterne Parallelität verwendet. Der Wert -1 bedeutet, dass das Optimierungsprogramm den Grad der partitions-internen Parallelität je nach Anzahl der Prozessoren und Art der Abfrage festlegt.

Der Grad der partitionsinternen Parallelität für eine SQL-Anweisung wird während der Kompilierung der Anweisung mit Hilfe des Sonderregisters CURRENT DEGREE oder der Bindeoption DEGREE angegeben. Der maximale Grad der partitionsinternen Parallelität zur Laufzeit für eine aktive Anwendung wird mit dem Befehl SET RUNTIME DEGREE angegeben. Der Konfigurationsparameter Maximaler Grad der Parallelität bei Abfragen (*max_querydegree*) gibt den maximalen Grad der partitionsinternen Parallelität für alle SQL-Abfragen an.

Der zur Laufzeit tatsächlich verwendete Parallelitätsgrad ist der niedrigste der folgenden Werte:

- Konfigurationsparameter *max_querydegree*
- Grad der Anwendung zur Laufzeit
- Parallelitätsgrad bei der Kompilierung der SQL-Anweisung

Zugehörige Referenzen:

- „max_querydegree - Maximaler Grad der Parallelität bei Abfragen “ auf Seite 559
- „Einschränkungen für den nativen XML-Datenspeicher“ in *XML-Handbuch*
- „GET DATABASE CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE CONFIGURATION command“ in *Command Reference*

dft_mttb_types - Standardtypen von verwalteten Tabellen zur Optimierung

Konfigurationstyp	Datenbank
Parametertyp	Konfigurierbar
Standardwert [Bereich]	SYSTEM [ALL, NONE, FEDERATED_TOOL, SYSTEM, USER oder eine Liste von Werten]

Mit diesem Parameter wird der Standardwert für das Sonderregister CURRENT MAINTAINED TABLE TYPES FOR OPTIMIZATION angegeben. Der Wert dieses Registers bestimmt, welche Typen mit REFRESH DEFERRED definierter MQTs der Abfrageoptimierung verwendet werden.

Sie können eine Liste von Werten durch Kommata getrennt angeben. Zum Beispiel: 'USER,FEDERATED_TOOL'. ALL und NONE können nicht zusammen mit anderen Werten angegeben werden, und Sie können denselben Wert nur einmal angeben.

Zugehörige Referenzen:

- „CURRENT MAINTAINED TABLE TYPES FOR OPTIMIZATION special register“ in *SQL Reference, Volume 1*

dft_queryopt - Standardabfrageoptimierungsklasse

Konfigurationstyp	Datenbank
Parametertyp	Online konfigurierbar
Weitergabeklasse	Sofort
Standardwert [Bereich]	5 [0 — 9]
Maßeinheit	Abfrageoptimierungsklasse (siehe unten)

Mit der Abfrageoptimierungsklasse können Sie das Optimierungsprogramm anweisen, beim Kompilieren von SQL- und XQuery-Abfragen verschiedene Grade der Optimierung zu verwenden. Dieser Parameter bietet zusätzliche Flexibilität, indem die Standardabfrageoptimierungsklasse für den Fall festgelegt wird, dass weder die Anweisung SET CURRENT QUERY OPTIMIZATION noch die Option QUERYOPT mit dem Bindebefehl verwendet werden.

Derzeit sind folgende Abfrageoptimierungsklassen definiert:

- 0 - Minimale Abfrageoptimierung
- 1 - Abfrageoptimierung in etwa vergleichbar mit DB2 Version 1
- 2 - Geringe Optimierung
- 3 - Mittlere Abfrageoptimierung
- 5 - Starke Abfrageoptimierung, die über heuristische Methoden den Aufwand bei der Auswahl eines Zugriffsplans reduziert. Dies ist der Standardwert.
- 7 - Starke Abfrageoptimierung
- 9 - Maximale Abfrageoptimierung

Zugehörige Referenzen:

- „SET CURRENT QUERY OPTIMIZATION statement“ in *SQL Reference, Volume 2*
- „BIND command“ in *Command Reference*
- „LIST DRDA INDOUBT TRANSACTIONS command“ in *Command Reference*
- „GET DATABASE CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE CONFIGURATION command“ in *Command Reference*

dft_refresh_age - Standardaktualisierungsalter

Konfigurationstyp	Datenbank
Parametertyp	Konfigurierbar
Standardwert [Bereich]	0 [0, 99999999999999 (ANY)]

Für diesen Parameter wird der für REFRESH AGE verwendete Standardwert eingesetzt, wenn das Sonderregister CURRENT REFRESH AGE nicht angegeben wird. Dieser Parameter gibt einen Zeitmarkendifferenzwert mit dem Datentyp DECIMAL(20,6) an. Diese Zeitdifferenz stellt die maximale Dauer seit der Verarbeitung einer Anweisung REFRESH TABLE für eine spezifische MQT (Materialized Query Table, gespeicherte Abfragetabelle) des Typs REFRESH DEFERRED dar, während diese Übersichtstabelle zum Optimieren der Verarbeitung einer Abfrage verwendet werden kann. Wenn für CURRENT REFRESH AGE der Wert 9999999999999999 (ANY) gilt und die Abfrageoptimierungsklasse (QUERY OPTIMIZATION) den Wert zwei (oder fünf und höher) hat, werden MQTs der Art REFRESH DEFERRED zum Optimieren der Verarbeitung einer dynamischen Abfrage herangezogen.

Zugehörige Referenzen:

- „GET DATABASE CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE CONFIGURATION command“ in *Command Reference*
- „Einschränkungen für den nativen XML-Datenspeicher“ in *XML-Handbuch*

dft_sqlmathwarn - Bei arithmetischen Ausnahmebedingungen fortsetzen

Konfigurationstyp	Datenbank
Parametertyp	Konfigurierbar
Standardwert [Bereich]	No [No, Yes]

Mit diesem Parameter wird der Standardwert festgelegt, der bestimmt, ob arithmetische Fehler und Fehler bei Abfrageumwandlungen beim Kompilieren von SQL-Anweisungen als Fehler oder als Warnungen behandelt werden. Bei statischen SQL-Anweisungen wird der Wert dieses Parameters dem Paket während des Bindens zugeordnet. Bei dynamischen SQL-DML-Anweisungen wird der Wert dieses Parameters verwendet, wenn die Anweisung vorbereitet wird (PREPARE).

Achtung: Wenn Sie den Wert des Parameters *dft_sqlmathwarn* für eine Datenbank ändern, kann sich die Funktionsweise von Prüfungen auf Integritätsbedingung, Triggern und Sichten, die arithmetische Ausdrücke enthalten, ändern. Dies wiederum kann sich auf die Datenintegrität der Datenbank auswirken. Sie sollten die Einstellung des Parameters *dft_sqlmathwarn* für eine Datenbank nur nach einer gründlichen Analyse der Auswirkungen der neuen Behandlung arithmetischer Ausnahmebedingungen auf Integritätsprüfungen, Trigger und Sichten ändern. Auch alle weiteren Änderungen sind mit derselben Sorgfalt vorzunehmen.

Betrachten Sie beispielsweise die folgende Prüfung auf Integritätsbedingung, die eine arithmetische Divisionsoperation enthält:

$A/B > 0$

Wenn *dft_sqlmathwarn* gleich „No“ ist und eine Einfügeoperation (INSERT) mit B=0 versucht wird, wird die Division durch 0 als arithmetischer Fehler verarbeitet. Die Einfügeoperation schlägt fehl, weil DB2 die Integritätsbedingung nicht prüfen kann. Wenn der Parameter *dft_sqlmathwarn* auf „Yes“ gesetzt wird, wird die Division durch 0 als arithmetische Warnung mit dem Ergebnis NULL verarbeitet. Das Ergebnis NULL führt dazu, dass das Vergleichselement mit dem Wert UNKNOWN ausgewertet wird und die Einfügeoperation erfolgreich ist. Wenn *dft_sqlmathwarn* wieder auf „No“ gesetzt wird, schlägt der Versuch, dieselbe Zeile einzufügen, fehl, weil der Fehler der Division durch 0 DB2 daran hindert, die Integritätsbedingung

auszuwerten. Die Zeile, die mit B=0 eingefügt wurde, als der Parameter *dft_sqlmathwarn* den Wert „Yes“ hatte, bleibt in der Tabelle und kann ausgewählt werden. Aktualisierungen der Zeile, für die die Integritätsbedingung ausgewertet wird, schlagen fehl, während Aktualisierungen, die keine erneute Prüfung der Integritätsbedingung erfordern, erfolgreich ausgeführt werden.

Bevor Sie den Parameter *dft_sqlmathwarn* von „No“ in „Yes“ ändern, sollten Sie in Betracht ziehen, die Integritätsbedingung so umzuschreiben, dass sie Nullwerte aus arithmetischen Ausdrücken gesondert behandelt. Zum Beispiel:

```
( A/B > 0 ) AND ( CASE
                    WHEN A IS NULL THEN 1
                    WHEN B IS NULL THEN 1
                    WHEN A/B IS NULL THEN 0
                    ELSE 1
                    END
                  = 1 )
```

Diese Bedingung kann verwendet werden, wenn A und B Nullwerte haben können, d. h. die Dateneingabe optional ist. Wenn A oder B keinen Nullwert haben kann, kann die entsprechende Klausel WHEN...IS NULL entfernt werden.

Bevor Sie den Parameter *dft_sqlmathwarn* von „Yes“ in „No“ ändern, sollten Sie zunächst prüfen, welche Daten inkonsistent werden könnten, indem Sie zum Beispiel Vergleichselemente wie die folgenden verwenden:

```
WHERE A IS NOT NULL AND B IS NOT NULL AND A/B IS NULL
```

Wenn inkonsistente Zeilen isoliert sind, können Sie geeignete Maßnahmen zur Korrektur der Inkonsistenz ergreifen, bevor Sie den Parameter *dft_sqlmathwarn* ändern. Sie können Integritätsbedingungen mit arithmetischen Ausdrücken nach der Änderung auch manuell gegenprüfen. Dazu setzen Sie die betroffenen Tabellen in den Status *Überprüfung anstehend* (mit der Klausel OFF der Anweisung SET CONSTRAINTS) und fordern anschließend eine Prüfung an (mit der Klausel IMMEDIATE CHECKED der Anweisung SET CONSTRAINTS). Inkonsistente Daten werden durch einen arithmetischen Fehler angezeigt, der verhindert, dass die Integritätsbedingung ausgewertet wird.

Empfehlung: Verwenden Sie die Standardeinstellung "No", sofern Sie keine Abfragen benötigen, deren Verarbeitung arithmetische Ausnahmebedingungen einschließt. In diesem Fall geben Sie den Wert "Yes" an. Dieser Fall kann eintreten, wenn Sie SQL-Anweisungen verarbeiten, die auf anderen Datenbankmanagern unabhängig von möglichen arithmetischen Ausnahmebedingungen Ergebnisse liefern.

Zugehörige Referenzen:

- „GET DATABASE CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE CONFIGURATION command“ in *Command Reference*

num_freqvalues - Anzahl der häufigsten Werte

Konfigurationstyp	Datenbank
Parametertyp	Online konfigurierbar
Weitergabeklasse	Sofort
Standardwert [Bereich]	10 [0 — 32 767]
Maßeinheit	Zähler

Mit diesem Parameter können Sie die Anzahl der „häufigsten Werte“ angeben, die gesammelt werden, wenn die Option WITH DISTRIBUTION im Befehl RUNSTATS angegeben wird. Wenn der Wert dieses Parameters erhöht wird, erhöht sich auch die Menge an Statistikzwischenspeicher (*stat_heap_sz*), die zum Sammeln statistischer Daten benötigt wird.

Die Statistik der „häufigsten Werte“ unterstützt das Optimierungsprogramm beim Feststellen der Verteilung der Datenwerte innerhalb einer Spalte. Wenn der Wert erhöht wird, stehen dem Abfrageoptimierungsprogramm mehr Daten zur Verfügung; allerdings ist zusätzlicher Katalogspeicher erforderlich. Wenn 0 angegeben wird, wird keine Statistik über häufige Werte erhoben, auch wenn Sie anfordern, dass statistische Informationen zur Datenverteilung gesammelt werden.

Sie können auch die Anzahl der häufigsten Werte angeben, die auf der Tabellen- oder Spaltenebene als Teil des Befehls RUNSTATS beibehalten werden. Wenn kein Wert angegeben wird, wird der Wert des Konfigurationsparameters *num_freqvalues* verwendet.

Durch Aktualisieren dieses Parameters kann das Optimierungsprogramm bessere Schätzwerte für die Auswahl von nicht gleichmäßig verteilten Daten mit Hilfe einiger Vergleichselemente (=, <, >, IS NULL, IS NOT NULL) erzielen. Exaktere Berechnungen der Selektivität können zur Auswahl effizienterer Zugriffspläne führen.

Nachdem Sie den Wert dieses Parameters verändert haben, müssen Sie wie folgt vorgehen:

- Führen Sie den Befehl RUNSTATS aus, nachdem alle Benutzer die Verbindung zur Datenbank getrennt und Sie die Verbindung zur Datenbank wiederhergestellt haben.
- Führen Sie für alle Pakete mit statischen SQL- oder XQuery-Anweisungen einen Rebind durch.

Der Befehl RUNSTATS ermöglicht es über die Option NUM_REQVALUES, die Anzahl der häufigsten Werte anzugeben. Es ist einfacher, die Anzahl der häufigsten Werte über den Befehl RUNSTATS zu ändern, als über den Datenbankkonfigurationsparameter *num_freqvalues*.

Beim Verwenden des Befehls RUNSTATS haben Sie die Möglichkeit, die Anzahl der häufigsten Werte zu begrenzen, die auf Tabellenebene oder auf Spaltenebene gesammelt werden. Dadurch können Sie den in den Katalogen belegten Speicherbereich optimieren, indem Sie die Verteilungsstatistik für Spalten reduzieren, in denen Sie nicht genutzt werden kann, und die Informationen jedoch weiter für kritische Spalten verwenden.

Empfehlung: Zur Aktualisierung dieses Parameters sollten Sie den Grad der Ungleichmäßigkeit der Daten in den wichtigsten Spalten (in den wichtigsten Tabellen) feststellen, für die in der Regel Auswahlvergleichselemente angegeben werden. Dies kann mit Hilfe einer SQL-Anweisung SELECT geschehen, die eine Rangfolge des Vorkommens jedes Werts in einer Spalte liefert. Dabei dürfen Sie einheitlich verteilte, eindeutige, lange oder LOB-Spalten nicht berücksichtigen. Ein geeigneter praktischer Wert für diesen Parameter liegt im Bereich zwischen 10 und 100.

Beachten Sie, dass das Sammeln statistischer Daten über die häufigsten Werte erhebliche CPU- und Speicherressourcen (*stat_heap_sz*) erfordert.

Zugehörige Referenzen:

- „num_quantiles - Anzahl der Quantile für Spalten ” auf Seite 546
- „stat_heap_sz - Größe des Statistikzwischenspeichers ” auf Seite 459
- „GET DATABASE CONFIGURATION command” in *Command Reference*
- „RESET DATABASE CONFIGURATION command” in *Command Reference*
- „UPDATE DATABASE CONFIGURATION command” in *Command Reference*

num_quantiles - Anzahl der Quantile für Spalten

Konfigurationstyp	Datenbank
Parametertyp	Online konfigurierbar
Weitergabeklasse	Sofort
Standardwert [Bereich]	20 [0 – 32 767]
Maßeinheit	Zähler

Mit diesem Parameter wird die Anzahl der Quantile gesteuert, die gesammelt werden, wenn die Option WITH DISTRIBUTION im Befehl RUNSTATS angegeben wird. Wenn der Wert dieses Parameters erhöht wird, erhöht sich auch die Menge an Statistikzwischenspeicher (*stat_heap_sz*), die zum Sammeln statistischer Daten benötigt wird.

Die Statistik der „Quantile” unterstützt das Optimierungsprogramm beim Feststellen der Verteilung der Datenwerte innerhalb einer Spalte. Wenn der Wert erhöht wird, stehen dem Abfrageoptimierungsprogramm mehr Daten zur Verfügung. Andererseits ist für den Katalog mehr Speicher erforderlich. Wenn die Werte 0 oder 1 angegeben werden, werden keine Quantil-Statistikdaten erhoben, auch wenn Sie anfordern, dass Verteilungsstatistikdaten gesammelt werden.

Sie können auch die Anzahl der Quantile angeben, die auf der Tabellen- oder Spaltenebene als Teil des Befehls RUNSTATS erfasst werden. Wenn kein Wert angegeben wird, wird der Wert des Konfigurationsparameters *num_quantiles* verwendet.

Durch Aktualisieren dieses Parameters können bessere Schätzwerte für die Auswahl von nicht gleichmäßig verteilten Daten mit Hilfe von Bereichsvergleichselementen erzielt werden. Unter anderem entscheidet das Optimierungsprogramm mit Hilfe dieser Informationen, ob eine Indexsuche oder eine Tabellensuche gewählt wird. (Beim Zugriff auf einen Bereich von Werten, die häufig vorkommen, ist eine Tabellensuche effizienter, während bei einem Bereich von Werten, die nicht häufig vorkommen, eine Indexsuche effizienter ist.)

Nachdem Sie den Wert dieses Parameters verändert haben, müssen Sie wie folgt vorgehen:

- Führen Sie den Befehl RUNSTATS aus, nachdem alle Benutzer die Verbindung zur Datenbank getrennt und Sie die Verbindung zur Datenbank wiederhergestellt haben.
- Führen Sie für alle Pakete mit statischen SQL- oder XQuery-Anweisungen einen Rebind durch.

Der Befehl RUNSTATS ermöglicht die Angabe der Anzahl der zu erfassenden Quantile über die Option NUM_QUANTILES. Es ist einfacher, die Anzahl der zu erfassenden Quantile über den Befehl RUNSTATS zu ändern, als über den Datenbankkonfigurationsparameter *num_quantiles*.

Beim Verwenden des Befehls RUNSTATS haben Sie die Möglichkeit, die Anzahl der erfassten Quantile sowohl auf Tabellenebene als auch auf Spaltenebene zu begrenzen. Dadurch können Sie den in den Katalogen belegten Speicherbereich optimieren, indem Sie die Verteilungsstatistik für Spalten reduzieren, in denen Sie nicht genutzt werden kann, und die Informationen jedoch weiter für kritische Spalten verwenden.

Empfehlung: Der Standardwert für diesen Parameter garantiert einen maximalen Schätzfehler von ungefähr 2,5 % für alle einseitigen Bereichsvergleichselemente (>, >=, < oder <=) und einen maximalen Fehler von 5 % für jedes BETWEEN-Vergleichselement. Nachfolgend eine einfache Möglichkeit, um die Anzahl der Quantile einzugrenzen:

- Bestimmen Sie den maximalen Fehler, der bei der Schätzung der Anzahl von Zeilen jeder Bereichsabfrage tolerierbar ist, als Prozentwert P.
- Die Anzahl der Quantile sollte ca. 100/P sein, wenn die meisten Ihrer Vergleichselemente BETWEEN-Vergleichselemente sind, und 50/P, wenn die meisten Ihrer Vergleichselemente andere Arten von Bereichsvergleichselementen (<, <=, > oder >=) sind.

Zum Beispiel ergeben 25 Quantile einen maximalen Schätzfehler von 4% bei BETWEEN-Vergleichselementen und 2% bei Vergleichselementen mit ">". Ein geeigneter praktischer Wert für diesen Parameter liegt im Bereich zwischen 10 und 50.

Zugehörige Referenzen:

- „num_freqvalues - Anzahl der häufigsten Werte “ auf Seite 544
- „stat_heap_sz - Größe des Statistikzwischenspeichers “ auf Seite 459
- „GET DATABASE CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE CONFIGURATION command“ in *Command Reference*

Automatische Verwaltung

Mit den folgenden Parametern können die automatischen Verwaltungsaktivitäten verschiedener DB2-Dienstprogramme gesteuert werden:

- „auto_maint - Automatische Verwaltung “

auto_maint - Automatische Verwaltung

Konfigurationstyp

Datenbank

Gilt für

- Datenbankserver mit lokalen und fernen Clients
- Client
- Datenbankserver mit lokalen Clients
- Partitionierten Datenbankserver mit lokalen und fernen Clients

Parametertyp

Online konfigurierbar

Weitergabeklasse

Sofort

Standardwert [Bereich]

ON [ON; OFF]

Dieser Parameter ist das übergeordnete Element für alle anderen Datenbankkonfigurationsparameter der automatischen Verwaltung (*auto_db_backup*, *auto_tbl_maint*, *auto_runstats*, *auto_stats_prof*, *auto_prof_upd* und *auto_reorg*). Wenn dieser Parameter

inaktiviert wird, werden alle untergeordneten Parameter ebenfalls inaktiviert, jedoch ändern sich ihre Einstellungen, die in der Datenbankkonfigurationsdatei aufgezeichnet sind, nicht. Wenn dieser übergeordnete Parameter aktiviert wird, werden die aufgezeichneten Werte für die untergeordneten Parameter wirksam. Auf diese Weise kann die automatische Verwaltung global aktiviert bzw. inaktiviert werden.

Standardmäßig ist dieser Parameter auf den Wert ON gesetzt.

Sie können einzelne Funktionen der automatischen Verwaltung unabhängig voneinander aktivieren oder inaktivieren, indem Sie die folgenden Parameter definieren:

auto_db_backup

Dieser Parameter der automatischen Verwaltung aktiviert bzw. inaktiviert automatische Backup-Operationen für eine Datenbank. Eine Backup-Richtlinie (eine definierte Gruppe von Regeln oder Bestimmungen) kann zur Angabe der automatischen Funktionsweise verwendet werden. Die Backup-Richtlinie hat den Zweck, sicherzustellen, dass die Datenbank regelmäßig gesichert wird. Die Backup-Richtlinie für eine Datenbank wird automatisch erstellt, wenn der DB2-Diagnosemonitor zum ersten Mal ausgeführt wird. Standardmäßig ist dieser Parameter auf den Wert OFF gesetzt. Zur Aktivierung muss dieser Parameter auf den Wert ON gesetzt werden und der übergeordnete Parameter muss ebenfalls aktiviert sein.

auto_tbl_maint

Dieser Parameter ist das übergeordnete Element für alle Tabellenverwaltungsparameter (*auto_runstats*, *auto_stats_prof*, *auto_prof_upd* und *auto_reorg*). Wenn dieser Parameter inaktiviert wird, werden alle untergeordneten Parameter ebenfalls inaktiviert, jedoch ändern sich ihre Einstellungen, die in der Datenbankkonfigurationsdatei aufgezeichnet sind, nicht. Wenn dieser übergeordnete Parameter aktiviert wird, werden die aufgezeichneten Werte für die untergeordneten Parameter wirksam. Auf diese Weise kann die Tabellenverwaltung global aktiviert bzw. inaktiviert werden.

Standardmäßig ist dieser Parameter auf den Wert ON gesetzt.

auto_runstats

Dieser Parameter der automatischen Tabellenverwaltung aktiviert bzw. inaktiviert automatische RUNSTATS-Operationen für Tabellen in einer Datenbank. Eine Richtlinie (eine definierte Gruppe von Regeln oder Bestimmungen) für die Statistikerstellung kann zur Angabe der automatischen Funktionsweise verwendet werden. Die vom Dienstprogramm RUNSTATS erfassten Statistiken werden vom Optimierungsprogramm zur Bestimmung des effizientesten Plans für den Zugriff auf die physischen Daten verwendet. Zur Aktivierung

muss dieser Parameter auf den Wert ON gesetzt werden und die übergeordneten Parameter müssen ebenfalls aktiviert sein.

Standardmäßig ist dieser Parameter auf den Wert ON gesetzt.

auto_stats_prof

Dieser Parameter der automatischen Tabellenverwaltung aktiviert die Statistikprofilgenerierung, die dazu dient, Anwendungen zu unterstützen, deren Auslastungen komplexe Abfragen, zahlreiche Vergleichselemente, Joins und Gruppierungen unter Einbeziehung verschiedener Tabellen beinhalten. Zur Aktivierung muss dieser Parameter auf den Wert ON gesetzt werden und die übergeordneten Parameter müssen ebenfalls aktiviert sein.

Standardmäßig ist dieser Parameter auf den Wert OFF gesetzt.

auto_prof_upd

Dieser Parameter der automatischen Tabellenverwaltung (ein untergeordneter Parameter von *auto_stats_prof*) gibt an, dass das RUNSTATS-Profil mit Empfehlungen zu aktualisieren ist. Wenn dieser Parameter inaktiviert wird, werden Empfehlungen in der Tabelle *opt_feedback_ranking* gespeichert, die Sie einsehen können, wenn Sie das RUNSTATS-Profil manuell aktualisieren. Zur Aktivierung muss dieser Parameter auf den Wert ON gesetzt werden und die übergeordneten Parameter müssen ebenfalls aktiviert sein.

Standardmäßig ist dieser Parameter auf den Wert OFF gesetzt.

auto_reorg

Dieser Parameter der automatischen Tabellenverwaltung aktiviert bzw. inaktiviert die automatische Tabellen- und Indexreorganisation für eine Datenbank. Eine Reorganisationsrichtlinie (eine definierte Gruppe von Regeln oder Bestimmungen) kann zur Angabe der automatischen Funktionsweise verwendet werden. Zur Aktivierung muss dieser Parameter auf den Wert ON gesetzt werden und die übergeordneten Parameter müssen ebenfalls aktiviert sein.

Standardmäßig ist dieser Parameter auf den Wert OFF gesetzt.

Zugehörige Referenzen:

- „GET DATABASE CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE CONFIGURATION command“ in *Command Reference*

Kommunikation

Die folgenden Gruppen von Parametern stellen Informationen zur Verwendung von DB2 in einer Client-/Serverumgebung bereit:

- „Konfiguration der Kommunikationsprotokolle“
- „DB2 Discovery“ auf Seite 552

Konfiguration der Kommunikationsprotokolle

Die folgenden Parameter dienen zur Konfiguration Ihrer Datenbankclients und Datenbankserver:

- „*nname* - Name der NetBIOS-Workstation “
- „*svcename* - TCP/IP-Servicename “ auf Seite 551

***nname* - Name der NetBIOS-Workstation**

Konfigurationstyp Datenbankmanager

Gilt für

- Datenbankserver mit lokalen und fernen Clients
- Client
- Datenbankserver mit lokalen Clients
- Partitionierten Datenbankserver mit lokalen und fernen Clients

Parametertyp Konfigurierbar

Standardwert Null

Mit diesem Parameter können Sie der Datenbankinstanz auf einer Workstation in der NetBIOS-LAN-Umgebung einen eindeutigen Namen zuordnen. Der Wert von *nname* ist die Basis für die tatsächlichen NetBIOS-Namen, die in NetBIOS für eine Workstation registriert werden.

Da das NetBIOS-Protokoll seine Verbindungen mit Hilfe dieser NetBIOS-Namen erstellt, muss der Parameter *nname* sowohl für den Client als auch für den Server festgelegt werden.

Client-Anwendungen müssen den *nname*-Wert des Servers mit der Datenbank kennen, auf die zugegriffen werden soll. Der *nname*-Wert des Servers muss im Client-Knotenverzeichnis als Parameter „server-*nname*“ mit dem Befehl CATALOG NETBIOS NODE katalogisiert werden.

Wenn der Knotenname *nname* auf dem Server-Knoten in einen neuen Namen geändert wird, müssen alle Clients, die auf Datenbanken dieses Servers zugreifen, diesen neuen Namen für den Server katalogisieren.

Zugehörige Referenzen:

- „GET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE MANAGER CONFIGURATION command“ in *Command Reference*

svcname - TCP/IP-Servicename

Konfigurationstyp Datenbankmanager

Gilt für

- Datenbankserver mit lokalen und fernen Clients
- Datenbankserver mit lokalen Clients
- Partitionierten Datenbankserver mit lokalen und fernen Clients

Parametertyp Konfigurierbar

Standardwert Null

Dieser Parameter enthält den Namen des TCP/IP-Ports, an dem ein Datenbankserver auf Datenübertragungen von fernen Clientknoten wartet. Dieser Name muss der erste zweier aufeinander folgender Ports sein, die für die Verwendung durch den Datenbankmanager reserviert sind. Der zweite Port wird zur Verarbeitung von Unterbrechungsanforderungen von Clients mit einer älteren Programmversion verwendet.

Damit Verbindungsanforderungen von einem Datenbankclient mit Hilfe von TCP/IP empfangen werden können, muss der Datenbankserver an einem Port empfangsbereit sein, der diesem Server zugewiesen wurde. Der Systemadministrator für den Datenbankserver muss einen Port (Nummer n) reservieren und den zugeordneten TCP/IP-Servicenamen in der Servicedatei auf dem Server definieren. Wenn der Datenbankserver Anforderungen von Clients mit einer älteren Programmversion unterstützen soll, muss ein zweiter Port (Nummer $n+1$ für Unterbrechungsanforderungen) in der Servicedatei auf dem Server definiert werden.

Der Port (Nummer n) des Datenbankservers und sein TCP/IP-Servicename müssen in der Servicedatei auf dem Datenbankclient definiert werden. Bei Clients mit einer älteren Programmversion muss außerdem der Unterbrechungsport (Nummer $n+1$) in der Servicedatei des Clients definiert werden.

Auf UNIX-basierten Systemen befindet sich die Servicedatei in folgendem Verzeichnis: `/etc/services`

Der Parameter *svcname* sollte auf den Servicennamen gesetzt werden, der dem Hauptverbindungsport zugeordnet ist, so dass der Datenbankserver nach dem Start ermitteln kann, an welchem Port er für eingehende Verbindungsanforderungen empfangsbereit sein muss. Wenn Sie einen Client mit einer älteren Programmversion unterstützen oder verwenden, wird der Servicename für den Unterbrechungsport nicht in der Konfigurationsdatei gespeichert. Die Nummer des Unterbrechungsports kann anhand der Nummer des Hauptverbindungsports ermittelt werden ($\text{unterbrechungsportnummer} = \text{hauptverbindungsport} + 1$).

Zugehörige Referenzen:

- „GET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE MANAGER CONFIGURATION command“ in *Command Reference*

DB2 Discovery

Die folgenden Parameter dienen zur Einrichtung von DB2 Discovery:

- „discover - Discovery-Modus “
- „discover_db - Discovery-Unterstützung für diese Datenbank “
- „discover_inst - Discovery-Serverinstanz “ auf Seite 553

discover - Discovery-Modus

Konfigurationstyp Datenbankmanager

Gilt für

- Datenbankserver mit lokalen und fernen Clients
- Client
- Datenbankserver mit lokalen Clients
- Partitionierten Datenbankserver mit lokalen und fernen Clients

Parametertyp Konfigurierbar

Standardwert [Bereich] SEARCH [DISABLE, KNOWN, SEARCH]

Aus der Perspektive eines Clients gilt einer der folgenden Fälle:

- Wenn SEARCH für *discover* angegeben ist, kann der Client Discovery-Anforderungen mit dem Parameter SEARCH zum Suchen nach DB2-Serversystemen im Netzwerk absetzen. Discovery mit dem Parameter SEARCH stellt eine Implikation der durch Discovery mit dem Parameter KNOWN bereitgestellten Funktionalität zur Verfügung. Wenn für *discover* SEARCH angegeben ist, können vom Client sowohl Discovery-Anforderungen mit dem Parameter SEARCH als auch Discovery-Anforderungen mit dem Parameter KNOWN abgesetzt werden.
- Wenn für *discover* KNOWN angegeben ist, können vom Client nur Discovery-Anforderungen mit dem Parameter KNOWN abgesetzt werden. Durch Angabe einiger Verbindungsinformationen für den Verwaltungsserver auf einem bestimmten System werden alle Instanz- und Datenbankinformationen auf dem DB2-System an den Client zurückgegeben.
- Wenn für *discover* DISABLE angegeben ist, ist Discovery auf dem Client inaktiviert.

Der Standard-Discovery-Modus ist SEARCH.

Zugehörige Referenzen:

- „GET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE MANAGER CONFIGURATION command“ in *Command Reference*

discover_db - Discovery-Unterstützung für diese Datenbank

Konfigurationstyp Datenbank

Parametertyp Online konfigurierbar

Weitergabeklasse Sofort

Standardwert [Bereich] Enable [Disable, Enable]

Mit diesem Parameter kann verhindert werden, dass Informationen zu einer Datenbank an einen Client zurückgegeben werden, wenn eine Discovery-Anforderung auf dem Server empfangen wird.

Standardmäßig ist dieser Parameter so eingestellt, dass Discovery-Unterstützung für diese Datenbank aktiviert ist.

Durch Ändern dieses Parameterwerts in „Disable“ können Datenbanken, die sensible Daten enthalten, vor dem Discovery-Prozess verdeckt werden. Diese Maßnahme kann zusätzlich zu anderen Datenbanksicherheitsfunktionen für die Datenbank ergriffen werden.

Zugehörige Referenzen:

- „GET DATABASE CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE CONFIGURATION command“ in *Command Reference*

discover_inst - Discovery-Serverinstanz

Konfigurationstyp Datenbankmanager

Gilt für

- Datenbankserver mit lokalen und fernen Clients
- Client
- Datenbankserver mit lokalen Clients
- Partitionierten Datenbankserver mit lokalen und fernen Clients

Parametertyp Online konfigurierbar

Weitergabeklasse Sofort

Standardwert [Bereich] ENABLE [ENABLE, DISABLE]

Dieser Parameter gibt an, ob diese Instanz von DB2 Discovery aufgespürt werden kann. Der Standardwert, "enable", gibt an, dass die Instanz aufgespürt werden kann, der Wert "disable" dagegen verhindert, dass die Instanz aufgespürt wird.

Zugehörige Referenzen:

- „GET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE MANAGER CONFIGURATION command“ in *Command Reference*

Umgebung mit partitionierter Datenbank

Die folgenden Gruppen von Parametern stellen Informationen zum Parallelbetrieb und zu Umgebungen mit partitionierten Datenbanken bereit:

- „Kommunikation“ auf Seite 554
- „Parallelverarbeitung“ auf Seite 559

Kommunikation

Die folgenden Parameter stellen Informationen zur Kommunikation in der Umgebung mit partitionierten Datenbanken bereit:

- „conn_elapse - Antwortzeit für Verbindung “
- „fcm_num_buffers - Anzahl FCM-Puffer “
- „fcm_num_channels - Anzahl FCM-Kanäle “ auf Seite 555
- „max_connretries - Anzahl Wiederholungen für Verbindungsaufbau zum Knoten “ auf Seite 556
- „max_time_diff - Maximale Zeitdifferenz zwischen Knoten “ auf Seite 557
- „start_stop_time - Zeitlimit für DB2START und DB2STOP “ auf Seite 557

conn_elapse - Antwortzeit für Verbindung

Konfigurationstyp	Datenbankmanager
Gilt für	Partitionierten Datenbankserver mit lokalen und fernen Clients
Parametertyp	Online konfigurierbar
Weitergabeklasse	Sofort
Standardwert [Bereich]	10 [0–100]
Maßeinheit	Sekunden

Dieser Parameter gibt an, innerhalb wie viel Sekunden eine TCP/IP-Verbindung zwischen zwei Datenbankpartitionsservern aufgebaut werden muss. Wird der Verbindungsaufbau innerhalb der angegebenen Zeit erfolgreich durchgeführt, wird der Datenaustausch freigegeben. Wird die Verbindung nicht rechtzeitig aufgebaut, wird ein weiterer Versuch zum Verbindungsaufbau durchgeführt. Kommt innerhalb der im Parameter *max_connretries* angegebenen Anzahl von Neuversuchen keine Verbindung zustande, wird eine Fehlermeldung ausgegeben.

Zugehörige Referenzen:

- „max_connretries - Anzahl Wiederholungen für Verbindungsaufbau zum Knoten “ auf Seite 556
- „GET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE MANAGER CONFIGURATION command“ in *Command Reference*

fcm_num_buffers - Anzahl FCM-Puffer

Konfigurationstyp	Datenbankmanager
Gilt für	• Datenbankserver mit lokalen und fernen Clients • Datenbankserver mit lokalen Clients • Partitionierten Datenbankserver mit lokalen und fernen Clients
Parametertyp	Online konfigurierbar
Weitergabeklasse	Sofort

Standardwert [Bereich]

32-Bit-Plattformen

Automatic [Automatic, 128 — 65 300]

64-Bit-Plattformen

Automatic [Automatic, 128 — 524 288]

- Datenbankserver mit lokalen und fernen Clients: Standardwert ist 1 024.
- Datenbankserver mit lokalen Clients: Standardwert ist 512.
- Partitionierter Datenbankserver mit lokalen und fernen Clients: Standardwert ist 4 096.

Auf Datenbanksystemen mit einer Partition wird dieser Parameter nur dann verwendet, wenn der Parameter *intra_parallel* aktiv ist.

Dieser Parameter gibt die Anzahl von 4-KB-Puffern an, die sowohl zwischen den als auch innerhalb der Datenbankserver für interne Kommunikation (Nachrichten) verwendet werden.

Wenn Sie über mehrere logische Knoten auf derselben Maschine verfügen, kann es erforderlich sein, den Wert für diesen Parameter zu erhöhen. Außerdem kann es erforderlich sein, den Wert für diesen Parameter zu erhöhen, wenn wegen der Anzahl von Benutzern im System, wegen der Anzahl von Datenbankpartitions- servern im System oder wegen der Komplexität der Anwendungen die verfügbaren Nachrichtenpuffer nicht ausreichen.

Wenn Sie mehrere logische Knoten verwenden, wird ein Pool mit der im Parameter *fcm_num_buffers* angegebenen Anzahl von Puffern von allen logischen Knoten auf derselben Maschine gemeinsam benutzt. Die Größe des Pufferpools wird durch die Multiplikation des Werts von *fcm_num_buffers* mit der Anzahl der logischen Knoten auf dieser physischen Maschine ermittelt. Überprüfen Sie daher den verwendeten Wert. Überlegen Sie, wie viele FCM-Puffer auf der Maschine (oder den Maschinen) mit mehreren logischen Knoten insgesamt zugeordnet werden.

Zugehörige Konzepte:

- „FCM-Kommunikation“ in *Systemverwaltung: Implementierung*

Zugehörige Referenzen:

- „GET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE MANAGER CONFIGURATION command“ in *Command Reference*

fcm_num_channels - Anzahl FCM-Kanäle

Konfigurationstyp

Datenbankmanager

Gilt für

- Datenbankserver mit lokalen und fernen Clients
- Datenbankserver mit lokalen Clients

Parametertyp	• Partitionierten Datenbankserver mit lokalen und fernen Clients • Satellitendatenbankserver mit lokalen Clients
Weitergabeklasse	Online konfigurierbar
Standardwert [Bereich]	Sofort

UNIX-Plattformen (32 Bit)

Automatic - mit den Anfangswerten 256, 512, 2 048 [128 — 120 000]

UNIX-Plattformen (64 Bit)

Automatic - mit den Anfangswerten 256, 512, 2 048 [128 — 524 288]

Windows (32 Bit)

Automatic - mit dem Anfangswert 10 000 [128 — 120 000]

Windows (64 Bit)

Automatic - mit den Anfangswerten 256, 512, 2 048 [128 — 524 288]

- Für einen Datenbankserver mit lokalen und fernen Clients ist der Anfangswert 512.
- Für einen Datenbankserver mit lokalen Clients ist der Anfangswert 256.
- Für einen partitionierten Datenbankserver mit lokalen und fernen Clients ist der Anfangswert 2 048.

In nicht partitionierten Datenbanksystemen muss der Parameter *intra_parallel* aktiviert werden, bevor der Parameter *fcm_num_channels* verwendet werden kann.

Ein FCM-Kanal stellt einen logischen Kommunikationsendpunkt zwischen EDUs (Engine Dispatchable Units) dar, die in der DB2-Steuerkomponente ausgeführt werden. Sowohl Steuerungsflüsse (Anforderung und Antwort) als auch Datenflüsse (Daten aus Tabellenwarteschlangen) können Daten zwischen Partitionen über Kanäle übertragen. Mit diesem Parameter wird die Anzahl der FCM-Kanäle für jede Datenbankpartition angegeben.

Zugehörige Referenzen:

- „ch_free - Channels Currently Free monitor element“ in *Systemmonitor Handbuch und Referenz*
- „ch_free_bottom - Minimum Channels Free monitor element“ in *Systemmonitor Handbuch und Referenz*

max_connretries - Anzahl Wiederholungen für Verbindungsaufbau zum Knoten

Konfigurationstyp	Datenbankmanager
Gilt für	Partitionierten Datenbankserver mit lokalen und fernen Clients
Parametertyp	Online konfigurierbar

Weitergabeklasse	Sofort
Standardwert [Bereich]	5 [0–100]

Wenn der Verbindungsaufbau zwischen zwei Datenbankpartitionsservern fehlschlägt (z. B. bei Erreichen des im Parameter *conn_elapse* angegebenen Werts), gibt *max_connretries* an, wie viele Wiederholungen durchgeführt werden dürfen, um die Verbindung zu einem Datenbankpartitionsserver herzustellen. Bei Überschreitung des für diesen Parameter angegebenen Werts wird eine Fehlermeldung zurückgegeben.

Zugehörige Referenzen:

- „conn_elapse - Antwortzeit für Verbindung ” auf Seite 554
- „GET DATABASE MANAGER CONFIGURATION command” in *Command Reference*
- „RESET DATABASE MANAGER CONFIGURATION command” in *Command Reference*
- „UPDATE DATABASE MANAGER CONFIGURATION command” in *Command Reference*

max_time_diff - Maximale Zeitdifferenz zwischen Knoten

Konfigurationstyp	Datenbankmanager
Gilt für	Partitionierten Datenbankserver mit lokalen und fernen Clients
Parametertyp	Konfigurierbar
Standardwert [Bereich]	60 [1–1 440]
Maßeinheit	Minuten

Jeder Datenbankpartitionsserver verfügt über eine eigene Systemuhr. Dieser Parameter gibt die maximale Zeitdifferenz (in Minuten) an, die zwischen den in der Knotenkonfigurationsdatei aufgelisteten Datenbankpartitionsservern zulässig ist.

Sind einer Transaktion zwei oder mehr Datenbankpartitionsserver zugeordnet, deren Systemzeiten sich um mehr als diese Zeitdifferenz unterscheiden, wird die Transaktion zurückgewiesen und ein SQLCODE-Wert zurückgegeben. (Die Transaktion wird nur zurückgewiesen, wenn eine Datenänderung mit ihr verbunden ist.)

DB2 verwendet die *Weltzeit* (UTC - Coordinated Universal Time), damit unterschiedliche Zeitzonen beim Festlegen dieses Parameters nicht ins Gewicht fallen. Die Weltzeit ist identisch mit der Westeuropäischen Zeit (Greenwich Mean Time).

Zugehörige Referenzen:

- „GET DATABASE MANAGER CONFIGURATION command” in *Command Reference*
- „RESET DATABASE MANAGER CONFIGURATION command” in *Command Reference*
- „UPDATE DATABASE MANAGER CONFIGURATION command” in *Command Reference*

start_stop_time - Zeitlimit für DB2START und DB2STOP

Konfigurationstyp	Datenbankmanager

Gilt für	Datenbankserver mit lokalen und fernen Clients
Parametertyp	Online konfigurierbar
Weitergabeklasse	Sofort
Standardwert [Bereich]	10 [1 - 1.440]
Maßeinheit	Minuten

Mit diesem Parameter wird die Zeit (in Minuten) angegeben, innerhalb deren alle Datenbankpartitionsserver auf den Befehl DB2START oder DB2STOP reagieren müssen. Außerdem wird er als Zeitlimit für ADD DBPARTITIONNUM-Operationen verwendet.

Datenbankpartitionsserver, die nicht innerhalb der angegebenen Zeit auf einen Befehl DB2START reagieren, senden eine Nachricht an das Fehlerprotokoll db2start, das sich im Unterverzeichnis log des Unterverzeichnisses sql1ib im Ausgangsverzeichnis für die Instanz befindet. Vor dem Neustart dieser Knoten sollten Sie auf diesen Knoten einen Befehl DB2STOP absetzen.

Datenbankpartitionsserver, die nicht innerhalb der angegebenen Zeit auf einen Befehl DB2STOP reagieren, senden eine Nachricht an das Fehlerprotokoll db2stop, das sich im Unterverzeichnis log des Unterverzeichnisses sql1ib im Benutzerverzeichnis für die Instanz befindet. Sie können db2stop entweder für jeden oder einmal für alle nicht reagierenden Datenbankpartitionsserver absetzen. (Die bereits gestoppten Server melden, dass sie bereits gestoppt sind.)

Wenn eine db2start- oder eine db2stop-Operation in einer Mehrpartitionsdatenbank nicht innerhalb des Zeitwerts ausgeführt wird, der durch den Konfigurationsparameter *start_stop_timeout* des Datenbankmanagers angegeben wurde, werden die Datenbankpartitionen intern abgebrochen, für die das zulässige Zeitlimit überschritten wurde. Bei Umgebungen mit vielen Datenbankpartitionen und einem geringen Wert für *start_stop_timeout* kann es zu diesem Verhalten kommen. Erhöhen Sie den Wert für *start_stop_timeout*, um diesem Verhalten entgegenzutreten.

Beim Hinzufügen einer neuen Datenbankpartition mit Hilfe des Befehls DB2START, START DATABASE MANAGER bzw. ADD DBPARTITIONNUM muss die Operation zum Hinzufügen der Datenbankpartition bestimmen, ob die einzelnen Datenbanken in der Instanz für dynamischen Speicher eingerichtet sind. Dies geschieht durch Kommunikation mit der Katalogpartition für jede einzelne Datenbank. Wenn der dynamische Speicher aktiviert ist, werden die Speicherpfaddefinitionen im Rahmen dieser Kommunikation abgerufen. Ähnlich gilt für den Fall, dass Tabellenbereiche für temporäre Tabellen zusammen mit den Datenbankpartitionen zu erstellen sind, dass die Operation möglicherweise mit einem anderen Datenbankpartitionsserver kommunizieren muss, um die Tabellenbereichsdefinitionen für die Datenbankpartitionen abzurufen, die sich auf diesem Server befinden. Diese Faktoren müssen bei der Festlegung des Werts für den Parameter *start_stop_time* berücksichtigt werden.

Zugehörige Referenzen:

- „ADD DBPARTITIONNUM command“ in *Command Reference*
- „GET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*

- „UPDATE DATABASE MANAGER CONFIGURATION command“ in *Command Reference*

Parallelverarbeitung

Die folgenden Parameter stellen Informationen zur Parallelverarbeitung bereit:

- „intra_parallel - Partitionsinterne Parallelität aktivieren “
- „max_querydegree - Maximaler Grad der Parallelität bei Abfragen “

intra_parallel - Partitionsinterne Parallelität aktivieren

Konfigurationstyp

Datenbankmanager

Gilt für

- Datenbankserver mit lokalen und fernen Clients
- Datenbankserver mit lokalen Clients
- Partitionierten Datenbankserver mit lokalen und fernen Clients

Parametertyp

Konfigurierbar

Standardwert [Bereich]

NO (0) [SYSTEM (-1), NO (0), YES (1)]

Der Wert -1 bewirkt, dass der Parameter auf „YES“ oder „NO“ gesetzt wird, abhängig von der Hardware, auf welcher der Datenbankmanager ausgeführt wird.

Dieser Parameter gibt an, ob der Datenbankmanager partitionsinterne Parallelität verwenden kann.

Ist dieser Parameter auf "YES" gesetzt, können die durch die Parallelverarbeitung erreichten Leistungssteigerungen unter anderem bei Datenbankabfragen und der Indexerstellung genutzt werden.

Anmerkung: Wenn Sie diesen Parameterwert ändern, werden Pakete möglicherweise erneut an die Datenbank gebunden und es kann zu Leistungseinbußen kommen.

Zugehörige Referenzen:

- „max_querydegree - Maximaler Grad der Parallelität bei Abfragen “ auf Seite 559
- „GET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE MANAGER CONFIGURATION command“ in *Command Reference*

max_querydegree - Maximaler Grad der Parallelität bei Abfragen

Konfigurationstyp

Datenbankmanager

Gilt für

- Datenbankserver mit lokalen und fernen Clients
- Datenbankserver mit lokalen Clients

- Partitionierten Datenbankserver mit lokalen und fernen Clients

Parametertyp	Online konfigurierbar
Weitergabeklasse	Anweisungsgrenzwert
Standardwert [Bereich]	-1 (ANY) [ANY, 1 — 32 767] (ANY bedeutet vom System ermittelt)

Dieser Parameter gibt den maximalen Grad partitionsinterner Parallelität an, der für SQL-Anweisungen verwendet wird, die auf dieser Instanz des Datenbankmanagers ausgeführt werden. Eine SQL-Anweisung verwendet bei ihrer Ausführung innerhalb einer Datenbankpartition nicht mehr als diese Anzahl paralleler Operationen. Der Konfigurationsparameter *intra_parallel* muss aktiviert sein, damit die Datenbankpartition die partitionsinterne Parallelität verwenden kann.

Der Standardwert für diesen Konfigurationsparameter lautet -1. Dieser Wert bedeutet, dass das System den vom Optimierungsprogramm festgelegten Parallelitätsgrad verwendet. Andernfalls wird der vom Benutzer angegebene Wert verwendet.

Anmerkung: Der Grad der Parallelität für eine SQL-Anweisung kann bei der Kompilierung der Anweisung mit Hilfe des Sonderregisters CURRENT DEGREE oder der Bindeoption DEGREE angegeben werden.

Der maximale Grad der Parallelität für eine aktive Anwendung bei Abfragen kann mit dem Befehl SET RUNTIME DEGREE geändert werden. Der zur Laufzeit tatsächlich verwendete Parallelitätsgrad ist der niedrigste der folgenden Werte:

- Konfigurationsparameter *max_querydegree*
- Parallelitätsgrad der Anwendung zur Laufzeit
- Parallelitätsgrad bei der Kompilierung der SQL-Anweisung

Eine Ausnahmesituation bezüglich der Festlegung des tatsächlichen Parallelitätsgrads bei Abfragen tritt beim Erstellen eines Index auf. Ist in diesem Fall die Parallelität *intra_parallel* aktiviert und die Tabelle so groß, dass die Verwendung mehrerer Prozessoren vorteilhaft wäre, wird beim Erstellen eines Index die Anzahl der Online-Prozessoren (maximal jedoch 6) plus 1 Prozessor verwendet. Die Angabe des anderen oben genannten Parameters, der Bindeoption oder des Sonderregisters hat keine Auswirkung.

Zugehörige Referenzen:

- „dft_degree - Standardgrad der Parallelität “ auf Seite 541
- „intra_parallel - Partitionsinterne Parallelität aktivieren “ auf Seite 559
- „Einschränkungen für den nativen XML-Datenspeicher“ in *XML-Handbuch*
- „GET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE MANAGER CONFIGURATION command“ in *Command Reference*

Instanzverwaltung

Eine Reihe von Parametern steht zur Verwaltung der Datenbankmanagerinstanzen zur Verfügung. Diese Parameter werden in folgende Kategorien unterteilt:

- „Diagnose“
- „Parameter für den Datenbanksystemmonitor“ auf Seite 564
- „Systemverwaltung“ auf Seite 566
- „Instanzverwaltung“ auf Seite 574

Diagnose

Mit den folgenden Parametern können die vom Datenbankmanager zur Verfügung gestellten Diagnoseinformationen gesteuert werden:

- „diaglevel - Aufzeichnungsebene bei Fehlerdiagnose “
- „diagpath - Verzeichnispfad für Diagnosedaten “ auf Seite 562
- „health_mon - Überwachung mit dem Diagnosemonitor “ auf Seite 563
- „notifylevel - Benachrichtigungsstufe “ auf Seite 563

diaglevel - Aufzeichnungsebene bei Fehlerdiagnose

Konfigurationstyp

Datenbankmanager

Gilt für

- Datenbankserver mit lokalen und fernen Clients
- Client
- Datenbankserver mit lokalen Clients
- Partitionierten Datenbankserver mit lokalen und fernen Clients

Parametertyp

Online konfigurierbar

Weitergabeklasse

Sofort

Standardwert [Bereich]

3 [0 — 4]

Durch diesen Parameter wird der Typ der Diagnosefehler festgelegt, die in der Datei 'db2diag.log' aufgezeichnet werden. Gültige Werte:

- 0 – Keine Aufzeichnung von Diagnosedaten
- 1 – Nur schwerwiegende Fehler
- 2 – Alle Fehler
- 3 – Alle Fehler und Warnungen
- 4 – Alle Fehler, Warnungen und Informationsnachrichten

Der Konfigurationsparameter *diagpath* wird zur Angabe des Verzeichnisses verwendet, in dem sich die Fehlerdatei, die Alertprotokolldatei und alle Speicherauszugsdateien befinden, die in Abhängigkeit vom Wert des Parameters *diaglevel* generiert werden.

Empfehlung: Sie können den Wert dieses Parameters erhöhen, um zusätzliche Fehlerbestimmungsdaten zu sammeln, die zur Lösung eines Problems beitragen können.

Zugehörige Referenzen:

- „diagpath - Verzeichnispfad für Diagnosedaten “ auf Seite 562

- „GET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE MANAGER CONFIGURATION command“ in *Command Reference*

diagpath - Verzeichnispfad für Diagnosedaten

Konfigurationstyp	Datenbankmanager
Gilt für	• Datenbankserver mit lokalen und fernen Clients • Client • Datenbankserver mit lokalen Clients • Partitionierten Datenbankserver mit lokalen und fernen Clients
Parametertyp	Online konfigurierbar
Weitergabeklasse	Sofort
Standardwert [Bereich]	NULL [beliebiger gültiger Pfadname]

Mit diesem Parameter können Sie einen vollständig qualifizierten Pfad für DB2-Diagnosedaten angeben. Im angegebenen Verzeichnis können abhängig von Ihrer Plattform Speicherauszugsdateien, Trapdateien, eine Fehlerprotokolldatei, eine Benachrichtigungsdatei und eine Alertprotokolldatei gespeichert werden.

Wenn dieser Parameter den Wert "Null" hat, werden die Diagnoseinformationen in Dateien eines der folgenden Verzeichnisse (bzw. Ordner) geschrieben:

- Für unterstützte Windows-Umgebungen:
 - Wenn die Umgebungsvariable bzw. das Schlüsselwort DB2INSTPROF **nicht** festgelegt ist, werden die Informationen in x:\SQLLIB\DB2INSTANCE geschrieben. Dabei ist x:\SQLLIB die Laufwerkangabe und das Verzeichnis, die in der Umgebungsvariablen bzw. dem Schlüsselwort DB2PATH angegeben sind, und DB2INSTANCE ist der Name des Instanzeigners.

Anmerkung: Das Verzeichnis muss nicht SQLLIB heißen.

- Wenn die Umgebungsvariable bzw. das Schlüsselwort DB2INSTPROF festgelegt ist, werden die Informationen in x:\DB2INSTPROF\DB2INSTANCE geschrieben. Dabei ist DB2INSTPROF der Name des Instanzprofilverzeichnisses und DB2INSTANCE der Name der Instanz.
- In auf UNIX basierenden Umgebungen gilt Folgendes: INSTHOME/sqllib/db2dump, wobei INSTHOME das Ausgangsverzeichnis (Home) der Instanz ist.

Empfehlung: Verwenden Sie den Standardwert, oder geben Sie eine zentrale Speicherposition für die Diagnosedaten mehrerer Instanzen an.

In einer Umgebung mit partitionierten Datenbanken muss der angegebene Pfad auf einem gemeinsam benutzten Dateisystem angelegt sein.

Zugehörige Referenzen:

- „diaglevel - Aufzeichnungsebene bei Fehlerdiagnose “ auf Seite 561
- „GET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*

- „RESET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE MANAGER CONFIGURATION command“ in *Command Reference*

health_mon - Überwachung mit dem Diagnosemonitor

Konfigurationstyp	Datenbankmanager
Parametertyp	Online konfigurierbar
Weitergabeklasse	Sofort
Standardwert [Bereich]	On [On; Off]

Zugehörige Parameter

Mit diesem Parameter können Sie angeben, ob Sie anhand verschiedener Diagnoseanzeiger eine Instanz, seine zugeordneten Datenbanken und Datenbankobjekte überwachen wollen. Wenn *health_mon* aktiviert ist (Standardeinstellung), erfasst ein Agent Informationen zum ordnungsgemäßen Betrieb der ausgewählten Objekte. Wenn auf der Basis der von Ihnen festgelegten Schwellenwerte der Betrieb eines Objekts als nicht ordnungsgemäß eingestuft wird, können automatisch Benachrichtigungen versendet und Aktionen unternommen werden. Wenn *health_mon* inaktiviert ist, wird der ordnungsgemäße Betrieb von Objekten nicht überwacht.

Sie können mit der Diagnosezentrale oder dem Befehlszeilenprozessor die Instanz und die Datenbankobjekte auswählen, die Sie überwachen wollen. Basierend auf den vom Diagnosemonitor erfassten Daten können Sie auch angeben, wann Benachrichtigungen gesendet und welche Aktionen unternommen werden sollen.

Zugehörige Referenzen:

- „GET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE MANAGER CONFIGURATION command“ in *Command Reference*

notifylevel - Benachrichtigungsstufe

Konfigurationstyp	Datenbankmanager

Gilt für

- Datenbankserver mit lokalen und fernen Clients
- Client
- Datenbankserver mit lokalen Clients
- Partitionierten Datenbankserver mit lokalen und fernen Clients

Parametertyp	Online konfigurierbar
Weitergabeklasse	Sofort
Standardwert [Bereich]	3 [0 — 4]

Dieser Parameter gibt den Typ der Hinweismeldungen zur Systemverwaltung an, die in das Protokoll mit Benachrichtigungen für die Systemverwaltung geschrieben werden. Auf UNIX-Plattformen ist das Protokoll mit Benachrichtigungen für die

Systemverwaltung eine Textdatei mit dem Namen *instanz.nfy*. Unter Windows werden alle Hinweismeldungen für die Systemverwaltung in das Ereignisprotokoll geschrieben. Die Fehler können von DB2, dem Diagnosemonitor, den Capture- und Apply-Programmen sowie von Benutzeranwendungen geschrieben werden.

Gültige Werte für diesen Parameter sind:

- 0 — Keine Aufzeichnung von Hinweismeldungen zur Systemverwaltung.
(Diese Einstellung wird nicht empfohlen.)
- 1 — Schwer wiegende oder nicht behebbare Fehler. Es werden nur schwer wiegende und nicht behebbare Fehler protokolliert. Zur Behebung einiger dieser Fehler benötigen Sie möglicherweise Unterstützung vom DB2-Service.
- 2 — Sofortmaßnahmen erforderlich. Es werden Bedingungen protokolliert, die Sofortmaßnahmen durch den Systemadministrator oder den Datenbankadministrator erfordern. Wenn die Bedingung nicht behoben wird, könnte dies zu einem schwer wiegenden Fehler führen. Auf dieser Ebene können ebenfalls äußerst wichtige, nicht fehlerbezogene Aktivitäten (wie zum Beispiel eine Recovery) protokolliert werden. Auf dieser Ebene werden Alarmmeldungen des Diagnosemonitors erfasst.
- 3 — Wichtige Informationen, keine Sofortmaßnahmen erforderlich. Es werden Bedingungen protokolliert, die nicht bedrohlich sind und keine Sofortmaßnahmen erfordern, die jedoch auf ein nicht optimales System hinweisen können. Auf dieser Ebene werden Alarmmeldungen, Warnungen und Informationsmeldungen des Diagnosemonitors erfasst.
- 4 — Informationsmeldungen.

Das Protokoll mit Benachrichtigungen für die Systemverwaltung enthält Nachrichten mit Werten bis einschließlich dem Wert von *notifylevel*. Wenn der Parameter *notifylevel* beispielsweise auf 3 gesetzt ist, enthält dieses Protokoll Nachrichten der Ebenen 1, 2 und 3.

Eine Benutzeranwendung muss die API `db2AdminMsgWrite` aufrufen, um in die Benachrichtigungsdatei oder das Windows-Ereignisprotokoll schreiben zu können.

Empfehlung: Sie können den Wert dieses Parameters erhöhen, um zusätzliche Fehlerbestimmungsdaten zu sammeln, die zur Lösung eines Problems beitragen können. Wenn der Diagnosemonitor Benachrichtigung an die in seiner Konfiguration angegebenen Ansprechpartner senden soll, müssen Sie für den Parameter *notifylevel* einen Wert von 2 oder höher angeben.

Zugehörige Referenzen:

- „GET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE MANAGER CONFIGURATION command“ in *Command Reference*

Parameter für den Datenbanksystemmonitor

Mit den folgenden Parametern können verschiedene Funktionen des Datenbanksystemmonitors gesteuert werden:

- „dft_monswitches - Monitorschalter des Standarddatenbanksystems “ auf Seite 565

dft_monswitches - Monitorschalter des Standard-datenbanksystems

Konfigurationstyp	Datenbankmanager
Gilt für	• Datenbankserver mit lokalen und fernen Clients • Datenbankserver mit lokalen Clients • Partitionierten Datenbankserver mit lokalen und fernen Clients
Parametertyp	Online konfigurierbar
Weitergabeklasse	Sofort
Standardwert	Alle Schalter ausgeschaltet außer dem standardmäßig eingeschalteten <i>dft_mon_timestamp</i>

Dieser Parameter ist insofern außergewöhnlich, als Sie mit ihm eine Reihe von Schaltern einstellen können, die intern jeweils als ein Bit des Parameters dargestellt werden. Sie können jeden dieser Schalter unabhängig aktualisieren, indem Sie die folgenden Parameter festlegen:

dft_mon_uow	Standardwert des UOW-Schalters (Unit of Work - Arbeitseinheit) von Snapshot Monitor
dft_mon_stmt	Standardwert des Schalters für SQL-Anweisungen von Snapshot Monitor
dft_mon_table	Standardwert des Schalters für Tabellen von Snapshot Monitor
dft_mon_bufpool	Standardwert des Schalters für Pufferpool von Snapshot Monitor
dft_mon_lock	Standardwert des Schalters für Sperren von Snapshot Monitor
dft_mon_sort	Standardwert des Schalters für Sortiervorgänge von Snapshot Monitor
dft_mon_timestamp	Standardwert des Schalters für die Zeitmarke des Überwachungsprogramms für Momentaufnahme

Empfehlung: Jeder auf ON gesetzte Schalter (außer *dft_mon_timestamp*) weist den Datenbankmanager an, die zu diesem Schalter gehörenden Monitordaten zu sammeln. Das Erfassen zusätzlicher Monitordaten erhöht den Systemaufwand des Datenbankmanagers, wodurch die Systemleistung beeinträchtigt werden kann. Das Setzen des Schalters *dft_mon_timestamp* auf OFF wird wichtig, sobald sich die CPU-Auslastung 100% nähert. Wenn dies eintritt, wird die zur Ausgabe von Zeitmarken benötigte CPU-Zeit drastisch erhöht. Wenn der Zeitmarkenschalter inaktiviert ist, reduziert dies zudem erheblich den Gesamtaufwand für andere Daten in der Monitorschaltersteuerung.

Alle Überwachungsanwendungen erhalten diese Standardeinstellungen für die Schalter, wenn die Anwendung die erste Überwachungsanforderung absetzt (z. B. einen Schalter einstellt, den Ereignismonitor aktiviert, eine Momentaufnahme macht). In der Konfigurationsdatei sollten Sie einen Schalter nur dann aktivieren, wenn Sie Daten sammeln möchten, sobald der Datenbankmanager gestartet wird.

(Andernfalls kann jede Überwachungsanwendung ihre eigenen Schalter einstellen, und die gesammelten Daten beziehen sich dann auf den Zeitpunkt, zu dem die Schalter eingestellt wurden.)

Zugehörige Referenzen:

- „GET MONITOR SWITCHES command“ in *Command Reference*
- „GET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE MANAGER CONFIGURATION command“ in *Command Reference*

Systemverwaltung

Die folgenden Parameter beziehen sich auf die Systemverwaltung:

- „comm_bandwidth - Kommunikationsbandbreite “
- „cpuspeed - CPU-Geschwindigkeit “ auf Seite 567
- „dft_account_str - Standardzeichenfolge für Abrechnung “ auf Seite 568
- „federated - Unterstützung für Systeme föderierter Datenbanken “ auf Seite 569
- „jdk_path - Installationspfad für das Software Developer's Kit für Java “ auf Seite 569
- „nodetype - Knotenart des Systems “ auf Seite 570
- „numdb - Maximale Anzahl gleichzeitig aktiver Datenbanken einschließlich Host- und iSeries-Datenbanken “ auf Seite 570
- „tp_mon_name - Name des Transaktionsprozessormonitors “ auf Seite 571
- „util_impact_lim - Richtlinie für Instanzauslastungswirkung “ auf Seite 573

comm_bandwidth - Kommunikationsbandbreite

Konfigurationstyp	Datenbankmanager
Gilt für	Partitionierten Datenbankserver mit lokalen und fernen Clients
Parametertyp	Online konfigurierbar
Weitergabeklasse	Anweisungsgrenzwert
Standardwert [Bereich]	-1 [.1 – 100 000]
	Durch den Wert -1 wird der Parameter auf den Standardwert zurückgesetzt. Der Standardwert wird abhängig davon berechnet, ob ein Hochgeschwindigkeitsschalter verwendet wird.
Maßeinheit	Megabyte pro Sekunde

Der Wert, der (in MB pro Sekunde) für die Übertragungsbandbreite berechnet wird, wird vom Abfrageoptimierungsprogramm zur Abschätzung des Aufwands für bestimmte Operationen zwischen den Datenbankpartitionsservern eines partitionierten Datenbanksystems herangezogen. Das Optimierungsprogramm modelliert nicht die Kosten der Datenfernverarbeitung zwischen einem Client und einem Server. Dieser Parameter sollte daher nur die nominale Bandbreite zwischen den Datenbankpartitionsservern darstellen.

Sie können diesen Wert explizit festlegen, um ein Modell einer Produktionsumgebung auf Ihrem Testsystem zu erstellen oder die Auswirkungen einer Hardwareaufrüstung zu bewerten.

Empfehlung: Sie sollten diesen Parameter nur anpassen, wenn Sie ein Modell einer anderen Umgebung erstellen wollen.

Die Übertragungsbandbreite wird vom Optimierungsprogramm bei der Bestimmung der Zugriffspfade verwendet. Wenn Sie diesen Parameter geändert haben, sollten Sie für die Anwendungen eventuell einen Rebind durchführen (mit dem Befehl REBIND PACKAGE).

Zugehörige Referenzen:

- „GET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE MANAGER CONFIGURATION command“ in *Command Reference*

cpuspeed - CPU-Geschwindigkeit

Konfigurationstyp	Datenbankmanager
Gilt für	• Datenbankserver mit lokalen und fernen Clients • Datenbankserver mit lokalen Clients • Partitionierten Datenbankserver mit lokalen und fernen Clients
Parametertyp	Online konfigurierbar
Weitergabeklasse	Anweisungsgrenzwert
Standardwert [Bereich]	-1 [1×10^{-10} - 1] Der Wert -1 gibt an, dass der Wert dieses Parameters gemäß den Ergebnissen eines Messprogramms neu eingestellt wird.
Maßeinheit	Sekunden

Die CPU-Geschwindigkeit (in Millisekunden pro Instruktion) wird vom Abfrageoptimierungsprogramm verwendet, um den Aufwand für die Ausführung bestimmter Operationen zu berechnen. Der Wert dieses Parameters wird automatisch bei der Installation des Datenbankmanagers auf der Grundlage der Ausgabe eines Programms zum Messen der CPU-Geschwindigkeit festgelegt. Dieses Programm wird ausgeführt, wenn Benchmark-Ergebnisse aus einem der folgenden Gründe nicht verfügbar sind:

- Auf der Plattform wird die Datei db2spec.dat nicht unterstützt.
- Die Datei db2spec.dat wird **nicht** gefunden.
- Die Daten für IBM RISC System/6000 Modell 530H werden in der Datei nicht gefunden.
- Die Daten für Ihr System werden in der Datei nicht gefunden.

Sie können diesen Wert explizit festlegen, um ein Modell einer Produktionsumgebung auf Ihrem Testsystem zu erstellen oder die Auswirkungen einer Hardwareaufrüstung zu bewerten. Wenn der Wert auf -1 gesetzt wird, wird *cpuspeed* erneut berechnet.

Empfehlung: Sie sollten diesen Parameter nur anpassen, wenn Sie ein Modell einer anderen Umgebung erstellen wollen.

Der Wert für die CPU-Geschwindigkeit wird vom Optimierungsprogramm bei der Bestimmung von Zugriffspfaden verwendet. Wenn Sie diesen Parameter geändert haben, sollten Sie für die Anwendungen eventuell einen Rebind durchführen (mit dem Befehl REBIND PACKAGE).

Zugehörige Referenzen:

- „GET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE MANAGER CONFIGURATION command“ in *Command Reference*

dft_account_str - Standardzeichenfolge für Abrechnung

Konfigurationstyp Datenbankmanager

Gilt für

- Datenbankserver mit lokalen und fernen Clients
- Client
- Datenbankserver mit lokalen Clients
- Partitionierten Datenbankserver mit lokalen und fernen Clients

Parametertyp Online konfigurierbar

Weitergabeklasse Sofort

Standardwert [Bereich] NULL [beliebige gültige Zeichenfolge]

Bei jeder Verbindungsanforderung durch eine Anwendung wird eine Abrechnungszeichenfolge vom Anwendungsrequester an einen DRDA-Anwendungsserver gesendet, die aus einem von DB2 Connect erstellten Präfix und einem vom Benutzer eingegebenen Suffix besteht. Anhand dieser Abrechnungsdaten kann ein Systemadministrator die Ressourcennutzung für jeden Benutzerzugriff bestimmen.

Anmerkung: Dieser Parameter ist nur für DB2 Connect gültig.

Das Suffix wird vom Anwendungsprogramm, das die API `sqlsact()` aufruft, oder dem Benutzer, der die Umgebungsvariable `DB2ACCOUNT` definiert, übergeben. Wenn von der API oder der Umgebungsvariablen kein Suffix bereitgestellt wird, verwendet DB2 Connect den Wert dieses Parameters als Standardsuffix. Dieser Parameter ist insbesondere für Datenbankclients früherer Versionen (alle vor Version 2) nützlich, die nicht über Funktionen zum Senden einer Abrechnungszeichenfolge an DB2 Connect verfügen.

Empfehlung: Verwenden Sie in der Abrechnungszeichenfolge folgende Zeichen:

- Alphabetische Zeichen (A - Z)
- Numerische Zeichen (0 - 9)
- Unterstreichungszeichen (_)

Zugehörige Referenzen:

- „GET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE MANAGER CONFIGURATION command“ in *Command Reference*

federated - Unterstützung für Systeme föderierter Datenbanken

Konfigurationstyp Datenbankmanager

Gilt für

- Datenbankserver mit lokalen und fernen Clients
- Datenbankserver mit lokalen Clients
- Partitionierten Datenbankserver mit lokalen und fernen Clients

Parametertyp Konfigurierbar

Standardwert [Bereich] No [Yes; No]

Dieser Parameter aktiviert bzw. inaktiviert die Unterstützung für Anwendungen, die verteilte Anforderungen für von Datenquellen (wie z. B. der DB2-Produktfamilie und Oracle) verwaltete Daten übergeben.

Zugehörige Referenzen:

- „GET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE MANAGER CONFIGURATION command“ in *Command Reference*

jdk_path - Installationspfad für das Software Developer's Kit für Java

Konfigurationstyp Datenbankmanager

Gilt für

- Datenbankserver mit lokalen und fernen Clients
- Datenbankserver mit lokalen Clients
- Partitionierten Datenbankserver mit lokalen und fernen Clients

Parametertyp Konfigurierbar

Standardwert [Bereich] NULL [gültiger Pfad]

Dieser Parameter gibt das Verzeichnis an, in dem das Software Developer's Kit (SDK) für Java installiert ist, das verwendet wird, um gespeicherte Java-Prozeduren und benutzerdefinierte Funktionen auszuführen. CLASSPATH und andere vom Java-Interpreter verwendete Umgebungsvariablen werden aus dem Wert dieses Parameters berechnet.

Da es für diesen Parameter keinen Standardwert gibt, sollten Sie beim Installieren des SDK für Java einen Wert angeben.

Zugehörige Referenzen:

- „java_heap_sz - Maximale Zwischenspeichergröße für Java-Interpreter ” auf Seite 469
- „GET DATABASE MANAGER CONFIGURATION command” in *Command Reference*
- „RESET DATABASE MANAGER CONFIGURATION command” in *Command Reference*
- „UPDATE DATABASE MANAGER CONFIGURATION command” in *Command Reference*

nodetype - Knotenart des Systems

Konfigurationstyp

Datenbankmanager

Gilt für

- Datenbankserver mit lokalen und fernen Clients
- Client
- Datenbankserver mit lokalen Clients
- Partitionierten Datenbankserver mit lokalen und fernen Clients

Parametertyp

Informativ

Dieser Parameter liefert Informationen zu den DB2-Produkten, die auf Ihrem System installiert sind, und daher auch zur Art Ihrer Datenbankmanagerkonfiguration. Die folgenden Werte können von diesem Parameter zurückgegeben werden, und dieser Knotenart können die folgenden Produkte zugeordnet sein:

- **Datenbankserver mit lokalen und fernen Clients:** Ein DB2-Serverprodukt, das lokale und ferne Datenbankclients unterstützt und auf andere ferne Datenbankserver zugreifen kann.
- **Client:** Ein Datenbankclient, der auf ferne Datenbankserver zugreifen kann.
- **Datenbankserver mit lokalen Clients:** Ein Verwaltungssystem für relationale DB2-Datenbanken, das lokale Datenbankclients unterstützt und auf andere ferne Datenbankserver zugreifen kann.
- **Partitionierter Datenbankserver mit lokalen und fernen Clients:** Ein DB2-Serverprodukt, das lokale und ferne Datenbankclients unterstützt, auf andere ferne Datenbankserver zugreifen kann und parallelitätsfähig ist.

Zugehörige Referenzen:

- „GET DATABASE MANAGER CONFIGURATION command” in *Command Reference*

numdb - Maximale Anzahl gleichzeitig aktiver Datenbanken einschließlich Host- und iSeries-Datenbanken

Konfigurationstyp

Datenbankmanager

Gilt für

- Datenbankserver mit lokalen und fernen Clients
- Datenbankserver mit lokalen Clients
- Partitionierten Datenbankserver mit lokalen und fernen Clients

Parametertyp

Konfigurierbar

Standardwert [Bereich]

UNIX

8 [1 — 256]

**Windows-Datenbankserver mit lokalen und fern-
en Clients** 8 [1 — 256]

Windows-Datenbankserver mit lokalen Clients
3 [1 — 256]

Maßeinheit

Zähler

Mit diesem Parameter wird die Anzahl der lokalen Datenbanken, die gleichzeitig aktiv sein können (d. h., auf die von Anwendungen zugegriffen werden kann), oder die maximale Anzahl verschiedener Datenbankaliasnamen angegeben, die in einem DB2 Connect-Server katalogisiert werden können. Jede Datenbank belegt Speicher, und eine aktive Datenbank verwendet ein neues Segment des gemeinsam benutzten Speichers.

Empfehlung: In der Regel ist es am besten, diesen Wert auf die tatsächliche Anzahl der bereits für den Datenbankmanager definierten Datenbanken zu setzen und um ca. 10 % für mögliches Wachstum zu erhöhen.

Eine Änderung des Parameters *numdb* kann Auswirkungen auf die Gesamtmenge an Speicher haben, der zugeordnet wird. Daher ist es nicht empfehlenswert, diesen Parameter häufig zu aktualisieren. Beim Aktualisieren dieses Parameters sollten Sie die anderen Konfigurationsparameter beachten, die Speicher für eine Datenbank oder für eine mit der Datenbank verbundenen Anwendung zuordnen können.

Zugehörige Referenzen:

- „app_ctl_heap_sz - Zwischenspeichergröße für Anwendungssteuerung “ auf Seite 449
- „applheapsz - Zwischenspeichergröße für Anwendungen “ auf Seite 454
- „aslheapsz - Zwischenspeichergröße für Anwendungsunterstützungsebene “ auf Seite 461
- „dbheap - Zwischenspeicher für Datenbank “ auf Seite 437
- „database_memory - Größe des gemeinsam genutzten Datenbankspeichers “ auf Seite 434
- „mon_heap_sz - Zwischenspeichergröße für Datenbanksystemmonitor “ auf Seite 470
- „locklist - Maximaler Speicher für Sperrenliste “ auf Seite 438
- „sortheap - Zwischenspeichergröße für Sortierlisten “ auf Seite 458
- „stmtheap - Größe des Anweisungszwischenspeichers “ auf Seite 460
- „stat_heap_sz - Größe des Statistikzwischenspeichers “ auf Seite 459
- „GET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE MANAGER CONFIGURATION command“ in *Command Reference*

tp_mon_name - Name des Transaktionsprozessormonitors

Konfigurationstyp

Datenbankmanager

Gilt für

- Datenbankserver mit lokalen und fern- Clients
- Client

	• Datenbankserver mit lokalen Clients • Partitionierten Datenbankserver mit lokalen und fernen Clients
Parametertyp	Konfigurierbar
Standardwert	Kein Standardwert
Gültige Werte	• CICS • MQ • ENCINA • CB • SF • TUXEDO • TOPEND • Kein Eintrag oder ein weiterer Wert (unter UNIX und Windows; keine weiteren gültigen Werte unter Solaris oder SINIX)

Mit diesem Parameter wird der Name des verwendeten Monitors für das Transaktionsprogramm (TP) angegeben.

- Sind Anwendungen in einer WebSphere Enterprise Edition CICS-Umgebung aktiv, muss für den Parameter „CICS“ angegeben werden.
- Sind Anwendungen in einer WebSphere Enterprise Edition Encina-Umgebung aktiv, muss für den Parameter „ENCINA“ angegeben werden.
- Sind Anwendungen in einer WebSphere Enterprise Edition Component Broker-Umgebung aktiv, muss für den Parameter „CB“ angegeben werden.
- Sind Anwendungen in einer IBM MQSeries-Umgebung aktiv, muss für den Parameter „MQ“ angegeben werden.
- Sind Anwendungen in einer BEA Tuxedo-Umgebung aktiv, muss für den Parameter „TUXEDO“ angegeben werden.
- Sind Anwendungen in einer IBM San Francisco-Umgebung aktiv, muss für den Parameter „SF“ angegeben werden.

Benutzer von **IBM WebSphere EJB** und **Microsoft Transaction Server** müssen für diesen Parameter keinen Wert konfigurieren.

Wenn keines der oben genannten Produkte verwendet wird, sollte dieser Parameter nicht konfiguriert, sondern ohne Angabe eines Werts belassen werden.

In vorangehenden Versionen von IBM DB2 unter Windows NT enthielt dieser Parameter den Pfad und den Namen der DLL, in der die Funktionen *ax_reg* und *ax_unreg* des XA-Transaktionsmanagers gespeichert waren. Dieses Format wird noch immer unterstützt. Wenn der Wert für diesen Parameter mit keinem der oben genannten TP-Monitornamen übereinstimmt, wird angenommen, dass der angegebene Wert der Name einer Bibliothek ist, die die Funktionen *ax_reg* und *ax_unreg* enthält. Dies gilt für UNIX- und Windows NT-Umgebungen.

Benutzer von TXSeries CICS und Encina: In vorangegangenen Versionen dieses Produkts unter Windows NT war es erforderlich, diesen Parameter als „libEncServer:C“ oder „libEncServer:E“ zu konfigurieren. Dies wird noch immer unterstützt, ist aber nicht mehr erforderlich. Wenn der Parameter als „CICS“ oder „ENCINA“ konfiguriert wird, ist dies ausreichend.

Benutzer von MQSeries: In vorangegangenen Versionen dieses Produkts unter Windows NT war es erforderlich, diesen Parameter als „mqmax“ zu konfigurieren. Dies wird noch immer unterstützt, ist aber nicht mehr erforderlich. Wenn der Parameter als „MQ“ konfiguriert wird, ist dies ausreichend.

Benutzer von Component Broker: In vorangegangenen Versionen dieses Produkts unter Windows NT war es erforderlich, diesen Parameter als „somtrx1i“ zu konfigurieren. Dies wird noch immer unterstützt, ist aber nicht mehr erforderlich. Wenn der Parameter als „CB“ konfiguriert wird, ist dies ausreichend.

Benutzer von San Francisco: In vorangegangenen Versionen dieses Produkts unter Windows NT war es erforderlich, diesen Parameter als „ibmsfDB2“ zu konfigurieren. Dies wird noch immer unterstützt, ist aber nicht mehr erforderlich. Wenn der Parameter als „SF“ konfiguriert wird, ist dies ausreichend.

Die maximale Länge der Zeichenfolge, die für diesen Parameter angegeben werden kann, beträgt 19 Zeichen.

Diese Informationen können auch in der Zeichenfolge XA OPEN von IBM DB2 Version 9.1 konfiguriert werden. Wenn mehrere TP-Monitore in einer einzigen DB2-Instanz verwendet werden, muss diese Funktion verwendet werden.

Zugehörige Referenzen:

- „xa_open-Zeichenfolgeformate“ in *Systemverwaltung: Konzept*
- „GET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE MANAGER CONFIGURATION command“ in *Command Reference*

util_impact_lim - Richtlinie für Instanzauslastungswirkung

Konfigurationstyp	Datenbankmanager
Gilt für	• Datenbankserver mit lokalen Clients • Datenbankserver mit lokalen und fernen Clients • Partitionierten Datenbankserver mit lokalen und fernen Clients
Parametertyp	Online konfigurierbar
Weitergabeklasse	Sofort
Standardwert [Bereich]	10 [1 - 100]
Maßeinheit	Prozentsatz der zulässigen Wirkung auf die Auslastung

Dieser Parameter ermöglicht dem Datenbankadministrator (DBA), die Leistungseinbußen für eine Auslastung durch ein gedrosseltes Dienstprogramm zu begrenzen. Der Datenbankadministrator kann so Onlinedienstprogramme während kritischer Auslastungszeiten des Produktionssystems ausführen und sicher sein, dass die Auswirkung auf die Produktionsleistung innerhalb annehmbarer Grenzen bleibt.

Zum Beispiel kann ein Datenbankadministrator, der den Parameter *util_impact_lim* (Auslastungswirkung) auf den Wert 10 setzt, davon ausgehen, dass ein gedrosselter BACKUP-Aufruf die Auslastung mit nicht mehr als 10 Prozent beansprucht.

Wenn der Parameter *util_impact_lim* auf den Wert 100 gesetzt ist, findet keine Drosselung von Dienstprogrammaufrufen statt. In diesem Fall können die Dienstprogramme die Auslastung in willkürlichem (und unerwünschtem) Ausmaß beanspruchen. Wenn der Parameter *util_impact_lim* auf einen Wert unter 100 gesetzt wird, ist es möglich, Dienstprogramme im gedrosselten Modus aufzurufen. Zur Ausführung im gedrosselten Modus muss ein Dienstprogramm außerdem mit einer Priorität ungleich Null aufgerufen werden.

Empfehlung: Für die meisten Benutzer ist es wahrscheinlich vorteilhaft, wenn der Parameter *util_impact_lim* auf einen niedrigen Wert (zum Beispiel zwischen 1 und 10) gesetzt wird.

Ein gedrosseltes Dienstprogramm benötigt in der Regel mehr Ausführungszeit als ein ungedrosseltes Dienstprogramm. Wenn Sie feststellen, dass ein Dienstprogramm extrem viel Zeit benötigt, erhöhen Sie den Wert des Parameters *util_impact_lim* oder inaktivieren die Drosselung völlig, indem Sie den Parameter *util_impact_lim* auf den Wert 100 setzen.

Zugehörige Referenzen:

- „GET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE MANAGER CONFIGURATION command“ in *Command Reference*

Instanzverwaltung

Die folgenden Parameter beziehen sich auf die Sicherheit und Verwaltung der Datenbankmanagerinstanz:

- „authentication - Authentifizierungstyp “ auf Seite 575
- „catalog_noauth - Katalogisieren ohne Berechtigung zulässig “ auf Seite 576
- „clnt_krb_plugin - Client-Kerberos-Plug-in “ auf Seite 577
- „clnt_pw_plugin - Client-Plug-in für Benutzer-ID und Kennwort “ auf Seite 578
- „dftdbpath - Standarddatenbankpfad “ auf Seite 578
- „fed_noauth - Authentifizierung bei Servern mit föderierten Datenbanken umgehen “ auf Seite 579
- „group_plugin - Gruppen-Plug-in “ auf Seite 580
- „local_gssplugin - GSS-API-Plug-in für lokale Berechtigung auf Instanzebene “ auf Seite 580
- „srvcon_auth - Authentifizierungstyp für ankommende Verbindungen auf dem Server “ auf Seite 581
- „srvcon_gssplugin_list - Liste der GSS-API-Plug-ins für ankommende Verbindungen auf dem Server “ auf Seite 581
- „srvcon_pw_plugin - Plug-in für Benutzer-ID/Kennwort für ankommende Verbindungen auf dem Server “ auf Seite 582
- „srv_plugin_mode - Server-Plug-in-Modus “ auf Seite 583
- „sysadm_group - SYSADM-Gruppenname “ auf Seite 583

- „sysctrl_group - SYSCTRL-Gruppenname “ auf Seite 584
- „sysmaint_group - SYSMAINT-Gruppenname “ auf Seite 585
- „sysmon_group - Berechtigungsgruppenname für Systemmonitor “ auf Seite 586
- „trust_allclnts - Alle Clients akzeptieren “ auf Seite 587
- „trust_clntauth - Authentifizierung gesicherter Clients “ auf Seite 588
- „restrict_access - Eingeschränkter Datenbankzugriff (Konfigurationsparameter)” auf Seite 589

authentication - Authentifizierungstyp

Konfigurationstyp

Datenbankmanager

Gilt für

- Datenbankserver mit lokalen und fernen Clients
- Client
- Datenbankserver mit lokalen Clients
- Partitionierten Datenbankserver mit lokalen und fernen Clients

Parametertyp

Konfigurierbar

Standardwert [Bereich]

SERVER [CLIENT; SERVER; SERVER_ENCRYPT;
DATA_ENCRYPT; DATA_ENCRYPT_CMP KERB-
EROS; KRB_SERVER_ENCRYPT; GSSPLUGIN;
GSS_SERVER_ENCRYPT]

Mit diesem Parameter wird angegeben und festgelegt, wie und wo die Authentifizierung eines Benutzers stattfindet.

Wird als Wert SERVER angegeben, werden die Benutzer-ID und das Kennwort vom Client an den Server gesendet, so dass die Authentifizierung auf dem Server ausgeführt werden kann. Der Wert SERVER_ENCRYPT unterscheidet sich vom Wert SERVER nur darin, dass über das Netzwerk gesendete Kennwörter verschlüsselt werden.

Der Wert DATA_ENCRYPT bedeutet, dass der Server verschlüsselte SERVER-Authentifizierungsschemata und die Verschlüsselung von Benutzerdaten akzeptiert. Die Authentifizierung funktioniert genau wie für SERVER_ENCRYPT.

Die folgenden Benutzerdaten werden bei Verwendung dieses Authentifizierungstyps verschlüsselt:

- SQL-Anweisungen.
- Daten von SQL-Programmvariablen.
- Ausgabedaten aus der Serververarbeitung einer SQL-Anweisung, einschließlich einer Beschreibung der Daten.
- Einige oder alle Antwortgruppendaten, die aus einer Abfrage resultieren.
- LOB-Streaming (LOB, Large Objects).
- SQLDA-Deskriptoren.

Der Wert DATA_ENCRYPT_CMP bedeutet, dass der Server verschlüsselte SERVER-Authentifizierungsschemata und die Verschlüsselung von Benutzerdaten akzeptiert. Darüber hinaus bietet dieser Authentifizierungstyp Kompatibilität mit Produkten früherer Versionen, die den Authentifizierungstyp DATA_ENCRYPT nicht unterstützen. Diese Produkte erhalten die Möglichkeit, die Verbindung mit dem Authen-

tifizierungstyp `SERVER_ENCRYPT` und ohne Verschlüsselung von Benutzerdaten herzustellen. Produkte, die den neuen Authentifizierungstyp unterstützen, müssen ihn verwenden. Dieser Authentifizierungstyp ist nur in der Konfigurationsdatei des Datenbankmanagers des Servers, jedoch nicht im Befehl `CATALOG DATABASE` gültig.

Anmerkung: Für eine Konfiguration, die der Common Criteria-Zertifizierung entspricht, ist `SERVER` der einzig unterstützte Wert.

Der Wert `CLIENT` gibt an, dass alle Authentifizierungen auf dem Client stattfinden. Auf dem Server muss keine Authentifizierung mehr ausgeführt werden.

Der Wert `KERBEROS` bedeutet, dass die Authentifizierung auf einem Kerberos-Server mit Hilfe des Kerberos-Sicherheitsprotokolls für Authentifizierung ausgeführt wird. Bei Verwendung des Authentifizierungstyps `KRB_SERVER_ENCRYPT` auf dem Server und Unterstützung des Kerberos-Sicherheitsystems durch die Clients ist der tatsächliche Systemauthentifizierungstyp `KERBEROS`. Unterstützen die Clients das Kerberos-Sicherheitssystem nicht, ist der Systemauthentifizierungstyp praktisch äquivalent zu `SERVER_ENCRYPT`.

Der Wert `GSSPLUGIN` bedeutet, dass die Authentifizierung durch einen externen GSSAPI-basierten Sicherheitsmechanismus ausgeführt wird. Bei Verwendung des Authentifizierungstyps `GSS_SERVER_ENCRYPT` auf dem Server und Unterstützung des `GSSPLUGIN`-Sicherheitsmechanismus durch die Clients ist der tatsächliche Systemauthentifizierungstyp `GSSPLUGIN` (d. h., wenn die Clients eines der Plug-ins des Servers unterstützen). Unterstützen die Clients den `GSSPLUGIN`-Sicherheitsmechanismus nicht, ist der Systemauthentifizierungstyp praktisch äquivalent zu `SERVER_ENCRYPT`.

Empfehlung: In der Regel ist der Standardwert (`SERVER`) geeignet.

Zugehörige Konzepte:

- „Authentifizierungsmethoden für den Server“ in *Systemverwaltung: Implementierung*

Zugehörige Referenzen:

- „Einschränkungen für den nativen XML-Datenspeicher“ in *XML-Handbuch*
- „GET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE MANAGER CONFIGURATION command“ in *Command Reference*

catalog_noauth - Katalogisieren ohne Berechtigung zulässig

Konfigurationstyp Datenbankmanager

Gilt für

- Datenbankserver mit lokalen und fernen Clients
- Client
- Datenbankserver mit lokalen Clients
- Partitionierten Datenbankserver mit lokalen und fernen Clients

Parametertyp	Online konfigurierbar
Weitergabeklasse	Sofort
Standardwert [Bereich]	Datenbankserver mit lokalen und fernen Clients NO [NO (0) — YES (1)] Client; Datenbankserver mit lokalen Clients YES [NO (0) — YES (1)]

Dieser Parameter gibt an, ob Benutzer Datenbanken und Knoten oder DCS- und ODBC-Verzeichnisse ohne SYSADM-Berechtigung katalogisieren und aus dem Katalog entfernen können. Der Standardwert (0) für diesen Parameter gibt an, dass SYSADM-Berechtigung erforderlich ist. Wenn dieser Parameter auf 1 gesetzt ist, ist keine SYSADM-Berechtigung erforderlich.

Zugehörige Referenzen:

- „GET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE MANAGER CONFIGURATION command“ in *Command Reference*

clnt_krb_plugin - Client-Kerberos-Plug-in

Konfigurationstyp	Datenbankmanager
Gilt für	• Datenbankserver mit lokalen und fernen Clients • Client • Datenbankserver mit lokalen Clients • Partitionierten Datenbankserver mit lokalen und fernen Clients
Parametertyp	Konfigurierbar
Standardwert [Bereich]	Null oder IBMkrb5 [beliebige gültige Zeichenfolge]

Mit diesem Parameter wird der Name der Standard-Plug-in-Bibliothek für Kerberos angegeben, die für die clientseitige Authentifizierung sowie für die lokale Berechtigung zu verwenden ist. Standardmäßig lautet der Wert null auf UNIX-basierten Systemen sowie IBMkrb5 auf Windows-Betriebssystemen. Dieses Plug-in wird verwendet, wenn der Client mit der KERBEROS-Authentifizierung authentifiziert wird.

Zugehörige Referenzen:

- „GET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE MANAGER CONFIGURATION command“ in *Command Reference*

clnt_pw_plugin - Client-Plug-in für Benutzer-ID und Kennwort

Konfigurationstyp

Datenbankmanager

Gilt für

- Datenbankserver mit lokalen und fernen Clients
- Client
- Datenbankserver mit lokalen Clients
- Partitionierten Datenbankserver mit lokalen und fernen Clients

Parametertyp

Konfigurierbar

Standardwert [Bereich]

NULL [beliebige gültige Zeichenfolge]

Mit diesem Parameter wird der Name der Benutzer-ID/Kennwort-Plug-in-Bibliothek angegeben, die für die clientseitige Authentifizierung sowie für die lokale Berechtigung zu verwenden ist. Standardmäßig ist dieser Wert null und die von DB2 bereitgestellte Benutzer-ID/Kennwort-Plug-in-Bibliothek wird verwendet. Das Plug-in wird verwendet, wenn der Client mit dem Authentifizierungstyp CLIENT, SERVER oder SERVER_ENCRYPT arbeitet.

Zugehörige Referenzen:

- „GET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE MANAGER CONFIGURATION command“ in *Command Reference*

dftdbpath - Standarddatenbankpfad

Konfigurationstyp

Datenbankmanager

Gilt für

- Datenbankserver mit lokalen und fernen Clients
- Datenbankserver mit lokalen Clients
- Partitionierten Datenbankserver mit lokalen und fernen Clients

Parametertyp

Online konfigurierbar

Weitergabeklasse

Sofort

Standardwert [Bereich]

UNIX

Benutzerverzeichnis des Instanzeigners [beliebiger vorhandener Pfad]

Windows

Laufwerk, auf dem DB2 installiert ist [beliebiger vorhandener Pfad]

Dieser Parameter enthält den Standarddateipfad, der zur Erstellung von Datenbanken unter dem Datenbankmanager verwendet wird. Wird beim Erstellen einer Datenbank kein Pfad angegeben, wird die Datenbank in dem Pfad erstellt, der durch den Parameter *dftdbpath* angegeben wird.

In einer Umgebung mit partitionierten Datenbanken müssen Sie sicherstellen, dass der Pfad, in dem die Datenbank erstellt wird, kein an das NFS angehängter Pfad (auf UNIX-Plattformen) bzw. kein Netzwerklaufwerk ist (in Windows-Umgebungen). Der angegebene Pfad muss physisch auf jedem Datenbankpartitionsserver vorhanden sein. Um Verwirrung zu vermeiden, ist es am besten, einen Pfad anzugeben, der auf jedem Datenbankpartitionsserver lokal angehängt ist. Die Länge des Pfads darf maximal 205 Zeichen betragen. Das System hängt den Knotennamen am Ende des Pfads an.

Weil Datenbanken auf beträchtliche Größen anwachsen und möglicherweise viele Benutzer Datenbanken erstellen können (je nach Umgebung und Zielsetzung), ist es häufig sehr praktisch, alle Datenbanken an einer einzigen definierten Position erstellen und speichern zu lassen. Außerdem ist es von Vorteil, Datenbanken von anderen Anwendungen und Daten zu trennen - sowohl aus Integritätsgründen als auch zur einfachen Durchführung von Backup und Recovery.

In UNIX-basierten Umgebungen darf die Länge des im Parameter *dftdbpath* definierten Namens 215 Zeichen nicht überschreiten, und der Name muss ein gültiger, absoluter Pfadname sein. Unter Windows kann der im Parameter *dftdbpath* angegebene Name ein Laufwerksbuchstabe sein, auf den optional ein Doppelpunkt folgt.

Empfehlung: Wenn die Möglichkeit besteht, legen Sie umfangreiche Datenbanken auf einer anderen Platte an als andere Daten, auf die häufig zugegriffen wird, wie z. B. Dateien des Betriebssystems oder die Datenbankprotokolldateien.

Zugehörige Referenzen:

- „GET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE MANAGER CONFIGURATION command“ in *Command Reference*

fed_noauth - Authentifizierung bei Servern mit föderierten Datenbanken umgehen

Konfigurationstyp Datenbankmanager

Gilt für

- Datenbankserver mit lokalen und fernen Clients
- Datenbankserver mit lokalen Clients
- Partitionierten Datenbankserver mit lokalen und fernen Clients

Parametertyp Online konfigurierbar

Weitergabeklasse Sofort

Standardwert [Bereich] No [Yes; No]

Wenn der Parameter *fed_noauth* auf *yes*, der Parameter *authentication* auf *server* oder auf *server_encrypt* und der Parameter *federated* auf *yes* gesetzt ist, wird die Authentifizierung auf der Instanz umgangen. Es wird dann angenommen, dass die Authentifizierung an der Datenquelle vorgenommen wird. Gehen Sie vorsichtig vor, wenn *fed_noauth* auf *yes* gesetzt ist. Es wird weder auf dem Client noch in DB2 eine

Authentifizierung durchgeführt. Jeder Benutzer, der den SYSADM-Authentifizierungsnamen kennt, kann für den Server mit föderierten Datenbanken SYSADM-Berechtigung erlangen.

Zugehörige Referenzen:

- „authentication - Authentifizierungstyp “ auf Seite 575
- „federated - Unterstützung für Systeme föderierter Datenbanken “ auf Seite 569
- „GET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE MANAGER CONFIGURATION command“ in *Command Reference*

group_plugin - Gruppen-Plug-in

Konfigurationstyp Datenbankmanager

Gilt für

- Datenbankserver mit lokalen und fernen Clients
- Client
- Datenbankserver mit lokalen Clients
- Partitionierten Datenbankserver mit lokalen und fernen Clients

Parametertyp Konfigurierbar

Standardwert [Bereich] NULL [beliebige gültige Zeichenfolge]

Mit diesem Parameter wird der Name der Gruppen-Plug-in-Bibliothek angegeben. Standardmäßig ist dieser Wert null, und DB2 verwendet die Gruppensuchfunktion (Lookup) des Betriebssystems. Das Plug-in wird für alle Gruppensuchen verwendet.

Zugehörige Referenzen:

- „GET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE MANAGER CONFIGURATION command“ in *Command Reference*

local_gssplugin - GSS-API-Plug-in für lokale Berechtigung auf Instanzebene

Konfigurationstyp Datenbankmanager

Gilt für

- Datenbankserver mit lokalen und fernen Clients
- Client
- Datenbankserver mit lokalen Clients
- Partitionierten Datenbankserver mit lokalen und fernen Clients

Parametertyp Konfigurierbar

Standardwert [Bereich] NULL [beliebige gültige Zeichenfolge]

Mit diesem Parameter wird der Name der GSS-API-Plug-in-Standardbibliothek angegeben, die zur lokalen Berechtigung auf Instanzebene zu verwenden ist, wenn der Konfigurationsparameter *authentication* des Datenbankmanagers auf den Wert GSSPLUGIN oder GSS_SERVER_ENCRYPT gesetzt ist.

Zugehörige Referenzen:

- „GET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE MANAGER CONFIGURATION command“ in *Command Reference*

srvcon_auth - Authentifizierungstyp für ankommende Verbindungen auf dem Server

Konfigurationstyp Datenbankmanager

Gilt für

- Datenbankserver mit lokalen und fernen Clients
- Datenbankserver mit lokalen Clients
- Partitionierten Datenbankserver mit lokalen und fernen Clients

Parametertyp Konfigurierbar

Standardwert [Bereich] Null [CLIENT; SERVER; SERVER_ENCRYPT; KERBEROS; KRB_SERVER_ENCRYPT; GSSPLUGIN; GSS_SERVER_ENCRYPT]

Mit diesem Parameter wird angegeben, wie und wo die Benutzerauthentifizierung bei der Verarbeitung ankommender Verbindungen auf dem Server stattfinden soll. Er dient zum Überschreiben des aktuellen Authentifizierungstyps. Wenn kein Wert angegeben wird, verwendet DB2 den Wert des Konfigurationsparameters *authentication* des Datenbankmanagers.

Eine Beschreibung der Authentifizierungstypen finden Sie unter „authentication - Authentifizierungstyp “ auf Seite 575.

Zugehörige Referenzen:

- „GET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE MANAGER CONFIGURATION command“ in *Command Reference*

srvcon_gssplugin_list - Liste der GSS-API-Plug-ins für ankommende Verbindungen auf dem Server

Konfigurationstyp Datenbankmanager

Gilt für

- Datenbankserver mit lokalen und fernen Clients
- Datenbankserver mit lokalen Clients

- Partitionierten Datenbankserver mit lokalen und fernen Clients

Parametertyp	Konfigurierbar
Standardwert [Bereich]	NULL [beliebige gültige Zeichenfolge]

Mit diesem Parameter werden die GSS-API-Plug-in-Bibliotheken angegeben, die vom Datenbankserver unterstützt werden. Dieser Wert ist standardmäßig null. Wenn der Authentifizierungstyp GSSPLUGIN ist und dieser Parameter NULL ist, wird ein Fehler zurückgegeben. Wenn der Authentifizierungstyp KERBEROS ist und dieser Parameter NULL ist, wird das von DB2 bereitgestellte Kerberos-Modul bzw. die Kerberos-Bibliothek verwendet. Bei einem anderen Authentifizierungstyp wird dieser Parameter nicht verwendet.

Wenn der Authentifizierungstyp KERBEROS ist und der Wert dieses Parameters nicht NULL ist, muss die angegebene Liste exakt ein Kerberos-Plug-in enthalten, und dieses Plug-in wird zur Authentifizierung verwendet (alle anderen GSS-Plug-ins in der Liste werden ignoriert). Enthält die Liste mehr als ein Kerberos-Plug-in, wird ein Fehler zurückgegeben.

Jeder Name eines GSS-API-Plug-ins muss durch ein Komma (,) ohne Leerzeichen vor oder nach dem Komma getrennt werden. Die Plug-in-Namen sollten in der bevorzugten Reihenfolge aufgelistet werden. Dieser Parameter verarbeitet ankommende Verbindungen auf dem Server, wenn der Parameter *srvcon_auth* mit dem Wert KERBEROS, KRB_SERVER_ENCRYPT, GSSPLUGIN oder GSS_SERVER_ENCRYPT angegeben ist oder wenn *srvcon_auth* nicht angegeben ist und *authentication* mit dem Wert KERBEROS, KRB_SERVER_ENCRYPT, GSSPLUGIN oder GSS_SERVER_ENCRYPT angegeben ist.

Zugehörige Referenzen:

- „GET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE MANAGER CONFIGURATION command“ in *Command Reference*

srvcon_pw_plugin - Plug-in für Benutzer-ID/Kennwort für ankommende Verbindungen auf dem Server

Konfigurationstyp	Datenbankmanager

Gilt für

- Datenbankserver mit lokalen und fernen Clients
- Datenbankserver mit lokalen Clients
- Partitionierten Datenbankserver mit lokalen und fernen Clients

Parametertyp	Konfigurierbar
Standardwert [Bereich]	NULL [beliebige gültige Zeichenfolge]

Mit diesem Parameter wird der Name der Benutzer-ID/Kennwort-Plug-in-Standardbibliothek angegeben, die für die serverseitige Authentifizierung zu verwenden ist. Standardmäßig ist dieser Wert null und die von DB2 bereitgestellte Benutzer-ID/Kennwort-Plug-in-Bibliothek wird verwendet.

Der Parameter verarbeitet ankommende Verbindungen auf dem Server, wenn der Parameter *srvcon_auth* mit dem Wert SERVER oder SERVER_ENCRYPT angegeben ist oder wenn *srvcon_auth* nicht angegeben ist und *authentication* mit dem Wert CLIENT, SERVER oder SERVER_ENCRYPT angegeben ist.

Zugehörige Referenzen:

- „GET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE MANAGER CONFIGURATION command“ in *Command Reference*

srv_plugin_mode - Server-Plug-in-Modus

Konfigurationstyp Datenbankmanager

Gilt für

- Datenbankserver mit lokalen und fernen Clients
- Datenbankserver mit lokalen Clients
- Partitionierten Datenbankserver mit lokalen und fernen Clients

Parametertyp Konfigurierbar

Standardwert [Bereich] UNFENCED

Mit diesem Parameter wird angegeben, ob Plug-ins im abgeschirmten oder nicht abgeschirmten Modus auszuführen sind. Es wird nur der nicht abgeschirmte Modus (UNFENCED) unterstützt.

Zugehörige Referenzen:

- „GET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE MANAGER CONFIGURATION command“ in *Command Reference*

sysadm_group - SYSADM-Gruppenname

Konfigurationstyp Datenbankmanager

Gilt für

- Datenbankserver mit lokalen und fernen Clients
- Client
- Datenbankserver mit lokalen Clients
- Partitionierten Datenbankserver mit lokalen und fernen Clients

Parametertyp Konfigurierbar

Standardwert Null

Die Berechtigung SYSADM (Systemadministrator) stellt die höchste Ebene der Berechtigungen innerhalb des Datenbankmanagers dar, von der aus sämtliche

Datenbankobjekte gesteuert werden. Mit diesem Parameter wird der Gruppenname definiert, der die Berechtigung SYSADM für die Datenbankmanagerinstanz besitzt.

Die Berechtigung SYSADM wird von den Sicherheitseinrichtungen festgelegt, die in einer bestimmten Betriebsumgebung verwendet werden.

- Unter dem Windows-Betriebssystem kann dieser Parameter auf eine beliebige lokale Gruppe gesetzt werden. Er wird in der Sicherheitsdatenbank von Windows 2000 definiert. Die Gruppennamen dürfen maximal 30 Byte lang sein. Wenn für diesen Parameter der Wert „NULL“ definiert wird, haben alle Mitglieder der Administratorengruppe die Berechtigung SYSADM.
- Wenn bei auf UNIX basierenden Systemen der Wert „NULL“ für diesen Parameter angegeben wird, gilt die Primärgruppe des Instanzeigners standardmäßig als SYSADM-Gruppe.
Ist der Wert nicht „NULL“, kann als SYSADM-Gruppe jeder gültige UNIX-Gruppenname definiert werden.

Wenn Sie den Parameter auf seinen Standardwert (NULL) zurücksetzen wollen, führen Sie den Befehl `UPDATE DBM CFG USING SYSADM_GROUP NULL` aus. Das Schlüsselwort „NULL“ muss in Großbuchstaben angegeben werden. Sie können auch das Notizbuch **Instanz konfigurieren** über die DB2-Steuerzentrale aufrufen und den Parameter über das Notizbuch definieren.

Zugehörige Referenzen:

- „GET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „sysctrl_group - SYSCTRL-Gruppenname “ auf Seite 584
- „sysmaint_group - SYSMAINT-Gruppenname “ auf Seite 585
- „sysmon_group - Berechtigungsgruppenname für Systemmonitor “ auf Seite 586

sysctrl_group - SYSCTRL-Gruppenname

Konfigurationstyp Datenbankmanager

Gilt für

- Datenbankserver mit lokalen und fernen Clients
- Client
- Datenbankserver mit lokalen Clients
- Partitionierten Datenbankserver mit lokalen und fernen Clients

Parametertyp Konfigurierbar

Standardwert Null

Mit diesem Parameter wird der Gruppenname definiert, der die Berechtigung SYSCTRL (Systemsteuerung) besitzt. Die Berechtigung SYSCTRL umfasst Zugriffsrechte, die sich auf Systemressourcen auswirkende Operationen ermöglichen, aber keinen direkten Zugriff auf Daten zulassen.

Gruppennamen werden auf allen Plattformen akzeptiert, solange sie höchstens eine Länge von 30 Byte haben.

Achtung: Dieser Parameter muss für Clients unter Windows 98 den Wert NULL aufweisen, wenn Systemsicherheit verwendet wird (d. h., die Authentifizierung ist CLIENT, SERVER, DCS oder eine andere gültige Authentifizierung). Der Grund hierfür ist, dass Windows 98 keine Gruppeninformationen speichert und somit keine Möglichkeit bietet, festzustellen, ob ein Benutzer ein Mitglied einer bestimmten SYSCTRL-Gruppe ist. Bei Angabe eines Gruppennamens kann keiner der Benutzer dieser Gruppe angehören.

Wenn Sie den Parameter auf seinen Standardwert (NULL) zurücksetzen wollen, führen Sie den Befehl UPDATE DBM CFG USING SYSCTRL_GROUP NULL aus. Das Schlüsselwort „NULL“ muss in Großbuchstaben angegeben werden. Sie können auch das Notizbuch **Instanz konfigurieren** über die DB2-Steuerzentrale aufrufen und den Parameter über das Notizbuch definieren.

Zugehörige Referenzen:

- „GET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „sysadm_group - SYSADM-Gruppenname “ auf Seite 583
- „sysmaint_group - SYSMANT-Gruppenname “ auf Seite 585
- „sysmon_group - Berechtigungsgruppenname für Systemmonitor “ auf Seite 586

sysmaint_group - SYSMANT-Gruppenname

Konfigurationstyp Datenbankmanager

Gilt für

- Datenbankserver mit lokalen und fernen Clients
- Client
- Datenbankserver mit lokalen Clients
- Partitionierten Datenbankserver mit lokalen und fernen Clients

Parametertyp Konfigurierbar

Standardwert Null

Mit diesem Parameter wird der Gruppenname definiert, der die Berechtigung SYSMANT (Systempflege) besitzt. Mit der Berechtigung SYSMANT können Operationen zur Pflege aller Datenbanken, die zu einer Instanz gehören, durchgeführt werden, jedoch ohne direkten Zugriff auf Daten.

Gruppennamen werden auf allen Plattformen akzeptiert, solange sie höchstens eine Länge von 30 Byte haben.

Achtung: Dieser Parameter muss für Clients unter Windows 98 den Wert NULL besitzen, wenn Systemsicherheit verwendet wird (d. h., die Authentifizierung ist CLIENT, SERVER, DCS oder eine andere gültige Authentifizierung). Der Grund hierfür ist, dass Windows 98 keine Gruppeninformationen speichert und somit keine Möglichkeit bietet, festzustellen, ob ein Benutzer ein Mitglied einer bestimmten SYSMANT-Gruppe ist. Bei Angabe eines Gruppennamens kann keiner der Benutzer dieser Gruppe angehören.

Wenn Sie den Parameter auf seinen Standardwert (NULL) zurücksetzen wollen, führen Sie den Befehl UPDATE DBM CFG USING SYSMANT_GROUP NULL aus. Das Schlüsselwort „NULL“ muss in Großbuchstaben angegeben werden. Sie können auch das Notizbuch **Instanz konfigurieren** über die DB2-Steuerzentrale aufrufen und den Parameter über das Notizbuch definieren.

Zugehörige Referenzen:

- „GET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „sysadm_group - SYSADM-Gruppenname “ auf Seite 583
- „sysctrl_group - SYSCTRL-Gruppenname “ auf Seite 584
- „sysmon_group - Berechtigungsgruppenname für Systemmonitor “ auf Seite 586

sysmon_group - Berechtigungsgruppenname für Systemmonitor

Konfigurationstyp Datenbankmanager

Gilt für

- Datenbankserver mit lokalen und fernen Clients
- Client
- Datenbankserver mit lokalen Clients
- Partitionierten Datenbankserver mit lokalen und fernen Clients

Parametertyp Konfigurierbar

Standardwert Null

Mit diesem Parameter wird der Gruppenname definiert, der die Berechtigung SYSMON (Systemmonitor) besitzt. Benutzer, die über die Berechtigung SYSMON auf der Instanzebene verfügen, können über den Datenbanksystemmonitor Momentaufnahmen von einer Datenbankmanagerinstanz oder den entstprechenden Datenbanken erstellen. Die Berechtigung SYSMON ermöglicht die Ausführung der folgenden Befehle:

- GET DATABASE MANAGER MONITOR SWITCHES
- GET MONITOR SWITCHES
- GET SNAPSHOT
- LIST ACTIVE DATABASES
- LIST APPLICATIONS
- LIST DCS APPLICATIONS
- RESET MONITOR
- UPDATE MONITOR SWITCHES

Benutzer mit der Berechtigung SYSADM, SYSCTRL oder SYSMANT haben automatisch die Möglichkeit, Momentaufnahmen über den Datenbanksystemmonitor zu erstellen und diese Befehle zu verwenden.

Gruppennamen werden auf allen Plattformen akzeptiert, solange sie höchstens eine Länge von 30 Byte haben.

Achtung: Dieser Parameter muss für Clients unter Windows 98 den Wert NULL besitzen, wenn Systemsicherheit verwendet wird (d. h., die Authentifizierung ist CLIENT, SERVER, DCS oder eine andere gültige Authentifizierung). Der Grund hierfür ist, dass Windows 98 keine Gruppeninformationen speichert und somit keine Möglichkeit bietet, festzustellen, ob ein Benutzer ein Mitglied einer bestimmten SYSMON-Gruppe ist. Bei Angabe eines Gruppennamens kann keiner der Benutzer dieser Gruppe angehören.

Wenn Sie den Parameter auf seinen Standardwert (NULL) zurücksetzen wollen, führen Sie den Befehl UPDATE DBM CFG USING SYSMON_GROUP NULL aus. Das Schlüsselwort „NULL“ muss in Großbuchstaben angegeben werden. Sie können auch das Notizbuch **Instanz konfigurieren** über die DB2-Steuerzentrale aufrufen und den Parameter über das Notizbuch definieren.

Zugehörige Referenzen:

- „GET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „sysadm_group - SYSADM-Gruppenname “ auf Seite 583
- „sysctrl_group - SYSCTRL-Gruppenname “ auf Seite 584
- „sysmaint_group - SYSMAINT-Gruppenname “ auf Seite 585

trust_allclnts - Alle Clients akzeptieren

Konfigurationstyp Datenbankmanager

Gilt für

- Datenbankserver mit lokalen und fernen Clients
- Datenbankserver mit lokalen Clients
- Partitionierten Datenbankserver mit lokalen und fernen Clients

Parametertyp Konfigurierbar

Standardwert [Bereich] YES [NO, YES, DRDAONLY]

Dieser Parameter ist nur aktiviert, wenn der Parameter *authentication* auf CLIENT gesetzt ist.

Dieser Parameter und der Parameter *trust_clntauth* werden verwendet, um festzustellen, wo Benutzerberechtigungen für die Datenbankumgebung überprüft werden.

Durch Übernehmen des Standardwerts „YES“ für diesen Parameter werden alle Clients als gesicherte Clients akzeptiert. Dies bedeutet, der Server geht davon aus, dass auf jedem Client eine Sicherheitsstufe vorhanden ist und auf jedem Client eine Gültigkeitsprüfung für Benutzer ausgeführt werden kann.

Dieser Parameter kann nur auf „NO“ gesetzt werden, wenn der Parameter *authentication* auf CLIENT gesetzt ist. Ist dieser Parameter auf „NO“ gesetzt, müssen die ungesicherten Clients beim Herstellen einer Verbindung zum Server eine Benut-

zer-ID mit Kennwort angeben. Ungesicherte Clients sind Betriebssystemplattformen, die nicht über ein Sicherheitssystem zur Gültigkeitsprüfung für Benutzer verfügen.

Das Setzen dieses Parameters auf „DRDAONLY“ schützt vor allen Clients mit Ausnahme von Clients von DB2 für OS/390 und z/OS, DB2 für VM und VSE sowie DB2 für OS/400. Nur diese Clients dürfen die Authentifizierung auf der Client-Seite ausführen. Alle anderen Clients müssen eine Benutzer-ID und ein Kennwort bereitstellen, deren Gültigkeit vom Server überprüft wird.

Wenn *trust_allclnts* auf „DRDAONLY“ gesetzt ist, wird mit dem Parameter *trust_clntauth* ermittelt, wo die Authentifizierung der Clients erfolgt. Wenn *trust_clntauth* auf „CLIENT“ gesetzt ist, findet die Authentifizierung auf dem Client statt. Wenn *trust_clntauth* auf „SERVER“ gesetzt ist, erfolgt die Authentifizierung auf dem Client, sofern kein Kennwort angegeben ist, und auf dem Server, sofern ein Kennwort angegeben ist.

Zugehörige Referenzen:

- „authentication - Authentifizierungstyp “ auf Seite 575
- „trust_clntauth - Authentifizierung gesicherter Clients “ auf Seite 588
- „GET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „RESET DATABASE MANAGER CONFIGURATION command“ in *Command Reference*
- „UPDATE DATABASE MANAGER CONFIGURATION command“ in *Command Reference*

trust_clntauth - Authentifizierung gesicherter Clients

Konfigurationstyp

Datenbankmanager

Gilt für

- Datenbankserver mit lokalen und fernen Clients
- Datenbankserver mit lokalen Clients
- Partitionierten Datenbankserver mit lokalen und fernen Clients

Parametertyp

Konfigurierbar

Standardwert [Bereich]

CLIENT [CLIENT, SERVER]

Dieser Parameter gibt an, ob eine Authentifizierung gesicherter Clients auf dem Server oder auf dem Client erfolgt, wenn der Client eine Benutzer-ID mit Kennwort für eine Verbindung angibt. Dieser Parameter (und *trust_allclnts*) ist nur aktiviert, wenn der Parameter *authentication* auf CLIENT gesetzt ist. Wird keine Benutzer-ID mit Kennwort angegeben, geht das System davon aus, dass die Authentifizierung des Benutzers vom Client ausgeführt wurde, d. h., auf dem Server erfolgt keine weitere Überprüfung.

Ist dieser Parameter auf CLIENT (die Standardeinstellung) gesetzt, kann der gesicherte Client eine Verbindung herstellen, ohne eine Benutzer-ID mit Kennwort anzugeben, und es wird angenommen, dass die Authentifizierung des Benutzers bereits vom Betriebssystem vorgenommen wurde. Ist der Parameter auf SERVER gesetzt, werden Benutzer-ID und Kennwort auf dem Server überprüft.

Der numerische Wert für CLIENT ist 0. Der numerische Wert für SERVER ist 1.

Zugehörige Referenzen:

- „authentication - Authentifizierungstyp ” auf Seite 575
- „trust_allclnts - Alle Clients akzeptieren ” auf Seite 587
- „GET DATABASE MANAGER CONFIGURATION command” in *Command Reference*
- „RESET DATABASE MANAGER CONFIGURATION command” in *Command Reference*
- „UPDATE DATABASE MANAGER CONFIGURATION command” in *Command Reference*

restrict_access - Eingeschränkter Datenbankzugriff (Konfigurationsparameter)

Konfigurationstyp	Datenbank
Parametertyp	Informativ

Dieser Parameter gibt an, ob die Datenbank mit dem eingeschränkten Satz an Standardaktionen erstellt wurde. Das heißt, ob sie mit der Klausel RESTRICTIVE im Befehl CREATE DATABASE erstellt wurde.

YES: Die Klausel RESTRICTIVE wurde bei der Erstellung dieser Datenbank im Befehl CREATE DATABASE verwendet.

NO: Die Klausel RESTRICTIVE wurde bei der Erstellung dieser Datenbank im Befehl CREATE DATABASE nicht verwendet.

DB2-Verwaltungsserver

Die folgenden Parameter beziehen sich auf den DB2-Verwaltungsserver (DAS):

- „authentication - DAS-Authentifizierungstyp ”
- „contact_host - Speicherposition der Liste mit Ansprechpartnern ” auf Seite 590
- „das_codepage - DAS-Codepage ” auf Seite 591
- „das_territory - DAS-Gebiet ” auf Seite 591
- „dasadm_group - DASADM-Gruppenname ” auf Seite 591
- „db2system - Name des DB2-Serversystems ” auf Seite 592
- „discover - DAS-Discovery-Modus ” auf Seite 593
- „exec_exp_task - Verfallene Tasks ausführen ” auf Seite 593
- „jdk_64_path - Installationspfad für Software Developer's Kit (64 Bit) für Java auf DAS ” auf Seite 594
- „jdk_path - Installationspfad für Software Developer's Kit für Java auf DAS ” auf Seite 595
- „sched_enable - Schedulermodus ” auf Seite 595
- „sched_userid - Benutzer-ID für Scheduler ” auf Seite 596
- „smtp_server - SMTP-Server ” auf Seite 596
- „toolscat_db - Toolskatalogdatenbank ” auf Seite 597
- „toolscat_inst - Instanz der Toolskatalogdatenbank ” auf Seite 597
- „toolscat_schema - Schema der Toolskatalogdatenbank ” auf Seite 598

authentication - DAS-Authentifizierungstyp

Konfigurationstyp	DB2-Verwaltungsserver

Gilt für	DB2-Verwaltungsserver
Parametertyp	Konfigurierbar
Standardwert [Bereich]	SERVER_ENCRYPT [SERVER_ENCRYPT; KERBEROS_ENCRYPT]

Mit diesem Parameter wird festgelegt, wie und wo die Authentifizierung eines Benutzers stattfindet.

Wird als Wert SERVER_ENCRYPT angegeben, werden die Benutzer-ID und das Kennwort vom Client an den Server gesendet, damit die Authentifizierung auf dem Server ausgeführt werden kann. Über das Netzwerk versendete Kennwörter werden verschlüsselt.

Der Wert KERBEROS_ENCRYPT bedeutet, dass die Authentifizierung auf einem Kerberos-Server mit Hilfe des Kerberos-Sicherheitsprotokolls für Authentifizierung ausgeführt wird.

Anmerkung: Der Authentifizierungstyp KERBEROS_ENCRYPT wird nur auf Servern unterstützt, auf denen Windows 2000 ausgeführt wird.

Dieser Parameter kann nur über einen Befehlszeilenprozessor (CLP) der Version 8 aktualisiert werden.

Zugehörige Referenzen:

- „GET ADMIN CONFIGURATION command“ in *Command Reference*
- „RESET ADMIN CONFIGURATION command“ in *Command Reference*
- „UPDATE ADMIN CONFIGURATION command“ in *Command Reference*

contact_host - Speicherposition der Liste mit Ansprechpartnern

Konfigurationstyp	DB2-Verwaltungsserver
Gilt für	DB2-Verwaltungsserver
Parametertyp	Online konfigurierbar
Weitergabeklasse	Sofort
Standardwert [Bereich]	NULL [beliebiger gültiger TCP/IP-Hostname eines DB2-Verwaltungsservers der Version 8]

Dieser Parameter gibt die Speicherposition der Ansprechpartnerinformationen an, die der Scheduler und der Diagnosemonitor für Benachrichtigungen verwenden. Die Position muss der TCP/IP-Hostname eines DB2-Verwaltungsservers sein. Wenn sich *contact_host* auf einem fernen Datenbankverwaltungsserver (DAS) befindet, kann eine Ansprechpartnerliste von mehreren DB2-Verwaltungsservern gemeinsam benutzt werden. Wenn für *contact_host* kein Wert angegeben wird, nimmt der DAS an, dass die Ansprechpartnerinformationen lokal sind.

Dieser Parameter kann nur über einen Befehlszeilenprozessor (CLP) der Version 8 aktualisiert werden.

Zugehörige Referenzen:

- „GET ADMIN CONFIGURATION command“ in *Command Reference*
- „RESET ADMIN CONFIGURATION command“ in *Command Reference*

- „UPDATE ADMIN CONFIGURATION command“ in *Command Reference*

das_codepage - DAS-Codepage

Konfigurationstyp	DB2-Verwaltungsserver
Gilt für	DB2-Verwaltungsserver
Parametertyp	Online konfigurierbar
Weitergabeklasse	Sofort
Standardwert [Bereich]	NULL [beliebige gültige DB2-Codepage]

Dieser Parameter zeigt die vom DB2-Verwaltungsserver verwendete Codepage an. Wenn der Parameter auf NULL gesetzt ist, wird die Standardcodepage des Systems verwendet. Dieser Parameter sollte mit der Ländereinstellung der lokalen DB2-Instanzen kompatibel sein. Andernfalls kann der DB2-Verwaltungsserver nicht mit den DB2-Instanzen kommunizieren.

Dieser Parameter kann nur über einen Befehlszeilenprozessor (CLP) der Version 8 aktualisiert werden.

Zugehörige Referenzen:

- „das_territory - DAS-Gebiet “ auf Seite 591
- „GET ADMIN CONFIGURATION command“ in *Command Reference*
- „RESET ADMIN CONFIGURATION command“ in *Command Reference*
- „UPDATE ADMIN CONFIGURATION command“ in *Command Reference*

das_territory - DAS-Gebiet

Konfigurationstyp	DB2-Verwaltungsserver
Gilt für	DB2-Verwaltungsserver
Parametertyp	Online konfigurierbar
Weitergabeklasse	Sofort
Standardwert [Bereich]	NULL [beliebiges gültiges DB2-Gebiet]

Dieser Parameter zeigt das vom DB2-Verwaltungsserver verwendete Gebiet an. Wenn der Parameter auf NULL gesetzt ist, wird das Standardgebiet des Systems verwendet.

Dieser Parameter kann nur über einen Befehlszeilenprozessor (CLP) der Version 8 aktualisiert werden.

Zugehörige Referenzen:

- „das_codepage - DAS-Codepage “ auf Seite 591
- „GET ADMIN CONFIGURATION command“ in *Command Reference*
- „RESET ADMIN CONFIGURATION command“ in *Command Reference*
- „UPDATE ADMIN CONFIGURATION command“ in *Command Reference*

dasadm_group - DASADM-Gruppenname

Konfigurationstyp	DB2-Verwaltungsserver
Gilt für	DB2-Verwaltungsserver

Parametertyp	Konfigurierbar
Standardwert [Bereich]	NULL [beliebiger gültiger Gruppenname]

Die Berechtigung DASADM (DAS-Verwaltung) ist die höchste Berechtigungsstufe innerhalb des Datenbankverwaltungsservers (DAS). Dieser Parameter definiert den Gruppennamen mit DASADM-Berechtigung für den DAS.

Die Berechtigung DASADM wird von den in einer bestimmten Betriebsumgebung verwendeten Sicherheitseinrichtungen ermittelt.

- Bei Windows-Betriebssystemen kann dieser Parameter auf eine beliebige lokale Gruppe gesetzt werden, die in der Sicherheitsdatenbank von Windows definiert ist. Gruppennamen werden akzeptiert, solange sie maximal 30 Byte lang sind. Wenn für diesen Parameter der Wert „NULL“ angegeben wird, haben alle Mitglieder der Gruppe **Administratoren** die Berechtigung DASADM.
- Wenn auf UNIX-basierten Systemen der Wert „NULL“ für diesen Parameter angegeben wird, wird die Primärgruppe des Instanzeigners standardmäßig als DASADM-Gruppe verwendet.
Ist der Wert nicht „NULL“, kann jeder gültige UNIX-Gruppenname als DASADM-Gruppe definiert werden.

Dieser Parameter kann nur über einen Befehlszeilenprozessor (CLP) der Version 8 aktualisiert werden.

Zugehörige Referenzen:

- „GET ADMIN CONFIGURATION command“ in *Command Reference*
- „RESET ADMIN CONFIGURATION command“ in *Command Reference*
- „UPDATE ADMIN CONFIGURATION command“ in *Command Reference*

db2system - Name des DB2-Serversystems

Konfigurationstyp	DB2-Verwaltungsserver
Gilt für	DB2-Verwaltungsserver
Parametertyp	Online konfigurierbar
Standardwert [Bereich]	TCP/IP-Hostname [beliebiger gültiger Systemname]

Dieser Parameter gibt den Namen an, der von Ihren Benutzern und Datenbankadministratoren verwendet wird, um das DB2-Serversystem anzugeben. Dieser Name sollte nach Möglichkeit innerhalb Ihres Netzwerks eindeutig sein.

Dieser Name wird auf der Systemebene der Objektbaumstruktur in der Steuerzentrale angezeigt, um Administratoren bei der Identifizierung von Server-Systemen zu helfen, die von der Steuerzentrale aus verwaltet werden können.

Bei der Verwendung der Funktion zum Durchsuchen des Netzwerks des Konfigurationsassistenten gibt DB2-Discovery diesen Namen zurück, und er wird auf der Systemebene der resultierenden Baumstruktur angezeigt. Dieser Name unterstützt Benutzer bei der Identifizierung des Systems, das die Datenbank enthält, auf die Sie zugreifen wollen. Bei der Installation wird für *db2system* wie folgt ein Wert festgelegt:

- Unter Windows setzt das Installationsprogramm diesen Parameter auf den Wert des Computernamens, der für das Windows-System angegeben ist.

- Auf UNIX-Systemen wird für diesen Parameter der TCP/IP-Hostname des UNIX-Systems definiert.

Zugehörige Referenzen:

- „discover - DAS-Discovery-Modus “ auf Seite 593
- „GET ADMIN CONFIGURATION command“ in *Command Reference*
- „RESET ADMIN CONFIGURATION command“ in *Command Reference*
- „UPDATE ADMIN CONFIGURATION command“ in *Command Reference*

discover - DAS-Discovery-Modus

Konfigurationstyp	DB2-Verwaltungsserver
Gilt für	DB2-Verwaltungsserver
Parametertyp	Online konfigurierbar
Weitergabeklasse	Sofort
Standardwert [Bereich]	SEARCH [DISABLE; KNOWN; SEARCH]

Aus der Perspektive eines Verwaltungsservers bestimmt dieser Konfigurationsparameter den Typ des Discovery-Modus, der beim Start des DB2-Verwaltungsservers gestartet wird.

- Wenn für *discover* SEARCH angegeben ist, bearbeitet der Verwaltungsserver Discovery-Anforderungen mit dem Parameterwert SEARCH von Clients. SEARCH stellt eine Implikation der durch Discovery mit dem Parameter KNOWN bereitgestellten Funktionalität zur Verfügung. Wenn für *discover* SEARCH angegeben ist, bearbeitet der Verwaltungsserver sowohl Discovery-Anforderungen mit dem Parameterwert SEARCH als auch Discovery-Anforderungen mit dem Parameterwert KNOWN von Clients.
- Wenn für *discover* KNOWN angegeben ist, bearbeitet der Verwaltungsserver nur Discovery-Anforderungen mit dem Parameterwert KNOWN von Clients.
- Wenn für *discover* DISABLE angegeben ist, bearbeitet der Verwaltungsserver keine Discovery-Anforderungen. Die Informationen für dieses Serversystem sind im Wesentlichen vor Clients verdeckt.

Der Standard-Discovery-Modus ist SEARCH.

Dieser Parameter kann nur über einen Befehlszeilenprozessor (CLP) der Version 8 aktualisiert werden.

Zugehörige Referenzen:

- „db2system - Name des DB2-Serversystems “ auf Seite 592
- „GET ADMIN CONFIGURATION command“ in *Command Reference*
- „RESET ADMIN CONFIGURATION command“ in *Command Reference*
- „UPDATE ADMIN CONFIGURATION command“ in *Command Reference*

exec_exp_task - Verfallene Tasks ausführen

Konfigurationstyp	DB2-Verwaltungsserver
Gilt für	DB2-Verwaltungsserver
Parametertyp	Konfigurierbar
Standardwert [Bereich]	No [Yes; No]

Dieser Parameter gibt an, ob der Scheduler Tasks ausführt, die in der Vergangenheit terminiert, bislang jedoch noch nicht ausgeführt wurden. Der Scheduler erkennt verfallene Tasks nur dann, wenn er gestartet wird.

Wenn Sie beispielsweise die Ausführung eines Jobs für jeden Samstag terminiert haben und der Scheduler am Freitag ausgeschaltet und am Montag erneut gestartet wird, ist der für Samstag terminierte Job nun ein in der Vergangenheit terminierter Job. Wenn der Parameter *exec_exp_task* auf "Yes" gesetzt ist, wird der für Samstag terminierte Job beim Start des Schedulers ausgeführt.

Dieser Parameter kann nur über einen Befehlszeilenprozessor (CLP) der Version 8 aktualisiert werden.

Zugehörige Referenzen:

- „smtp_server - SMTP-Server “ auf Seite 596
- „sched_enable - Schedulermodus “ auf Seite 595
- „sched_userid - Benutzer-ID für Scheduler “ auf Seite 596
- „toolscat_db - Toolskatalogdatenbank “ auf Seite 597
- „toolscat_inst - Instanz der Toolskatalogdatenbank “ auf Seite 597
- „toolscat_schema - Schema der Toolskatalogdatenbank “ auf Seite 598
- „GET ADMIN CONFIGURATION command“ in *Command Reference*
- „RESET ADMIN CONFIGURATION command“ in *Command Reference*
- „UPDATE ADMIN CONFIGURATION command“ in *Command Reference*

jdk_64_path - Installationspfad für Software Developer's Kit (64 Bit) für Java auf DAS

Konfigurationstyp	DB2-Verwaltungsserver
Gilt für	DB2-Verwaltungsserver
Parametertyp	Online konfigurierbar
Weitergabeklasse	Sofort
Standardwert [Bereich]	NULL [beliebiger gültiger Pfad]

Dieser Parameter gibt das Verzeichnis an, in dem das 64-Bit-Software Developer's Kit (SDK) für Java installiert ist, das zu verwenden ist, um DB2-Verwaltungsserverfunktionen auszuführen.

Anmerkung: Dieser Parameter unterscheidet sich vom Konfigurationsparameter *jdk_path*, der ein 32-Bit-SDK für Java angibt.

Die vom Java-Interpreter verwendeten Umgebungsvariablen werden aus dem Wert dieses Parameters errechnet. Dieser Parameter wird nur auf den Plattformen verwendet, die 32- und 64-Bit-Instanzen unterstützen.

Bei den Plattformen handelt es sich um folgende:

- 64-Bit-Kernel für AIX, HP-UX und Solaris Operating System
- 64-Bit-Windows unter X64 und IPF
- 64-Bit-Linux-Kernel unter x86-64, POWER(TM) und zSeries(R).

Bei allen anderen Plattformen wird nur *jdk_path* verwendet.

Da es für diesen Parameter keinen Standardwert gibt, sollten Sie beim Installieren des SDK für Java einen Wert angeben.

Dieser Parameter kann nur über einen Befehlszeilenprozessor (CLP) der Version 8 aktualisiert werden.

Zugehörige Referenzen:

- „GET ADMIN CONFIGURATION command“ in *Command Reference*
- „RESET ADMIN CONFIGURATION command“ in *Command Reference*
- „UPDATE ADMIN CONFIGURATION command“ in *Command Reference*

jdk_path - Installationspfad für Software Developer's Kit für Java auf DAS

Konfigurationstyp	DB2-Verwaltungsserver
Gilt für	DB2-Verwaltungsserver
Parametertyp	Online konfigurierbar
Weitergabeklasse	Sofort
Standardwert [Bereich]	Java-Standardinstallationspfad [beliebiger gültiger Pfad]

Dieser Parameter gibt das Verzeichnis an, in dem das Software Developer's Kit (SDK) für Java installiert ist, das zu verwenden ist, um DB2-Verwaltungsserverfunktionen auszuführen. Die vom Java-Interpreter verwendeten Umgebungsvariablen werden aus dem Wert dieses Parameters errechnet.

Auf Windows-Betriebssystemen werden Java-Dateien (falls erforderlich) während der DB2-Installation im Verzeichnis sqllib (in java\jdk) gespeichert. Der Konfigurationsparameter *jdk_path* wird anschließend auf sqllib\java\jdk festgelegt. Java wird tatsächlich nicht durch DB2 auf Windows-Plattformen installiert. Die Dateien werden lediglich im Verzeichnis sqllib abgelegt, und zwar unabhängig davon, ob Java bereits installiert ist.

Dieser Parameter kann nur über einen Befehlszeilenprozessor (CLP) der Version 8 aktualisiert werden.

Zugehörige Referenzen:

- „GET ADMIN CONFIGURATION command“ in *Command Reference*
- „RESET ADMIN CONFIGURATION command“ in *Command Reference*
- „UPDATE ADMIN CONFIGURATION command“ in *Command Reference*

sched_enable - Schedulermodus

Konfigurationstyp	DB2-Verwaltungsserver
Gilt für	DB2-Verwaltungsserver
Parametertyp	Konfigurierbar
Standardwert [Bereich]	Off [On; Off]

Mit diesem Parameter wird angegeben, ob der Scheduler vom Verwaltungsserver gestartet wird oder nicht. Der Scheduler ermöglicht Tools wie der Taskzentrale das Terminieren und Ausführen von Tasks auf dem Verwaltungsserver.

Dieser Parameter kann nur über einen Befehlszeilenprozessor (CLP) der Version 8 aktualisiert werden.

Zugehörige Referenzen:

- „exec_exp_task - Verfallene Tasks ausführen “ auf Seite 593
- „sched_userid - Benutzer-ID für Scheduler “ auf Seite 596
- „smtp_server - SMTP-Server “ auf Seite 596
- „toolscat_db - Toolskatalogdatenbank “ auf Seite 597
- „toolscat_inst - Instanz der Toolskatalogdatenbank “ auf Seite 597
- „toolscat_schema - Schema der Toolskatalogdatenbank “ auf Seite 598
- „GET ADMIN CONFIGURATION command“ in *Command Reference*
- „RESET ADMIN CONFIGURATION command“ in *Command Reference*
- „UPDATE ADMIN CONFIGURATION command“ in *Command Reference*

sched_userid - Benutzer-ID für Scheduler

Konfigurationstyp	DB2-Verwaltungsserver
Gilt für	DB2-Verwaltungsserver
Parametertyp	Informativ
Standardwert [Bereich]	NULL [beliebige gültige Benutzer-ID]

Dieser Parameter gibt die Benutzer-ID an, die der Scheduler verwendet, um eine Verbindung zur Toolskatalogdatenbank herzustellen. Dieser Parameter ist nur dann relevant, wenn die Toolskatalogdatenbank eine ferne Datenbank für den DB2-Verwaltungsserver ist.

Die vom Scheduler für eine Verbindung zur fernen Toolskatalogdatenbank verwendete Benutzer-ID und das Kennwort werden mit dem Befehl **db2admin** angegeben.

Zugehörige Referenzen:

- „exec_exp_task - Verfallene Tasks ausführen “ auf Seite 593
- „sched_enable - Schedulermodus “ auf Seite 595
- „smtp_server - SMTP-Server “ auf Seite 596
- „toolscat_db - Toolskatalogdatenbank “ auf Seite 597
- „toolscat_inst - Instanz der Toolskatalogdatenbank “ auf Seite 597
- „toolscat_schema - Schema der Toolskatalogdatenbank “ auf Seite 598
- „GET ADMIN CONFIGURATION command“ in *Command Reference*

smtp_server - SMTP-Server

Konfigurationstyp	DB2-Verwaltungsserver
Gilt für	DB2-Verwaltungsserver
Parametertyp	Online konfigurierbar
Weitergabeklasse	Sofort
Standardwert [Bereich]	NULL [beliebiger gültiger TCP/IP-Hostname für SMTP-Server]

Dieser Parameter wird vom Scheduler und dem Diagnosemonitor verwendet.

Wenn der Scheduler aktiviert ist, gibt dieser Parameter den SMTP-Server an, den der Scheduler zum Versenden von E-Mail- und Pagerbenachrichtigungen verwendet.

Dieser Parameter kann nur über einen Befehlszeilenprozessor (CLP) der Version 8 aktualisiert werden.

Zugehörige Referenzen:

- „exec_exp_task - Verfallene Tasks ausführen “ auf Seite 593
- „sched_enable - Schedulermodus “ auf Seite 595
- „sched_userid - Benutzer-ID für Scheduler “ auf Seite 596
- „toolscat_db - Toolskatalogdatenbank “ auf Seite 597
- „toolscat_inst - Instanz der Toolskatalogdatenbank “ auf Seite 597
- „toolscat_schema - Schema der Toolskatalogdatenbank “ auf Seite 598
- „GET ADMIN CONFIGURATION command“ in *Command Reference*
- „RESET ADMIN CONFIGURATION command“ in *Command Reference*
- „UPDATE ADMIN CONFIGURATION command“ in *Command Reference*

toolscat_db - Toolskatalogdatenbank

Konfigurationstyp	DB2-Verwaltungsserver
Gilt für	DB2-Verwaltungsserver
Parametertyp	Konfigurierbar
Standardwert [Bereich]	NULL [beliebiger gültiger Datenbankaliasname]

Dieser Parameter gibt die vom Scheduler verwendete Toolskatalogdatenbank an. Diese Datenbank muss sich im Datenbankverzeichnis der Instanz befinden, das vom Parameter *toolscat_inst* angegeben ist.

Dieser Parameter kann nur über einen Befehlszeilenprozessor (CLP) der Version 8 aktualisiert werden.

Zugehörige Referenzen:

- „exec_exp_task - Verfallene Tasks ausführen “ auf Seite 593
- „sched_enable - Schedulermodus “ auf Seite 595
- „sched_userid - Benutzer-ID für Scheduler “ auf Seite 596
- „smtp_server - SMTP-Server “ auf Seite 596
- „toolscat_inst - Instanz der Toolskatalogdatenbank “ auf Seite 597
- „toolscat_schema - Schema der Toolskatalogdatenbank “ auf Seite 598
- „GET ADMIN CONFIGURATION command“ in *Command Reference*
- „RESET ADMIN CONFIGURATION command“ in *Command Reference*
- „UPDATE ADMIN CONFIGURATION command“ in *Command Reference*

toolscat_inst - Instanz der Toolskatalogdatenbank

Konfigurationstyp	DB2-Verwaltungsserver
Gilt für	DB2-Verwaltungsserver
Parametertyp	Konfigurierbar
Standardwert [Bereich]	NULL [beliebige gültige Instanz]

Dieser Parameter gibt den Instanznamen an, der vom Scheduler verwendet wird. Außerdem werden `toolscat_db` und `toolscat_schema` angegeben, um die Toolskatalogdatenbank zu kennzeichnen. Die Toolskatalogdatenbank (Tools Catalog) enthält die Taskinformationen, die von der Taskzentrale und der Steuerzentrale erstellt werden. Diese Toolskatalogdatenbank muss sich im Datenbankverzeichnis der Instanz befinden, das von diesem Konfigurationsparameter angegeben wird. Die Datenbank kann lokal oder fern sein. Bei Toolskatalogdatenbank muss die Instanz für TCP/IP konfiguriert sein. Bei einer fernen Datenbank muss der Knoten, der im Datenbankverzeichnis katalogisiert wird, ein TCP/IP-Knoten sein.

Dieser Parameter kann nur über einen Befehlszeilenprozessor (CLP) der Version 8 aktualisiert werden.

Zugehörige Tasks:

- „Einrichten und Konfigurieren der Toolskatalogdatenbank und des Schedulers des DB2-Verwaltungsservers (DAS)“ in *Systemverwaltung: Implementierung*

Zugehörige Referenzen:

- „exec_exp_task - Verfallene Tasks ausführen “ auf Seite 593
- „sched_enable - Schedulermodus “ auf Seite 595
- „sched_userid - Benutzer-ID für Scheduler “ auf Seite 596
- „smtp_server - SMTP-Server “ auf Seite 596
- „toolscat_db - Toolskatalogdatenbank “ auf Seite 597
- „toolscat_schema - Schema der Toolskatalogdatenbank “ auf Seite 598
- „GET ADMIN CONFIGURATION command“ in *Command Reference*
- „RESET ADMIN CONFIGURATION command“ in *Command Reference*
- „UPDATE ADMIN CONFIGURATION command“ in *Command Reference*

toolscat_schema - Schema der Toolskatalogdatenbank

Konfigurationstyp	DB2-Verwaltungsserver
Gilt für	DB2-Verwaltungsserver
Parametertyp	Konfigurierbar
Standardwert [Bereich]	NULL [beliebiges gültiges Schema]

Dieser Parameter gibt das Schema der vom Scheduler verwendeten Toolskatalogdatenbank an. Das Schema wird verwendet, um eine Reihe von Toolskatalogtabellen und -sichten innerhalb der Datenbank eindeutig anzugeben.

Dieser Parameter kann nur über einen Befehlszeilenprozessor (CLP) der Version 8 aktualisiert werden.

Zugehörige Referenzen:

- „exec_exp_task - Verfallene Tasks ausführen “ auf Seite 593
- „sched_enable - Schedulermodus “ auf Seite 595
- „sched_userid - Benutzer-ID für Scheduler “ auf Seite 596
- „smtp_server - SMTP-Server “ auf Seite 596
- „toolscat_db - Toolskatalogdatenbank “ auf Seite 597
- „toolscat_inst - Instanz der Toolskatalogdatenbank “ auf Seite 597
- „GET ADMIN CONFIGURATION command“ in *Command Reference*

- „RESET ADMIN CONFIGURATION command” in *Command Reference*
- „UPDATE ADMIN CONFIGURATION command” in *Command Reference*

Anhang B. Registrierdatenbank- und Umgebungsvariablen von DB2

In diesem Kapitel wird die Verwendung der Registrierdatenbankvariablen und Umgebungsvariablen beschrieben. Darüber hinaus enthält es eine nach Kategorien geordnete Liste der Variablen, in der die Syntax und die Verwendung der einzelnen Variablen erläutert wird.

Registrierdatenbank- und Umgebungsvariablen von DB2

DB2 stellt eine Reihe von Registrierdatenbankvariablen und Umgebungsvariablen bereit, die Sie kennen sollten, um DB2 betriebsbereit zu machen.

Zum Anzeigen einer Liste aller unterstützten Registrierdatenbankvariablen führen Sie den folgenden Befehl aus:

```
db2set -lr
```

Zum Ändern des Werts für eine Variable in der aktuellen Instanz bzw. in der Standardinstanz führen Sie den folgenden Befehl aus:

```
db2set name_der_registrierdatenbankvariablen=neuer_wert
```

Ob die DB2-Umgebungsvariablen DB2INSTANCE, DB2NODE, DB2PATH und DB2INSTPROF in den DB2-Profilregistrierdatenbanken gespeichert werden, hängt vom Betriebssystem ab. Zur Aktualisierung dieser Umgebungsvariablen verwenden Sie den Befehl `set`. Diese Änderungen werden nur für das lokale (aktuelle) Befehlsfenster wirksam und bleiben bis zum nächsten Neustart des Systems wirksam. Auf UNIX-Plattformen können Sie den Befehl `export` an Stelle des Befehls `set` verwenden.

Sie müssen die geänderten Werte für die Registrierdatenbankvariablen definieren, bevor Sie den Befehl `DB2START` ausführen.

Anmerkung: Wenn eine Registrierdatenbankvariable Boolesche Werte als Argumente erfordert, sind einerseits die Werte YES, 1 und ON sowie andererseits die Werte NO, 0 und OFF äquivalent. Für jede Variable können Sie einen beliebigen der entsprechenden äquivalenten Werte angeben.

Zugehörige Konzepte:

- „Umgebungsvariablen und die Profilregistrierdatenbank“ in *Systemverwaltung: Implementierung*

Zugehörige Tasks:

- „Definieren von DB2-Registrierdatenbankvariablen auf Benutzerebene in der LDAP-Umgebung“ in *Systemverwaltung: Implementierung*

Zugehörige Referenzen:

- „Befehlszeilenvariablen“ auf Seite 621
- „Kommunikationsvariablen“ auf Seite 615
- „Verschiedene Variablen“ auf Seite 649

- „MPP-Konfigurationsvariablen“ auf Seite 623
- „Leistungsvariablen“ auf Seite 630
- „Abfragecompilervariablen“ auf Seite 624
- „Systemumgebungsvariablen“ auf Seite 607

Registrierdatenbank- und Umgebungsvariablen nach Kategorie

Die folgenden Abschnitte listen die Registrierdatenbank- und Umgebungsvariablen nach den verschiedenen Aspekten des Datenbankmanagers bzw. der Datenbankfunktion auf, die sie beeinflussen.

Allgemeine Registrierdatenbankvariablen

DB2ACCOUNT

- Betriebssystem: Alle
- Standardwert=null
- Diese Variable definiert die Abrechnungszeichenfolge, die an den fernen Host gesendet wird. Einzelheiten siehe *DB2 Connect Benutzerhandbuch*.

DB2BIDI

- Betriebssystem: Alle
- Standardwert=NO, Werte: YES oder NO
- Diese Variable aktiviert die bidirektionale Unterstützung von Sprachen, und die Variable DB2CODEPAGE dient zur Deklaration der zu verwendenden Codepage.

DB2CODEPAGE

- Betriebssystem: Alle
- Standardwert: abgeleitet aus dem vom Betriebssystem definierten Landescode
- Diese Variable gibt die Codepage der Daten an, die an DB2 für Datenbankclientanwendungen übergeben werden. Der Benutzer sollte DB2CODEPAGE nicht definieren, sofern dies nicht explizit in DB2-Dokumenten angegeben ist oder er vom DB2-Service dazu angeleitet wird. Wird DB2CODEPAGE auf einen Wert gesetzt, der vom Betriebssystem nicht unterstützt wird, können unerwartete Ergebnisse auftreten. Im Normalfall brauchen Sie DB2CODEPAGE nicht zu definieren, weil DB2 die Informationen zur Codepage automatisch aus dem Betriebssystem abrufen.

Anmerkung: Da Windows keine Unicode-Codepage (in den Ländereinstellungen von Windows) an Stelle der ANSI-Codepage meldet, kann sich eine Windows-Anwendung nicht wie ein Unicode-Client verhalten. Um dieses Verhalten außer Kraft zu setzen, setzen Sie die Registrierdatenbankvariable DB2CODEPAGE auf 1208 (für die Unicode-Codepage), damit die Anwendung sich wie eine Unicode-Anwendung verhält.

DB2_COLLECT_TS_REC_INFO

- Betriebssystem: Alle
- Standardwert=ON, Werte: ON oder OFF
- Diese Variable gibt an, ob DB2 alle Protokolldateien bei einer aktualisierenden Recovery eines Tabellenbereichs verarbeitet, unabhängig davon,

ob die Protokolldateien Protokollsätze enthalten, die den Tabellenbereich betreffen. Wenn die Protokolldateien übersprungen werden sollen, die bekanntermaßen keine Protokollsätze für diesen Tabellenbereich enthalten, setzen Sie diese Variable auf den Wert ON.

DB2_COLLECT_TS_REC_INFO muss definiert werden, bevor die Protokolldateien erstellt und verwendet werden, so dass die Informationen, die zum Überspringen von Protokolldateien erforderlich sind, erfasst werden.

DB2_CONNRETRIES_INTERVAL

- Betriebssystem: Alle
- Standardwert= nicht gesetzt, Werte: ganzzahlige Werte in Sekunden
- Die Variable gibt die Ruhezeit (in Sekunden) zwischen aufeinanderfolgenden Verbindungsversuchen für die automatische Clientweiterleitungsfunktion an. Sie können diese Variable in Verbindung mit DB2_MAX_CLIENT_CONNRETRIES verwenden, um das Wiederholungsverhalten für die automatische Clientweiterleitung zu konfigurieren.

Wenn DB2_MAX_CLIENT_CONNRETRIES definiert ist, jedoch DB2_CONNRETRIES_INTERVAL nicht, nimmt DB2_CONNRETRIES_INTERVAL den Standardwert 30 an. Wenn DB2_MAX_CLIENT_CONNRETRIES nicht definiert ist, jedoch DB2_CONNRETRIES_INTERVAL definiert ist, gilt für DB2_MAX_CLIENT_CONNRETRIES der Standardwert 10. Wenn weder DB2_MAX_CLIENT_CONNRETRIES noch DB2_CONNRETRIES_INTERVAL definiert ist, kehrt die Funktion der automatischen Clientweiterleitung zur Standardfunktionsweise zurück. Das heißt, sie versucht die Verbindung zu einer Datenbank über eine Dauer von bis zu 10 Minuten wiederholt herzustellen.

DB2CONSOLECP

- Betriebssystem: Windows
- Standardwert= null, Werte: alle gültigen Codepagewerte
- Gibt die Codepage für die Anzeige von DB2-Nachrichtentext an. Wenn angegeben, überschreibt dieser Wert die Einstellung für die Codepage des Betriebssystems.

DB2COUNTRY

- Betriebssystem: Windows
- Standardwert=null, Werte: alle gültigen numerischen Codes für Land, Gebiet oder Region
- Diese Variable gibt den Code für Land, Gebiet oder Region der Clientanwendung an. Wenn angegeben, überschreibt dieser Wert die Einstellung des Betriebssystems.

DB2DBDFT

- Betriebssystem: Alle
- Standardwert=null
- Diese Variable gibt den Aliasnamen der Datenbank an, die für implizite Verbindungen zu verwenden ist. Wenn eine Anwendung keine Datenbankverbindung hat, aber SQL- oder XQuery-Anweisungen abgesetzt werden, wird eine implizite Verbindung hergestellt, wenn die Umgebungsvariable DB2DBDFT mit einer Standarddatenbank definiert wurde.

DB2DBMSADDR

- Betriebssystem: Windows (32-Bit)
- Standardwert= 0x20000000, Werte: 0x20000000 bis 0xB0000000 in Inkrementen von 0x10000
- Diese Variable gibt die Standardadresse des Datenbankmanagers für gemeinsam benutzten Speicher im Hexadezimalformat an. Wenn der Befehl *db2start* auf Grund einer Adresskollision beim gemeinsam benutzen Speicher fehlschlägt, kann diese Registrierdatenbankvariable geändert werden, um die Datenbankmanagerinstanz zu zwingen, den gemeinsam benutzen Speicher an einer anderen Adresse anzulegen.

DB2DISCOVERYTIME

- Betriebssystem: Windows
- Standardwert=40 Sekunden, Minimalwert=20 Sekunden
- Diese Variable gibt die Zeitdauer an, die der Discovery-Prozess SEARCH nach DB2-Systemen sucht.

DB2FFDC

- Betriebssystem: Alle
- Standardwert: ON, Werte: ON, CORE:OFF
- Bietet die Möglichkeit, die Generierung der Kerndatei zu inaktivieren. Standardmäßig ist diese Registrierdatenbankvariable auf ON gesetzt. Wenn diese Registrierdatenbankvariable nicht gesetzt oder auf einen anderen Wert als CORE:OFF gesetzt ist, werden die Kerndateien möglicherweise generiert, wenn der DB2-Server abnormal beendet wird. (Kerndateien werden zur Fehlerbestimmung verwendet und im DIAGPATH erstellt.)

Anmerkung: Auf Linux-Plattformen ist die standardmäßige Größengrenzung für Kerndateien auf 0 gesetzt (d. h., ulimit -c). Bei dieser Einstellung werden keine Kerndateien generiert. Wenn Kerndateien auf Linux-Plattformen erstellt werden sollen, muss der Wert unbegrenzt sein.

Kerndateien enthalten das gesamte Prozessimage des DB2-Prozesses, der beendet wird. Ein besonderes Augenmerk sollte dem verfügbaren Dateispeicherspeicher gelten, da Kerndateien ziemlich groß sein können. Die Größe hängt von der DB2-Konfiguration und dem Status des Prozesses zum Zeitpunkt des Auftretens des Problems ab.

DB2_FORCE_APP_ON_MAX_LOG

- Betriebssystem: Alle
- Standardwert: TRUE, Werte: TRUE, FALSE
- Gibt an, was geschieht, wenn der Wert des Konfigurationsparameters MAX_LOG überschritten wird. Wenn TRUE, wird die Anwendung zwangsweise von der Datenbank getrennt und die UOW (Unit of Work, Arbeitseinheit) mit ROLLBACK rückgängig gemacht.

Bei FALSE schlägt die aktuelle Anweisung fehl. Die Anwendung kann weiterhin von vorherigen Anweisungen in der UOW fertiggestellte Arbeit mit COMMIT festschreiben, oder sie kann die ausgeführte Arbeit mit ROLLBACK rückgängig machen, um die UOW zurückzusetzen.

DB2GRAPHICUNICODESERVER

- Betriebssystem: Alle
- Standardwert=OFF, Werte: ON oder OFF

- Diese Registrierdatenbankvariable dient zur Zulassung vorhandener Anwendungen, die zum Einfügen von Grafikdaten in eine Unicode-Datenbank geschrieben sind. Ihre Verwendung ist nur für Anwendungen erforderlich, die sqldbchar-Daten (Grafikdaten) speziell in Unicode anstatt in der Codepage des Clients senden. (sqldbchar ist ein unterstützter SQL-Datentyp in C und C++, der ein einzelnes Doppelbytezeichen aufnehmen kann.) Wenn Sie diese Registrierdatenbankvariable auf den Wert „ON“ setzen, teilen Sie der Datenbank mit, dass der Eingang von Unicode-Grafikdaten zu erwarten ist, und die Anwendung erwartet ebenfalls, Unicode-Grafikdaten zu empfangen.

DB2INCLUDE

- Betriebssystem: Alle
- Standardwert = aktuelles Verzeichnis
- Gibt einen Pfad an, der bei der Verarbeitung der SQL-Anweisung INCLUDE textdatei während der Ausführung des DB2-Befehls PREP zu verwenden ist. Er gibt eine Liste von Verzeichnissen an, in denen sich die INCLUDE-Datei befinden könnte. Im Handbuch *Developing Embedded SQL Applications* finden Sie Beschreibungen, wie DB2INCLUDE in den verschiedenen vorkompilierten Sprachen verwendet wird.

DB2INSTDEF

- Betriebssystem: Windows
- Standardwert = DB2
- Diese Variable definiert den Wert, der zu verwenden ist, wenn DB2INSTANCE nicht definiert ist.

DB2INSTOWNER

- Betriebssystem: Windows
- Standardwert=null
- Die Registrierdatenbankvariable, die bei der ersten Erstellung der Instanz in der DB2-Profilregistrierdatenbank erstellt wird. Diese Variable wird auf den Namen der Instanzeignermaschine gesetzt.

DB2_LIC_STAT_SIZE

- Betriebssystem: Alle
- Standardwert=null, Bereich: 0 bis 32 767
- Diese Variable legt die Maximalgröße (in MB) der Datei mit den Lizenzstatistikdaten für das System fest. Der Wert 0 schaltet das Sammeln von Lizenzstatistikdaten aus. Wenn die Variable nicht erkannt wird oder nicht definiert ist, wird standardmäßig eine uneingeschränkte Größe angenommen. Die Statistikdaten werden über die Lizenzzentrale angezeigt.

DB2LOCALE

- Betriebssystem: Alle
- Standardwert=NO, Werte: YES oder NO
- Diese Variable gibt an, ob das Standardlocale "C" eines Prozesses nach dem Aufrufen von DB2 auf das Standardlocale "C" zurückgesetzt wird und ob das Prozesslocale nach dem Aufrufen einer DB2-Funktion auf das ursprüngliche 'C' zurückzusetzen ist. Wenn das ursprüngliche Locale nicht 'C' war, wird diese Registrierdatenbankvariable ignoriert.

DB2_MAX_CLIENT_CONNRETRIES

- Betriebssystem: Alle

- Standardwert=nicht gesetzt, Werte: eine ganze Zahl für die maximale Anzahl von Wiederholungsversuchen zur Herstellung der Verbindung
- Diese Variable gibt die maximale Anzahl von Verbindungsversuchen an, die von der Funktion zur automatischen Clientweiterleitung unternommen werden sollen. Sie können diese Variable in Verbindung mit DB2_CONNRETRIES_INTERVAL verwenden, um das Wiederholungsverhalten für die automatische Clientweiterleitung zu konfigurieren.

Wenn DB2_MAX_CLIENT_CONNRETRIES definiert ist, jedoch DB2_CONNRETRIES_INTERVAL nicht, nimmt DB2_CONNRETRIES_INTERVAL den Standardwert 30 an. Wenn DB2_MAX_CLIENT_CONNRETRIES nicht definiert ist, jedoch DB2_CONNRETRIES_INTERVAL definiert ist, gilt für DB2_MAX_CLIENT_CONNRETRIES der Standardwert 10. Wenn weder DB2_MAX_CLIENT_CONNRETRIES noch DB2_CONNRETRIES_INTERVAL definiert ist, kehrt die Funktion der automatischen Clientweiterleitung zur Standardfunktionsweise zurück. Das heißt, sie versucht die Verbindung zu einer Datenbank über eine Dauer von bis zu 10 Minuten wiederholt herzustellen.

DB2NBDISCOVERRCVBUFS

- Betriebssystem: Alle
- Standardwert=16 Puffer, Minimalwert=16 Puffer
- Diese Variable wird für den NetBIOS-Discovery-Prozess (SEARCH) verwendet. Die Variable gibt die Anzahl der gleichzeitigen Discovery-Antworten an, die von einem Client empfangen werden können. Wenn der Client mehr gleichzeitige Antworten empfängt, als von dieser Variable festgelegt, werden die überzähligen Antworten von der NetBIOS-Schicht gelöscht. Der Standardwert gibt 16 NetBIOS-Empfangspuffer an. Wenn ein kleinerer Wert als der Standardwert gewählt wird, wird der Standardwert verwendet.

DB2_OBJECT_TABLE_ENTRIES

- Betriebssystem: Alle
 - Standardwert=0, Werte: 0–65532
- Der tatsächliche Maximalwert, der auf Ihrem System möglich ist, hängt von der Seitengröße und der Speicherbereichsgröße (EXTENTSIZE) ab, kann jedoch den Wert 65.532 nicht überschreiten.
- Diese Variable gibt die erwartete Anzahl von Objekten in einem Tabellenbereich an. Wenn Sie wissen, dass in einem DMS-Tabellenbereich eine große Anzahl von Objekten (z. B. 1000 oder mehr) erstellt wird, sollten Sie diese Registrierdatenbankvariable auf die ungefähre Anzahl setzen, bevor Sie den Tabellenbereich erstellen. Dadurch wird zusammenhängender Speicherplatz für Objektmetadaten bei der Erstellung des Tabellenbereichs reserviert. Die Reservierung von Speicher verringert die Wahrscheinlichkeit, dass ein Online-Backup Operationen blockiert, die Einträge in den Metadaten aktualisieren (wie z. B. CREATE INDEX, IMPORT REPLACE). Sie erleichtert außerdem eine spätere Änderung der Größe des Tabellenbereichs, weil die Metadaten am Anfang des Tabellenbereichs gespeichert werden.

Wenn die Anfangsgröße des Tabellenbereichs nicht ausreichend groß ist, um zusammenhängenden Speicher zu reservieren, wird die Erstellung des Tabellenbereichs fortgesetzt, ohne den zusätzlichen Speicher zu reservieren.

DB2TERRITORY

- Betriebssystem: Alle
- Standardwert: abgeleitet aus dem vom Betriebssystem definierten Landescode
- Diese Variable gibt den Regions- oder Gebietscode der Clientanwendung an, was sich auf die Formate für Datum und Uhrzeit auswirkt.

DB2_VIEW_REOPT_VALUES

- Betriebssystem: Alle
- Standardwert=NO, Werte: YES, NO
- Diese Variable gibt allen Benutzern die Möglichkeit, die im Cache abgelegten Werte einer reoptimierten SQL- oder XQuery-Anweisung in der Tabelle EXPLAIN_PREDICATE zu speichern, wenn die Anweisung mit EXPLAIN bearbeitet wird. Wenn diese Variable auf NO gesetzt ist, kann nur der DBADM diese Werte in der Tabelle EXPLAIN_PREDICATE speichern.

Zugehörige Konzepte:

- „Registrierdatenbank- und Umgebungsvariablen von DB2“ auf Seite 601

Systemumgebungsvariablen

DB2_ALTERNATE_GROUP_LOOKUP

- Betriebssystem: AIX
- Standardwert: NULL, Werte: NULL oder GETGRSET
- Diese Variable ermöglicht es DB2, Gruppeninformationen aus einer alternativen Quelle abzurufen, die vom Betriebssystem bereitgestellt wird. Unter AIX wird die Funktion 'getgrset' verwendet. Diese Funktion bietet die Möglichkeit, Gruppen aus einer anderen Position als lokalen Dateien über ladbare Authentifizierungsmodule (Loadable Authentication Modules) abzurufen.

DB2_CLP_EDITOR

- Betriebssystem: Alle
- Standardwert: Windows-Plattformen: 'Notepad', UNIX: 'vi', Werte: Alle gültigen Editoren, die sich im Betriebssystempfad befinden.

Anmerkung: Diese Registrierdatenbankvariable wird während der Installation nicht auf den Standardwert gesetzt. Stattdessen verwendet der Code, der diese Variable verwendet, einen Standardwert, wenn die Registrierdatenbankvariable nicht festgelegt ist.

- Diese Variable legt den Editor fest, der bei der Ausführung des Befehls **EDIT** verwendet wird. In einer interaktiven CLP-Sitzung startet der Befehl **EDIT** einen Editor, der mit einem benutzerdefinierten Befehl vorinstalliert ist, der bearbeitet und ausgeführt werden kann.

DB2_CLPHISTSIZE

- Betriebssystem: Alle
- Standardwert: 20, Werte: 1 - 500 einschließlich

Anmerkung: Diese Registrierdatenbankvariable wird während der Installation nicht auf den Standardwert festgelegt. Stattdessen verwendet der Code, der diese Variable verwendet, einen Standardwert von 20, wenn die Registrierdatenbank-

variable nicht festgelegt ist oder wenn sie auf einen Wert festgelegt wurde, der sich außerhalb des gültigen Bereichs befindet.

- Diese Variable legt die Anzahl der Befehle fest, die im Befehlsprotokoll während interaktiver CLP-Sitzungen gespeichert werden. Da das Befehlsprotokoll im Speicher gehalten wird, kann ein sehr hoher Wert für diese Variable zu Auswirkungen auf die Leistung führen. Diese hängen von der Anzahl und Länge der Befehle ab, die in einer Sitzung ausgeführt werden.

DB2CONNECT_IN_APP_PROCESS

- Betriebssystem: Alle
- Standardwert=YES, Werte: YES oder NO
- Wenn Sie diese Variable auf den Wert NO setzen, werden lokale DB2 Connect-Clients auf einer Maschine mit DB2 Connect Enterprise Edition gezwungen, innerhalb eines Agenten aktiv zu sein. Einige Vorteile der Ausführung innerhalb eines Agenten bestehen darin, dass lokale Clients auf diese Weise überwacht werden und die SYSPLEX-Unterstützung nutzen können.

DB2_COPY_NAME

- Betriebssystem: Windows
- Standardwert: Der Name der Standardkopie von DB2, die auf Ihrer Maschine installiert ist. Werte: Der Name einer Kopie von DB2, die auf Ihrer Maschine installiert ist. Der Name kann maximal 128 Zeichen lang sein.
- Die Variable DB2_COPY_NAME speichert den Namen der DB2-Kopie, die momentan verwendet wird. Wenn Sie mehrere DB2-Kopien auf Ihrer Maschine installiert haben, können Sie DB2_COPY_NAME nicht zum Wechseln zu einer anderen Kopie von DB2 verwenden. Sie müssen den Befehl [INSTALLPATH]\bin\db2envvars.bat ausführen, um die momentan verwendete Kopie zu wechseln.

DB2DBMSADDR

- Betriebssystem: Linux unter x86 und Linux unter zSeries (31-Bit)
- Standardwert: null, Werte: virtuelle Adressen aus dem Bereich von 0x09000000 bis 0xB0000000 in Inkrementen von 0x10000
- Die Registrierdatenbankvariable DB2DBMSADDR gibt die Standardadresse der Datenbank für gemeinsam genutzten Speicher im Hexadezimalformat an.

Anmerkung: Eine falsche Adresse kann schwer wiegende Probleme auf dem DB2-System verursachen, die von einem Fehlschlagen des Starts einer DB2-Instanz bis hin zum Fehlschlagen der Verbindung zur Datenbank reichen können. Eine falsche Adresse ist eine Adresse, die einen Konflikt mit einem Adressbereich verursacht, der bereits verwendet wird oder der für andere Daten oder Programme reserviert ist. Dieses Problem können Sie beheben, indem Sie die Registrierdatenbankvariable DB2DBMSADDR mit dem folgenden Befehl auf null setzen:

```
db2set DB2DBMSADDR=
```

Diese Variable können Sie in Verbindung mit DB2_MAPPED_BASE oder alleine setzen, um die Adressraumbelegung von DB2-Prozessen feinabzustimmen. Diese Variable ändert die Position des gemeinsam genutzten

Speichers für die Instanz von der aktuellen Position bei der virtuellen Adresse 0x10000000 in den neuen Wert.

DB2DOMAINLIST

- Betriebssystem: Nur Windows-Server
- Standardwert=null, Werte: Eine Liste mit Windows-Domänennamen, die durch Kommata (,,,"") getrennt werden.
- Diese Variable definiert eine oder mehrere Windows-Domänen. Diese Liste definiert die Domänen, für die die anfordernde Benutzer-ID authentifiziert wird. Nur CONNECT- oder ATTACH-Verbindungsanforderungen von Benutzern, die zu diesen Domänen gehören, werden akzeptiert.

Diese Variable ist nur wirksam, wenn in der Konfiguration des Datenbankmanagers die Clientauthentifizierung definiert ist, und wird benötigt, wenn eine einzige Anmeldung über einen Windows-Desktop in einer Windows-Domänenumgebung erforderlich ist.

Diese Registrierdatenbankvariable sollte nur in einer reinen Windows-Domänenumgebung mit DB2-Servern und -Clients verwendet werden, die DB2 Universal Database Version 7.1 (oder höher) ausführen.

DB2ENVLIST

- Betriebssystem: UNIX
- Standardwert: null
- Diese Variable listet spezifische Variablennamen für gespeicherte Prozeduren oder für benutzerdefinierte Funktionen auf. Standardmäßig filtert der Befehl **db2start** alle Benutzerumgebungsvariablen außer die heraus, die das Präfix **DB2** oder **db2** haben. Wenn bestimmte Umgebungsvariablen entweder an gespeicherte Prozeduren oder an benutzerdefinierte Funktionen übergeben werden müssen, können Sie die Variablennamen in der Umgebungsvariablen DB2ENVLIST auflisten. Trennen Sie die einzelnen Variablennamen durch ein oder mehrere Leerzeichen.

DB2INSTANCE

- Betriebssystem: Alle
- Standardwert=DB2INSTDEF bei 31-Bit-Windows-Betriebssystemen
- Die Umgebungsvariable, die zur Definition der Instanz dient, die standardmäßig aktiv ist. Unter UNIX müssen Benutzer einen Wert für DB2INSTANCE angeben.

DB2INSTPROF

- Betriebssystem: Windows
- Standardwert: null
- Die Umgebungsvariable, die zur Angabe der Position des Instanzverzeichnisses auf Windows-Betriebssystemen dient, wenn sie von DB2PATH abweicht.

DB2LIBPATH

- Betriebssystem: UNIX
- Standardwert: null
- DB2 bildet einen eigenen gemeinsamen Bibliothekspfad. Wenn Sie einen Pfad (PATH) zu dem Bibliothekspfad der Steuerkomponente hinzufügen möchten (z. B. erfordert eine benutzerdefinierte Funktion unter AIX einen bestimmten Eintrag in LIBPATH), müssen Sie die Variable

DB2LIBPATH definieren. Der tatsächliche Wert der Variablen DB2LIBPATH wird an das Ende des von DB2 gebildeten gemeinsam genutzten Bibliothekspfad angehängt.

DB2LOGINRESTRICTIONS

- Betriebssystem: AIX
- Standardwert: LOCAL, Werte: LOCAL, REMOTE, SU, NONE
- Durch diese Registrierdatenbankvariable können Sie eine AIX-Betriebssystem-API mit der Bezeichnung `loginrestrictions()` verwenden. Diese API bestimmt, ob ein Benutzer für den Zugriff auf das System berechtigt ist. Durch Aufrufen dieser API kann die DB2-Datenbanksicherheitsfunktion die Anmeldeeinschränkungen umsetzen, die durch das Betriebssystem definiert sind. Es gibt unterschiedliche Werte, die an diese API übergeben werden können, wenn diese Registrierdatenbankvariable verwendet wird. Diese Werte sind:
 - REMOTE
DB2 setzt Anmeldeeinschränkungen nur um, um zu überprüfen, ob das Konto für ferne Anmeldevorgänge über die Programme *rlogind* bzw. *telnetd* verwendet werden kann.
 - SU
DB2 Version 9.1 setzt SU-Einschränkungen nur um, um zu überprüfen, ob der Befehl „su“ erlaubt ist und ob der aktuelle Prozess über eine Gruppen-ID verfügt, die den Befehl „su“ aufrufen kann, um zu dem Konto umzuschalten.
 - NONE
DB2 setzt keinerlei Anmeldeeinschränkungen um.
 - LOCAL (oder die Variable ist nicht gesetzt)
DB2 setzt Anmeldeeinschränkungen nur um, um zu überprüfen, ob für dieses Konto lokale Anmeldungen erlaubt sind. Dies ist das normale Verhalten bei einer Anmeldung.

Ganz gleich, welche dieser Optionen Sie setzen, Benutzerkonten bzw. -IDs, die über die angegebenen Zugriffsrechte verfügen, können DB2 sowohl lokal auf dem Server als auch über ferne Clients erfolgreich verwenden. Eine Beschreibung der API `loginrestrictions()` finden Sie in der AIX-Dokumentation.

DB2_MAPPED_BASE

- Betriebssystem: Linux unter x86 und Linux unter zSeries (31-Bit)
- Standardwert: null, Werte: 0 oder (hexadezimale) virtuelle Adresse im 31-Bit- und 32-Bit-Bereich oder null (nicht gesetzt).
- Mit der Registrierdatenbankvariablen DB2_MAPPED_BASE können Sie den für einen DB2-Prozess verfügbaren zusammenhängenden virtuellen Adressraum erhöhen, indem Sie die Anschlussadresse der gemeinsam genutzten Bibliotheken für den bestimmten Prozess verlagern. Der zusammenhängende virtuelle Adressraum ist wichtig, um die Größe des für die gemeinsame Datenbanknutzung verfügbaren Speichers für das DB2-Datenbanksystem zu maximieren. Diese Variable gilt nur bei Verteilungen, für die sich im Dateisystem *proc* die Datei 'mapped-base' im Verzeichnis für die Prozessidentifikation befindet.

Wenn diese Variable nicht gesetzt ist, versucht DB2, die gemeinsam genutzten Bibliotheken an die virtuelle Adresse 0x20000000 zu verlagern. Die Registrierdatenbankvariable kann auch auf eine beliebige virtuelle Adresse (hexadezimal) im Bereich des 31- und 32-Bit-Adressraums

gesetzt werden, sofern die neue Adresse die gemeinsam genutzten Bibliotheken im Adressraum niedriger platziert.

Anmerkung: Eine falsche Adresse kann schwer wiegende Probleme auf einem DB2-Datenbanksystem verursachen, die von einem Fehlschlagen des Starts von DB2-Datenbankinstanzen bis hin zum Fehlschlagen der Verbindung zur Datenbank reichen können. Eine falsche Adresse ist eine Adresse, die einen Konflikt mit einem Adressbereich verursacht, der bereits verwendet wird oder der für andere Daten oder Programme reserviert ist. Dieses Problem können Sie beheben, indem Sie die Registrierdatenbankvariable DB2_MAPPED_BASE mit dem folgenden Befehl auf null setzen:

```
db2set DB2_MAPPED_BASE=
```

Die folgende Nachricht kann mehrmals in der Datei 'db2diag.log' vorkommen, da diese Änderung jeweils einmal pro logischen Knoten erforderlich ist:

```
ADM0506I DB2 hat den Kernelparameter "mapped_base"
automatisch von "0x40000000(hex) 1073741824(dec)" auf
den empfohlenen Wert "0x20000000(hex) 536870912(dec)" aktualisiert.
```

Diese Nachricht wird nur angezeigt, wenn das Setzen der Registrierdatenbankvariablen erfolgreich ist. In der Nachricht ist die Adresse enthalten, unter der sich die gemeinsam genutzten Dateien befinden.

DB2NODE

- Betriebssystem: Alle
- Standardwert: null, Werte: 1 bis 999
- Dient zur Angabe des logischen Zielknotens eines Datenbankpartitions-servers unter DB2 Enterprise Server Edition, zu dem eine Verbindung über CONNECT oder ATTACH hergestellt werden soll. Wenn diese Variable nicht definiert wird, wird als logischer Zielknoten standardmäßig der logische Knoten angenommen, der auf der Maschine mit Port 0 definiert ist.

DB2OPTIONS

- Betriebssystem: Alle
- Standardwert: null
- Dient zum Festlegen der Optionen für den Befehlszeilenprozessor.

DB2_PARALLEL_IO

- Betriebssystem: Alle
- Standardwert: null, Werte: *Tabellenbereichs-ID[:n],...* – eine durch Kom-mata getrennte Liste definierter Tabellenbereiche (angegeben durch ihre numerische Tabellenbereichs-ID). Wenn PREFETCHSIZE eines Tabellenbereichs auf AUTOMATIC gesetzt ist, können Sie dem DB2-Datenbank-manager die Anzahl Platten pro Container für diesen Tabellenbereich mitteilen, indem Sie die Tabellenbereichs-ID, gefolgt von einem Doppelpunkt, gefolgt von der Anzahl Platten pro Container *n* angeben. Wird *n* nicht angegeben, ist der Standardwert 6.

Sie können 'Tabellenbereichs-ID' durch einen Stern (*) ersetzen, um alle Tabellenbereiche anzugeben. Wenn Sie zum Beispiel 'DB2_PARALLEL_IO=*' angeben, verwenden alle Tabellenbereiche 6 als Anzahl von Platten pro Container. Wenn Sie sowohl einen Stern als auch eine Tabellenbereichs-ID angeben, hat die Einstellung der Tabellen-

bereichs-ID Vorrang. Beispiel: Bei der Angabe 'DB2_PARALLEL_IO=*,1:3' verwenden alle Tabellenbereiche 6 als Anzahl der Platten pro Container mit Ausnahme des Tabellenbereichs 1, der 3 Platten pro Container verwendet.

- Diese Registrierdatenbankvariable dient zur Änderung der Art und Weise, wie DB2 die E/A-Parallelität eines Tabellenbereichs berechnet. Wenn die E/A-Parallelität aktiviert ist (entweder implizit durch die Verwendung mehrerer Container oder explizit durch die Einstellung der Variablen DB2_PARALLEL_IO), wird sie umgesetzt, indem die richtige Anzahl von Vorablesezugriffsanforderungen abgesetzt wird. Jede Vorablesezugriffsanforderung ist eine Anforderung für einen EXTENTSIZE großen Speicherbereich von Seiten.

Wenn diese Registrierdatenbankvariable nicht definiert ist, entspricht der Grad der Parallelität eines Tabellenbereichs der Anzahl der Container des Tabellenbereichs. Wenn DB2_PARALLEL_IO zum Beispiel auf den Wert null gesetzt wird und ein Tabellenbereich über vier Container verfügt, werden vier Vorablesezugriffsanforderungen in der Größe von EXTENTSIZE abgesetzt.

Wenn diese Registrierdatenbankvariable definiert ist und PREFETCHSIZE für die Tabelle nicht auf AUTOMATIC gesetzt ist, ist der Grad der Parallelität des Tabellenbereichs gleich dem Wert für PREFETCHSIZE dividiert durch den Wert von EXTENTSIZE. Wenn DB2_PARALLEL_IO zum Beispiel für einen Tabellenbereich definiert ist, dessen PREFETCHSIZE-Wert 160 und dessen EXTENTSIZE-Wert 32 Seiten beträgt, werden fünf EXTENTSIZE große Vorablesezugriffsanforderungen abgesetzt.

Wenn diese Registrierdatenbankvariable definiert ist und PREFETCHSIZE des Tabellenbereichs auf den Wert AUTOMATIC gesetzt ist, berechnet DB2 die Größe des Vorablesezugriffs des Tabellenbereichs automatisch anhand folgender Gleichung:

$$\text{PREFETCHSIZE} = (\text{Anzahl Container}) * (\text{Anzahl Platten pro Container}) * \text{EXTENTSIZE}$$

Die Zahl nach dem Doppelpunkt wird von DB2 als *Anzahl Platten pro Container* in der Gleichung verwendet. Wenn nur ein Stern und keine Zahl angegeben wird, werden standardmäßig 6 Platten pro Container verwendet.

In der folgenden Tabelle sind die verschiedenen verfügbaren Optionen sowie die Berechnung der Parallelität für die einzelnen Fälle zusammengefasst:

Tabelle 76. Berechnung der Parallelität

PREFETCHSIZE des Tabellenbereichs	DB2_PARALLEL_IO-Einstellung	Die Parallelität ist gleich:
AUTOMATIC (PREFETCHSIZE = Anzahl Container * 1 * EXTENTSIZE)	Nicht definiert	Anzahl Container
AUTOMATIC (PREFETCHSIZE = Anzahl Container * 6 * EXTENTSIZE)	Tabellenbereichs-ID	Anzahl Container * 6
AUTOMATIC (PREFETCHSIZE = Anzahl Container * n * EXTENTSIZE)	Tabellenbereichs-ID:n	Anzahl Container * n
Nicht AUTOMATIC	Nicht definiert	Anzahl Container
Nicht AUTOMATIC	Tabellenbereichs-ID	PREFETCHSIZE / EXTENTSIZE

Tabelle 76. Berechnung der Parallelität (Forts.)

PREFETCHSIZE des Tabellenbereichs	DB2_PARALLEL_IO-Einstellung	Die Parallelität ist gleich:
Nicht AUTOMATIC	Tabellenbereichs-ID:n	PREFETCHSIZE / EXTENTSIZE

Nehmen Sie zum Beispiel an, Sie haben drei Tabellenbereiche, deren IDs 3, 4 und 5 lauten. Der EXTENTSIZE-Wert für alle Tabellenbereiche beträgt 4096 Byte, und alle Tabellenbereiche haben je zwei Container. Der PREFETCHSIZE-Wert der Tabellenbereiche 3 und 4 ist für beide auf AUTOMATIC gesetzt, der von Tabellenbereich 5 auf 16384 Byte. Wenn nun 'DB2_PARALLEL_IO=*:5,4:10' angegeben wird, leitet sich der Grad der Parallelität für die Tabellenbereiche wie folgt ab:

- Tabellenbereich 3: 'n' (Anzahl Platten pro Container) hat den Wert 5, EXTENTSIZE den Wert 4096, die Anzahl Container ist 2, und PREFETCHSIZE hat den Wert AUTOMATIC. Dies führt zu den Gleichungen $PREFETCHSIZE = 2 \cdot 5 \cdot 4096$ und Grad der Parallelität = Anzahl Container * n = $2 \cdot 5 = 10$.
- Tabellenbereich 4: Beachten Sie, dass für diesen Tabellenbereich 'n' (Anzahl Platten pro Container) explizit auf den Wert 10 gesetzt ist. Es gilt: EXTENTSIZE = 4096, Anzahl Container = 2, n = 10 und PREFETCHSIZE = AUTOMATIC. Dies führt zu den Gleichungen $PREFETCHSIZE = 2 \cdot 10 \cdot 4096$ und Grad der Parallelität = Anzahl Container * n = $2 \cdot 10 = 20$.
- Tabellenbereich 5: 'n' hat weiterhin den Wert 5, jedoch ist dies nicht von Belang, da PREFETCHSIZE nicht den Wert AUTOMATIC hat. Es gilt: EXTENTSIZE = 4096, Anzahl Container = 2 und PREFETCHSIZE = 16384. Daraus ergibt sich die Gleichung: Grad der Parallelität = $PREFETCHSIZE / EXTENTSIZE = 16384 / 4096 = 4$.

DB2PATH

- Betriebssystem: Windows
- Standardwert: (je nach Betriebssystem unterschiedlich)
- Diese Umgebungsvariable wird zur Angabe des Verzeichnisses verwendet, in dem das Produkt auf 32-Bit-Windows-Betriebssystemen installiert ist.

DB2PROCESSORS

- Betriebssystem: Windows
- Standardwert: null, Werte: 0 - n-1 (hierbei ist n = Anzahl der Prozessoren)
- Diese Variable definiert die Prozessaffinitätsmaske für einen bestimmten db2syscs-Prozess. In Umgebungen mit mehreren logischen Knoten wird diese Variable verwendet, um einem Prozessor bzw. einer Gruppe von Prozessoren einen logischen Knoten zuzuordnen.

Wird diese Variable angegeben, gibt DB2 die API 'SetProcessAffinityMask()' aus. Wird diese Variable nicht angegeben, wird der Prozess 'db2syscs' allen Prozessoren auf dem Server zugeordnet.

DB2RCMD_LEGACY_MODE

- Betriebssystem: Windows
- Standardwert=null, Werte: YES, ON, TRUE oder 1 bzw. NO, OFF, FALSE oder 0

- Diese Variable gibt Benutzern die Möglichkeit, die erweiterten Sicherheitsmerkmale für den DB2 Remote Command Service zu aktivieren oder zu inaktivieren. Zur Ausführung des DB2 Remote Command Service im sicheren Modus setzen Sie die Variable DB2RCMD_LEGACY_MODE auf den Wert NO, OFF, FALSE, 0 oder null. Zur Ausführung im traditionellen Modus (d. h., ohne erweiterte Sicherheit) setzen Sie die Variable DB2RCMD_LEGACY_MODE auf den Wert YES, ON, TRUE oder 1. Der sichere Modus steht nur zur Verfügung, wenn Ihr Domänencontroller mit Windows 2000 oder einer späteren Version des Betriebssystems ausgeführt wird.

Anmerkung: Wenn die Variable auf den Wert YES, ON, TRUE oder 1 gesetzt ist, werden alle Anforderungen, die an den DB2 Remote Command Service gesendet werden, unter dem Kontext des Anforderers verarbeitet. Zu diesem Zweck müssen Sie zulassen, dass entweder die Maschine oder das Serviceanmeldekonto die Identität des Clients annehmen, indem Sie das Maschinenkonto und das Serviceanmeldekonto auf dem Domänencontroller aktivieren.

Anmerkung: Wenn die Variable DB2RCMD_LEGACY_MODE auf den Wert NO, OFF, FALSE oder 0 gesetzt ist, müssen Sie über die Berechtigung SYSADM verfügen, damit der DB2 Remote Command Service Befehle für Sie ausführen kann.

DB2SYSTEM

- Betriebssystem: Windows und UNIX
- Standardwert=null
- Gibt den Namen an, der von Ihren Benutzern und Datenbankadministratoren zur Identifizierung des DB2-Serversystems verwendet wird. Dieser Name sollte nach Möglichkeit innerhalb Ihres Netzwerks eindeutig sein.

Dieser Name wird auf der Systemebene der Objektbaumstruktur in der Steuerzentrale angezeigt, um Administratoren bei der Identifizierung von Server-Systemen zu helfen, die von der Steuerzentrale aus verwaltet werden können.

Bei der Verwendung der Funktion zum Durchsuchen des Netzwerks des Clientkonfigurationsassistenten gibt DB2-Discovery diesen Namen zurück, und er wird auf der Systemebene in der resultierenden Objektbaumstruktur angezeigt. Dieser Name unterstützt Benutzer bei der Identifizierung des Systems, das die Datenbank enthält, auf die Sie zugreifen wollen. Bei der Installation wird für DB2SYSTEM wie folgt ein Wert festgelegt:

- Unter Windows setzt das Installationsprogramm diesen Parameter auf den Wert des Computernamens, der für das Windows-System angegeben ist.
- Auf UNIX-Systemen wird für diesen Parameter der TCP/IP-Hostname des UNIX-Systems definiert.

DB2_USE_PAGE_CONTAINER_TAG

- Betriebssystem: Alle
- Standardwert: null, Werte: ON, null
- Standardmäßig speichert DB2 eine Containerkennung im ersten Speicherbereich jedes DMS-Containers, unabhängig davon, ob es sich

um eine Datei oder eine Einheit handelt. Die Containerkennung bildet die Metadaten für den Container. Vor DB2 Version 8.1 wurde die Containerkennung auf einer einzigen Seite gespeichert und erforderte so weniger Speicherplatz im Container. Wenn die Containerkennung weiterhin in einer einzigen Seite gespeichert werden soll, setzen Sie die Variable DB2_USE_PAGE_CONTAINER_TAG auf den Wert ON.

Wenn Sie jedoch diese Registrierdatenbankvariable auf ON setzen, wenn Sie RAID-Einheiten als Container verwenden, kann sich die E/A-Leistung verschlechtern. Da Sie für RAID-Einheiten Tabellenbereiche mit einem EXTENTSIZE-Wert erstellen, der der Stripegröße oder einem Vielfachen der Stripegröße der RAID-Einheiten entspricht, führt die Einstellung der Variablen DB2_USE_PAGE_CONTAINER_TAG auf ON dazu, dass sich die EXTENTSIZE großen Speicherbereiche nicht an den RAID-Stripes (einheitenübergreifend gespeicherten Datenblöcken) ausrichten. Infolgedessen muss eine E/A-Anforderung eventuell auf mehr physische Platten zugreifen, als es optimal der Fall wäre. Benutzern wird ausdrücklich davon abgeraten, diese Registrierdatenbankvariable zu aktivieren.

Zur Aktivierung von Änderungen an dieser Registrierdatenbankvariablen führen Sie den Befehl DB2STOP und anschließend den Befehl DB2START aus.

DB2_WORKLOAD

- Betriebssystem: Alle
- Standardwert=nicht gesetzt, Werte: Auf SAP setzen, um DB2 für die SAP-Umgebung zu konfigurieren.
- Wenn DB2_WORKLOAD auf SAP gesetzt wird, wird automatisch eine Anzahl von DB2-Registrierdatenbankvariablen für die SAP-Umgebung konfiguriert.

Zugehörige Konzepte:

- „Registrierdatenbank- und Umgebungsvariablen von DB2“ auf Seite 601

Zugehörige Referenzen:

- „Command line processor options“ in *Command Reference*

Kommunikationsvariablen

DB2CHECKCLIENTINTERVAL

- Betriebssystem: Alle, nur Server
- Standardwert=50, Werte: ein numerischer Wert größer-gleich null.
- Diese Variable gibt die Häufigkeit von TCP/IP-Clientverbindungsprüfungen an. Sie ermöglicht das frühe Erkennen einer Clientbeendigung, anstatt zu warten, bis die Abfrage abgeschlossen ist. Wird diese Variable auf 0 gesetzt, wird keine Prüfung durchgeführt.

Niedrigere Werte bedeuten häufigere Überprüfungen. Verwenden Sie 100 als Richtwert für geringe Häufigkeit, 50 für mittlere Häufigkeit und 10 für hohe Häufigkeit. Der Wert wird in einer internen DB2-Metrik gemessen. Die Werte stellen eine lineare Skala dar. Dies bedeutet, dass eine Erhöhung des Werts von 50 auf 100 die Länge des Intervalls verdoppelt. Je häufiger Sie den Clientstatus während der Ausführung einer Datenbankanforderung prüfen, desto länger dauert die Ausführung von Abfragen. Wenn die DB2-Auslastung hoch ist (d. h., wenn viele interne Anforderungen vorliegen), hat das Einstellen von DB2CHECKCLIENT-

INTERVAL auf einen niedrigen Wert eine größere Auswirkung auf die Leistung als in einer Situation, in der die Auslastung gering ist.

In DB2 Universal Database Version 8.1.4 hat DB2CHECKCLIENTINTERVAL den Standardwert 50. Vor Version 8.1.4 war der Standardwert 0.

DB2COMM

- Betriebssystem: Alle, nur Server
- Standardwert=null, Werte: NPIPE, TCPIP
- Diese Variable gibt die Kommunikationsmanager an, die gestartet werden, wenn der Datenbankmanager gestartet wird. Wenn diese Variable nicht definiert wird, werden auf dem Server keine DB2-Kommunikationsmanager gestartet.

DB2_FORCE_NLS_CACHE

- Betriebssystem: AIX, HP_UX, Solaris
- Standardwert=FALSE, Werte: TRUE oder FALSE
- Diese Variable dient zur Verhinderung der Möglichkeit von Sperrkonflikten in Multithread-Anwendungen. Wenn diese Registrierdatenbankvariable den Wert „TRUE“ hat, werden die Informationen zur Codepage und zum Gebietscode beim Erstzugriff eines Threads auf die Informationen im Cache gespeichert. Von da an werden die im Cache gespeicherten Informationen für jeden anderen Thread verwendet, der diese Informationen anfordert. Dadurch wird der Sperrkonflikt beseitigt, was in bestimmten Situationen zu einem Leistungsvorteil führt. Diese Einstellung sollte nicht verwendet werden, wenn die Anwendung die länderspezifischen Angaben (Locale) zwischen Verbindungen ändert. In einer solchen Situation ist sie wahrscheinlich kaum erforderlich, da Multithread-Anwendungen ihre länderspezifischen Angaben in der Regel nicht ändern, weil dies nicht „threadsicher“ ist.

DB2JD_PORT_NUMBER

- Betriebssystem: Alle
- Standardwert=6789, Werte: 1 - 65535
- Diese Variable kann zum Überschreiben der Standardportnummer von db2jd verwendet werden. Wenn diese Registrierdatenbankvariable definiert ist, versucht 'db2jd', diesen Wert als Empfangsport von 'db2jd' zu verwenden, wenn 'db2jstrt' ohne Parameter ausgeführt wird. Wenn diese Registrierdatenbankvariable nicht definiert ist und kein Portparameter angegeben wird, wird 'db2jd' am Standardport 6789 gestartet.

DB2NBADAPTERS

- Betriebssystem: Windows
- Standardwert=0, Bereich: 0 - 15, mehrere Werte müssen durch Kommata getrennt werden
- Dient zur Definition, welche lokalen Adapter für die DB2-NetBIOS-LAN-Kommunikation zu verwenden sind. Jeder lokale Adapter wird mit Hilfe seiner logischen Adapternummer angegeben.

DB2NBCHECKUPTIME

- Betriebssystem: nur Windows-Server
- Standardwert=1 Minute, Werte: 1 - 720

- Definiert das Zeitintervall zwischen den Aufrufen der Prüfprozedur des NetBIOS-Protokolls. Die Prüfzeit wird in Minuten angegeben.
Niedrigere Werte stellen sicher, dass die Prüfprozedur des NetBIOS-Protokolls häufiger aktiv wird und Speicher und andere Systemressourcen freigibt, die zurückbleiben, wenn Agenten oder Sitzungen unerwartet beendet werden.

DB2NBINTRLISTENS

- Betriebssystem: nur Windows-Server
- Standardwert=1, Werte: 1 - 10, mehrere Werte müssen durch Kommata getrennt werden
- Diese Variable gibt die Anzahl der NetBIOS-Befehle 'listen send' (NCBs - NetBIOS-Steuerblöcke) an, die asynchron in Empfangsbereitschaft für Unterbrechungen ferner Clients abgesetzt werden. Diese Flexibilität ist für "unterbrechungsaktive" Umgebungen gedacht, um sicherzustellen, dass Unterbrechungsaufrufe von fernen Clients eine Verbindung herstellen können, wenn die Server mit der Bedienung anderer ferner Unterbrechungen beschäftigt sind.

Durch Definieren eines niedrigeren Werts für DB2NBINTRLISTENS werden NetBIOS-Sitzungen und NetBIOS-Steuerblöcke (NCBs) auf dem Server geschont. In einer Umgebung jedoch, in der Clientunterbrechungen regelmäßig vorkommen, ist ein höherer Wert für DB2NBINTRLISTENS eventuell sinnvoll, um auf unterbrechende Clients flexibler reagieren zu können.

Anmerkung: Die angegebenen Werte sind positionsabhängig. Sie beziehen sich jeweils auf die entsprechenden Wertpositionen für DB2NBADAPTERS.

DB2NBRECVBUFSIZE

- Betriebssystem: nur Windows-Server
- Standardwert=4096 Byte, Bereich: 4096 - 65536
- Diese Variable gibt die Größe der Empfangspuffer des DB2 NetBIOS-Protokolls an. Diese Puffer werden den NetBIOS-Empfangssteuerblöcken zugeordnet (NCBs). Niedrigere Werte sparen Serverspeicher, während höhere Werte erforderlich sein können, wenn die Clientdatenübertragungen umfangreicher sind.

DB2NBRECVNCBS

- Betriebssystem: nur Windows-Server
- Standardwert=10, Bereich: 1 - 99
- Diese Variable gibt die Anzahl der NetBIOS-Befehle "receive_any" (NCBs) an, die der Server beim Betrieb absetzt und verwaltet. Dieser Wert wird je nach der Anzahl ferner Clients, mit denen der Server verbunden ist, angepasst. Niedrigere Werte sparen Serverressourcen.

Anmerkung: Für jeden verwendeten Adapter kann ein eigener eindeutiger Empfangs-NCB-Wert im Parameter DB2NBRECVNCBS angegeben werden. Die angegebenen Werte sind positionsabhängig. Sie beziehen sich jeweils auf die entsprechenden Wertpositionen für DB2NBADAPTERS.

DB2NBRESOURCES

- Betriebssystem: nur Windows-Server
- Standardwert=null

- Definiert die Anzahl der NetBIOS-Ressourcen, die zur Verwendung durch DB2 in einer Mehrkontextumgebung zuzuordnen sind. Diese Variable ist auf den Clientbetrieb in einer Mehrkontextumgebung beschränkt.

DB2NBSENDNCBS

- Betriebssystem: nur Windows-Server
- Standardwert=6, Bereich: 1 - 720
- Diese Variable gibt die Anzahl der NetBIOS-Befehle send (NCBs) an, die der Server zur Verwendung empfängt. Dieser Wert kann angepasst werden je nach der Anzahl ferner Clients, mit denen der Server verbunden ist. Durch Definieren eines niedrigeren Werts für DB2NBSENDNCBS werden Serverressourcen gespart. Jedoch kann ein höherer Wert erforderlich werden, um zu vermeiden, dass der Server mit einem send-Befehl an einen fernen Client warten muss, wenn alle anderen send-Befehle in Gebrauch sind.

DB2NBSESSIONS

- Betriebssystem: nur Windows-Server
- Standardwert=null, Bereich: 5 - 254
- Definiert die Anzahl von Sitzungen, die DB2 zur Reservierung für DB2 anfordern soll. Der Wert von DB2NBSESSIONS kann definiert werden, um eine bestimmte Sitzung für jeden in DB2NBADAPTERS angegebenen Adapter anzufordern.

Anmerkung: Die angegebenen Werte sind positionsabhängig. Sie beziehen sich jeweils auf die entsprechenden Wertpositionen für DB2NBADAPTERS.

DB2NBXTRANCBS

- Betriebssystem: nur Windows-Server
- Standardwert=5 pro Adapter, Bereich: 5 - 254
- Gibt die Anzahl von NetBIOS-Befehlen "extra" (NCBs) an, die der Server reservieren muss, wenn der Befehl **db2start** ausgeführt wird. Der Wert von DB2NBXTRANCBS kann definiert werden, um eine bestimmte Sitzung für jeden in DB2NBADAPTERS angegebenen Adapter anzufordern.

DB2RETRY

- Betriebssystem: Windows
- Standardwert=0, Bereich: 0 - 20 000
- Diese Variable definiert die Anzahl der DB2-Versuche, die APPC-Empfangsfunktion erneut zu starten. Wenn das SNA-Subsystem auf dem Server/Gateway abgestürzt ist, kann die APPC-Empfangsfunktion mit Hilfe dieser Profilvariablen zusammen mit DB2RETRYTIME automatisch neu gestartet werden, ohne die Clientkommunikation unter Verwendung anderer Protokolle zu unterbrechen. In einem derartigen Szenario braucht DB2 nicht gestoppt und erneut gestartet zu werden, um die APPC-Clientkommunikation wiederherzustellen.

DB2RETRYTIME

- Betriebssystem: Windows
- Standardwert=1 Minute, Bereich: 0 - 7 200 Minuten
- In Inkrementen von einer Minute die Anzahl Minuten, die DB2 zwischen dem Ausführen aufeinander folgender Wiederholungen zum Starten der APPC-Empfangsfunktion zulässt. Wenn das SNA-Subsystem auf dem

Server/Gateway abgestürzt ist, kann die APPC-Empfangsfunktion mit Hilfe dieser Profilvariablen zusammen mit DB2RETRY automatisch neu gestartet werden, ohne die Clientkommunikation unter Verwendung anderer Protokolle zu unterbrechen. In einem derartigen Szenario braucht DB2 nicht gestoppt und erneut gestartet zu werden, um die APPC-Clientkommunikation wiederherzustellen.

DB2RSHCMD

- Betriebssystem: UNIX
- Standardwert=rsh (remsh unter HP-UX), Werte sind vollständige Pfadnamen für rsh, remsh oder ssh
- Standardmäßig verwendet das DB2-Datenbanksystem beim Starten ferner Datenbankpartitionen 'rsh' als Kommunikationsprotokoll. Das Script 'db2_all' wird verwendet, um Dienstprogramme und Befehle für alle Datenbankpartitionen auszuführen. Das Setzen dieser Registrierdatenbankvariablen auf den vollständigen Pfadnamen für 'ssh' veranlasst DB2-Datenbankprodukte zum Beispiel zur Verwendung von 'ssh' als Kommunikationsprotokoll für die angeforderte Ausführung der Dienstprogramme und Befehle. Sie kann auch auf den vollständigen Pfadnamen eines Scripts gesetzt werden, das das ferne Befehlsprogramm mit entsprechenden Standardparametern aufruft. Diese Variable ist nur für partitionierte Datenbanken oder für Umgebungen mit einer einzigen Partition erforderlich, wenn der Befehl 'db2start' auf einem Server ausgeführt wird, auf dem das DB2-Produkt nicht installiert ist. Der Instanzeigner muss das angegebene Programm der fernen Shell verwenden können, um sich von jedem DB2-Datenbankknoten aus bei jedem anderen DB2-Datenbankknoten anzumelden, ohne dass eine weitere Prüfung oder Authentifizierung (d. h. Kennwörter oder Kennwortausdrücke) erforderlich wird.

DB2RSHTIMEOUT

- Betriebssystem: UNIX
- Standardwert=30, Werte werden in Sekunden angezeigt; akzeptierte Werte sind 1 bis 120
- Diese Variable ist nur gültig, wenn DB2RSHCMD auf einen Wert ungleich null gesetzt ist. Diese Registrierdatenbankvariable wird für die Steuerung des Zeitlimitintervalls verwendet, während dessen das DB2-Datenbanksystem auf einen fernen Befehl wartet. Wenn bis zum Ablauf dieses Zeitlimits keine Antwort empfangen wurde, wird angenommen, dass die ferne Datenbankpartition nicht erreichbar und die Operation fehlgeschlagen ist.

Anmerkung: Bei dem angegebenen Zeitwert handelt es sich nicht um die für die Ausführung des fernen Befehls erforderliche Zeit, sondern um die Zeit, die für die Authentifizierung der Anforderung benötigt wird.

DB2SERVICETPINSTANCE

- Betriebssystem: Windows, AIX, Solaris
- Standardwert=null
- Dient zur Lösung des durch folgende Ursachen ausgelösten Problems:
 - Auf einer Maschine sind mehrere Instanzen aktiv.
 - Eine Instanz der Version 6 oder der Version 7 auf derselben Maschine versucht, dieselben TP-Namen zu registrieren.

Wenn der Befehl **db2start** aufgerufen wird, startet die angegebene Instanz die APPC-Empfangsfunktionen (Listeners) für die folgenden TP-Namen:

- DB2DRDA
- x'07'6DB

DB2SORCVBUF

- Betriebssystem: Alle
- Standardwert=65536
- Definiert den Wert von TCP/IP-Empfangspuffern unter Windows-Betriebssystemen.

DB2SOSNDBUF

- Betriebssystem: Alle
- Standardwert=65536
- Definiert den Wert von TCP/IP-Sendepuffern unter Windows-Betriebssystemen.

DB2SYSPLEX_SERVER

- Betriebssystem: Windows und UNIX
- Standardwert=null
- Gibt an, ob beim Herstellen einer Verbindung zu DB2 für OS/390 oder z/OS die SYSPLEX-Ausnutzung aktiviert wird. Ist diese Registrierdatenbankvariable nicht vorhanden (Standardeinstellung) oder wird sie auf einen anderen Wert als null eingestellt, wird die Ausnutzung aktiviert. Wird diese Registrierdatenbankvariable auf null (0) gesetzt, wird die Ausnutzung inaktiviert. Durch den Wert null wird die SYSPLEX-Ausnutzung für das Gateway inaktiviert, unabhängig davon, wie der DCS-Datenbankkatalogeintrag angegeben wurde. Weitere Informationen finden Sie in der Beschreibung des Befehls **CATALOG DCS DATABASE** für den Befehlszeilenprozessor.

DB2TCP_CLIENT_CONTIMEOUT

- Betriebssystem: Alle, nur Client
- Standardwert=0 (kein Zeitlimit), Werte: 0 bis 32 767 Sekunden
- Die Registrierdatenbankvariable DB2TCP_CLIENT_CONTIMEOUT gibt die Anzahl an Sekunden an, die ein Client auf den Abschluss einer TCP/IP-Verbindungsoperation wartet. Wenn innerhalb der angegebenen Sekunden keine Verbindung hergestellt wird, gibt der DB2-Datenbankmanager den Fehler -30081 selectForConnectTimeout zurück.

Es gibt keine Zeitlimitüberschreitung, wenn die Registrierdatenbankvariable nicht definiert oder auf den Wert 0 gesetzt ist.

Anmerkung: Auch Betriebssysteme haben einen Wert für ein Verbindungszeitlimit, der früher greifen könnte als das Zeitlimit, das Sie durch DB2TCP_CLIENT_CONTIMEOUT definieren. Zum Beispiel hat AIX den Standardwert tcp_keepinit=150 (in Halbsekunden), der den Verbindungsaufbau nach 75 Sekunden beenden würde.

Die Variable DB2TCP_CLIENT_CONTIMEOUT ist ab Version 9.1 des DB2-Datenbankmanagers verfügbar.

DB2TCP_CLIENT_RCVTIMEOUT

- Betriebssystem: Alle, nur Client

- Standardwert=0 (kein Zeitlimit), Werte: 0 bis 32 767 Sekunden
- Die Registrierdatenbankvariable DB2TCP_CLIENT_RCVTIMEOUT gibt die Anzahl Sekunden an, die ein Client auf den Empfang von Daten durch eine TCP/IP-Operation wartet. Wenn innerhalb der angegebenen Sekunden keine Daten vom Server empfangen werden, gibt der DB2-Datenbankmanager den Fehler -30081 selectForRecvTimeout zurück.
Es gibt keine Zeitlimitüberschreitung, wenn die Registrierdatenbankvariable nicht definiert oder auf den Wert 0 gesetzt ist.
Die Variable DB2TCP_CLIENT_RCVTIMEOUT ist ab Version 9.1 des DB2-Datenbankmanagers verfügbar.

DB2TCPCONNMGERS

- Betriebssystem: Alle
- Standardwert=1 auf seriellen Maschinen; auf symmetrischen Multiprozessormaschinen die aufgerundete Quadratwurzel aus der Anzahl Prozessoren bis zu maximal 16 Verbindungsmanagern. Werte: 1 bis 16.
- Wenn die Registrierdatenbankvariable nicht definiert ist, wird die Standardzahl von Verbindungsmanagern erstellt. Wenn die Registrierdatenbankvariable definiert ist, setzt der zugeordnete Wert den Standardwert außer Kraft. Die Zahl der angegebenen TCP/IP-Verbindungsmanager wird bis zu einem Maximum von 16 erstellt. Wenn weniger als einer angegeben wird, wird DB2TCPCONNMGERS auf den Wert 1 gesetzt und eine Warnung protokolliert, dass der Wert nicht innerhalb des gültigen Bereichs liegt. Wenn ein Wert größer als 16 angegeben wird, wird DB2TCPCONNMGERS auf den Wert 16 gesetzt und eine Warnung protokolliert, dass der Wert nicht innerhalb des gültigen Bereichs liegt. Werte zwischen 1 und 16 werden wie angegeben verwendet. Wenn mehr als ein Verbindungsmanager erstellt wird, sollte sich der Verbindungsdurchsatz verbessern, wenn mehrere Clientverbindungen gleichzeitig empfangen werden. Wenn der Benutzer eine SMP-Maschine verwendet oder die Registriervariable DB2TCPCONNMGERS geändert hat, können für den TCP/IP-Verbindungsmanager zusätzliche Prozesse (unter UNIX) bzw. Threads (unter Windows-Betriebssystemen) auftreten. Zusätzliche Prozesse oder Threads erfordern zusätzlichen Speicher.

Anmerkung: Wenn die Anzahl der Verbindungsmanager auf den Wert 1 gesetzt wird, kommt es bei Fernverbindungen in Systemen mit zahlreichen Benutzern und/oder häufigem Verbindungsauf- und -abbau zu einem Leistungsabfall.

Zugehörige Konzepte:

- „Registrierdatenbank- und Umgebungsvariablen von DB2“ auf Seite 601

Befehlszeilenvariablen

DB2BQTIME

- Betriebssystem: Alle
- Standardwert=1 Sekunde, Maximalwert: 1 Sekunde
- Diese Variable gibt die Zeitdauer an, die das Front-End des Befehlszeilenprozessors inaktiv bleibt, bevor es prüft, ob der Back-End-Prozess aktiv ist, und eine Verbindung zu diesem Prozess herstellt.

DB2BQTRY

- Betriebssystem: Alle

- Standardwert=60 Wiederholungen, Minimalwert: 0 Wiederholungen
- Diese Variable gibt die Anzahl der Wiederholungen an, mit denen das Front-End des Befehlszeilenprozessors festzustellen versucht, ob der Back-End-Prozess bereits aktiv ist. Dieser Parameter arbeitet in Verbindung mit dem Parameter DB2BQTIME.

DB2_CLPPROMPT

- Betriebssystem: Alle
- Standardwert=Keiner (falls kein Standardwert definiert ist, wird „db2 =>“ als interaktive Standard-CLP-Eingabeaufforderung verwendet). Werte: Alle Textzeichenfolgen mit einer Länge von weniger als 100 Zeichen, die keinen oder mehrere der folgenden Tokens enthalten: %i, %d, %ia, %da oder %n. Diese Variable braucht nur dann definiert zu werden, wenn die standardmäßige interaktive CLP-Eingabeaufforderung (db2 =>) explizit geändert werden soll.
- Mit dieser Registrierdatenbankvariablen können Benutzer die Eingabeaufforderung definieren, die im interaktiven Modus des Befehlszeilenprozessors (CLP) verwendet werden soll. Gültige Werte für die Variable: Alle Textzeichenfolgen mit einer Länge von weniger als 100 Zeichen, die keinen oder mehrere der folgenden optionalen Tokens enthalten: %i, %d, %ia, %da oder %n. Wird der interaktive CLP-Modus ausgeführt, wird die zu verwendende Eingabeaufforderung erstellt, indem die in der Registrierdatenbankvariablen DB2_CLPPROMPT angegebene Textzeichenfolge verwendet wird und alle Vorkommnisse der Tokens %i, %d, %ia, %da oder %n durch den lokalen Aliasnamen der jeweils zugeordneten Instanz, den lokalen Aliasnamen der aktuellen Datenbankverbindung, die Berechtigungs-ID der jeweils zugeordneten Instanz, die Berechtigungs-ID der aktuellen Datenbankverbindung bzw. eine neue Zeile (d. h. eine Zeilenschaltung) ersetzt wird.

Anmerkungen:

1. Wird die Registrierdatenbankvariable DB2_CLPPROMPT innerhalb des interaktiven CLP-Modus geändert, tritt der neue Wert für DB2_CLPPROMPT erst in Kraft, nachdem der interaktive CLP-Modus geschlossen und erneut geöffnet wurde.
2. Ist keine Instanz zugeordnet, wird %ia durch eine leere Zeichenfolge und %i durch den Wert der Registrierdatenbankvariablen DB2INSTANCE ersetzt. Nur auf Windows-Plattformen: Wird die Variable DB2INSTANCE nicht definiert, wird %i durch den Wert der Registrierdatenbankvariablen DB2INSTDEF ersetzt. Wird keine dieser Variablen definiert, wird %i durch eine leere Zeichenfolge ersetzt.
3. Ist keine Datenbankverbindung vorhanden, wird %da durch eine leere Zeichenfolge und %d durch den Wert der Registrierdatenbankvariablen DB2DBDFT ersetzt. Wird die Variable DB2DBDFT nicht definiert, wird %d durch eine leere Zeichenfolge ersetzt.
4. Die Eingabeaufforderung für interaktive Eingabe zeigt die Werte für die Berechtigungs-IDs, Datenbanknamen und Instanznamen stets in Großbuchstaben an.

DB2IQTIME

- Betriebssystem: Alle
- Standardwert=5 Sekunden, Minimalwert: 1 Sekunde
- Definiert die Zeitdauer, die der Back-End-Prozess des Befehlszeilenprozessors an der Eingabewarteschlange darauf wartet, dass der Front-End-Prozess Befehle übergibt.

DB2RQTIME

- Betriebssystem: Alle
- Standardwert=5 Sekunden, Minimalwert: 1 Sekunde
- Definiert die Zeitdauer, die der Back-End-Prozess des Befehlszeilenprozessors auf eine Anforderung vom Front-End-Prozess wartet.

Zugehörige Konzepte:

- „Registrierdatenbank- und Umgebungsvariablen von DB2“ auf Seite 601

MPP-Konfigurationsvariablen

DB2CHGPWD_EEE

- Betriebssystem: DB2 ESE unter AIX und Windows
- Standardwert=null, Werte: YES oder NO
- Diese Variable gibt an, ob Sie zulassen, dass andere Benutzer Kennwörter auf ESE-Systemen unter AIX oder Windows ändern. Es muss sichergestellt werden, dass die Kennwörter für alle Datenbankpartitionen oder Knoten mit Hilfe eines Windows-Domänencontrollers unter Windows oder NIS unter AIX zentral verwaltet werden. Wenn Kennwörter nicht zentral verwaltet werden, bleiben sie möglicherweise nicht für alle Datenbankpartitionen bzw. Knoten konsistent. Dies könnte dazu führen, dass ein Kennwort nur in der Datenbankpartition geändert wird, mit der der Benutzer zur Durchführung der Änderung verbunden ist.

DB2_NUM_FAILOVER_NODES

- Betriebssystem: Alle
- Standardwert: 2, Werte: 0 bis zur Anzahl der logischen Knoten
- Gibt die Anzahl der Knoten an, die als Funktionsübernahmeknoten in einer Umgebung mit hoher Verfügbarkeit verwendet werden können. Bei hoher Verfügbarkeit kann ein Knoten beim Auftreten eines Fehlers als zweiter logischer Knoten auf einem anderen Host neu gestartet werden. Die für diese Variable verwendete Zahl legt fest, wie viel Speicher für FCM-Ressourcen für Funktionsübernahmeknoten reserviert ist.
Beispielsweise verfügt Host A über zwei logische Knoten: 1 und 2, und Host B verfügt über zwei logische Knoten: 3 und 4. Nehmen Sie an, dass DB2_NUM_FAILOVER_NODES auf den Wert 2 gesetzt ist. Während der Ausführung von DB2START reservieren Host A und auch Host B genügend Speicher für den FCM, um bis zu vier logische Knoten verwalten zu können. Wenn dann bei einem Host ein Fehler auftritt, können die logischen Knoten für den fehlerhaften Host auf dem anderen Host erneut gestartet werden.

DB2_PARTITIONEDLOAD_DEFAULT

- Betriebssystem: Alle unterstützten ESE-Plattformen
- Standardwert=YES, Werte: YES oder NO
- Die Registrierdatenbankvariable DB2_PARTITIONEDLOAD_DEFAULT ermöglicht Benutzern, die Standardfunktionsweise des Dienstprogramms LOAD in einer ESE-Umgebung zu ändern, wenn keine ESE-spezifischen LOAD-Optionen angegeben werden. Der Standardwert YES gibt an, dass in einer ESE-Umgebung, wenn Sie keine ESE-spezifischen LOAD-Optionen angeben, das Laden in allen Datenbankpartitionen versucht wird, in denen die Zieltabelle definiert ist.

Wenn der Wert NO ist, wird das Laden nur in der Datenbankpartition versucht, mit der das Dienstprogramm LOAD gegenwärtig verbunden ist.

DB2PORTRANGE

- Betriebssystem: Windows
- Werte: nnnn:nnnn
- Dieser Wert wird auf den TCP/IP-Portbereich gesetzt, der vom FCM verwendet wird, so dass alle zusätzlichen auf einer anderen Maschine erstellten Datenbankpartitionen den gleichen Portbereich haben.

Zugehörige Konzepte:

- „Registrierdatenbank- und Umgebungsvariablen von DB2“ auf Seite 601

Abfragecompilervariablen

DB2_ANTIJOIN

- Betriebssystem: Alle
- Standardwert=NO in einer ESE-Umgebung, Standardwert=YES in einer anderen Umgebung (nicht ESE), Werte: YES, NO oder EXTEND
- Für DB2 Enterprise Server Edition: Bei Angabe von "YES" sucht das Optimierungsprogramm nach Möglichkeiten, „NOT EXISTS“-Unterabfragen in Antijoins umzusetzen, die von DB2 effizienter verarbeitet werden können. In anderen Umgebungen (nicht ESE): Bei Angabe von "NO" schränkt das Optimierungsprogramm die Möglichkeiten zur Umsetzung von „NOT EXISTS“-Unterabfragen in Antijoins ein.
Wenn EXTEND angegeben ist, sucht das Optimierungsprogramm sowohl in ESE-Umgebungen als auch in anderen Umgebungen nach Möglichkeiten, NOT IN- und NOT EXISTS-Unterabfragen in Antijoins umzusetzen.

DB2_INLIST_TO_NLJN

- Betriebssystem: Alle
- Standardwert=NO, Werte: YES oder NO
- In einigen Fällen kann der SQL- und XQuery-Compiler ein Listenvergleichselement IN in einen Join umschreiben. Betrachten Sie zum Beispiel die folgende Abfrage:

```
SELECT *  
FROM EMPLOYEE  
WHERE DEPTNO IN ('D11', 'D21', 'E21')
```

Diese Abfrage könnte folgendermaßen geschrieben werden:

```
SELECT *  
FROM EMPLOYEE, (VALUES 'D11', 'D21', 'E21') AS V(DNO)  
WHERE DEPTNO = V.DNO
```

Diese Überarbeitung könnte eine bessere Leistung erzielen, wenn es einen Index für die Spalte DEPTNO gibt. Auf die Liste von Werten würde zuerst zugegriffen und die Liste mit der Tabelle EMPLOYEE durch einen Join mit Verschachtelungsschleife mit Hilfe des Index zur Anwendung des Joinvergleichselements verknüpft.

Manchmal verfügt das Optimierungsprogramm über keine genauen Informationen, um die beste Joinmethode für die umgeschriebene Version der Abfrage zu bestimmen. Dies kann der Fall sein, wenn die IN-Liste Parametermarken oder Hostvariablen enthält, die verhindern,

dass das Optimierungsprogramm die Selektivität mit Hilfe der Katalogstatistiken ermitteln kann. Diese Registrierdatenbankvariable veranlasst das Optimierungsprogramm, Joins mit Verschachtelungsschleifen zu bevorzugen, um die Liste von Werten zu verknüpfen und dabei die Tabelle, die die IN-Liste beisteuert, als innere Tabelle im Join zu verwenden.

Anmerkung: Wenn mindestens eine der beiden DB2-Abfragecompilervariablen DB2_MINIMIZE_LISTPREFETCH und DB2_INLIST_TO_NLJN auf YES gesetzt ist, bleibt sie selbst dann aktiv, wenn REOPT(ONCE) angegeben ist.

DB2_LIKE_VARCHAR

- Betriebssystem: Alle
- Standardwert=Y,Y,
- Steuert die Verwendung von Statistikdaten zu Unterelementen. Dabei handelt es sich um Statistikdaten zum Inhalt von Spaltendaten, wenn diese eine Struktur in Form einer Folge von Unterfeldern oder Unterelementen aufweisen, die durch Leerzeichen getrennt sind. Die Erfassung von Statistikdaten für Unterelemente ist optional und wird durch Optionen im Befehl RUNSTATS oder des API gesteuert.

Diese Registrierdatenbankvariable beeinflusst, wie das Optimierungsprogramm ein Vergleichselement der folgenden Form behandelt:

```
COLUMN LIKE '%xxxxxx%'
```

Dabei ist xxxxxx eine beliebige Zeichenfolge.

Die folgende Syntax zeigt, wie diese Registrierdatenbankvariable verwendet wird:

```
db2set DB2_LIKE_VARCHAR=[Y|N|S|num1] [,Y|N|S|num2]
```

Dabei gilt Folgendes:

- Der Term vor dem Komma bzw. der einzige Term rechts des Vergleichselements hat folgende Bedeutung, allerdings nur, wenn für den zweiten Term der Wert N angegeben wurde oder die Spalte keine positiven Statistikdaten zu Unterelementen hat:
 - S – Das Optimierungsprogramm schätzt die Länge der einzelnen Elemente einer Folge miteinander verknüpfter Elemente, die eine Spalte bilden, ausgehend von der Länge der Zeichenfolge, die von den %-Zeichen eingeschlossen wird.
 - Y – Der Standardwert. Verwenden des Standardwerts 1,9 als Algorithmusparameter. Verwenden des Algorithmus für Unterelemente mit variabler Länge mit dem Algorithmusparameter.
 - N – Verwenden eines Algorithmus für Unterelemente mit fester Länge.
 - num1 – Verwenden des Werts von num1 als Algorithmusparameter für den Algorithmus für Unterelemente mit variabler Länge.
- Der Term nach dem Komma hat folgende Bedeutung, jedoch nur für Spalten, die positive Statistikdaten zu Unterelementen haben:
 - N – Kein Verwenden von Statistikdaten zu Unterelementen. Der erste Term tritt in Kraft.
 - Y – Der Standardwert. Verwenden eines Algorithmus für Unterelemente mit variabler Länge, der Statistikdaten zu Unterelementen

zusammen mit dem Standardwert 1,9 als Algorithmusparameter verwendet, wenn Spalten mit positiven Statistikdaten zu Unterelementen vorliegen.

- num2 – Verwenden eines Algorithmus für Unterelemente mit variabler Länge, der Statistikdaten zu Unterelementen zusammen mit dem Wert num2 als Algorithmusparameter verwendet, wenn Spalten mit positiven Statistikdaten zu Unterelementen vorliegen.

DB2_MINIMIZE_LISTPREFETCH

- Betriebssystem: Alle
- Standardwert=NO, Werte: YES oder NO
- Der Vorabesezugriff über Listen ist eine spezielle Zugriffsmethode auf Tabellen, bei der die den Bedingungen entsprechenden Satz-IDs (RIDs) aus einem Index abgerufen, nach Seitennummer sortiert und anschließend die Datenseiten vorab gelesen werden. Manchmal verfügt das Optimierungsprogramm über keine genauen Informationen, um zu ermitteln, ob ein Vorabesezugriff über Listen eine gute Zugriffsmethode ist. Dies kann der Fall sein, wenn die Selektivitäten von Vergleichselementen Parametermarken oder Hostvariablen enthalten, die verhindern, dass das Optimierungsprogramm die Selektivität mit Hilfe der Katalogstatistiken ermitteln kann.

Diese Registrierdatenbankvariable verhindert, dass das Optimierungsprogramm in solchen Situationen einen Vorabesezugriff über Listen in Betracht zieht.

Anmerkung: Wenn mindestens eine der beiden DB2-Abfragecompilervariablen DB2_MINIMIZE_LISTPREFETCH und DB2_INLIST_TO_NLJN auf YES gesetzt ist, bleibt sie selbst dann aktiv, wenn REOPT(ONCE) angegeben ist.

DB2_NEW_CORR_SQ_FF

- Betriebssystem: Alle
- Standardwert=OFF, Werte: ON oder OFF
- Beeinflusst, wenn auf „ON “ gesetzt, den Selektivitätswert, der vom Abfrageoptimierungsprogramm für bestimmte Vergleichselemente von Unterabfragen berechnet wird. Diese Variable dient zur Verbesserung der Genauigkeit des Selektivitätswerts von Gleichheitsvergleichselementen in Unterabfragen, die die Spaltenfunktion MIN oder MAX in der SELECT-Liste der Unterabfrage enthalten. Zum Beispiel:

```
SELECT * FROM T WHERE  
T.COL = (SELECT MIN(T.COL)  
FROM T WHERE ...)
```

DB2_OPT_MAX_TEMP_SIZE

- Betriebssystem: Alle
- Standardwert=null, Wert: Größe des Speicherbereichs in Megabyte, der von einer Abfrage in allen temporären Tabellenbereichen verwendet werden kann
- Begrenzt die Größe des Speicherbereichs, den Abfragen in den temporären Tabellenbereichen nutzen können. Wenn die Variable DB2_OPT_MAX_TEMP_SIZE definiert wird, kann sie das Optimierungsprogramm veranlassen, einen aufwendigeren Plan als sonst auszuwählen, der jedoch weniger Speicherplatz in den temporären Tabellenbereichen erfordert. Wenn Sie die Variable DB2_OPT_MAX_TEMP_SIZE definieren, stellen Sie sicher, dass Sie die Notwendigkeit, den Bedarf an

temporärem Tabellenbereich einzuschränken, gegen die Effizienz des Plans, der durch Ihre Einstellung ausgewählt wird, geeignet abwägen. Wenn DB2_WORKLOAD=SAP definiert wird, wird die Variable DB2_OPT_MAX_TEMP_SIZE automatisch auf den Wert 10240 (10 GB) gesetzt.

Wenn Sie eine Abfrage ausführen, die über den für DB2_OPT_MAX_TEMP_SIZE festgelegten Wert hinaus temporären Tabellenbereich belegt, schlägt die Abfrage nicht fehl, jedoch empfangen Sie eine Warnung, dass die Leistung vielleicht nicht optimal ist, da möglicherweise nicht alle Ressourcen verfügbar sind.

Die folgenden vom Optimierungsprogramm in Betracht gezogenen Operationen sind von dem durch DB2_OPT_MAX_TEMP_SIZE definierten Grenzwert betroffen:

- Explizite Sortierungen für Operationen wie ORDER BY, DISTINCT, GROUP BY, Mischsuchjoins und Joins mit Verschachtelungsschleife
- Explizite temporäre Tabellen
- Implizite temporäre Tabellen für Hash-Joins und duplizierte Mischjoins

DB2_PRED_FACTORIZE

- Betriebssystem: Alle
- Standardwert=NO, Wert: YES oder NO
- Gibt an, ob das Optimierungsprogramm nach Gelegenheiten suchen soll, um zusätzliche Vergleichselemente aus Disjunktionen zu extrahieren. In manchen Fällen können die zusätzlichen Vergleichselemente die geschätzte Kardinalität der vorläufigen und endgültigen Ergebnismengen verändern. Mit der folgenden Abfrage:

```
SELECT n1.empno,  
       n1.lastname  
FROM employee n1,  
     employee n2  
WHERE  
  ((n1.lastname='SMITH'  
   AND n2.lastname='JONES')  
   OR (n1.lastname='JONES'  
   AND n2.lastname='SMITH'))
```

kann das Optimierungsprogramm die folgenden zusätzlichen Vergleichselemente generieren:

```
SELECT n1.empno,  
       n1.lastname  
FROM employee n1,  
     employee n2  
WHERE n1.lastname IN  
  ('SMITH', 'JONES')  
   AND n2.lastname IN  
  ('SMITH', 'JONES')  
   AND  
  ((n1.lastname='SMITH'  
   AND n2.lastname='JONES')  
   OR (n1.lastname='JONES'  
   AND n2.lastname='SMITH'))
```

DB2_REDUCED_OPTIMIZATION

- Betriebssystem: Alle
- Standardwert=NO, Werte: NO, YES, beliebige Ganzzahl, DISABLE, NO_SORT_NLJOIN, NO_SORT_MGJOIN

- Mit dieser Registrierdatenbankvariablen können Sie entweder reduzierte Optimierungsfunktionen oder die strenge Anwendung der Optimierungsfunktionen der angegebenen Optimierungsklasse anfordern. Wenn Sie die Anzahl der verwendeten Optimierungstechniken reduzieren, können Sie dadurch die Zeit und die Ressourcenbelastung bei der Optimierung verringern.

Anmerkung: Die Optimierungszeit und die Ressourcenbelastung wird zwar reduziert, jedoch erhöht sich das Risiko, dass ein nicht optimaler Datenzugriffsplan generiert wird. Verwenden Sie diese Registrierdatenbankvariable nur, wenn Sie von IBM oder einem IBM Partner dazu aufgefordert werden.

- Wenn der Wert NO definiert ist
Das Optimierungsprogramm ändert seine Optimierungstechniken nicht.
- Wenn der Wert YES definiert ist
Wenn die Optimierungsklasse 5 (Standardwert) oder eine niedrigere verwendet wird, inaktiviert das Optimierungsprogramm einige Optimierungstechniken, die möglicherweise viel Vorbereitungszeit und Ressourcen beanspruchen, jedoch in der Regel zur Generierung besserer Zugriffspläne führen.
Wenn die Optimierungsklasse exakt 5 ist, reduziert das Optimierungsprogramm einige zusätzliche Techniken oder inaktiviert sie, wodurch sich die Optimierungszeit und die Ressourcenbelastung weiter verringern können, jedoch gleichzeitig die Gefahr wächst, dass ein nicht optimaler Zugriffsplan generiert wird. Bei Optimierungsklassen unter 5 kommen einige dieser Techniken möglicherweise ohnehin nicht zur Anwendung. Wenn sie dennoch angewandt werden, bleiben sie jedoch wirksam.
- Wenn eine beliebige Ganzzahl definiert ist.
Der Effekt ist der gleiche wie bei YES, jedoch mit folgenden zusätzlichen Merkmalen für dynamisch vorbereitete Abfragen, die mit Klasse 5 optimiert werden. Wenn die Gesamtanzahl von Joins in einem beliebigen Abfrageblock die Einstellung überschreitet, wechselt das Optimierungsprogramm zur schnellen Joinaufzählung (Greedy Join Enumeration), anstatt zusätzliche Optimierungstechniken wie oben für die Optimierungsklasse 5 beschrieben zu inaktivieren. Dies impliziert, dass die Abfrage mit einer ähnlichen Klasse wie Optimierungsklasse 2 optimiert wird.
- Wenn der Wert DISABLE definiert ist
Wenn keine Einschränkung durch diese Variable DB2_REDUCED_OPTIMIZATION definiert ist, sieht die Funktionsweise des Optimierungsprogramms zuweilen vor, die Optimierung für dynamische Abfragen in Optimierungsklasse 5 dynamisch zu reduzieren. Diese Einstellung setzt diese Funktionsweise außer Kraft und zwingt das Optimierungsprogramm, die volle Optimierung der Klasse 5 durchzuführen.
- Wenn der Wert NO_SORT_NLJOIN definiert ist
Das Optimierungsprogramm generiert keine Abfragepläne, die Sortierungen für Joins mit Verschachtelungsschleife (NLJN, Nested Loop Join) erzwingen. Diese Typen von Sortierungen können zur Steige-

rung der Leistung nützlich sein. Daher ist bei der Verwendung der Option NO_SORT_NLJOIN Vorsicht geboten, da die Leistung erheblich beeinträchtigt werden kann.

- Wenn der Wert NO_SORT_MGJOIN definiert ist

Das Optimierungsprogramm generiert keine Abfragepläne, die Sortierungen für Mischsuchjoins (MSJN, Merge-Scan-Joins) erzwingen. Diese Typen von Sortierungen können zur Steigerung der Leistung nützlich sein. Daher ist bei der Verwendung der Option NO_SORT_MGJOIN Vorsicht geboten, da die Leistung erheblich beeinträchtigt werden kann.

Beachten Sie, dass die dynamische Reduzierung der Optimierung in Optimierungsklasse 5 Vorrang vor der Funktionsweise hat, die für die exakte Optimierungsklasse 5 beschrieben ist, wenn die Variable DB2_REDUCED_OPTIMIZATION auf den Wert YES gesetzt ist, sowie vor der Funktionsweise, die für die ganzzahlige Einstellung beschrieben ist.

DB2_SELECTIVITY

- Betriebssystem: Alle
- Standardwert=No, Werte: Yes oder No
- Diese Registrierdatenbankvariable steuert, wo die Klausel SELECTIVITY in Suchbedingungen in SQL-Anweisungen verwendet werden kann.

Wenn diese Registrierdatenbankvariable auf "Yes" gesetzt ist, kann die Klausel SELECTIVITY für die folgenden Vergleichselemente angegeben werden:

- Ein einfaches Vergleichselement, in dem mindestens ein Ausdruck Hostvariablen enthält
- Ein LIKE-Vergleichselement, in dem Abgleichausdruck (MATCH), der Vergleichselementausdruck oder der Escape-Ausdruck Hostvariablen enthält

DB2_SQLROUTINE_PREPOPTS

- Betriebssystem: Alle
- Standardwert=leere Zeichenfolge, Werte:
 - BLOCKING {UNAMBIG | ALL | NO}
 - DATETIME {DEF | USA | EUR | ISO | JIS | LOC}
 - DEGREE {1 | *grad-der-parallelität* | ANY}
 - DYNAMICRULES {BIND | RUN}
 - EXPLAIN {NO | YES | ALL}
 - EXPLSNAP {NO | YES | ALL}
 - FEDERATED {NO | YES}
 - INSERT {DEF | BUF}
 - ISOLATION {CS | RR | UR | RS | NC}
 - QUERYOPT *optimierungsgrad*
 - VALIDATE {RUN | BIND}
- Die Registrierdatenbankvariable DB2_SQLROUTINE_PREPOPTS kann zur Anpassung der Vorkompilierungs- und Bindeoptionen für SQL- und XQuery-Prozeduren verwendet werden.

Zugehörige Konzepte:

- „Registrierdatenbank- und Umgebungsvariablen von DB2“ auf Seite 601

- „Auswählen von Optimierungsklassen“ auf Seite 179
- „Strategien zur Auswahl optimaler Joins“ auf Seite 224

Zugehörige Referenzen:

- „Optimierungsklassen“ auf Seite 181

Leistungsvariablen

DB2_ALLOCATION_SIZE

- Betriebssystem: Alle
- Standardwert=128 KB, Bereich: 64 KB - 256 MB
- Gibt die Größe von Speicherzuordnungen für Pufferpools an.

Wenn diese Registrierdatenbankvariable auf einen höheren Wert gesetzt wird, besteht der potenzielle Vorteil darin, dass weniger Zuordnungen erforderlich sind, um eine gewünschte Größe des Speichers für einen Pufferpool zu erreichen.

Der potenzielle Nachteil eines höheren Werts für diese Registrierdatenbankvariable besteht darin, dass Speicher verschwendet werden kann, wenn der Pufferpool um einen Wert geändert wird, der nicht einem Vielfachen der Zuordnungsgröße entspricht. Wenn zum Beispiel die Variable DB2_ALLOCATION_SIZE auf 8 MB gesetzt ist und ein Pufferpool um 4 MB verkleinert wird, werden diese 4 MB verschwendet, weil kein ganzes 8-MB-Segment freigegeben werden kann.

DB2_APM_PERFORMANCE

- Betriebssystem: Alle
- Standardwert=OFF, Werte: ON, OFF
- Setzen Sie diese Variable auf den Wert ON, um die leistungsrelevanten Änderungen am Zugriffsplanmanager (APM) zu aktivieren, die sich auf die Funktionsweise des Abfragecache (Paketcache) auswirken. Diese Einstellungen werden für Produktionssysteme im Normalfall nicht empfohlen. Sie verursachen einige Einschränkungen, wie zum Beispiel die Möglichkeit von Fehlern auf Grund nicht ausreichenden Paketcacheplatzes oder erhöhter Speicherauslastung (oder beider Ursachen).

Die Einstellung der Variablen DB2_APM_PERFORMANCE auf den Wert ON aktiviert außerdem den Modus 'Keine Paketsperre' ('No Package Lock'). Dieser Modus ermöglicht den Betrieb des globalen Abfragecache ohne Verwendung von Paketsperren, bei denen es sich um interne Systemsperren handelt, die Paketeinträge im Cache davor schützen, entfernt zu werden. Der Modus 'Keine Paketsperre' kann zwar zu einer etwas verbesserten Leistung führen, jedoch sind bestimmte Datenbankoperationen nicht zulässig. Zu diesen unzulässigen Operationen können gehören: Operationen, die Pakete ungültig machen, Operationen, die Pakete funktionsunfähig machen, sowie die Operationen PRECOMPILE, BIND und REBIND.

DB2ASSUMEUPDATE

- Betriebssystem: Alle
- Standardwert=OFF, Werte: ON, OFF
- Wenn diese Variable aktiviert ist, kann das DB2-Datenbanksystem annehmen, dass alle Spalten fester Länge, die in einer UPDATE-Anweisung angegeben werden, geändert werden. Dadurch muss das DB2-Datenbanksystem keinen Vergleich zwischen den vorhandenen Spaltenwerten und den angegebenen neuen Spaltenwerten durchführen, um

festzustellen, ob sich die Spalte tatsächlich ändert. Wenn bei Verwendung dieser Registrierdatenbankvariablen Spalten zur Aktualisierung angegeben werden (z. B. in einer SET-Klausel), jedoch nicht wirklich geändert werden, kann dies zusätzliche Protokoll- und Indexpflegeaktivitäten zur Folge haben.

Diese Registrierdatenbankvariable wird zum Zeitpunkt einer Aktualisierung (UPDATE-Operation) geprüft.

DB2_AVOID_PREFETCH

- Betriebssystem: Alle
- Standardwert=OFF, Werte: ON oder OFF
- Definiert, ob ein Vorablesezugriff bei der Recovery nach einem Systemabsturz verwendet werden soll. Wenn DB2_AVOID_PREFETCH=ON definiert ist, wird der Vorablesezugriff nicht verwendet.

DB2_AWE

- Betriebssystem: Windows 2000
- Standardwert=null, Werte: <eintrag>[;<eintrag>...] mit <eintrag>=<pufferpool-id>,<anzahl physischer seiten>, <anzahl von adressfenstern>
- Mit dieser Variablen können DB2-Datenbanksysteme auf 32-Bit-Plattformen unter Windows 2000 Pufferpools zuordnen, die bis zu 64 GB Speicher belegen. Die Unterstützung von AWE-Pufferpools (AWE, Address Windowing Extensions) erfordert eine entsprechende Konfiguration von Windows 2000. Dies umfasst die Zuordnung der Berechtigung „Seiten im Speicher sperren“ für den Benutzer, die Zuordnung der physischen Seiten und der Adressfensterseiten sowie die Einstellung dieser Registrierdatenbankvariablen. Zur Einstellung dieser Variablen benötigen Sie die Pufferpool-ID des Pufferpools, der für die AWE-Unterstützung verwendet werden soll. Die ID des Pufferpools können Sie der Spalte BUFFERPOOLID in der Systemkatalogsicht SYSCAT.BUFFERPOOLS entnehmen.

Anmerkung:

- Pufferpools, auf die diese Registrierdatenbankvariable verweist, müssen bereits im Systemkatalog SYSCAT.SYSBUFFERPOOLS vorhanden sein.
- Pufferpools, die für AWE aktiviert sind, haben Vorrang vor Pufferpools, die für blockbasierte E/A aktiviert sind. Wenn ein Pufferpool für AWE und blockbasierte E/A konfiguriert ist, hat AWE Vorrang vor der blockbasierten E/A.

DB2BPVARS

- Betriebssystem: Wie für jeden Parameter angegeben
- Standardwert=Pfad
- Es sind zwei Gruppen von Parametern zur Optimierung von Pufferpools verfügbar. Die eine Gruppe von Parametern, die nur unter Windows zur Verfügung steht, gibt an, dass Pufferpools SCATTER-Lesezugriffe für bestimmte Typen von Containern verwenden sollen. Die andere Gruppe von Parametern, die auf allen Plattformen verfügbar ist, beeinflusst die Funktionsweise des Vorablesezugriffs.

Die Parameter werden in einer ASCII-Datei, jeweils ein Parameter in einer Zeile, in der Form `parameter=wert` angegeben. Zum Beispiel könnte eine Datei mit dem Namen `bpvars.vars` die folgenden Zeilen enthalten:

```
NO_NT_SCATTER = 1
NUMPREFETCHQUEUES = 2
```

Wenn die Datei `bpvars.vars` zum Beispiel im Pfad `F:\vars\` gespeichert ist, geben Sie zur Definition dieser Variablen den folgenden Befehl ein:

```
db2set DB2BPVARS=F:\vars\bpvars.vars
```

Parameter für SCATTER-Lesezugriff

Die Parameter für den SCATTER-Lesezugriff werden für Systeme mit einem hohen Aufkommen an sequenziellen Vorableseoperationen für die jeweilige Art von Container empfohlen, für die Sie bereits den Parameter `DB2NTNOCACHE` auf den Wert `ON` gesetzt haben. Diese Parameter, die nur auf Windows-Plattformen verfügbar sind, heißen `NT_SCATTER_DMSFILE`, `NT_SCATTER_DMSDEVICE` und `NT_SCATTER_SMS`. Geben Sie den Parameter `NO_NT_SCATTER` an, um den SCATTER-Lesezugriff für einen Container explizit auszuschließen. Bestimmte Parameter werden dazu verwendet, den SCATTER-Lesezugriff für alle Container des angegebenen Typs zu aktivieren. Für jeden dieser Parameter ist der Standardwert 0 (bzw. OFF). Die möglichen Werte sind: 0 (oder OFF) und 1 (oder ON).

Anmerkung: Sie können den SCATTER-Lesezugriff nur aktivieren, wenn `DB2NTNOCACHE` auf den `ON` gesetzt ist, um die Windows-Dateicachefunktion abzuschalten. Wenn `DB2NTNOCACHE` auf den Wert `OFF` gesetzt oder nicht definiert ist, wird eine Warnung in das Benachrichtigungsprotokoll für die Systemverwaltung geschrieben, wenn Sie versuchen, den SCATTER-Lesezugriff für einen Container zu aktivieren. Der SCATTER-Lesezugriff bleibt in diesem Fall inaktiviert.

Parameter zur Anpassung des Vorablesezugriffs

Die Parameter zur Anpassung des Vorablesezugriffs heißen `NUMPREFETCHQUEUES` und `PREFETCHQUEUESIZE`. Diese Parameter stehen auf allen Plattformen zur Verfügung und können zur Verbesserung des Datenvorabzugriffs für Pufferpools verwendet werden. Betrachten Sie zum Beispiel den sequenziellen Vorablesezugriff, bei dem der gewünschte `PREFETCHSIZE`-Wert in `PREFETCHSIZE/EXTENTSIZE` Vorableseanforderungen dividiert wird. In diesem Fall werden die Anforderungen in Vorablesewarteschlangen gestellt, aus denen E/A-Server zugeteilt werden, um asynchrone E/A-Operationen auszuführen. Standardmäßig verwaltet der DB2-Datenbankmanager eine Warteschlange in der Größe `max( 100 , 2*NUM_IOSERVERS )` für jede Datenbankpartition. In einigen Umgebungen lässt sich die Leistung entweder durch mehr Warteschlangen und/oder durch Warteschlangen einer anderen Größe verbessern. Die Anzahl der Warteschlangen für den Vorablesezugriff sollte höchstens die Hälfte der Anzahl von E/A-Servern betragen. Berücksichtigen Sie bei der Einstellung dieser Parameter andere Parameter wie beispielsweise `PREFETCHSIZE`, `EXTENTSIZE`, `NUM_IOSERVERS` und die Pufferpoolgröße sowie Auslastungsmerkmale wie z. B. die Anzahl der aktuellen Benutzer.

Wenn Sie meinen, dass die Standardwerte für Ihre Umgebung zu klein sind, erhöhen Sie die Werte zunächst nur leicht. Zum Beispiel könnten Sie NumprefetchQueues=4 und PrefetchQueueSize=200 definieren. Nehmen Sie Änderungen an diesen Parametern kontrolliert vor, so dass Sie die Wirkungen der Änderung überwachen und bewerten können.

Für NumprefetchQueues ist der Standardwert 1, wobei der Wertebereich von 1 bis NumIoservers geht. Wenn Sie NumprefetchQueues auf einen kleineren Wert als 1 setzen, wird er auf den Wert 1 angepasst. Wenn Sie ihn auf einen höheren Wert als NumIoservers setzen, wird er auf den Wert von NumIoservers angepasst.

Der Standardwert für PrefetchQueueSize ist $\max(100, 2 * \text{NumIoservers})$. Der Wertebereich liegt zwischen 1 und 32.767 einschließlich. Wenn Sie PrefetchQueueSize auf einen kleineren Wert als 1 setzen, wird er auf den Standardwert angepasst. Wenn er auf einen höheren Wert als 32.767 gesetzt wird, wird er auf 32.767 angepasst.

DB2CHKPTR

- Betriebssystem: Alle
- Standardwert=OFF, Werte: ON oder OFF
- Definiert, ob die Zeigerprüfung für Eingaben erforderlich ist oder nicht.

DB2CHKSQLDA

- Betriebssystem: Alle
- Standardwert=ON, Werte: ON oder OFF
- Definiert, ob die Prüfung von SQL-Deskriptorbereichen (SQLDA) für Eingaben erforderlich ist oder nicht.

DB2_ENABLE_BUFPD

- Betriebssystem: Alle
- Standardwert=YES, Werte: ON oder OFF
- Gibt an, ob der DB2-Datenbankmanager eine Zwischenpufferung zur Verbesserung der Abfrageleistung einsetzt. Die Pufferung führt möglicherweise nicht in allen Umgebungen zu einer Verbesserung der Abfrageleistung. Um einzelne Verbesserungen in der Abfrageleistung zu ermitteln, sollten Tests durchgeführt werden.

DB2_EVALUNCOMMITTED

- Betriebssystem: Alle
- Standardwert=OFF, Werte: ON, OFF
- Wenn sie aktiviert ist, ermöglicht diese Variable, dass bei Tabellen- oder Indexsuchzugriffen Zeilensperren nach Möglichkeit so lange verzögert bzw. vermieden werden, bis feststeht, dass ein Datensatz der Auswertung der Vergleichselemente entspricht.

Wenn diese Variable aktiviert ist, findet möglicherweise auch eine Auswertung von Vergleichselementen für nicht festgeschriebene Daten statt.

DB2_EVALUNCOMMITTED gilt nur für Anweisungen, die mit der Isolationsstufe Cursorstabilität oder Lesestabilität arbeiten. Bei Indexsuchen muss es sich um einen Index des Typs 2 handeln.

Darüber hinaus werden beim Tabellensuchzugriff gelöschte Zeilen bedingungslos übersprungen, während gelöschte Schlüssel beim Durchsuchen von Indizes des Typs 2 nicht übersprungen werden, sofern nicht die Registrierdatenbankvariable DB2_SKIPDELETED ebenfalls definiert ist.

Die Aktivierung der Registrierdatenbankvariablen DB2_EVALUNCOMMITTED erfolgt mit dem Befehl **db2start**. Die Entscheidung, ob ein verzögertes Sperren anwendbar ist, wird beim Kompilieren oder Binden der Anweisung getroffen.

DB2_EXTENDED_IO_FEATURES

- Betriebssystem: AIX
- Standardwert=OFF, Werte: [ON|OFF],
[ONLINE_BACKUP_IOPRIORITY=[HIGH|MEDIUM|LOW]]
- Setzen Sie diese Variable auf den Wert ON, um die Funktionsmerkmale zur Verbesserung der E/A-Leistung zu aktivieren. Zu diesen Verbesserungen gehören eine Erhöhung der Trefferrate von Hauptspeichercaches sowie eine Verringerung der Latenzzeit für E/A-Operationen mit hoher Priorität. Diese Funktionsmerkmale sind nur für bestimmte Kombinationen aus Software- und Hardwarekonfigurationen verfügbar. Für andere Konfigurationen wird die Einstellung ON entweder vom DB2-Datenbankmanagementsystem oder vom Betriebssystem ignoriert. Eine Konfiguration muss die folgenden Mindestvoraussetzungen erfüllen:
 - Datenbankversion: DB2 Version 9.1.
 - Eine Roheinheit (RAW) muss für Datenbankcontainer verwendet werden (Container in Dateisystemen werden nicht unterstützt).
 - Betriebssystem: AIX 5.3 TL4.
 - Speichersubsystem: Shark DS8000 unterstützt alle erweiterten E/A-Leistungsmerkmale. Shark DS6000 unterstützt nur einen Teil dieser Merkmale. Informationen zu Voraussetzungen und zur Konfiguration finden Sie in der jeweiligen Benutzerdokumentation.

Wenn Sie die Variable DB2_EXTENDED_IO_FEATURES auf den Wert ON setzen, können Sie optional das Schlüsselwort ONLINE_BACKUP_IOPRIORITY verwenden, um die E/A-Priorität für Online-Backups auf den Wert HIGH, MEDIUM oder LOW zu setzen. Die Standardeinstellungen der E/A-Priorität für HIGH, MEDIUM und LOW sind 3, 8 bzw. 12. Mit Hilfe der Registrierdatenbankvariablen DB2_IO_PRIORITY_SETTINGS können Sie diese Einstellungen ändern. Wenn Sie die Variable DB2_EXTENDED_IO_FEATURES auf den Wert ON setzen, ohne das Schlüsselwort ONLINE_BACKUP_IOPRIORITY anzugeben, wird die E/A-Priorität auf den Wert LOW gesetzt.

DB2_EXTENDED_OPTIMIZATION

- Betriebssystem: Alle
- Standardwert=OFF, Werte: ON oder OFF
- Gibt an, ob das Abfrageoptimierungsprogramm die Optimierungserweiterungen zum Verbessern der Abfrageleistung verwendet. Die Erweiterungen verbessern die Abfrageleistung möglicherweise nicht in allen Umgebungen. Um einzelne Verbesserungen in der Abfrageleistung zu ermitteln, sollten Tests durchgeführt werden.

DB2_IO_PRIORITY_SETTING

- Betriebssystem: AIX
- Werte: HIGH:#,MEDIUM:#,LOW:# mit # aus dem Bereich von 1 bis 15
- Diese Variable wird in Kombination mit der Registrierdatenbankvariablen DB2_EXTENDED_IO_FEATURES verwendet. Diese Registrierdatenbankvariable bietet die Möglichkeit, die Standardeinstellungen der Werte HIGH, MEDIUM und LOW für die E/A-Priorität für das DB2-Datenbanksystem zu ändern. Die Standardwerte sind 3, 8 bzw. 12. Der Bereich

der gültigen Einstellungen für die E/A-Priorität reicht von 1 bis 15. Diese Registrierdatenbankvariable muss vor dem Start einer Instanz definiert werden. Jede Änderung erfordert einen Neustart der Instanz. Beachten Sie, dass die Einstellung dieser Registrierdatenbankvariablen allein die erweiterten E/A-Funktionen nicht aktiviert. Diese müssen mit der Variablen DB2_EXTENDED_IO_FEATURES aktiviert werden. Alle Systemvoraussetzungen für DB2_EXTENDED_IO_FEATURES gelten auch für diese Registrierdatenbankvariable.

DB2_KEEPTABLELOCK

- Betriebssystem: Alle
- Standardwert=OFF, Werte: ON, OFF
- Wenn sie aktiviert ist, kann das DB2-Datenbanksystem mit dieser Variablen die Tabellensperre beibehalten, wenn eine Isolationsstufe UR (Nicht festgeschriebener Lesevorgang) oder CS (Cursorstabilität) geschlossen wird. Die Tabellensperre, die beibehalten wird, wird am Ende der Transaktion freigegeben, wie dies bei Suchvorgängen mit der Isolationsstufe RS (Lesestabilität) und RR (Wiederholtes Lesen) der Fall ist.

Diese Registrierdatenbankvariable wird zum Zeitpunkt des Kompilierens oder Bindens einer Anweisung geprüft.

DB2_LARGE_PAGE_MEM

- Betriebssystem: Nur AIX 5.x 64-Bit, Linux, Windows Server 2003
- Standardwert=NULL, Werte: Geben Sie mit Hilfe eines Sterns (*) an, dass alle gültigen Speicherbereiche große Seitenspeicher verwenden sollen, oder geben Sie eine durch Kommas getrennte Liste bestimmter Speicherbereiche an, die den großen Seitenspeicher verwenden sollen. Die verfügbaren Bereiche variieren je nach dem verwendeten Betriebssystem. Unter AIX 5.x (64-Bit) können die folgenden Bereiche angegeben werden: DB, DBMS, FCM oder PRIVATE. Unter Linux kann der folgende Bereich angegeben werden: DB. Unter Windows Server 2003 kann der folgende Bereich angegeben werden: DB.
- Die Registrierdatenbankvariable DB2_LARGE_PAGE_MEM wird verwendet, um die Unterstützung für große Seiten zu aktivieren. Diese Registrierdatenbankvariable macht die Registrierdatenbankvariable DB2_LGPAGE_BP überflüssig und ersetzt sie. DB2_LGPAGE_BP wurde lediglich zur Aktivierung des Speichers mit großen Seiten für den gemeinsam genutzten Datenbankspeicherbereich verwendet. Dieser kann jetzt durch die Angabe von DB2_LARGE_PAGE_MEM=DB aktiviert werden. In einigen Dokumentationen wird erläutert, dass große Seiten mit der Registrierdatenbankvariablen DB2_LGPAGE_BP aktiviert werden. Dies kann als synonym zur Angabe DB2_LARGE_PAGE_MEM=DB betrachtet werden.

Die Verwendung großer Seiten dient hauptsächlich zur Leistungsverbesserung hochleistungsfähiger Datenverarbeitungsanwendungen. Bei Anwendungen, die einen intensiven Speicherzugriff erfordern und große Mengen an virtuellem Speicher verwenden, kann die Verwendung großer Seiten zu einer Leistungsverbesserung führen. Um das DB2-Datenbanksystem für die Verwendung großer Seiten zu aktivieren, müssen Sie zunächst das Betriebssystem für die Verwendung solcher Seiten konfigurieren.

Durch die Aktivierung großer privater Seiten nimmt die Speicherbelegung des DB2-Datenbanksystems deutlich zu, da jeder DB2-Agent mindestens eine große Seite (16 MB) an physischem Speicher belegt. Zum Aktivieren großer Seiten für privaten Agentenspeicher unter DB2

für AIX (64 Bit) über die Einstellung DB2_LARGE_PAGE_MEM=PRIVATE müssen außer der Konfiguration großer Seiten im Betriebssystem die folgenden Voraussetzungen erfüllt sein:

- Der Instanzeigner muss über die Funktionen CAP_BYPASS-_RAC_VMM und CAP_PROPAGATE verfügen.
- Der Kernel muss Schnittstellen unterstützen, mit denen ein Prozess die zugehörige Seitengröße während der Ausführung modifizieren kann.

Unter DB2 für AIX (64 Bit) reduziert diese Variable die Größe des gemeinsam genutzten Speichersegments, das den Datenbankspeicher unterstützt, auf den Mindestbedarf. Standardmäßig wird ein 64-GB-Segment erstellt. Weitere Informationen hierzu finden Sie unter dem Konfigurationsparameter für die Größe des gemeinsam genutzten Datenbankspeichers (*database_memory*). Dadurch wird vermieden, dass mehr gemeinsam genutzter Speicher im Arbeitsspeicher belassen wird, als voraussichtlich verwendet wird.

Das Definieren dieser Variablen schränkt die Möglichkeit ein, die Konfiguration für den gesamten gemeinsam genutzten Datenbankspeicher dynamisch zu erhöhen. Es besteht zum Beispiel eine Einschränkung in Bezug auf die Erhöhung der Größe von Pufferpools.

Wenn diese Variable unter DB2 für AIX (64 Bit) auf 'DB' gesetzt wird, kann die automatische Optimierung für den gemeinsam genutzten Datenbankspeicher (durch die Einstellung des Konfigurationsparameters *database_memory* auf den Wert AUTOMATIC) nicht aktiviert werden.

Unter DB2 für AIX 5L (64 Bit) können Sie diese Variable auf 'FCM' setzen. Da sich der FCM-Speicher in einer eigenen Speichergruppe befindet, müssen Sie das Schlüsselwort FCM dem Wert der Registrierdatenbankvariablen DB2_LARGE_PAGE_MEM hinzufügen, um große Seiten für den FCM-Speicher zu aktivieren.

Unter Linux besteht die zusätzliche Voraussetzung, dass die Bibliothek *libcap.so.1* verfügbar sein muss. Diese Bibliothek muss installiert sein, damit diese Option funktioniert. Wenn diese Option aktiviert wird und die Bibliothek im System nicht vorhanden ist, inaktiviert die DB2-Datenbank die großen Kernelseiten und setzt die Verarbeitung wie ohne sie fort.

Zum Prüfen, ob unter Linux große Kernelseiten verfügbar sind, können Sie den folgenden Befehl absetzen:

```
cat /proc/meminfo
```

Wenn große Kernelseiten verfügbar sind, sollten die drei folgenden Zeilen (mit je nach Größe des auf Ihrem Server konfigurierten Speichers unterschiedlichen Werten) angezeigt werden:

```
HugePages_Total: 200
HugePages_Free: 200
Hugepagesize: 16384 kB
```

Wenn diese Zeilen nicht angezeigt werden oder HugePages_Total den Wert 0 hat, ist eine Konfiguration des Betriebssystems oder des Kernels erforderlich.

Unter Windows ist die Menge an Speicher mit großen Seiten, die auf dem System verfügbar ist, kleiner als der insgesamt verfügbare Speicher. Wenn das System einige Zeit aktiv war, kann der Speicher fragmentiert werden, so dass die Menge an Speicher mit großen Seiten abnimmt.

DB2_LGPAGE_BP

- Betriebssystem: Nur AIX 5.x 64-Bit, Linux
- Standardwert=OFF, Werte: ON oder OFF
- Die Registrierdatenbankvariable DB2_LGPAGE_BP wird nicht weiter unterstützt. Nun wird die Registrierdatenbankvariable DB2_LARGE_PAGE_MEM verwendet.

DB2MAXFSCRSEARCH

- Betriebssystem: Alle
- Standardwert=5, Werte: -1, 1 bis 33 554
- Gibt an, wie viele Steuersätze für freien Speicherbereich beim Hinzufügen eines Eintrags zu einer Tabelle durchsucht werden. Standardmäßig werden fünf Steuersätze für freien Speicherbereich durchsucht. Mit diesem Wert können Sie die Gewichtung zwischen Einfügeschwindigkeit und Speicherwiederverwendung beeinflussen. Mit hohen Werten optimieren Sie die Speicherwiederverwendung. Mit niedrigen Werten optimieren Sie die Einfügeschwindigkeit. Der Wert -1 zwingt den Datenbankmanager, alle Steuersätze für freien Speicherbereich zu durchsuchen.

DB2_MAX_NON_TABLE_LOCKS

- Betriebssystem: Alle
- Standardwert=YES, Werte: Siehe Beschreibung
- Diese Variable definiert die maximale Anzahl von NON-Tabellensperren, die eine Transaktion haben kann, bevor sie alle diese Sperren freigibt. NON-Tabellensperren sind Tabellensperren, die in der Hashtabelle und der Transaktionskette verbleiben, selbst wenn sie von der Transaktion nicht mehr verwendet werden. Da Transaktionen häufig mehrmals auf die gleiche Tabelle zugreifen, kann das Beibehalten der Sperren und das Ändern ihres Status in NON die Leistung verbessern.

Im Hinblick auf optimale Ergebnisse entspricht der für diese Variable empfohlene Wert der maximalen Anzahl von Tabellen, auf die voraussichtlich durch Verbindungen zugegriffen wird. Wenn kein benutzerdefinierter Wert angegeben wird, gilt folgender Standardwert: Wenn die Sperrenliste größer oder gleich

`SQLP_THRESHOLD_VAL_OF_LRG_LOCKLIST_SZ_FOR_MAX_NON_LOCKS`

(zurzeit 8000) ist, ist der Standardwert

`SQLP_DEFAULT_MAX_NON_TABLE_LOCKS_LARGE`

(zurzeit 150). Ansonsten ist der Standardwert

`SQLP_DEFAULT_MAX_NON_TABLE_LOCKS_SMALL`

(zurzeit 0).

DB2_MDC_ROLLOUT

- Betriebssystem: Alle
- Standardwert=ON, Werte: ON oder OFF
- Mit dieser Variablen wird eine Leistungsverbesserung für Löschvorgänge in MDC-Tabellen aktiviert, die als „Rollout“ bezeichnet wird.

Ein Rollout ist eine schnellere Möglichkeit, um Zeilen in einer MDC-Tabelle zu löschen, wenn ganze Zellen (Schnittpunkte von Dimensionswerten) in einer Suchanweisung mit DELETE gelöscht werden. Die Vorteile sind eine reduzierte Protokollierung und eine effizientere Verar-

beitung. Ein Commit sollte nach einer solchen Löschaktion und vor Einfügeaktionen in die Tabelle ausgeführt werden, damit die geleerten Blöcke wiederverwendet werden können.

DB2MEMDISCLAIM

- Betriebssystem: AIX
- Standardwert=YES, Werte: YES oder NO
- Unter AIX verfügt der von Prozessen des DB2-Datenbanksystems verwendete Speicher möglicherweise über einen zugeordneten Paging-Bereich. Die Reservierung dieses Paging-Bereichs bleibt möglicherweise auch dann bestehen, wenn der zugehörige Speicher freigegeben wird. Ob dies der Fall ist, hängt von der (optimierbaren) Richtlinie für die Speicherzuordnung der Verwaltung des virtuellen Speichers auf dem AIX-System ab. Die Registrierdatenbankvariable DB2MEMDISCLAIM steuert, ob DB2-Agenten AIX explizit auffordern, die Zuordnung des reservierten Paging-Bereichs zu dem freigegebenen Speicher aufzuheben. Wenn DB2MEMDISCLAIM auf YES gesetzt wird, sinkt der Bedarf an Paging-Bereich und möglicherweise auch die auf Seitenwechsel (Paging) zurückzuführende Plattenaktivität. Wenn DB2MEMDISCLAIM auf NO gesetzt wird, steigt der Bedarf an Paging-Bereich und möglicherweise auch die auf Seitenwechsel zurückzuführende Plattenaktivität. In einigen Fällen, in denen reichlich Paging-Bereich und so viel Realspeicher verfügbar ist, dass keine Seitenwechsel auftreten, bietet die Einstellung NO eine geringfügige Leistungssteigerung.

DB2MEMMAXFREE

- Betriebssystem: Alle
- Standardwert=8 388 608 Byte, Werte: 0 bis $2^{32}-1$ Byte
- Gibt die maximale Anzahl von Byte an ungenutztem privaten Speicher an, der von Prozessen des DB2-Datenbanksystems zurückbehalten wird, bevor ungenutzter Speicher an das Betriebssystem zurückgegeben wird.

DB2_MEM_TUNING_RANGE

- Betriebssystem: AIX und Windows
- Standardwert=null, Werte: eine Folge von Prozentsätzen n, m mit $n=minfree$ und $m=maxfree$
- Die Größe des physischen Speichers, der von der DB2-Instanz nicht beansprucht wird, spielt eine wichtige Rolle, weil durch sie festgelegt wird, wie viel Speicher andere Anwendungen, die auf derselben Maschine aktiv sind, nutzen können. Wenn die automatische Optimierung für den gemeinsam genutzten Speicher aktiviert ist, hängt die Größe des physischen Speichers, der von einer bestimmten Instanz nicht genutzt wird, vom Speicherbedarf der Instanz (und ihrer aktiven Datenbanken) ab. Wenn eine Instanz dringend zusätzlichen Speichers bedarf, ordnet sie Speicher zu, bis der freie physische Speicher auf dem System den Prozentsatz erreicht, der durch den Parameter *minfree* angegeben wird. Wenn die Instanz weniger Speicher bedarf, behält sie eine größere Kapazität an freiem physischen Speicher bei, die als Prozentsatz durch den Parameter *maxfree* angegeben wird. Daher ist es erforderlich, dass der für *minfree* definierte Wert kleiner als der Wert für *maxfree* ist.

Wenn diese Variable nicht festgelegt wird, berechnet der DB2-Datenbankmanager Werte für *minfree* und *maxfree* auf der Basis der Speicherkapazität auf dem Server. Es wird empfohlen, diese Variable *nicht* festzulegen, sofern Sie nicht mit aktiviertem Self-Tuning Memory Mana-

ger (STMM) arbeiten, DATABASE_MEMORY auf den Wert AUTOMATIC gesetzt haben und Probleme wegen unzureichenden freien physischen Speichers auftreten.

DB2_MMAP_READ

- Betriebssystem: AIX
- Standardwert=ON, Werte: ON oder OFF
- Diese Variable wird in Verbindung mit DB2_MMAP_WRITE verwendet, damit das DB2-Datenbanksystem 'mmap' als alternative E/A-Methode verwenden kann. In den meisten Umgebungen sollte 'mmap' verwendet werden, um Sperren des Betriebssystems zu vermeiden, wenn mehrere Prozesse parallel in verschiedene Abschnitte derselben Datei schreiben. Werden diese Variablen auf ON gesetzt, umgehen Daten, die in die DB2-Pufferpools gelesen und aus diesen geschrieben werden, den AIX-Speichercache. Wenn der DB2-Pufferpool relativ klein ist und Sie diesen Pufferpool nicht vergrößern können oder wollen, sollten Sie die Nutzung des AIX-Speichercaches in Erwägung ziehen, indem Sie die Variablen DB2_MMAP_READ und DB2_MMAP_WRITE auf OFF setzen.

DB2_MMAP_WRITE

- Betriebssystem: AIX
- Standardwert=ON, Werte: ON oder OFF
- Diese Variable wird in Verbindung mit DB2_MMAP_READ verwendet, damit das DB2-Datenbanksystem 'mmap' als alternative E/A-Methode verwenden kann. In den meisten Umgebungen sollte 'mmap' verwendet werden, um Sperren des Betriebssystems zu vermeiden, wenn mehrere Prozesse parallel in verschiedene Abschnitte derselben Datei schreiben. Werden diese Variablen auf ON gesetzt, umgehen Daten, die in die DB2-Pufferpools gelesen und aus diesen geschrieben werden, den AIX-Speichercache. Wenn der DB2-Pufferpool relativ klein ist und Sie diesen Pufferpool nicht vergrößern können oder wollen, sollten Sie die Nutzung des AIX-Speichercaches in Erwägung ziehen, indem Sie die Variablen DB2_MMAP_READ und DB2_MMAP_WRITE auf OFF setzen.

DB2_NO_FORK_CHECK

- Betriebssystem: UNIX
- Standardwert=OFF, Werte: ON, OFF
- Wenn diese Variable aktiviert ist, minimiert der DB2-Laufzeitclient Prüfungen zur Bestimmung, ob der aktuelle Prozess ein Ergebnis eines fork-Aufrufs ist. Dies kann die Leistung von DB2-Anwendungen erhöhen, die nicht mit der API fork() arbeiten.

DB2_NO_MPFA_FOR_NEW_DB

- Betriebssystem: Alle
- Standardwert= nicht gesetzt, Werte: YES
- Für Datenbanken, die durch den Befehl CREATE DATABASE oder die äquivalente API erstellt werden, ist die mehrseitige Dateizuordnung (MPFA) aktiviert. Wenn die MPFA für eine Datenbank aktiviert ist, kann sie nicht mehr inaktiviert werden. Zur Erstellung einer Datenbank mit inaktivierter MPFA setzen Sie diese Registriervariable auf YES und starten die Instanz vor der Erstellung der Datenbank erneut. Wenn diese Registrierdatenbankvariable definiert ist, werden alle Datenbanken mit inaktivierter MPFA erstellt.

Zur Aktivierung der MPFA für eine Datenbank mit inaktivierter MPFA verwenden Sie den Befehl **db2empfa**.

DB2NTMEMSIZE

- Betriebssystem: Windows
- Standardwert =(je nach Speichersegment unterschiedlich)
- Windows erfordert, dass alle gemeinsam genutzten Speichersegmente während der DLL-Initialisierung reserviert werden, um übereinstimmende Adressen in allen Prozessen zu gewährleisten. DB2NTMEMSIZE ermöglicht Benutzern das Überschreiben der DB2-Standardwerte unter Windows (falls erforderlich). In den meisten Situationen sollte der Standardwert ausreichen. Die Angaben für Speichersegmente, Standardgrößen und Überschreibungsoptionen lauten wie folgt: 1) Datenbankkernel: Standardgröße ist 16777216 (16 MB); Überschreibungsoption ist DBMS:<anzahl_byte>. 2) Parallele FCM-Puffer: Standardgröße ist 22020096 (21 MB); Überschreibungsoption ist FCM:<anzahl_byte>. 3) GUI für Datenbankverwaltung: Standardgröße ist 33554432 (32 MB); Überschreibungsoption ist DBAT:<anzahl_byte>. 4) Abgeschirmte gespeicherte Prozeduren: Standardgröße ist 16777216 (16 MB); Überschreibungsoption ist APLD:<anzahl_byte> Sie können mehrere Segmente überschreiben, indem Sie die Überschreibungsoptionen durch ein Semikolon (;) voneinander trennen. Wenn Sie zum Beispiel den Datenbankkernel auf ungefähr 256 KB und die FCM-Puffer auf ungefähr 64 MB begrenzen wollen, geben Sie Folgendes an:

```
db2set DB2NTMEMSIZE=DBMS:256000;FCM:64000000
```

DB2NTNOCACHE

- Betriebssystem: Windows
- Standardwert=OFF, Werte: ON oder OFF
- Die Registrierdatenbankvariable DB2NTNOCACHE gibt an, ob das DB2-Datenbanksystem Datenbankdateien mit der Option NOCACHE öffnet. Wenn DB2NTNOCACHE=ON definiert ist, wird das Zwischenspeichern im Dateisystemcache inaktiviert. Wenn DB2NTNOCACHE=OFF definiert ist, speichert das Betriebssystem DB2-Dateien im Cache. Dies gilt für alle Daten außer für Dateien, die Langfelddaten oder LOB-Daten enthalten. Durch Inaktivieren der Cachefunktion des Betriebssystems steht mehr Speicher für die Datenbank zur Verfügung, so dass der Pufferpool oder der Sortierspeicher vergrößert werden können.

Unter Windows werden Dateien im Cache zwischengespeichert, sobald sie geöffnet werden. Dies ist das Standardverhalten. Für jedes GB in der Datei wird aus einem Systempool ein MB reserviert. Verwenden Sie diese Registrierdatenbankvariable, um die nicht dokumentierte Begrenzung von 192 MB für den Cache außer Kraft zu setzen. Wenn die Cachebegrenzung erreicht ist, wird ein Fehler wegen nicht ausreichender Resource ausgegeben.

DB2NTPRICCLASS

- Betriebssystem: Windows
- Standardwert=null, Wert: R, H, (beliebiger anderer Wert)
- Stellt die Prioritätsklasse für die DB2-Instanz ein (Programm DB2SYSCS.EXE). Es gibt drei Prioritätsklassen:
 - NORMAL_PRIORITY_CLASS (die Standardprioritätsklasse)
 - REALTIME_PRIORITY_CLASS (Wert „R“)
 - HIGH_PRIORITY_CLASS (Wert „H“)

Diese Variable wird in Verbindung mit einzelnen Threadprioritäten (mit DB2PRIORITIES eingestellt) verwendet, um die absolute Priorität von DB2-Threads relativ zu anderen Threads im System zu bestimmen.

Anmerkung: Bei Verwendung dieser Variablen ist Sorgfalt geboten. Eine falsche Verwendung dieser Variablen kann sich negativ auf die Gesamtleistung des Systems auswirken.

Weitere Informationen finden Sie unter der API *SetPriorityClass()* in der Win32-Dokumentation.

DB2NTWORKSET

- Betriebssystem: Windows
- Standardwert=1,1
- Dient zur Änderung der minimalen und der maximalen Größe des Arbeitsbereichs, der für den DB2-Datenbankmanager verfügbar ist. Standardmäßig kann der Arbeitsbereich eines Prozesses, wenn unter Windows keine Seitenwechsel auftreten, nach Bedarf wachsen. Wenn es jedoch zum Paging kommt, liegt der maximale Arbeitsbereich, den ein Prozess haben kann, bei annähernd 1 MB. Mit DB2NTWORKSET kann dieser Standardwert überschrieben werden.

Geben Sie DB2NTWORKSET in der Syntax DB2NTWORKSET=min,max an, wobei 'min' und 'max' MB-Werte sind.

DB2_OVERRIDE_BPF

- Betriebssystem: Alle
- Standardwert=nicht gesetzt, Werte: eine positive Anzahl von Seiten ODER <eintrag>[:<eintrag>...], wobei <eintrag>=<pufferpool-ID>,<anzahl der seiten> ist.
- Diese Variable gibt die Größe des Pufferpools (in Seiten) an, der bei der Aktivierung der Datenbank oder bei der Herstellung der ersten Verbindung zu erstellen ist. Diese Variable ist nützlich, wenn es während der Datenbankaktivierung oder der ersten Verbindung auf Grund begrenzter Speicherkapazitäten zu Fehlern kommt. Die Speicherknappheit kann infolge des seltenen Falls eines Mangels an Realspeicher auftreten oder durch den Versuch seitens des Datenbankmanagers verursacht werden, einen großen Pufferpool für nicht korrekt konfigurierte Pufferpools zuzuordnen. Wenn z. B. selbst ein Pufferpool der Minimalgröße von 16 Seiten nicht vom Datenbankmanager bereitgestellt werden kann, haben Sie die Möglichkeit, mit Hilfe dieser Umgebungsvariablen eine kleinere Anzahl von Seiten anzugeben und den Versuch zu wiederholen. Der für diese Variable angegebene Wert überschreibt die aktuelle Pufferpoolgröße.

Sie können auch <eintrag>[:<eintrag>...] (mit <eintrag>=<pufferpool-ID>,<anzahl-der-seiten>) angeben, um die Größe aller oder einer Teilmenge der Pufferpools temporär zu ändern, so dass sie gestartet werden können.

DB2_PINNED_BP

- Betriebssystem: AIX, HP-UX, Linux
- Standardwert=NO, Werte: YES oder NO
- Mit dieser Variablen wird der globale Datenbankspeicher (einschließlich Pufferpools), der der Datenbank im Hauptspeicher zugeordnet ist, unter

einigen AIX-Betriebssystemen angegeben. Wenn dieser globale Datenbankspeicher im Hauptspeicher des Systems bleibt, ist eine konsistentere Datenbankleistung möglich.

Zum Beispiel führt eine Auslagerung des Pufferpools aus dem Systemhauptspeicher zu einer verminderten Datenbankleistung. Die geringere Zahl von Lese- und Schreibvorgängen auf der Festplatte, wenn sich die Pufferpools im Systemspeicher befinden, verbessert die Datenbankleistung. Falls andere Anwendungen mehr Hauptspeicher benötigen, sollten Sie abhängig vom Systemhauptspeicherbedarf die Auslagerung des globalen Datenbankspeichers aus dem Hauptspeicher zulassen.

Unter Linux ist neben einer Modifizierung dieser Registrierdatenbankvariablen auch die Bibliothek *libcap.so.1* erforderlich.

Unter DB2 für AIX (64-Bit) reduziert diese Variable die Größe des gemeinsam genutzten Speichersegments. Dabei wird der Datenbankspeicher auf den Mindestbedarf reduziert. (Standardmäßig wird ein 64-GB-Segment erstellt. Weitere Informationen finden Sie unter dem Konfigurationsparameter *database_memory*.) Dadurch wird vermieden, dass mehr gemeinsam genutzter Speicher im Arbeitsspeicher belassen wird, als voraussichtlich verwendet wird.

Das Definieren dieser Variablen schränkt die Möglichkeit ein, die Konfiguration für den gesamten gemeinsam genutzten Datenbankspeicher dynamisch zu erhöhen. Es besteht zum Beispiel eine Einschränkung in Bezug auf die Erhöhung der Größe von Pufferpools.

Wenn diese Variable unter DB2 für AIX (64-Bit) auf 'YES' gesetzt wird, kann die automatische Optimierung für den gemeinsam genutzten Datenbankspeicher (durch die Einstellung des Konfigurationsparameters *database_memory* auf den Wert AUTOMATIC) nicht aktiviert werden.

Für HP-UX in einer 64-Bit-Umgebung müssen Sie zusätzlich zur Änderung dieser Registrierdatenbankvariablen der DB2-Instanzgruppe das Zugriffsrecht MLOCK erteilen. Dazu führt ein Benutzer mit Rootberechtigung die folgenden Aktionen aus:

1. Er fügt die DB2-Instanzgruppe der Datei */etc/privgroup* hinzu. Wenn die DB2-Instanzgruppe beispielsweise zur Gruppe *db2iadm1* gehört, müssen Sie der Datei */etc/privgroup* die folgende Zeile hinzufügen:

```
db2iadm1 MLOCK
```

2. Führen Sie den folgenden Befehl aus:

```
setprivgrp -f /etc/privgroup
```

DB2PRIORITIES

- Betriebssystem: Alle
- Einstellung der Werte von der Plattform abhängig
- Steuert die Prioritäten von DB2-Prozessen und -Threads.

DB2_RESOURCE_POLICY

- Betriebssystem: AIX 5 oder höher, alle Linux-Versionen außer zSeries (32-Bit), Windows Server 2003 oder höher
- Standardwert=nicht gesetzt, Werte: gültiger Pfad zur Konfigurationsdatei
- Definiert eine Ressourcenrichtlinie, mit deren Hilfe die von der DB2-Datenbank genutzten Betriebssystemressourcen begrenzt werden können, oder sie enthält Regeln für die Zuordnung bestimmter Betriebssystemressourcen zu bestimmten DB2-Datenbankobjekten. Unter AIX, Linux oder Windows kann diese Registrierdatenbankvariable zum Beispiel zur

Begrenzung der vom DB2-Datenbanksystem genutzten Gruppe von Prozessoren verwendet werden. Der Speicherbereich der Ressourcensteuerung variiert in Abhängigkeit vom Betriebssystem.

Auf AIX-Maschinen mit aktiviertem NUMA kann eine Richtlinie definiert werden, die angibt, welche Ressourcengruppen das DB2-Datenbanksystem verwendet. Wenn die Ressourcengruppenbindung verwendet wird, wird jeder einzelne DB2-Prozess an eine bestimmte Ressourcengruppe gebunden. Dies kann in einigen Leistungs-optimierungsszenarios von Vorteil sein.

Sie können die Registrierdatenbankvariable festlegen, um den Pfad zu einer Konfigurationsdatei anzugeben, die eine Richtlinie für das Binden von DB2-Prozessen an Betriebssystemressourcen definiert. Die Ressourcenrichtlinie ermöglicht Ihnen die Angabe einer Gruppe von Betriebssystemressourcen, auf die das DB2-Datenbanksystem zu beschränken ist. Jeder DB2-Prozess wird an eine einzige Ressource der Gruppe gebunden. Die Ressourcenzuordnung erfolgt in einem zirkulären Umlauf.

Beispiele für Konfigurationsdateien:

Beispiel 1: Binden aller DB2-Prozesse entweder an CPU 1 oder an CPU 3.

```
<RESOURCE_POLICY>
<METHOD>CPU</METHOD>
<RESOURCE>1</RESOURCE>
<RESOURCE>3</RESOURCE>
</RESOURCE_POLICY>
```

Beispiel 2: Binden von DB2-Prozessen an eine der folgenden Ressourcengruppen:

```
sys/node.03.00000, sys/node.03.00001, sys/node.03.00002,
sys/node.03.00003
<RESOURCE_POLICY>
<METHOD>RSET</METHOD>
<RESOURCE>sys/node.03.00000</RESOURCE>
<RESOURCE>sys/node.03.00001</RESOURCE>
<RESOURCE>sys/node.03.00002</RESOURCE>
<RESOURCE>sys/node.03.00003</RESOURCE>
</RESOURCE_POLICY>
```

Anmerkung: Die Verwendung der Methode RSET setzt die CAP_NUMA_ATTACH-Funktionalität voraus und wird unter Linux nicht unterstützt.

Die in der Registrierdatenbankvariablen DB2_RESOURCE_POLICY angegebene Konfigurationsdatei akzeptiert ein SCHEDULING_POLICY-Element. Über das SCHEDULING_POLICY-Element können Sie auf einigen Plattformen Folgendes auswählen:

- Die Planungsrichtlinie des Betriebssystems, die vom DB2-Server verwendet wird

Sie können eine Betriebssystemplanungsrichtlinie für DB2 unter AIX und für DB2 unter Windows über die Registrierdatenbankvariable DB2NTPRICLASS definieren.

- Die Betriebssystemprioritäten, die von einzelnen DB2-Serveragenten verwendet werden

Alternativ können Sie zur Steuerung der Betriebssystemplanungsrichtlinie und zur Festlegung der DB2-Agentenprioritäten die Registrierdatenbankvariablen DB2PRIORITIES und DB2NTPRICLASS verwenden. Die Spezifikation eines SCHEDULING_POLICY-Elements in der Konfigurationsdatei für Ressourcenrichtlinien stellt eine gemeinsame Stelle zur Angabe sowohl der Planungsrichtlinie als auch der zugeordneten Agentenprioritäten zur Verfügung.

Beispiel 1: Auswahl der AIX-Planungsrichtlinie SCHED_FIFO2 mit Zuordnung zu einer höheren Priorität für die Prozesse zum Schreiben und Lesen des DB2-Protokolls.

```
<RESOURCE_POLICY>
  <SCHEDULING_POLICY>
    <POLICY_TYPE>SCHED_FIFO2</POLICY_TYPE>
    <PRIORITY_VALUE>60</PRIORITY_VALUE>

    <EDU_PRIORITY>
      <EDU_NAME>db2loggr</EDU_NAME>
      <PRIORITY_VALUE>56</PRIORITY_VALUE>
    </EDU_PRIORITY>

    <EDU_PRIORITY>
      <EDU_NAME>db2loggr</EDU_NAME>
      <PRIORITY_VALUE>56</PRIORITY_VALUE>
    </EDU_PRIORITY>
  </SCHEDULING_POLICY>
</RESOURCE_POLICY>
```

Beispiel 2: Ersatz für DB2NTPRCLASS=H unter Windows.

```
<RESOURCE_POLICY>
  <SCHEDULING_POLICY>
    <POLICY_TYPE>HIGH_PRIORITY_CLASS</POLICY_TYPE>
  </SCHEDULING_POLICY>
</RESOURCE_POLICY>
```

DB2_SKIPDELETED

- Betriebssystem: Alle
- Standardwert=OFF, Werte: ON, OFF
- Wenn diese Variable aktiviert ist, ermöglicht sie Anweisungen mit den Isolationsstufen Cursorstabilität oder Lesestabilität, gelöschte Schlüssel bei einem Indexzugriff und gelöschte Zeilen bei einem Tabellenzugriff bedingungslos zu überspringen. Wenn DB2_EVALUNCOMMITTED aktiviert ist, werden gelöschte Zeilen automatisch übersprungen. Nicht festgeschriebene Pseudolöschungen von Schlüssel in Indizes des Typs 2 werden jedoch nur übersprungen, wenn auch DB2_SKIPDELETED aktiviert ist.

Diese Registrierdatenbankvariable wirkt sich nicht auf das Verhalten von Cursors in den DB2-Katalogtabellen aus.

Diese Registrierdatenbankvariable wird mit dem Befehl 'db2start' aktiviert.

DB2_SKIPINSERTED

- Betriebssystem: Alle
- Standardwert=OFF, Werte: ON, OFF
- Wenn die Registrierdatenbankvariable DB2_SKIPINSERTED aktiviert ist, können Anweisungen mit der Isolationsstufe Cursorstabilität oder Lesestabilität nicht festgeschriebene eingefügte Zeilen so überspringen, als wären sie nicht eingefügt worden. Diese Registrierdatenbankvariable wirkt sich nicht auf das Verhalten von Cursors in den DB2-Katalogtabellen aus. Diese Registrierdatenbankvariable wird beim Datenbankstart aktiviert; die Entscheidung darüber, nicht festgeschriebene eingefügte Zeilen zu überspringen, wird bei der Kompilierung oder dem Bindevorgang gefällt.

DB2_SMS_TRUNC_TMPTABLE_THRESH

- Betriebssystem: Alle
- Standardwert=0, Werte: -1, 0 - n, wobei n=Anzahl der EXTENTSIZE großen Speicherbereiche pro temporärer Tabelle im SMS-Tabellenbereichscontainer ist, die zu verwalten sind
- Diese Variable gibt einen minimalen Schwellenwert für die Dateigröße an, mit der die Datei, die eine temporäre Tabelle darstellt, in SMS-Tabellenbereichen verwaltet wird.

Diese Variable wird standardmäßig auf den Wert 0 gesetzt, d. h. es wird keine spezielle Schwellenwertbehandlung ausgeführt. Stattdessen wird die Datei auf die Größe 0 abgeschnitten, wenn die Tabelle nicht mehr benötigt wird.

Wenn der Wert dieser Variablen größer als 0 ist, wird eine größere Datei behalten. Dadurch kann der Systemaufwand verringert werden, der mit dem Löschen und erneuten Erstellen der Datei bei jeder Verwendung einer temporären Tabelle verbunden ist.

Wenn diese Variable auf den Wert -1 gesetzt wird, wird die Datei nicht abgeschnitten und kann uneingeschränkt an Größe zunehmen, soweit die Systemressourcen dies zulassen.

DB2_SORT_AFTER_TQ

- Betriebssystem: Alle
- Standardwert=NO, Werte: YES oder NO
- Gibt an, wie das Optimierungsprogramm mit gezielt übertragenen Tabellenwarteschlangen in einer Umgebung mit partitionierten Datenbanken arbeitet, wenn die Daten für den Empfänger sortiert sein müssen und die Anzahl der Empfängerknoten der Anzahl der Senderknoten entspricht.

Wenn DB2_SORT_AFTER_TQ= NO ist, sortiert das Optimierungsprogramm Zeilen in der Regel auf der sendenden Seite und mischt die Zeilen auf der empfangenden Seite zusammen.

Wenn DB2_SORT_AFTER_TQ= YES ist, überträgt das Optimierungsprogramm in der Regel die Zeilen unsortiert, mischt sie auf der empfangenden Seite nicht zusammen, sondern sortiert die Zeilen nach Empfang aller Zeilen auf der empfangenden Seite.

DB2_SELUDI_COMM_BUFFER

- Betriebssystem: Alle
- Standardwert=OFF, Werte=ON, OFF
- Diese Variable wird bei der Verarbeitung von Blockcursoren über Abfragen mit SELECT from UPDATE/INSERT/DELETE (UDI) verwendet. Wenn aktiviert, verhindert diese Registrierdatenbankvariable, dass das Ergebnis einer Abfrage in einer temporären Tabelle gespeichert wird. Stattdessen versucht das DB2-Datenbanksystem bei der OPEN-Verarbeitung eines Blockcursors für eine Abfrage SELECT from UDI, das gesamte Ergebnis der Abfrage direkt im Speicherbereich des Kommunikationspuffers zu puffern.

Anmerkung: Wenn der Kommunikationspufferbereich für das gesamte Ergebnis der Abfrage nicht ausreicht, wird der Fehler SQL-CODE -906 ausgegeben und die Transaktion mit ROLLBACK rückgängig gemacht. Informationen zur Anpassung der Größe des Kommunikationspufferbereichs für lokale und ferne Anwendungen finden Sie in den Beschreibungen der Konfigurationsparameter *aslheapsz* bzw. *rqrioblk* des Datenbankmanagers.

Diese Registrierdatenbankvariable wird in partitionierten Datenbankumgebungen oder bei aktivierter partitionsinterner Parallelität nicht unterstützt.

DB2_TRUSTED_BINDIN

- Betriebssystem: Alle

- Standardwert=OFF, Werte=OFF, ON, CHECK
- Wenn die Variable DB2_TRUSTED_BINDIN aktiviert ist, beschleunigt sie die Ausführung von Abfrageanweisungen, die Hostvariablen innerhalb einer eingebetteten, nicht abgeschränkten gespeicherten Prozedur enthalten.

Wenn diese Variable aktiviert ist, findet keine Konvertierung vom externen SQLDA-Format in ein internes DB2-Format beim Binden von SQL- und XQuery-Anweisungen statt, die in einer eingebetteten, nicht abgeschränkten gespeicherten Prozedur enthalten sind. Dies führt zu einer beschleunigten Verarbeitung eingebetteter SQL- und XQuery-Anweisungen.

Die folgenden Datentypen werden in eingebetteten, nicht abgeschränkten gespeicherten Prozeduren nicht unterstützt, wenn diese Variable aktiviert ist:

- SQL_TYP_DATE
- SQL_TYP_TIME
- SQL_TYP_STAMP
- SQL_TYP_DATA LINK
- SQL_TYP_CGSTR
- SQL_TYP_BLOB
- SQL_TYP_CLOB
- SQL_TYP_DBCLOB
- SQL_TYP_CSTR
- SQL_TYP_LSTR
- SQL_TYP_BLOB_LOCATOR
- SQL_TYP_CLOB_LOCATOR
- SQL_TYP_DCLOB_LOCATOR
- SQL_TYP_BLOB_FILE
- SQL_TYP_CLOB_FILE
- SQL_TYP_DCLOB_FILE
- SQL_TYP_BLOB_FILE_OBSOLETE
- SQL_TYP_CLOB_FILE_OBSOLETE
- SQL_TYP_DCLOB_FILE_OBSOLETE

Wenn diese Datentypen auftreten, wird der Fehler SQLCODE –804, SQL-STATE 07002 zurückgegeben.

Anmerkung: Der Datentyp und die Länge der Eingabehostvariablen muss exakt mit dem internen Datentyp und der Länge des entsprechenden Elements übereinstimmen. Für Hostvariablen wird diese Voraussetzung immer erfüllt. Bei Parametermarken muss jedoch sorgfältig darauf geachtet werden, dass übereinstimmende Datentypen verwendet werden. Zur Sicherstellung, dass die Datentypen und Längen für alle Eingabehostvariablen übereinstimmen, kann die Option CHECK verwendet werden. Allerdings macht diese Option die meisten Leistungsvorteile wieder zunichte.

DB2_USE_ALTERNATE_PAGE_CLEANING

- Betriebssystem: Alle
- Standardwert=nicht gesetzt, Werte: ON, OFF

- Diese Variable gibt an, ob eine DB2-Datenbank die alternative Methode der Seitenlöschalgorithmen oder die Standardmethode der Seitenlöschfunktion verwendet. Wenn diese Variable auf den Wert ON gesetzt ist, verwendet das DB2-System eine Seitenlöschmethode, die geänderte Seiten auf die Platte schreibt, dem Wert von LSN_GAP voraus bleibt und proaktiv Seiten ermittelt, die bereinigt werden können. Dadurch können die Seitenlöschfunktionen die verfügbare E/A-Bandbreite der Platte besser nutzen. Wenn diese Variable auf den Wert ON gesetzt ist, ist der Datenbankkonfigurationsparameter *chnpgps_thresh* nicht länger relevant, weil er keinen Einfluss auf die Aktivitäten der Seitenlöschfunktionen hat.

Zugehörige Konzepte:

- „Registrierdatenbank- und Umgebungsvariablen von DB2“ auf Seite 601

Data Links-Variablen

Tabelle 77. Data Links-Variablen

Variablenname	Betriebssystem	Werte
Beschreibung		
DLFM_BACKUP_DIR_NAME	AIX, Windows, Solaris	Standardwert: Null Werte: ein beliebiger gültiger Pfad
Gibt den Pfad des Verzeichnisses an, in dem archivierte Dateien gesichert werden, wenn DLFM_BACKUP_TARGET auf LOCAL eingestellt ist.		
DLFM_BACKUP_TARGET	AIX, Windows, Solaris	Standardwert: Null Werte: LOCAL, TSM, XBSA
Gibt den zu verwendenden Backup-Datenträger an. Wird diese Variable auf LOCAL eingestellt, muss die Variable DLFM_BACKUP_DIR_NAME definiert werden. Wird diese Variable auf XBSA eingestellt, muss die Variable DLFM_BACKUP_TARGET_LIBRARY definiert werden. Wenn Sie die Einstellung dieser Registrierdatenbankvariablen von einem Ziel in ein anderes ändern, werden die archivierten Dateien nicht verschoben. Nur neue Backups werden an die neue Position platziert. Zuvor archivierte Dateien werden nicht verschoben.		
DLFM_BACKUP_TARGET_LIBRARY	AIX, Windows, Solaris	Standardwert: Null Werte: ein beliebiger gültiger Pfad zum Namen der DLL oder der gemeinsam benutzten Bibliothek
Gibt den vollständig qualifizierten Pfad zur DLL des XBSA-kompatiblen Archivierungsserver oder zur gemeinsam benutzten Bibliothek an. Diese Bibliothek wird unter Verwendung der Bibliothek <i>libdfmxbsa.a</i> geladen. Diese Variable muss definiert werden, wenn DLFM_BACKUP_TARGET auf XBSA eingestellt ist. Diese Variable gilt nicht, wenn DLFM_BACKUP_TARGET auf einen anderen Wert eingestellt wird.		
DLFM_GC_MODE	AIX, Windows, Solaris	Standardwert: PASSIVE Werte: SLEEP, PASSIVE oder ACTIVE
Gibt die Steuerung der Bereinigung des Speichers von ungenutzten Dateien (Garbage File Collection) auf dem Data Links-Server an. Beim Wert SLEEP erfolgt keine Bereinigung. Wenn auf den Wert PASSIVE gesetzt, wird die Bereinigung nur dann ausgeführt, wenn keine anderen Transaktionen aktiv sind. Wenn auf ACTIVE gesetzt, wird die Speicherbereinigung durchgeführt, auch wenn andere Anwendungen aktiv sind.		

Tabelle 77. Data Links-Variablen (Forts.)

Variablenname	Betriebssystem	Werte
Beschreibung		
DLFM_INSTALL_PATH	AIX, Windows, Solaris	Standardwert Unter AIX und Solaris: /home/<instanz>/sqllib/bin; dabei ist <instanz> die Instanz-ID von Data Links Manager. Unter Windows: %DB2PATH%\bin (sofern %DB2PATH% definiert ist) oder c:\sqllib\bin (sofern %DB2PATH% nicht definiert ist) Bereich: beliebiger gültiger Pfad
Gibt den Pfad an, in dem die ausführbaren Data Links-Dateien installiert sind.		
DLFM_PORT	AIX, Windows, Solaris	Standardwert: 50100 Werte: jede gültige Portnummer
Gibt die Portnummer an, die zur Kommunikation mit dem Data Links-Server verwendet wird, auf dem DB2 Data Links Manager aktiv ist.		
DLFM_TSM_MGMTCLASS	AIX, Windows, Solaris	Standardwert: die Standard-TSM-Verwaltungsklasse Werte: jede gültige TSM-Verwaltungsklasse
Gibt an, welche TSM-Verwaltungsklasse zum Archivieren und Abrufen von verknüpften Dateien verwendet wird. Wenn für diese Variable kein Wert definiert ist, wird die TSM-Standardverwaltungsklasse verwendet.		
DLFM_START_ASCOPYD	AIX, Windows, Solaris	Standardwert: NO Werte: YES, NO
Gibt an, ob der Data Links Manager-Replikationsdämon (DLFM_ASCOPYD) immer beim Start von DLFM gestartet wird. Der Data Links Manager-Replikationsdämon muss gestartet werden, wenn DB2 Replication verwendet wird, um DATALINK-Dateien zum bzw. vom Data Links Manager-Server zu kopieren. Wird diese Variable auf YES eingestellt, muss DLFM_ASCOPYD_PORT definiert werden.		
DLFM_ASCOPYD_PORT	AIX, Windows, Solaris	Standardwert: Null Werte: jede gültige Portnummer
Gibt die Nummer des TCP/IP-Ports an, an dem der Data Links Manager-Replikationsdämon (DLFM_ASCOPYD) Anforderungen für Dateireplikationen empfängt. Diese Variable muss definiert werden, wenn die Variable DLFM_START_ASCOPYD auf YES eingestellt ist.		
DLFM_NUM_ARCHIVE_SUBSYSTEMS	AIX, Windows, Solaris	Standardwert: 2 Werte: alle Zahlen größer-gleich 1.
Gibt die Anzahl der DLFM Copy Daemon-Prozesse an, die auf einem bestimmten DLFM-Server ausgeführt werden sollen. Je größer die Anzahl der Kopierprozesse, desto größer der Durchsatz beim Backup verbundener Dateien. Allerdings sollte dieser Wert der Anzahl der E/A-Kanäle entsprechen, die zum Kopieren verbundener Dateien in den ausgewiesenen Archivierungsbereich verfügbar sind. Ist der Wert zu hoch, kann die Menge der verbrauchten Systemressourcen den Nutzen der E/A-Parallelität mindern.		
DLFM_AUTOSTART	AIX, Solaris	Standardwert: NO Werte: YES, NO

Tabelle 77. Data Links-Variablen (Forts.)

Variablenname	Betriebssystem	Werte
Beschreibung		
Gibt an, ob der DLFM-Server bei einem Warmstart des Betriebssystems immer automatisch gestartet wird. Diese Variable wird vom Script dlfsmount überprüft. Dieses Script wird während der Verarbeitung des Warmstarts aus der Datei /etc/inittab aufgerufen.		

Zugehörige Konzepte:

- „Registrierdatenbank- und Umgebungsvariablen von DB2“ auf Seite 601

Verschiedene Variablen

DB2ADMINSERVER

- Betriebssystem: Windows und UNIX
- Standardwert=null
- Gibt den DB2-Verwaltungsserver (DAS) an.

DB2CLIINIPATH

- Betriebssystem: Alle
- Standardwert=null
- Dient zum Überschreiben des Standardpfads der DB2-CLI/ODBC-Konfigurationsdatei (db2cli.ini) und zum Angeben einer alternativen Speicherposition auf dem Client. Der definierte Wert muss ein gültiger Pfad auf dem Clientsystem sein.

DB2_COMMIT_ON_EXIT

- Betriebssystem: UNIX
- Standardwert=OFF, Werte: OFF/NO/0 oder ON/YES/1
- Auf UNIX-Plattformen vor Version 8 schrieb DB2 alle verbleibenden unvollständigen Transaktionen bei erfolgreicher Beendigung der Anweisung fest. In Version 8 wurde diese Funktionsweise geändert, so dass unvollständige Transaktionen bei Beendigung mit ROLLBACK rückgängig gemacht wurden. Diese Registrierdatenbankvariable gibt Benutzern mit Anwendungen, die von der früheren Funktionsweise abhängig sind, die Möglichkeit, diese Funktionsweise in Version 8 zu aktivieren.

Beachten Sie, dass in Version 10 diese Registrierdatenbankvariable und die Funktionsweise mit Commit bei Beendigung nicht weiter unterstützt werden. Benutzer sollten bestimmen, ob Anwendungen von ihnen, die vor Version 8 entwickelt wurden, weiterhin von dieser Funktionsweise abhängig sein sollten, und die entsprechenden expliziten COMMIT-Anweisungen nach Bedarf in die Anwendungen einfügen. Wenn die Registrierdatenbankvariable aktiviert ist, sollte sorgsam vermieden werden, neue Anwendungen zu implementieren, die keine expliziten Commits vor ihrer Beendigung ausführen.

Die meisten Benutzer sollten die Standardeinstellung dieser Registrierdatenbankvariablen belassen.

DB2_CREATE_DB_ON_PATH

- Betriebssystem: Windows
- Standardwert=null, Werte: YES oder NO
- Setzen Sie diese Registrierdatenbankvariable auf den Wert YES, um die Unterstützung für die Verwendung eines Pfads (und eines Laufwerks)

als Datenbankpfad zu aktivieren. Die Einstellung der Variablen DB2_CREATE_DB_ON_PATH wird überprüft, wenn eine Datenbank erstellt wird, wenn der Konfigurationsparameter DFTDBPATH des Datenbankmanagers definiert wird und wenn ein Restore einer Datenbank durchgeführt wird. Der vollständig qualifizierte Datenbankpfad kann bis zu 215 Zeichen lang sein.

Wenn DB2_CREATE_DB_ON_PATH nicht definiert wird (oder auf den Wert NO gesetzt wird) und Sie beim Erstellen oder beim Restore einer Datenbank einen Pfad für den Datenbankpfad angeben, wird der Fehler SQL1052N zurückgegeben.

Wenn DB2_CREATE_DB_ON_PATH nicht definiert wird (oder auf den Wert NO gesetzt wird) und Sie den Konfigurationsparameter DFTDBPATH der Datenbank aktualisieren, wird der Fehler SQL5136N zurückgegeben.

Achtung:

Wenn die Pfadunterstützung zur Erstellung neuer Datenbanken verwendet wird, ist es möglich, dass vor Version 9.1 geschriebene Anwendungen, die die API 'db2DbDirGetNextEntry()' oder eine ältere Version dieser API verwenden, nicht korrekt funktionieren. Detaillierte Informationen zu verschiedenen Szenarios sowie zur geeigneten Vorgehensweise finden Sie unter <http://www.ibm.com/software/data/db2/udb/support/>.

DB2DEFPREP

- Betriebssystem: Alle
- Standardwert=NO, Werte: ALL, YES oder NO
- Simuliert das Laufzeitverhalten der Precompiler-Option DEFERRED_PREPARE für Anwendungen, die vor der Verfügbarkeit dieser Option vorkompiliert wurden. Wenn zum Beispiel eine Anwendung von DB2 Version 2.1.1 oder früher in einer Umgebung von DB2 Version 2.1.2 oder später ausgeführt würde, könnte der Parameter DB2DEFPREP verwendet werden, um das gewünschte Verhalten einer „verzögerten Vorbereitung“ (Deferred Prepare) zu simulieren.

DB2_DISABLE_FLUSH_LOG

- Betriebssystem: Alle
- Standardwert=OFF, Werte: ON oder OFF
- Gibt an, ob das Schließen der aktiven Protokolldatei nach Abschluss eines Online-Backups inaktiviert werden soll.

Wenn ein Online-Backup abgeschlossen ist, wird die letzte aktive Protokolldatei abgeschnitten, geschlossen und zur Archivierung verfügbar gemacht. Hierdurch wird sichergestellt, dass Ihrem Online-Backup alle zur Recovery benötigten archivierten Protokolldateien zur Verfügung stehen. Sie möchten möglicherweise das Schließen der letzten aktiven Protokolldatei inaktivieren, wenn Sie Sorge haben, dass Sie Teile des Protokollfolgennummernbereichs (LSN) verschwenden. Jedes Mal, wenn eine aktive Protokolldatei abgeschnitten wird, wird die Protokollfolgennummer um einem zum abgeschnittenen Speicherplatz proportionalen Betrag erhöht. Wenn Sie jeden Tag eine große Anzahl von Online-Backups durchführen, ist es vielleicht sinnvoll, das Schließen der letzten aktiven Protokolldatei zu inaktivieren.

Darüber hinaus kann es sinnvoll sein, das Schließen der letzten aktiven Protokolldatei zu inaktivieren, wenn Sie feststellen, dass Sie Nachrichten über ein volles Protokoll kurz nach Abschluss des Online-Backups emp-

fangen. Wenn eine Protokolldatei abgeschnitten wird, wird der reservierte Speicherbereich für aktive Protokolle um einen Betrag vergrößert, der proportional zur Größe des abgeschnittenen Protokolls ist. Der Speicherbereich für aktive Protokolle wird freigegeben, wenn die abgeschnittene Protokolldatei wieder zur Verwendung verfügbar gemacht wird. Die Wiederverfügbarmachung erfolgt kurz nach dem Zeitpunkt, zu dem die Protokolldatei inaktiv wird. Während des kurzen Intervalls zwischen diesen beiden Ereignissen empfangen Sie möglicherweise Nachrichten über ein volles Protokoll.

Bei jedem Backup, das Protokolle mit einbezieht, wird diese Registrierdatenbankvariable ignoriert, da die aktive Protokolldatei abgeschnitten und geschlossen werden muss, um die Protokolle mit in das Backup einzubeziehen.

DB2_DISCONNECT_ON_INTERRUPT

- Betriebssystem: Alle
- Standardwert= NO, Werte: YES, TRUE, 1 oder NO, FALSE, 0
- Wenn diese Variable auf den Wert YES (TRUE bzw. 1) gesetzt ist, gibt sie an, dass die Verbindung zu einem Server mit DB2 UDB z/OS Version 8 sofort abbrechen ist, wenn ein Interrupt auftritt. Sie können diese Variable in den folgenden Konfigurationen verwenden:
 - Wenn Sie einen DB2-Client der Version 8 mit einem Server unter DB2 UDB z/OS Version 8 betreiben, setzen Sie die Variable DB2_DISCONNECT_ON_INTERRUPT auf dem Client auf den Wert YES.
 - Wenn Sie einen DB2-Client der Version 8 über ein DB2 Connect-Gateway der Version 8 mit einem Server unter DB2 UDB z/OS verbinden, setzen Sie DB2_DISCONNECT_ON_INTERRUPT auf dem Gateway auf den Wert YES.

DB2_DISPATCHER_PEEKTIMEOUT

- Betriebssystem: Alle
- Standardwert: 1, Werte: 0 - 32767 Sekunden; 0 gibt an, dass das Zeitlimit sofort abläuft
- Mit DB2_DISPATCHER_PEEKTIMEOUT können Sie die Zeit (in Sekunden) anpassen, die eine Zuteileroutine (Dispatcher) auf die Verbindungsanforderung eines Clients wartet, bevor sie den Client an einen Agenten weitergibt. In den meisten Fällen sollte eine Anpassung dieser Registrierdatenbankvariablen nicht nötig sein. Diese Registrierdatenbankvariable wirkt sich nur auf Instanzen aus, für die der DB2 Connect-Verbindungskonzentrator aktiviert ist.

Diese Registrierdatenbankvariable und die Registrierdatenbankvariable DB2_SERVER_CONTIMEOUT konfigurieren beide die Behandlung eines neuen Clients während der Verbindungszeit. Wenn viele langsame Clients eine Verbindung zu einer Instanz herstellen, wird die Zuteileroutine möglicherweise bis zu 1 Sekunde aufgehalten, um das Zeitlimit jedes Clients abzuwarten. Auf diese Weise wird die Zuteileroutine zu einem Engpass, wenn viele Clients gleichzeitig eine Verbindung herstellen. Wenn in einer Instanz mit mehreren aktiven Datenbanken sehr langsame Verbindungszeiten festgestellt werden, kann DB2_DISPATCHER_PEEKTIMEOUT auf den Wert 0 herabgesetzt werden. Die Herabsetzung von DB2_DISPATCHER_PEEKTIMEOUT bewirkt, dass die Zuteileroutine nur kurz prüft, ob die Verbindungsanforderung des Clients bereits da ist, ohne auf das Eintreffen der Verbindungsanforderung zu

warten. Wenn ein ungültiger Wert definiert wird, wird der Standardwert verwendet. Diese Registrierdatenbankvariable ist nicht dynamisch.

DB2_DJ_COMM

- Betriebssystem: Alle
- Standardwert=null, Werte: libdb2drda.a, libdb2net8.a, libdb2informix.a db2drda.dll, db2net8.dll, db2informix.a usw.
- Gibt die Wrapper-Bibliotheken an, die beim Start des Datenbankmanagers geladen werden. Durch Angeben dieser Variablen wird der Laufzeitaufwand für das Laden häufig benutzter Wrapper-Bibliotheken verringert. Andere Werte für andere Betriebssysteme werden unterstützt (die Erweiterung .dll ist für das Betriebssystem Windows, die Erweiterung .a für das Betriebssystem AIX). Die Bibliotheksnamen sind je nach Protokoll und Betriebssystem verschieden.

Diese Variable wird ignoriert, sofern nicht der Parameter FEDERATED des Datenbankmanagers den Wert YES hat.

DB2_DJ_INI

- Betriebssystem: Alle
- Standardwert:
 - UNIX: db2_instanzverzeichnis/cfg/db2dj.ini
 - Windows: db2_installationsverzeichnis\cfg\db2dj.ini
- Gibt den absoluten Pfadnamen der Konfigurationsdatei für den Zusammenschluss von Datenquellen an, zum Beispiel: db2set DB2_DJ_INI=\$HOME/sql1lib/cfg/my_db2dj.ini. Diese Datei enthält die Einstellungen für Umgebungsvariablen von Datenquellen. Diese Umgebungsvariablen werden vom Informix-Wrapper und von den durch WebSphere Federation Server bereitgestellten Wrappern verwendet.

Das folgende Beispiel zeigt eine Konfigurationsdatei für die Föderation von Datenquellen:

```
INFORMIXDIR=/informix/client_sdk
INFORMIXSERVER=inf93
ORACLE_HOME=/usr/oracle9i
SYBASE=/sybase/V12
SYBASE_OCS=OCS-12_5
```

Für die Datei 'db2dj.ini' gelten die folgenden Einschränkungen:

- Einträge müssen das Format *umgebvarname=wert* haben, wobei umgebvarname der Name der Umgebungsvariablen und wert der entsprechende Wert ist.
- Der Name der Umgebungsvariablen kann maximal 255 Byte lang sein.
- Der Wert der Umgebungsvariablen kann maximal 765 Byte lang sein.

Diese Variable wird ignoriert, sofern nicht der Parameter FEDERATED des Datenbankmanagers den Wert YES hat.

DB2DMNBCKCTL

- Betriebssystem: Windows
- Standardwert=null, Werte: ? oder ein Domänenname
- Wenn Sie den Namen der Domäne kennen, für die der DB2-Server der Backup-Domänen-Controller ist, stellen Sie DB2DMNBCKCTL=DOMÄNENNAME ein. Die Angabe für DOMÄNENNAME muss in Großbuchstaben erfolgen. Sie können DB2 die Domäne

ermitteln lassen, für die die lokale Maschine ein Backup-Domänen-Controller ist, indem Sie DB2DMNBCKCTLR=? einstellen. Wenn die Profilvariable DB2DMNBCKCTLR nicht oder mit leerem Wert definiert ist, führt DB2 die Authentifizierung auf dem primären Domänencontroller aus.

Anmerkung: DB2 verwendet standardmäßig keinen vorhandenen Backup-Domänen-Controller, weil ein Backup-Domänen-Controller die Synchronisation mit dem primären Domänencontroller verlieren und damit ein Sicherheitsrisiko darstellen kann. Es kann zum Verlust der Synchronisation kommen, wenn die Sicherheitsdatenbank des primären Domänencontrollers aktualisiert wird, die Änderungen jedoch nicht an einen Backup-Domänen-Controller weitergegeben werden. Der Grund hierfür könnten Netzwerklatenzzeiten oder ein nicht funktionierender Computersuchdienst sein.

DB2_DOCHOST

- Betriebssystem: Alle
- Standardwert: <http://publib.boulder.ibm.com/infocenter/db2help/> (<http://hostname>; dabei ist *hostname* der gültige Hostname oder die gültige IP-Adresse)
- Gibt den Namen des Hosts an, auf dem die DB2-Informationszentrale installiert ist. Diese Variable kann bei der Installation der DB2-Informationszentrale automatisch definiert werden, wenn die Option zur automatischen Konfiguration im DB2-Installationsassistenten ausgewählt wird.

DB2_DOCPORT

- Betriebssystem: Alle
- Standardwert: NULL, Werte: jede gültige Portnummer
- Gibt die Portnummer an, über die das DB2-Hilfesystem die DB2-Dokumentation bereitstellt. Diese Variable kann bei der Installation der DB2-Informationszentrale automatisch definiert werden, wenn die Option zur automatischen Konfiguration im DB2-Installationsassistenten ausgewählt wird.

DB2_ENABLE_AUTOCONFIG_DEFAULT

- Betriebssystem: Alle
- Standardwert=null, Werte: YES oder NO
- Mit dieser Variable wird gesteuert, ob der Konfigurationsadvisor automatisch bei der Datenbankerstellung ausgeführt wird. Wenn die Variable DB2_ENABLE_AUTOCONFIG_DEFAULT nicht definiert (null) ist, bewirkt dies dasselbe wie die Einstellung der Variablen auf den Wert YES. Das heißt, der Konfigurationsadvisor wird bei der Datenbankerstellung ausgeführt. Sie brauchen die Instanz nach dem Definieren dieser Variablen nicht erneut zu starten. Wenn Sie den Befehl AUTOCONFIGURE oder die Anweisung CREATE DB AUTOCONFIGURE ausführen, überschreiben diese die Einstellung von DB2_ENABLE_AUTOCONFIG_DEFAULT.

DB2_ENABLE_LDAP

- Betriebssystem: Alle
- Standardwert=NO, Werte: YES oder NO

- Gibt an, ob LDAP (Lightweight Directory Access Protocol) verwendet wird. LDAP ist eine Zugriffsmethode auf Verzeichnisservices.

DB2_EXTSECURITY

- Betriebssystem: Windows
- Standardwert=ON, Werte: ON oder OFF
- Verhindert unbefugten Zugriff auf DB2 durch Sperren der DB2-Systemdateien. Zur Vermeidung potenzieller Probleme sollte diese Registrierdatenbankvariable nicht auf OFF gesetzt werden.

DB2_FALLBACK

- Betriebssystem: Windows
- Standardwert=OFF, Werte: ON oder OFF
- Diese Variable erlaubt Ihnen, alle Datenbankverbindungen bei der FALLBACK-Verarbeitung zu trennen. Sie wird in Verbindung mit der Unterstützung für die Funktionsübernahme in der Windows-Umgebung mit Microsoft Cluster Server (MSCS) verwendet. Wenn DB2_FALLBACK nicht oder auf OFF gesetzt ist und eine Datenbankverbindung bei der FALLBACK-Operation besteht, kann die DB2-Ressource nicht offline genommen werden. Das bedeutet, dass die FALLBACK-Verarbeitung fehlschlägt.

DB2_FMP_COMM_HEAPSZ

- Betriebssystem: Windows, UNIX
- Standardwert: 20 MB oder ausreichend Speicherbereich, um 10 abgeschirmte Routinen ausführen zu können (je nachdem, welcher Wert größer ist). Unter AIX ist der Standardwert 256 MB.
- Diese Variable gibt die Größe des Pools (in 4-KB-Seiten) an, der für Aufrufe abgeschirmter Routinen, wie zum Beispiel Aufrufe gespeicherter Prozeduren oder benutzerdefinierter Funktionen, verwendet wird. Der von den einzelnen abgeschirmten Routinen verwendete Speicherbereich entspricht dem zweifachen Wert des Konfigurationsparameters *aslheapsz*. Wenn Sie auf Ihrem System eine große Anzahl an abgeschirmten Routinen ausführen, müssen Sie den Wert dieser Variablen unter Umständen erhöhen. Wenn Sie eine sehr kleine Anzahl an abgeschirmten Routinen ausführen, können Sie den entsprechenden Wert senken.

Wird dieser Wert auf 0 gesetzt, wird keine Gruppe erstellt, so dass keine abgeschirmten Routinen aufgerufen werden können. Dies bedeutet außerdem, dass der Diagnosemonitor und die Funktionalität zur automatischen Datenbankpflege (z. B. automatische Backups, Statistikerfassungen und Reorganisationen) inaktiviert wird, da diese Funktionalität von der Infrastruktur für abgeschirmte Routinen abhängig ist.

DB2_GRP_LOOKUP

- Betriebssystem: Windows
- Standardwert=null, Werte: LOCAL, DOMAIN
- Diese Variable gibt an, welcher Windows-Sicherheitsmechanismus zur Aufzählung der Gruppen verwendet wird, denen ein Benutzer angehört.

DB2_HADR_BUF_SIZE

- Betriebssystem: Alle
- Standardwert=2*LOGBUFSZ

- Diese Variable gibt die Protokollempfangspuffergröße des Bereitschaftssystems in Einheiten von Protokollseiten an. Wenn nicht definiert, verwendet DB2 das Zweifache des Werts des Konfigurationsparameters LOGBUFSZ für die Empfangspuffergröße des Bereitschaftssystems. Diese Variable muss in der Bereitschaftsinstanz definiert werden. Sie wird von der Primärdatenbank ignoriert.

Wenn der HADR-Synchronisationsmodus (Datenbankkonfigurationsparameter HADR_SYNCMODE) auf den Wert ASYNC gesetzt ist, kann ein langsames Bereitschaftssystem im Peerstatus die Sendeoperation auf dem Primärsystem aufhalten und so die Transaktionsverarbeitung in der Primärdatenbank blockieren. In einer Bereitschaftsdatenbank kann ein Protokollempfangspuffer mit einer größeren als der Standardgröße konfiguriert werden, um die Aufnahme einer größeren Menge unverarbeiteter Protokolldaten zu ermöglichen. Dadurch lassen sich kurze Zeiträume auffangen, in denen die Primärdatenbank schneller Protokolldaten generiert, als die Bereitschaftsdatenbank sie entgegennehmen kann, ohne dass die Transaktionsverarbeitung in der Primärdatenbank blockiert wird.

DB2LDAP_BASEDN

- Betriebssystem: Alle
- Standardwert=null, Werte: alle gültigen definierten Basisdomänennamen.
- Wenn diese Variable definiert ist, werden die LDAP-Objekte für DB2 im LDAP-Verzeichnis unter

CN=System CN=IBM CN=DB2

unter dem angegebenen definierten Basisnamen (Basis-DN) gespeichert.

Wenn diese Variable für Microsoft Active Directory Server verwendet wird, stellen Sie sicher, dass die Werte CN=DB2, CN=IBM und CN=System unter diesem definierten Namen (DN) definiert sind.

DB2LDAPCACHE

- Betriebssystem: Alle
- Standardwert=YES, Werte: YES oder NO
- Gibt an, dass der LDAP-Cache aktiviert werden muss. Dieser Cache wird zum Katalogisieren der Datenbank-, Knoten- und DCS-Verzeichnisse auf der lokalen Maschine verwendet.

Führen Sie die folgenden Befehle aus, um sicherzustellen, dass sich in Ihrem Cache die aktuellsten Einträge befinden:

```
REFRESH LDAP DB DIR
REFRESH LDAP NODE DIR
```

Diese Befehle aktualisieren das Datenbank- und das Knotenverzeichnis und entfernen daraus falsche Einträge.

DB2LDAP_CLIENT_PROVIDER

- Betriebssystem: Windows
- Standardwert=Null (wenn verfügbar, wird Microsoft verwendet; andernfalls wird IBM verwendet.) Werte: IBM oder Microsoft
- Bei der Ausführung in einer Windows-Umgebung unterstützt DB2 die Verwendung von Microsoft LDAP-Clients oder IBM LDAP-Clients zum Zugriff auf das LDAP-Verzeichnis. Diese Registrierdatenbankvariable wird dazu verwendet, den von DB2 zu verwendenden LDAP-Client explizit auszuwählen.

Anmerkung: Verwenden Sie zum Anzeigen des aktuellen Werts dieser Registrierdatenbankvariablen den Befehl `db2set`:

```
db2set DB2LDAP_CLIENT_PROVIDER
```

DB2LDAPHOST

- Betriebssystem: Alle
- Standardwert=null, Werte: Beliebiger gültiger Hostname
- Gibt den Hostnamen des Standorts für das LDAP-Verzeichnis an.

DB2LDAP_KEEP_CONNECTION

- Betriebssystem: Alle
- Standardwert= YES, Werte: YES, NO
- Gibt an, ob DB2 die internen LDAP-Verbindungskennungen in den Cache stellt. Wenn diese Variable auf NO gesetzt wird, stellt DB2 die Kennungen für LDAP-Verbindungen zum Verzeichnisserver nicht in den Cache. Dies führt mit einiger Wahrscheinlichkeit zu einer Leistungsbeeinträchtigung. Es kann jedoch wünschenswert sein, die Variable DB2LDAP_KEEP_CONNECTION auf den Wert NO zu setzen, wenn die Anzahl simultan aktiver LDAP-Clientverbindungen zum Verzeichnisserver auf ein Minimum reduziert werden muss.

Setzen Sie diese Variable standardmäßig auf den Wert YES, um die optimale Leistung zu erhalten.

Die Registrierdatenbankvariable DB2LDAP_KEEP_CONNECTION ist nur als Profilregistrierdatenbankvariable der globalen Ebene in LDAP implementiert, so dass sie mit dem Befehl **db2set** unter Angabe der Option `-gl` wie folgt definiert werden muss:

```
db2set -gl DB2LDAP_KEEP_CONNECTION=NO
```

DB2LDAP_SEARCH_SCOPE

- Betriebssystem: Alle
- Standardwert= DOMAIN, Werte: LOCAL, DOMAIN, GLOBAL
- Gibt den Bereich für die Suche nach Informationen an, die sich in Datenbankpartitionen oder Domänen im Lightweight Directory Access Protocol (LDAP) befinden. Der Wert LOCAL inaktiviert das Suchen im LDAP-Verzeichnis. Beim Wert „DOMAIN“ wird das LDAP-Verzeichnis nur nach der aktuellen Verzeichnispartmention durchsucht. Beim Wert „GLOBAL“ wird das LDAP-Verzeichnis in allen Verzeichnispartmentionen durchsucht, bis das Objekt gefunden wird.

DB2_LOAD_COPY_NO_OVERRIDE

- Betriebssystem: Alle
- Standardwert: NONRECOVERABLE, Werte: COPY YES, NONRECOVERABLE
- Diese Variable konvertiert jeden Befehl LOAD COPY NO je nach Wert der Variablen entweder in LOAD COPY YES oder NONRECOVERABLE. Diese Variable gilt für HADR-Primärdatenbanken und für Standarddatenbanken (nicht HADR). Sie wird in einer HADR-Bereitschaftsdatenbank ignoriert. Wenn in einer HADR-Primärdatenbank diese Variable nicht definiert ist, wird LOAD COPY NO in LOAD NONRECOVERABLE konvertiert. Der Wert dieser Variablen gibt entweder eine nicht wiederherstellbare Ladeoperation oder das Kopierziel an, wobei die gleiche Syntax wie bei der Klausel COPY YES zu verwenden ist.

DB2LOADREC

- Betriebssystem: Alle
- Standardwert=null
- Dient zum Überschreiben der Speicherposition der Ladekopie bei einer aktualisierenden Recovery (ROLLFORWARD). Wenn der Benutzer die physische Speicherposition der Ladekopie geändert hat, muss die Variable DB2LOADREC vor dem Ausführen der aktualisierenden Recovery definiert werden.

DB2LOCK_TO_RB

- Betriebssystem: Alle
- Standardwert=null, Werte: STATEMENT
- Definiert, ob durch Sperrenzeitlimits für die gesamte Transaktion oder nur für die aktuelle Anweisung ein Rollback durchgeführt werden soll. Wenn DB2LOCK_TO_RB den Wert STATEMENT hat, wird aufgrund eines Zeitlimits für Sperren nur für die aktuelle Anweisung ein Rollback durchgeführt. Jeder andere Wert für diesen Parameter sorgt dafür, dass für die gesamte Transaktion ein Rollback durchgeführt wird.

DB2_MAP_XML_AS_CLOB_FOR_DLC

- Betriebssystem: Alle
- Standardwert=NO, Werte: YES oder NO
- Die Registrierdatenbankvariable DB2_MAP_XML_AS_CLOB_FOR_DLC bietet die Möglichkeit, das DESCRIBE- und FETCH-Standardverhalten für XML-Werte für Clients (bzw. DRDA-Anwendungsrequester) zu überschreiben, die XML als Datentyp nicht unterstützen. (Diese werden im Allgemeinen als Clients einer *älteren Version* bezeichnet.)

Der Standardwert NO gibt an, dass für Clients einer älteren Version ein DESCRIBE von XML-Werten den Ergebnistyp BLOB(2GB) liefert und ein FETCH von XML-Werten eine implizite XML-Serialisierung in BLOB bewirkt, die eine XML-Deklaration enthält, die eine UTF-8-Codierung angibt.

Wenn die Variable auf den Wert YES gesetzt ist, liefert ein DESCRIBE von XML-Werten den Ergebnistyp CLOB(2GB) und ein FETCH von XML-Werten bewirkt eine implizite XML-Serialisierung in CLOB, die keine XML-Deklaration enthält.

DB2_MAX_LOB_BLOCK_SIZE

- Betriebssystem: Alle
- Standardwert=0 (kein Grenzwert), Werte: 0 bis 21487483647
- Legt die maximale Größe von LOB- oder XML-Daten fest, die in einem Block zurückgegeben werden sollen. Dieser Wert ist kein fixer Maximalwert. Wenn dieser Maximalwert auf dem Server während eines Datenabrufs erreicht wird, beendet der Server das Schreiben der aktuellen Zeile, bevor er eine Antwort für den Befehl, zum Beispiel FETCH, an den Client generiert.

DB2_NEWLOGPATH2

- Betriebssystem: UNIX
- Standardwert=0, Werte: 0 oder 1
- Mit diesem Parameter können Sie angeben, ob über einen sekundären Pfad eine doppelte Protokollierung eingerichtet werden soll. Der sekundäre Pfadname wird generiert, indem eine „2“ an den aktuellen Wert des Datenbankkonfigurationsparameters *logpath* angefügt wird.

DB2NOEXITLIST

- Betriebssystem: Alle
- Standardwert=OFF, Werte: ON oder OFF
- Wenn diese Variable definiert ist, weist sie DB2 an, keine Exitlistenroutine in Anwendungen zu installieren und keine Commits auszuführen. Normalerweise installiert DB2 eine Prozessexitlistenroutine in Anwendungen, und die Exitlistenroutine führt eine COMMIT-Operation aus, wenn die Anwendung normal beendet wird.

Bei Anwendungen, die die DB2-Bibliothek dynamisch laden und aus dem Speicher entfernen, bevor die Anwendung beendet ist, schlägt der Aufruf der Exitlistenroutine fehl, weil die Routine nicht mehr in der Anwendung geladen ist. Wenn Ihre Anwendung so arbeitet, müssen Sie die Variable DB2NOEXITLIST angeben und sicherstellen, dass Ihre Anwendung alle erforderlichen COMMIT-Operationen explizit aufruft.

DB2_NUM_CKPW_DAEMONS

- Betriebssystem: UNIX
- Standardwert=3, Werte: 0 bis 100
- Mit Hilfe der Registrierdatenbankvariablen DB2_NUM_CKPW_DAEMONS können Sie eine konfigurierbare Anzahl von Kennwortprüfdämonen starten. Die Dämonen werden bei Ausführung von 'db2start' erstellt und unter der Berechtigung 'root' ausgeführt. Sie verarbeiten Anforderungen zum Prüfen von Kennwörtern. Eine Erhöhung des Werts von DB2_NUM_CKPW_DAEMONS kann die Zeit verkürzen, die zur Herstellung einer Datenbankverbindung benötigt wird. Dies gilt jedoch nur in Szenarios, in denen viele Verbindungen gleichzeitig hergestellt werden und in denen die Authentifizierung aufwendig ist (z. B. wenn über Network Information Service (NIS) oder NIS+ auf ferne Kennwortdateien zugegriffen wird).

Durch Setzen von DB2_NUM_CKPW_DAEMONS auf den Wert 0 werden die Kennwortprüfdämonen inaktiviert, so dass der DB2-Datenbankmanager jedes Mal ein separates Programm zum Prüfen eines Kennworts starten muss (Funktionsweise wie Version 7).

DB2REMOTEPREG

- Betriebssystem: Windows
- Standardwert=null, Werte: alle gültigen Windows-Maschinennamen
- Definiert den Namen der fernen Maschine, die die Win32-Registrierdatenbankliste von DB2-Instanzprofilen und DB2-Instanzen enthält. Der Wert für DB2REMOTEPREG sollte nur einmal nach der Installation von DB2 definiert und anschließend nicht mehr geändert werden. Verwenden Sie diese Variable mit großer Vorsicht.

DB2_RESOLVE_CALL_CONFLICT

- Betriebssystem: AIX, HP-UX, Solaris, Linux, Windows
- Standardwert=YES, Werte: YES oder NO
- Eliminiert Fehler mit SQLCODE-Wert -746 im Triggerkontext. Beim Absetzen einer Anweisung CALL in einem Trigger wird während der Laufzeit möglicherweise ein SQLCODE-Wert SQL0746 ausgegeben. Durch den Fehler SQL0746 werden Prozeduren, die von einem Trigger aufgerufen werden, am Zugriff auf Tabellen gehindert, die zuvor im Kontext der Aufrufanweisung geändert wurden. Wenn diese Variable gesetzt ist, erzwingt der DB2-Datenbankmanager die Ausführung sämtli-

cher Änderungen an Tabellen entsprechend den SQL-Standardregeln für Trigger, bevor die Anweisung CALL ausgeführt wird.

Sie müssen die Instanz stoppen, bevor Sie DB2_RESOLVE_CALL_CONFLICT neu definieren, und sie anschließend neu starten. Binden Sie anschließend sämtliche Pakete neu, die den Aufruf von Triggern verursachen. SQL-Prozeduren binden Sie wie folgt erneut: `CALL SYSPROC.REBIND_ROUTINE_PACKAGE ('P','prozedur-schema.prozedurname','CONSERVATIVE');`

Beachten Sie, dass sich DB2_RESOLVE_CALL_CONFLICT auf die Leistung auswirken kann. Wenn DB2_RESOLVE_CALL_CONFLICT auf YES gesetzt wird, löst der DB2-Datenbankmanager alle potenziellen Lese- und Schreibkonflikte durch Einschaltung temporärer Tabellen (nach Bedarf). In der Regel ist die Wirkung gering, da maximal eine temporäre Tabelle eingefügt wird. Die Wirkung in einer OLTP-Umgebung ist gering, da nur eine Zeile bzw. eine kleine Anzahl von Zeilen durch die Triggeranweisung geändert wird. In der Regel ist der Einfluss auf die Leistung durch DB2_RESOLVE_CALL_CONFLICT gering, wenn die allgemeinen Empfehlungen zur Verwendung von SMS (systemverwalteter Speicher, System Managed Storage) für temporäre Tabellenbereiche befolgt werden.

DB2ROUTINE_DEBUG

- Betriebssystem: AIX und Windows
- Standardwert=OFF, Werte: ON, OFF
- Gibt an, ob die Fehlerbehebungsfunktion (Debug) für gespeicherte Java-Prozeduren aktiviert wird. Wenn Sie nicht gerade Fehler in gespeicherten Java-Prozeduren beheben, sollten Sie den Standardwert OFF verwenden. Die Aktivierung des Fehlerbehebungsmodus hat Auswirkungen auf die Leistung.

DB2SATELLITEID

- Betriebssystem: Alle
- Standardwert=null, Werte: eine gültige Satelliten-ID, die in der Satellitensteuerungsdatenbank deklariert ist
- Gibt die Satelliten-ID an, die an den Satellitensteuerungsserver übergeben wird, wenn ein Satellit eine Synchronisation durchführt. Wenn für diese Variable kein Wert angegeben ist, wird die Anmelde-ID als Satelliten-ID verwendet.

DB2_SERVER_CONTIMEOUT

- Betriebssystem: Alle
- Standardwert=180, Werte: 1 - 32767 Sekunden
- Diese Registrierdatenbankvariable und die Registrierdatenbankvariable DB2_DISPATCHER_PEEKTIMEOUT konfigurieren beide die Behandlung eines neuen Clients während der Verbindungszeit. Mit DB2_SERVER_CONTIMEOUT können Sie die Zeit (in Sekunden) anpassen, die ein Agent auf die Verbindungsanforderung eines Clients wartet, bevor er die Verbindung beendet. In den meisten Fällen sollte es nicht nötig sein, diese Registrierdatenbankvariable anzupassen. Wenn DB2-Clients beim Verbindungsaufbau jedoch beständig das vom Server definierte Zeitlimit überschreiten, können Sie einen höheren Wert für DB2_SERVER_CONTIMEOUT angeben, um das Zeitlimitintervall zu verlängern. Wenn ein ungültiger Wert definiert wird, wird der Standardwert verwendet. Diese Registrierdatenbankvariable ist nicht dynamisch.

DB2SORT

- Betriebssystem: Alle, nur Server
- Standardwert=null
- Definiert die Speicherposition einer während der Laufzeit durch das Dienstprogramm LOAD zu ladenden Bibliothek. Die Bibliothek enthält den Eingangspunkt für Funktionen, die beim Sortieren von Indexdaten verwendet werden. Verwenden Sie DB2SORT, um Sortierprogrammprodukte anderer Lieferanten mit dem Dienstprogramm LOAD zur Generierung von Tabellenindizes zu nutzen. Der angegebene Pfad muss relativ zum Datenbankserver definiert werden.

DB2_TRUNCATE_REUSESTORAGE

- Betriebssystem: Alle
- Standardwert=null (nicht gesetzt), Werte: IMPORT, import
- Mit Hilfe dieser Variablen können Sie Sperrenkonflikte zwischen dem Befehl IMPORT with REPLACE und dem Befehl BACKUP ... ONLINE auflösen. In einigen Fällen können ein Online-Backup und Abschneideoperationen (TRUNCATE) nicht gleichzeitig ausgeführt werden. Wenn dieser Fall eintritt, können Sie die Variable DB2_TRUNCATE_REUSESTORAGE auf den Wert "IMPORT" bzw. "import" setzen. Dadurch wird das physische Abschneiden des Objekts, einschließlich Daten, Indizes, Langfelddaten, große Objekte (LOBs) und Blockzuordnungen (für MDC-Tabellen) übersprungen und nur ein logisches Abschneiden ausgeführt. Das heißt, der Befehl IMPORT with REPLACE leert die Tabelle, so dass sich die logische Größe des Objekts verringert, der Plattenspeicherplatz jedoch zugeordnet bleibt.

Diese Registrierdatenbankvariable ist dynamisch. Sie können sie definieren bzw. ihren Wert löschen, ohne die Instanz stoppen und starten zu müssen. Sie können die Variable DB2_TRUNCATE_REUSESTORAGE vor dem Start eines Online-Backups definieren und nach Abschluss des Online-Backups ihren Wert wieder löschen. In Umgebungen mit mehreren Datenbankpartitionen ist die Registrierdatenbankvariable nur auf den Knoten aktiv, auf denen sie definiert ist. Die Variable DB2_TRUNCATE_REUSESTORAGE ist nur auf permanenten DMS-Objekten wirksam.

Wenn in SAP-Umgebungen DB2_WORKLOAD=SAP definiert ist, hat diese Registrierdatenbankvariable den Standardwert IMPORT.

DB2_USE_DB2JCCT2_JROUTINE

- Betriebssystem: Alle
- Standardwert=null (nicht gesetzt), Werte: OFF/NO/0/FALSE oder ON/YES/1/TRUE
- Wenn DB2_USE_DB2JCCT2_JROUTINE auf ON, YES, 1 oder TRUE gesetzt wird, wird der IBM DB2-Treiber für JDBC und SQLJ für SQL-Anforderungen für Java-Routinen verwendet. Andernfalls wird der DB2 JDBC-Treiber vom Typ 2 für Linux, UNIX und Windows verwendet.

DB2_UTIL_MSGPATH

- Betriebssystem: Alle
- Standardwert=Verzeichnis *instanzname/tmp*
- Die Registrierdatenbankvariable DB2_UTIL_MSGPATH wird in Verbindung mit der Prozedur SYSPROC.ADMIN_CMD, der Prozedur SYSPROC.ADMIN_REMOVE_MSGS und der benutzerdefinierten Funktion (UDF) SYSPROC.ADMIN_GET_MSGS verwendet. Sie gilt für die Instanzebene. Die Variable DB2_UTIL_MSGPATH kann definiert werden,

um einen Verzeichnispfad auf dem Server anzugeben, in dem die abgeschirmte Benutzer-ID Dateien lesen, schreiben und löschen kann. Dieses Verzeichnis muss von allen Koordinatorpartitionen aus zugänglich sein und genügend Speicherplatz besitzen, um die Nachrichtendateien von Dienstprogrammen aufzunehmen.

Wenn dieser Pfad nicht festgelegt ist, wird das Verzeichnis *instanzname/tmp* als Standard verwendet. (Beachten Sie, dass das Verzeichnis *instanzname/tmp* bei der Deinstallation von DB2 bereinigt wird.)

Wenn dieser Pfad geändert wird, werden die Dateien, die sich in dem durch die vorige Einstellung der Variablen bezeichneten Verzeichnis befanden, nicht automatisch versetzt oder gelöscht. Wenn Sie den Inhalt der Nachrichten, die unter dem alten Pfad erstellt wurden, abrufen wollen, müssen Sie diese Nachrichten (die den Dienstprogrammnamen als Präfix und die Benutzer-ID als Suffix aufweisen) manuell in das neue Verzeichnis versetzen, das durch DB2_UTIL_MSGPATH angegeben wird. Die nächste Nachrichtendatei eines Dienstprogramms wird an der neuen Position erstellt, gelesen und bereinigt.

Die Dateien unter dem DB2_UTIL_MSGPATH-Verzeichnis sind für die jeweiligen Dienstprogramme spezifisch und nicht transaktionsabhängig. Sie sind kein Teil des Backup-Images. Die Dateien unter dem DB2_UTIL_MSGPATH-Verzeichnis werden vom Benutzer verwaltet. Dies bedeutet, dass ein Benutzer die Nachrichtendateien mit Hilfe der Prozedur SYSPROC.ADMIN_REMOVE_UTILMSG löschen kann. Diese Dateien werden bei der Deinstallation von DB2 nicht bereinigt.

DB2_VENDOR_INI

- Betriebssystem: AIX, HP-UX, Solaris und Windows
- Standardwert=null, Werte: eine beliebige gültige Angabe für Pfad und Datei
- Verweist auf eine Datei, die alle herstellersistenspezifischen Umgebungseinstellungen enthält. Der Wert wird beim Start des Datenbankmanagers gelesen.

DB2_XBSA_LIBRARY

- Betriebssystem: AIX, HP-UX, Solaris und Windows
- Standardwert=null, Werte: eine beliebige gültige Angabe für Pfad und Datei
- Diese Registrierdatenbankvariable verweist auf die werkseitige XBSA-Bibliothek. Unter AIX muss die Einstellung das gemeinsam benutzte Objekt enthalten, sofern es nicht 'shr.o' heißt. Für HP-UX, Solaris und Windows wird der Name des gemeinsam benutzten Objekts nicht benötigt. Wenn z. B. das NetWorker Business Suite Module für DB2 von Legato verwendet werden soll, muss diese Registrierdatenbankvariable wie folgt eingestellt werden:

```
db2set DB2_XBSA_LIBRARY="/usr/lib/libxdb2.a(bsashr10.o)"
```

Die XBSA-Schnittstelle kann mit dem Befehl BACKUP DATABASE oder RESTORE DATABASE aufgerufen werden. Zum Beispiel:

```
db2 backup db sample use XBSA
db2 restore db sample use XBSA
```

Zugehörige Konzepte:

- „Registrierdatenbank- und Umgebungsvariablen von DB2“ auf Seite 601
- „Verbindungskonzentrator“ in *DB2 Connect Benutzerhandbuch*

Anhang C. EXPLAIN-Tabellen

EXPLAIN-Tabellen

Die EXPLAIN-Tabellen erfassen Zugriffspläne, wenn das Programm EXPLAIN aktiviert ist. Die EXPLAIN-Tabellen müssen erstellt werden, bevor EXPLAIN aufgerufen werden kann. Sie können sie mit Hilfe der dokumentierten Tabellendefinitionen oder durch Aufrufen des Beispielscripts für den Befehlszeilenprozessor erstellen, das in der Datei EXPLAIN.DDL bereitgestellt wird, die sich im Unterverzeichnis 'misc' des Verzeichnisses 'sqllib' befindet. Zum Aufrufen des Scripts stellen Sie eine Verbindung zu der Datenbank her, in der die EXPLAIN-Tabellen benötigt werden, und geben den folgenden Befehl ein:

```
db2 -tf EXPLAIN.DDL
```

Die EXPLAIN-Einrichtung verwendet die folgenden IDs als Schema, wenn sie EXPLAIN-Tabellen, die sie mit Daten füllt, mit einem Qualifikationsmerkmal versieht:

- Die Sitzungsberechtigungs-ID für dynamisches SQL
- Die Anweisungsberechtigungs-ID für statisches SQL

Das Schema kann einer Gruppe von EXPLAIN-Tabellen oder Aliasnamen zugeordnet werden, die auf eine Gruppe von EXPLAIN-Tabellen unter einem anderen Schema verweisen. Wenn keine EXPLAIN-Tabellen unter dem Schema gefunden werden, sucht die EXPLAIN-Einrichtung nach EXPLAIN-Tabellen unter dem Schema SYSTOOLS und versucht, die gefundenen Tabellen zu verwenden.

Das Füllen der EXPLAIN-Tabellen mit Werten durch die EXPLAIN-Einrichtung aktiviert weder Trigger noch referenzielle Integritätsbedingungen oder Prüfungen auf Integritätsbedingungen. Wenn zum Beispiel ein Einfügetrigger für die Tabelle EXPLAIN_INSTANCE definiert wäre und EXPLAIN-Informationen über eine auswählbare Anweisung erfasst würden, würde der Trigger nicht aktiviert.

Zur Verbesserung der Leistung der EXPLAIN-Einrichtung in einem partitionierten Datenbanksystem wird empfohlen, die EXPLAIN-Tabellen in einer Datenbank-partitionsgruppe mit nur einer Datenbankpartition zu erstellen, und zwar vorzugsweise in derselben Datenbankpartition, zu der auch die Verbindung hergestellt wird, wenn die Abfrage kompiliert wird.

Zugehörige Referenzen:

- „Die Tabelle ADVISE_INDEX“ auf Seite 688
- „Die Tabelle ADVISE_INSTANCE“ auf Seite 692
- „Die Tabelle ADVISE_MQT“ auf Seite 693
- „Die Tabelle ADVISE_PARTITION“ auf Seite 695
- „Die Tabelle ADVISE_TABLE“ auf Seite 697
- „Die Tabelle ADVISE_WORKLOAD“ auf Seite 698
- „Die Tabelle EXPLAIN_ARGUMENT“ auf Seite 665
- „Die Tabelle EXPLAIN_INSTANCE“ auf Seite 673
- „Die Tabelle EXPLAIN_OBJECT“ auf Seite 676
- „Die Tabelle EXPLAIN_OPERATOR“ auf Seite 679
- „Die Tabelle EXPLAIN_PREDICATE“ auf Seite 681

EXPLAIN-Tabellen

- „Die Tabelle EXPLAIN_STATEMENT“ auf Seite 684
- „Die Tabelle EXPLAIN_STREAM“ auf Seite 686

Die Tabelle EXPLAIN_ARGUMENT

Die Tabelle EXPLAIN_ARGUMENT enthält gegebenenfalls vorhandene, eindeutige Kenndaten für jeden einzelnen Operator.

Tabelle 78. Die Tabelle EXPLAIN_ARGUMENT. PK bedeutet, dass die Spalte Teil eines Primärschlüssels ist. FK bedeutet, dass die Spalte Teil eines Fremdschlüssels ist.

Spaltenname	Datentyp	Nullwert?	Schlüssel?	Beschreibung
EXPLAIN_REQUESTER	VARCHAR(128)	Nein	FK	Berechtigungs-ID des Initiators dieser EXPLAIN-Anforderung
EXPLAIN_TIME	TIMESTAMP	Nein	FK	Initialisierungszeitpunkt für die EXPLAIN-Anforderung
SOURCE_NAME	VARCHAR(128)	Nein	FK	Name des Pakets, das ausgeführt wurde, als die dynamische Anweisung mit EXPLAIN bearbeitet wurde, oder der Name der Quellendatei, als das statische SQL mit EXPLAIN bearbeitet wurde.
SOURCE_SCHEMA	VARCHAR(128)	Nein	FK	Schema oder Qualifikationsmerkmal der Quelle der EXPLAIN-Anforderung
SOURCE_VERSION	VARCHAR(64)	Nein	FK	Version der Quelle der EXPLAIN-Anforderung
EXPLAIN_LEVEL	CHAR(1)	Nein	FK	Ebene der EXPLAIN-Informationen, für die diese Zeile relevant ist
STMTNO	INTEGER	Nein	FK	Anweisungsnummer im Paket, auf die sich diese EXPLAIN-Informationen beziehen
SECTNO	INTEGER	Nein	FK	Abschnittsnummer im Paket, auf die sich diese EXPLAIN-Informationen beziehen
OPERATOR_ID	INTEGER	Nein	Nein	Eindeutige ID für diesen Operator in dieser Abfrage
ARGUMENT_TYPE	CHAR(8)	Nein	Nein	Der Argumenttyp für diesen Operator
ARGUMENT_VALUE	VARCHAR(1024)	Ja	Nein	Der Argumentwert für diesen Operator. NULL, wenn sich der Wert in LONG_ARGUMENT_VALUE befindet.
LONG_ARGUMENT_VALUE	CLOB(2M)	Ja	Nein	Der Argumentwert für diesen Operator, wenn der Text nicht in ARGUMENT_VALUE passt. NULL, wenn sich der Wert in ARGUMENT_VALUE befindet.

Tabelle 79. Werte der Spalten ARGUMENT_TYPE und ARGUMENT_VALUE

Wert für ARGUMENT_TYPE	Gültige Werte für ARGUMENT_VALUE	Beschreibung
AGGMODE	COMPLETE PARTIAL INTERMEDIATE FINAL	Partielle Bezugswerte für Spaltenberechnung
BITFLTR	INTEGER FALSE	Größe des von einem Hash-Join verwendeten Bit-filters
BLD_LEVEL	DB2-Build-Kennung	Interne ID-Zeichenfolge für die Quellcodeversion
BLKLOCK	EXCLUSIVE INTENT EXCLUSIVE INTENT SHARE NONE SHARE UPDATE	Vorgesehene Sperre auf Blockebene
CSERQY	TRUE FALSE	Ferne Abfrage ist ein allgemeiner Unterausdruck (common subexpression).
CSETEMP	TRUE FALSE	Markierung für temporäre Tabelle über allgemeinen Unterausdruck
DIRECT	TRUE	Anzeiger für direkten Abruf

Tabelle EXPLAIN_ARGUMENT

Tabelle 79. Werte der Spalten ARGUMENT_TYPE und ARGUMENT_VALUE (Forts.)

Wert für ARGUMENT- _TYPE	Gültige Werte für ARGUMENT_VALUE	Beschreibung
DPESTFLG	TRUE FALSE	Gibt an, ob der DPNUMPRT-Wert auf einem Schätzwert beruht oder nicht. Mögliche Werte sind 'TRUE' (DPNUMPRT stellt eine geschätzte Anzahl von Datenpartitionen dar, auf die zugegriffen wird) oder 'FALSE' (DPNUMPRT stellt die tatsächliche Anzahl von Datenpartitionen dar, auf die zugegriffen wird).
DPLSTPRT	NONE CHARACTER	Stellt die Datenpartitionen dar, auf die zugegriffen wird. Besteht aus einer Liste, deren Einträge durch Komma (z. B: 1,3,5) oder Bindestrich (z. B.: 1-5) getrennt sind. Der Wert 'NONE' bedeutet, dass keine Datenpartition verbleibt, nachdem die angegebenen Vergleichselemente angewendet wurden.
DPNUMPRT	INTEGER	Stellt die tatsächliche oder geschätzte Anzahl von Datenpartitionen dar, auf die zugegriffen wird.
DSTSEVER	Servename	Zielservers (Absendeserver)
DUPLWARN	TRUE FALSE	Markierung für Warnung, dass gleiche Werte auftraten
EARLYOUT	LEFT RIGHT NONE	Anzeiger für frühes Verlassen (Early out)
ENVVAR	Jede Zeile dieses Typs enthält die folgenden Angaben: - Name der Umgebungsvariable - Wert der Umgebungsvariable	Sich auf das Optimierungsprogramm auswirkende Umgebungsvariable
ERRTOL	Jede Zeile dieses Typs enthält ein Paar aus SQLSTATE-Wert und SQLCODE-Wert.	Eine Liste von zu tolerierenden Fehlern
FETCHMAX	IGNORE INTEGER	Außerkräftsetzen des Werts für das Argument MAXPAGES im Operator FETCH
GREEDY	TRUE	Gibt an, dass das Optimierungsprogramm den GREEDY-Algorithmus zur Zugriffsplanung verwendete.
GLOBLOCK	EXCLUSIVE INTENT EXCLUSIVE INTENT NONE INTENT SHARE NO LOCK OBTAINED SHARE SHARE INTENT EXCLUSIVE SUPER EXCLUSIVE UPDATE	Stellt Informationen zu vorgesehenen globalen Sperren für ein partitioniertes Tabellenobjekt dar.
GROUPBYC	TRUE FALSE	Gibt an, ob Spalten mit der Klausel Group By angegeben wurden.
GROUPBYN	Ganze Zahl	Anzahl der Vergleichsspalten
GROUPBYR	Jede Zeile dieses Typs enthält die folgenden Angaben: - Ordinalzahl der Spalte in der Klausel Group By (gefolgt von einem Doppelpunkt und einem Leerzeichen) - Name der Spalte	Anforderung der Klausel Group By
HASHCODE	24 32	Größe (in Bit) des Hash-Codes, der für einen Hash-Join verwendet wird

Tabelle 79. Werte der Spalten ARGUMENT_TYPE und ARGUMENT_VALUE (Forts.)

Wert für ARGUMENT- _TYPE	Gültige Werte für ARGUMENT_VALUE	Beschreibung
INNERCOL	Jede Zeile dieses Typs hat folgenden Inhalt: - Ordinalzahl der Spalte in der Reihenfolge (gefolgt von einem Doppelpunkt und einem Leerzeichen) - Name der Spalte - Sortierungswert (A) Aufsteigend (D) Absteigend	Innere Sortierspalten
INPUTXID	Eine Kontextknotenkennung (ID)	INPUTXID gibt den Eingabekontextknoten an, der vom Operator XSCAN verwendet wird.
ISCANMAX	IGNORE INTEGER	Außerkräftsetzen des Werts für das Argument MAXPAGES im Operator ISCAN
JN_INPUT	INNER OUTER	Gibt an, ob Operator der Operator ist, der INNER oder OUTER für eine JOIN-Operation bereitstellt.
LISTENER	TRUE FALSE	Anzeiger für empfangsbereite Tabellenwarteschlange (Listener)
MAXPAGES	ALL NONE INTEGER	Maximale Anzahl der erwarteten Seiten für Vorab- lesezugriff
MAXRIDS	NONE INTEGER	Die maximale Anzahl der Zeilen-IDs muss in jeder Anforderung für Vorablesezugriff über Listen enthalten sein.
NUMROWS	INTEGER	Anzahl der erwarteten, zu sortierenden Zeilen
ONEFETCH	TRUE FALSE	Anzeiger für eine Abrufoperation (FETCH)
OUTERCOL	Jede Zeile dieses Typs hat folgenden Inhalt: - Ordinalzahl der Spalte in der Reihenfolge (gefolgt von einem Doppelpunkt und einem Leerzeichen) - Name der Spalte - Sortierungswert (A) Aufsteigend (D) Absteigend	Äußere Ordnungsspalten
OUTERJN	LEFT RIGHT FULL LEFT (ANTI) RIGHT (ANTI)	Anzeiger für Outer Join.
PARTCOLS	Name der Spalte	Partitionierungsspalten für den Operator
PREFETCH	LIST NONE SEQUENTIAL	Art des verfügbaren Vorablesezugriffs
REOPT	ALWAYS ONCE	Die Anweisung wird unter Verwendung von Einbindewerten für Parametermarken, Hostvariablen und Sonderregistern optimiert.
RMTQTEXT	Abfragetext	Ferner Abfragetext
RNG_PROD	Funktionsname	Wertebereich erzeugende Funktion für erweiterten Indexzugriff

Tabelle EXPLAIN_ARGUMENT

Tabelle 79. Werte der Spalten ARGUMENT_TYPE und ARGUMENT_VALUE (Forts.)

Wert für ARGUMENT- _TYPE	Gültige Werte für ARGUMENT_VALUE	Beschreibung
ROWLOCK	EXCLUSIVE NONE REUSE SHARE SHORT (INSTANT) SHARE UPDATE	Vorgesehene Zeilensperre (Row Lock Intent)
ROWWIDTH	INTEGER	Länge der zu sortierenden Zeile
RSUFFIX	Abfragetext	Suffix für fernes SQL
SCANDIR	FORWARD REVERSE	Suchrichtung
SCANGRAN	INTEGER	Partitionsinterne Parallelität, Granularität der partitionsinternen Parallelsuche, ausgedrückt in SCANUNITs
SCANTYPE	LOCAL PARALLEL	Partitionsinterne Parallelität, Index- oder Tabellen-suche
SCANUNIT	ROW PAGE	Partitionsinterne Parallelität, Einheit für Suchgranularität
SHARED	TRUE	Partitionsinterne Parallelität, Anzeiger für gemeinsam benutztes TEMP
SLOWMAT	TRUE FALSE	Markierung für langsame Materialisierung (Slow Materialization)
SNGLPROD	TRUE FALSE	Sortierung durch partitionsinterne Parallelität oder Tabelle TEMP von einem Agenten erstellt.
SORTKEY	Jede Zeile dieses Typs hat folgenden Inhalt: - Ordinalzahl der Spalte im Schlüssel (gefolgt von einem Doppelpunkt und einem Leerzeichen) - Name der Spalte - Sortierungswert (A) Aufsteigend (D) Absteigend	Sortierschlüsselspalten
SORTTYPE	PARTITIONED SHARED ROUND ROBIN REPLICATED	Partitionsinterne Parallelität, Sortiertyp
SRCSEVER	Servername	Quellenserver (Empfangsserver)
SPILED	INTEGER	Geschätzte Anzahl von Seiten im Sortierungsüberlauf
SQLCA	Informationen zur Warnung	Warnungen und Ursachencodes, die während der EXPLAIN-Operation ausgegeben wurden
STMTHEAP	INTEGER	Größe des Anweisungszwischenspeichers beim Start der Anweisungskompilierung
STREAM	TRUE FALSE	Ferne Quelle im Datenstrommodus (STREAMING)
TABLOCK	EXCLUSIVE INTENT EXCLUSIVE INTENT NONE INTENT SHARE REUSE SHARE SHARE INTENT EXCLUSIVE SUPER EXCLUSIVE UPDATE	Vorgesehene Tabellensperre (Table Lock Intent)
TEMPSIZE	INTEGER	Seitengröße der temporären Tabelle

Tabelle 79. Werte der Spalten ARGUMENT_TYPE und ARGUMENT_VALUE (Forts.)

Wert für ARGUMENT- _TYPE	Gültige Werte für ARGUMENT_VALUE	Beschreibung
TQDEGREE	INTEGER	Partitionsinterne Parallelität, Anzahl von Sub-agenten, die auf die Tabellenwarteschlange zugreifen.
TQMERGE	TRUE FALSE	Angabe für (sortiertes) Mischen von Tabellenwarteschlangen
TQREAD	READ AHEAD STEPPING SUBQUERY STEPPING	Lesemerkmale für Tabellenwarteschlange
TQSEND	BROADCAST DIRECTED SCATTER SUBQUERY DIRECTED	Sendemerkmale für Tabellenwarteschlange
TQTYPE	LOCAL	Partitionsinterne Parallelität, Tabellenwarteschlange
TQ_ORIGIN	ASYNCHRONY	Die Grund, aus dem die Tabellenwarteschlange in den Zugriffsplan eingeführt wurde
TRUNCSTRT	TRUE	Einschränkendes Sortieren (beschränkt die Anzahl erzeugter Zeilen)
UNIQUE	TRUE FALSE	Anzeiger für Eindeutigkeit
UNIQUEKEY	Jede Zeile dieses Typs enthält die folgenden Angaben: - Ordinalzahl der Spalte im Schlüssel (gefolgt von einem Doppelpunkt und einem Leerzeichen) - Name der Spalte	Eindeutige Spalten
VOLATILE	TRUE	Flüchtige Tabelle
XDFOUT	DECIMAL	XDFOUT gibt die erwartete Anzahl von Dokumenten an, die vom Operator XISCAN für jeden Kontextknoten zuzurückzugeben sind.
XLOGID	Eine Kennung, die aus einem SQL-Schemanamen und dem Namen eines Index für XML-Daten besteht	XLOGID gibt den Index für XML-Daten an, der vom Optimierungsprogramm für den Operator XISCAN ausgewählt wurde.
XPATH	Ein XPATH-Ausdruck und eine Ergebnismenge in einem internen Format	Dieses Argument gibt die Auswertung eines XPATH-Ausdrucks durch den Operator XSCAN an.
XPHYID	Eine Kennung, die aus einem SQL-Schemanamen und dem Namen eines physischen Index für XML-Daten besteht	XPHYID gibt den physischen Index an, der einem vom Operator XISCAN verwendeten Index für XML-Daten zugeordnet ist.

Tabelle EXPLAIN_DIAGNOSTIC

Die Tabelle EXPLAIN_DIAGNOSTIC enthält einen Eintrag für jede einzelne Diagnosenachricht, die für eine bestimmte Instanz einer EXPLAIN-Anweisung in der Tabelle EXPLAIN_STATEMENT erstellt wird.

Die Tabellenfunktion EXPLAIN_GET_MSGS fragt die EXPLAIN-Tabellen EXPLAIN_DIAGNOSTIC und EXPLAIN_DIAGNOSTIC_DATA ab und gibt formatierte Nachrichten zurück.

Tabelle 80. Tabelle EXPLAIN_DIAGNOSTIC. PK bedeutet, dass die Spalte Teil eines Primärschlüssels ist. FK bedeutet, dass die Spalte Teil eines Fremdschlüssels ist.

Spaltenname	Datentyp	Dateneingabe optional?	Schlüssel?	Beschreibung
EXPLAIN_REQUESTER	VARCHAR(128)	Nein	PK, FK	Berechtigungs-ID des Initiators dieser EXPLAIN-Anforderung.
EXPLAIN_TIME	TIMESTAMP	Nein	PK, FK	Initialisierungszeitpunkt für die EXPLAIN-Anforderung.
SOURCE_NAME	VARCHAR(128)	Nein	PK, FK	Name des Pakets, das ausgeführt wurde, als die dynamische Anweisung mit EXPLAIN bearbeitet wurde, oder der Name der Quelldatei, als das statische SQL mit EXPLAIN bearbeitet wurde.
SOURCE_SCHEMA	VARCHAR(128)	Nein	PK, FK	Schema oder Qualifikationsmerkmal der Quelle der EXPLAIN-Anforderung.
SOURCE_VERSION	VARCHAR(64)	Nein	PK, FK	Version der Quelle der EXPLAIN-Anforderung.
EXPLAIN_LEVEL	CHAR(1)	Nein	PK, FK	Ebene der EXPLAIN-Informationen, für die diese Zeile relevant ist Gültige Werte: O Ursprünglicher Text (wie vom Benutzer eingegeben) P PLAN AUSWAHL
STMTNO	INTEGER	Nein	PK, FK	Anweisungsnummer im Paket, auf die sich diese EXPLAIN-Informationen beziehen. Bei dynamischen EXPLAIN-SQL-Anweisungen wird dieser Wert auf 1 gesetzt. Bei statischen SQL-Anweisungen stimmt dieser Wert mit dem Wert überein, der für die Systemkatalogsicht SYSCAT.STATEMENTS verwendet wird.
SECTNO	INTEGER	Nein	PK, FK	Abschnittsnummer in dem Paket, die diese SQL-Anweisung enthält. Bei Dynamic Explain-SQL-Anweisungen wird hier die Abschnittsnummer angegeben, die zur Speicherung des Abschnitts für diese Anweisung während der Laufzeit verwendet wird. Bei statischen SQL-Anweisungen stimmt dieser Wert mit dem Wert überein, der für die Systemkatalogsicht SYSCAT.STATEMENTS verwendet wird.
DIAGNOSTIC_ID	INTEGER	Nein	PK	Diagnose-ID für eine bestimmte Instanz einer Anweisung in der Tabelle EXPLAIN_STATEMENT.
CODE	INTEGER	Nein	Nein	Ein eindeutiger numerischer Wert, der allen Diagnosenachrichten zugeordnet wird. Dieser Wert kann von einer Nachrichten-API zum Abrufen des vollständigen Textes der Diagnosenachricht verwendet werden.

Zugehörige Referenzen:

- „Tabelle EXPLAIN_DIAGNOSTIC_DATA“ auf Seite 671
- „EXPLAIN_GET_MSGS table function“ in *Administrative SQL Routines and Views*
- „Die Tabelle EXPLAIN_STATEMENT“ auf Seite 684
- „SYSCAT.STATEMENTS catalog view“ in *SQL Reference, Volume 1*

Tabelle EXPLAIN_DIAGNOSTIC_DATA

Die Tabelle EXPLAIN_DIAGNOSTIC_DATA enthält Nachrichtentoken für bestimmte Diagnosenachrichten, die in der Tabelle EXPLAIN_DIAGNOSTIC aufgezeichnet werden. Die Nachrichtentoken enthalten weitere Informationen, die sich auf die Ausführung der durch die Nachricht generierten SQL-Anweisung beziehen.

Die Tabellenfunktion EXPLAIN_GET_MSGS fragt die EXPLAIN-Tabellen EXPLAIN_DIAGNOSTIC und EXPLAIN_DIAGNOSTIC_DATA ab und gibt formatierte Nachrichten zurück.

Tabelle 81. Tabelle EXPLAIN_DIAGNOSTIC_DATA. PK bedeutet, dass die Spalte Teil eines Primärschlüssels ist. FK bedeutet, dass die Spalte Teil eines Fremdschlüssels ist.

Spaltenname	Datentyp	Dateneingabe optional?	Schlüssel?	Beschreibung
EXPLAIN_REQUESTER	VARCHAR(128)	Nein	FK	Berechtigungs-ID des Initiators dieser EXPLAIN-Anforderung.
EXPLAIN_TIME	TIMESTAMP	Nein	FK	Initialisierungszeitpunkt für die EXPLAIN-Anforderung.
SOURCE_NAME	VARCHAR(128)	Nein	FK	Name des Pakets, das ausgeführt wurde, als die dynamische Anweisung mit EXPLAIN bearbeitet wurde, oder der Name der Quelldatei, als das statische SQL mit EXPLAIN bearbeitet wurde.
SOURCE_SCHEMA	VARCHAR(128)	Nein	FK	Schema oder Qualifikationsmerkmal der Quelle der EXPLAIN-Anforderung.
SOURCE_VERSION	VARCHAR(64)	Nein	FK	Version der Quelle der EXPLAIN-Anforderung.
EXPLAIN_LEVEL	CHAR(1)	Nein	FK	Ebene der EXPLAIN-Informationen, für die diese Zeile relevant ist Gültige Werte: O Ursprünglicher Text (wie vom Benutzer eingegeben) P PLANAUSSWAHL
STMTNO	INTEGER	Nein	FK	Anweisungsnummer im Paket, auf die sich diese EXPLAIN-Informationen beziehen. Bei dynamischen EXPLAIN-SQL-Anweisungen wird dieser Wert auf 1 gesetzt. Bei statischen SQL-Anweisungen stimmt dieser Wert mit dem Wert überein, der für die Systemkatalogsicht SYSCAT.STATEMENTS verwendet wird.
SECTNO	INTEGER	Nein	FK	Abschnittsnummer in dem Paket, die diese SQL-Anweisung enthält. Bei Dynamic Explain-SQL-Anweisungen wird hier die Abschnittsnummer angegeben, die zur Speicherung des Abschnitts für diese Anweisung während der Laufzeit verwendet wird. Bei statischen SQL-Anweisungen stimmt dieser Wert mit dem Wert überein, der für die Systemkatalogsicht SYSCAT.STATEMENTS verwendet wird.
DIAGNOSTIC_ID	INTEGER	Nein	PK	Diagnose-ID für eine bestimmte Instanz einer Anweisung in der Tabelle EXPLAIN_STATEMENT.
ORDINAL	INTEGER	Nein	Nein	Position des Tokens im vollständigen Nachrichtentext.
TOKEN	VARCHAR(1000)	Ja	Nein	In den vollständigen Nachrichtentext einzufügendes Token; eventuell abgeschnitten.
TOKEN_LONG	BLOB(3M)	Ja	Nein	Genauere Informationen, sofern verfügbar.

Zugehörige Referenzen:

- „Tabelle EXPLAIN_DIAGNOSTIC“ auf Seite 670
- „EXPLAIN_GET_MSGS table function“ in *Administrative SQL Routines and Views*
- „Die Tabelle EXPLAIN_STATEMENT“ auf Seite 684

EXPLAIN_DIAGNOSTIC_DATA, Tabelle

- „SYSCAT.STATEMENTS catalog view“ in *SQL Reference, Volume 1*

Die Tabelle EXPLAIN_INSTANCE

Die Tabelle EXPLAIN_INSTANCE ist die Hauptsteuertabelle für alle EXPLAIN-Informationen. Jede Datenzeile in den EXPLAIN-Tabellen ist explizit mit einer eindeutigen Zeile in dieser Tabelle verbunden. Die Tabelle EXPLAIN_INSTANCE enthält grundlegende Informationen über die Quelle der SQL-Anweisungen, für die EXPLAIN-Informationen ausgegeben werden, sowie Informationen über die Umgebung, in der die Ausgabe von EXPLAIN-Informationen stattfand.

Tabelle 82. Tabelle EXPLAIN_INSTANCE. PK bedeutet, dass die Spalte Teil eines Primärschlüssels ist. FK bedeutet, dass die Spalte Teil eines Fremdschlüssels ist.

Spaltenname	Datentyp	Nullwert?	Schlüssel?	Beschreibung
EXPLAIN_REQUESTER	VARCHAR(128)	Nein	PK	Berechtigungs-ID des Initiators dieser EXPLAIN-Anforderung
EXPLAIN_TIME	TIMESTAMP	Nein	PK	Initialisierungszeitpunkt für die EXPLAIN-Anforderung
SOURCE_NAME	VARCHAR(128)	Nein	PK	Name des Pakets, das ausgeführt wurde, als die dynamische Anweisung mit EXPLAIN bearbeitet wurde, oder der Name der Quellendatei, als das statische SQL mit EXPLAIN bearbeitet wurde.
SOURCE_SCHEMA	VARCHAR(128)	Nein	PK	Schema oder Qualifikationsmerkmal der Quelle der EXPLAIN-Anforderung
SOURCE_VERSION	VARCHAR(64)	Nein	PK	Version der Quelle der EXPLAIN-Anforderung
EXPLAIN_OPTION	CHAR(1)	Nein	Nein	Gibt an, welche EXPLAIN-Informationen bei dieser Anforderung angefordert wurden. Mögliche Werte: P PLANAUSWAHL
SNAPSHOT_TAKEN	CHAR(1)	Nein	Nein	Gibt an, ob eine EXPLAIN-Momentaufnahme für diese Anforderung erstellt wurde. Mögliche Werte: Y Es wurde eine oder mehrere EXPLAIN-Momentaufnahmen erstellt und in der Tabelle EXPLAIN_STATEMENT gespeichert. Darüber hinaus wurden auch reguläre EXPLAIN-Informationen erfasst. N Es wurde keine EXPLAIN-Momentaufnahme erstellt. Die regulären EXPLAIN-Informationen wurden jedoch erfasst. O Es wurde nur eine EXPLAIN-Momentaufnahme erstellt. Die regulären EXPLAIN-Informationen wurden nicht erfasst.
DB2_VERSION	CHAR(7)	Nein	Nein	Die Releasenummer für das DB2-Produkt, das diese EXPLAIN-Anforderung verarbeitet hat. Das Format ist <i>vv.rr.m</i> , wobei Folgendes gilt: vv Versionsnummer rr Releasenummer m Nummer des Wartungsrelease
SQL_TYPE	CHAR(1)	Nein	Nein	Gibt an, ob die EXPLAIN-Instanz für statisches oder dynamisches SQL galt. Mögliche Werte: S Statisches SQL D Dynamisches SQL

Tabelle EXPLAIN_INSTANCE

Tabelle 82. Tabelle EXPLAIN_INSTANCE (Forts.). PK bedeutet, dass die Spalte Teil eines Primärschlüssels ist. FK bedeutet, dass die Spalte Teil eines Fremdschlüssels ist.

Spaltenname	Datentyp	Nullwert?	Schlüssel?	Beschreibung
QUERYOPT	INTEGER	Nein	Nein	Gibt die vom SQL-Compiler zum Zeitpunkt des EXPLAIN-Aufrufs verwendete Optimierungsklasse an. Der Wert gibt an, welche Stufe der Abfrageoptimierung durch den SQL-Compiler für die mit EXPLAIN bearbeiteten SQL-Anweisungen ausgeführt wurde.
BLOCK	CHAR(1)	Nein	Nein	Gibt an, welche Art von Cursorblockung beim Kompilieren der SQL-Anweisungen ausgeführt wurde. Weitere Informationen finden Sie in der Spalte BLOCK in der Katalogsicht SYSCAT.PACKAGES. Mögliche Werte: N Keine Blockung U Blockung eindeutiger Cursor B Blockung aller Cursor
ISOLATION	CHAR(2)	Nein	Nein	Gibt an, welche Isolationsstufe beim Kompilieren der SQL-Anweisungen verwendet wurde. Weitere Informationen finden Sie in der Spalte ISOLATION in der Katalogsicht SYSCAT.PACKAGES. Mögliche Werte: RR Wiederholtes Lesen (Repeatable Read) RS Lesestabilität (Read Stability) CS Cursorstabilität (Cursor Stability) UR Nicht festgeschriebener Lesevorgang (Uncommitted Read)
BUFFPAGE	INTEGER	Nein	Nein	Enthält den Wert des Konfigurationsparameters BUFFPAGE für die Datenbank zum Zeitpunkt des Aufrufs von EXPLAIN.
AVG_APPLS	INTEGER	Nein	Nein	Enthält den Wert des Konfigurationsparameters AVG_APPLS zum Zeitpunkt des Aufrufs von EXPLAIN.
SORTHEAP	INTEGER	Nein	Nein	Enthält den Wert des Konfigurationsparameters SORTHEAP für die Datenbank zum Zeitpunkt des Aufrufs von EXPLAIN.
LOCKLIST	INTEGER	Nein	Nein	Enthält den Wert des Konfigurationsparameters LOCKLIST für die Datenbank zum Zeitpunkt des Aufrufs von EXPLAIN.
MAXLOCKS	SMALLINT	Nein	Nein	Enthält den Wert des Konfigurationsparameters MAXLOCKS für die Datenbank zum Zeitpunkt des Aufrufs von EXPLAIN.
LOCKS_AVAIL	INTEGER	Nein	Nein	Enthält die Anzahl der Sperren, die nach Annahme des Optimierungsprogramms für jeden Benutzer verfügbar sind. (Von den Konfigurationsparametern LOCKLIST und MAXLOCKS abgeleitet.)
CPU_SPEED	DOUBLE	Nein	Nein	Enthält den Wert des Konfigurationsparameters CPUSPEED für den Datenbankmanager zum Zeitpunkt des Aufrufs von EXPLAIN.
REMARKS	VARCHAR(254)	Ja	Nein	Vom Benutzer angegebener Kommentar
DBHEAP	INTEGER	Nein	Nein	Enthält den Wert des Konfigurationsparameters DBHEAP für die Datenbank zum Zeitpunkt des Aufrufs von EXPLAIN.
COMM_SPEED	DOUBLE	Nein	Nein	Enthält den Wert des Konfigurationsparameters COMM_BANDWIDTH für die Datenbank zum Zeitpunkt des Aufrufs von EXPLAIN.

Tabelle 82. Tabelle EXPLAIN_INSTANCE (Forts.). PK bedeutet, dass die Spalte Teil eines Primärschlüssels ist. FK bedeutet, dass die Spalte Teil eines Fremdschlüssels ist.

Spaltenname	Datentyp	Nullwert?	Schlüssel?	Beschreibung
PARALLELISM	CHAR(2)	Nein	Nein	Mögliche Werte: • N=Keine Parallelität • P=Partitionsinterne Parallelität • IP=Partitionsübergreifende Parallelität • BP=Partitionsinterne Parallelität und partitionsübergreifende Parallelität
DATAJOINER	CHAR(1)	Nein	Nein	Mögliche Werte: • N=Zugriffsplan von nicht föderierten Systemen • Y=Zugriffsplan von föderierten Systemen

Die Tabelle EXPLAIN_OBJECT

Die Tabelle EXPLAIN_OBJECT identifiziert die Datenobjekte, die für den Zugriffsplan erforderlich sind, der zur Ausführung der SQL-Anweisung generiert wurde.

Tabelle 83. Tabelle EXPLAIN_OBJECT. PK bedeutet, dass die Spalte Teil eines Primärschlüssels ist. FK bedeutet, dass die Spalte Teil eines Fremdschlüssels ist.

Spaltenname	Datentyp	Nullwert?	Schlüssel?	Beschreibung
EXPLAIN_REQUESTER	VARCHAR(128)	Nein	FK	Berechtigungs-ID des Initiators dieser EXPLAIN-Anforderung
EXPLAIN_TIME	TIMESTAMP	Nein	FK	Initialisierungszeitpunkt für die EXPLAIN-Anforderung
SOURCE_NAME	VARCHAR(128)	Nein	FK	Name des Pakets, das ausgeführt wurde, als die dynamische Anweisung mit EXPLAIN bearbeitet wurde, oder der Name der Quelldatei, als das statische SQL mit EXPLAIN bearbeitet wurde.
SOURCE_SCHEMA	VARCHAR(128)	Nein	FK	Schema oder Qualifikationsmerkmal der Quelle der EXPLAIN-Anforderung
SOURCE_VERSION	VARCHAR(64)	Nein	FK	Version der Quelle der EXPLAIN-Anforderung
EXPLAIN_LEVEL	CHAR(1)	Nein	FK	Ebene der EXPLAIN-Informationen, für die diese Zeile relevant ist
STMTNO	INTEGER	Nein	FK	Anweisungsnummer im Paket, auf die sich diese EXPLAIN-Informationen beziehen.
SECTNO	INTEGER	Nein	FK	Abschnittsnummer im Paket, auf die sich diese EXPLAIN-Informationen beziehen
OBJECT_SCHEMA	VARCHAR(128)	Nein	Nein	Schema, zu dem dieses Objekt gehört
OBJECT_NAME	VARCHAR(128)	Nein	Nein	Name des Objekts
OBJECT_TYPE	CHAR(2)	Nein	Nein	Beschreibende Kennung für den Objekttyp.
CREATE_TIME	TIMESTAMP	Ja	Nein	Zeitpunkt der Erstellung des Objekts. Null, wenn es sich um eine Tabellenfunktion handelt.
STATISTICS_TIME	TIMESTAMP	Ja	Nein	Zeitpunkt der letzten Statistikaktualisierung für dieses Objekt. Null, wenn keine Statistik für dieses Objekt vorhanden ist.
COLUMN_COUNT	SMALLINT	Nein	Nein	Anzahl der Spalten in diesem Objekt
ROW_COUNT	INTEGER	Nein	Nein	Geschätzte Anzahl Zeilen in diesem Objekt.
WIDTH	INTEGER	Nein	Nein	Durchschnittliche Länge des Objekts in Byte. Bei einem Index auf -1 gesetzt.
PAGES	BIGINT	Nein	Nein	Geschätzte Anzahl Seiten, die das Objekt im Pufferpool einnimmt. Für eine Tabellenfunktion auf -1 gesetzt.
DISTINCT	CHAR(1)	Nein	Nein	Gibt an, ob die Zeilen in dem Objekt eindeutig sind (d. h. keine doppelten Werte enthalten).
				Mögliche Werte:
				Y Ja
				N Nein
TABSPACE_NAME	VARCHAR(128)	Ja	Nein	Name des Tabellenbereichs, in dem dieses Objekt gespeichert ist. Null, wenn kein Tabellenbereich benutzt wird.
OVERHEAD	DOUBLE	Nein	Nein	Geschätzter Gesamtaufwand in Millisekunden für eine einzelne wahlfreie Ein-/Ausgabeoperation mit dem angegebenen Tabellenbereich. Berücksichtigt die Aufwände für die Steuereinheit, für Plattensuchen und die Latenzzeit. Wert -1, wenn kein Tabellenbereich benutzt wird.

Tabelle 83. Tabelle EXPLAIN_OBJECT (Forts.). PK bedeutet, dass die Spalte Teil eines Primärschlüssels ist. FK bedeutet, dass die Spalte Teil eines Fremdschlüssels ist.

Spaltenname	Datentyp	Nullwert?	Schlüssel?	Beschreibung
TRANSFER_RATE	DOUBLE	Nein	Nein	Geschätzte Zeit (in Millisekunden) zum Lesen einer Datenseite vom angegebenen Tabellenbereich. Wert -1, wenn kein Tabellenbereich benutzt wird.
PREFETCHSIZE	INTEGER	Nein	Nein	Die Anzahl der Datenseiten, die gelesen werden, wenn der Vorabsehzugriff (Prefetch) aktiv ist. Für eine Tabellenfunktion auf -1 gesetzt.
EXTENTSIZE	INTEGER	Nein	Nein	Größe des Speicherbereichs in Datenseiten. Dies ist die Anzahl der 4-KB-Seiten, die in einen Container des Tabellenbereichs geschrieben werden, bevor zum nächsten Container gewechselt wird. Für eine Tabellenfunktion auf -1 gesetzt.
CLUSTER	DOUBLE	Nein	Nein	Grad der Datenclusterbildung bezüglich des Index. Wenn ≥ 1 , ist dies der Wert CLUSTERRATIO. Wenn ≥ 0 und < 1 , ist dies der Wert CLUSTERFACTOR. Für eine Tabelle, Tabellenfunktion oder wenn diese Statistik nicht verfügbar ist, auf den Wert -1 gesetzt.
NLEAF	BIGINT	Nein	Nein	Die Anzahl der Blattseiten (Leaf pages), die von den Werten dieses Indexobjekts eingenommen werden. Für eine Tabelle, Tabellenfunktion oder wenn diese Statistik nicht verfügbar ist, auf den Wert -1 gesetzt.
NLEVELS	INTEGER	Nein	Nein	Die Anzahl von Indexstufen in der Baumstruktur dieses Indexobjekts. Für eine Tabelle, Tabellenfunktion oder wenn diese Statistik nicht verfügbar ist, auf den Wert -1 gesetzt.
FULLKEYCARD	BIGINT	Nein	Nein	Anzahl der unterschiedlichen vollständigen Schlüsselwerte in diesem Indexobjekt. Für eine Tabelle, Tabellenfunktion oder wenn diese Statistik nicht verfügbar ist, auf den Wert -1 gesetzt.
OVERFLOW	BIGINT	Nein	Nein	Gesamtzahl der Überlaufsätze in der Tabelle. Für einen Index, eine Tabellenfunktion oder wenn diese Statistik nicht verfügbar ist, auf den Wert -1 gesetzt.
FIRSTKEYCARD	BIGINT	Nein	Nein	Anzahl der eindeutigen ersten Schlüsselwerte. Für eine Tabelle, Tabellenfunktion oder wenn diese Statistik nicht verfügbar ist, auf den Wert -1 gesetzt.
FIRST2KEYCARD	BIGINT	Nein	Nein	Anzahl der eindeutigen ersten Schlüsselwerte bei Verwendung der ersten {2,3,4} Spalten des Index. Für eine Tabelle, Tabellenfunktion oder wenn diese Statistik nicht verfügbar ist, auf den Wert -1 gesetzt.
FIRST3KEYCARD	BIGINT	Nein	Nein	
FIRST4KEYCARD	BIGINT	Nein	Nein	
SEQUENTIAL_PAGES	BIGINT	Nein	Nein	Anzahl der Blattseiten (Leaf pages), die sich mit wenigen oder keinen größeren Abständen nach Indexschlüsseln sortiert auf der Platte befinden. Für eine Tabelle, Tabellenfunktion oder wenn diese Statistik nicht verfügbar ist, auf den Wert -1 gesetzt.
DENSITY	INTEGER	Nein	Nein	Verhältnis der SEQUENTIAL_PAGES zur Anzahl der Seiten im durch den Index belegten Seitenbereich; ausgedrückt als Prozentwert (ganze Zahl zwischen 0 und 100). Für eine Tabelle, Tabellenfunktion oder wenn diese Statistik nicht verfügbar ist, auf den Wert -1 gesetzt.
STATS_SRC	CHAR(1)	Nein	Nein	Gibt die Quelle für die Statistiken an. Auf den Wert 1 gesetzt, wenn von einem einzigen Knoten.
AVERAGE_SEQUENCE_GAP	DOUBLE	Nein	Nein	Lücke zwischen den Seitenfolgen
AVERAGE_SEQUENCE_FETCH_GAP	DOUBLE	Nein	Nein	Lücke zwischen Seitenfolgen beim Abrufen des Index
AVERAGE_SEQUENCE_PAGES	DOUBLE	Nein	Nein	Durchschnittliche Anzahl von Indexseiten, auf die in Reihenfolge zugegriffen werden kann

Tabelle EXPLAIN_OBJECT

Tabelle 83. Tabelle EXPLAIN_OBJECT (Forts.). PK bedeutet, dass die Spalte Teil eines Primärschlüssels ist. FK bedeutet, dass die Spalte Teil eines Fremdschlüssels ist.

Spaltenname	Datentyp	Nullwert?	Schlüssel?	Beschreibung
AVERAGE_SEQUENCE_FETCH_PAGES	DOUBLE	Nein	Nein	Durchschnittliche Anzahl von Tabellenseiten, auf die in Reihenfolge zugegriffen werden kann
AVERAGE_RANDOM_PAGES	DOUBLE	Nein	Nein	Durchschnittliche Anzahl von Indexseiten, auf die zwischen den sequenziellen Seitenzugriffen wahlfrei zugegriffen werden muss
AVERAGE_RANDOM_FETCH_PAGES	DOUBLE	Nein	Nein	Durchschnittliche Anzahl von Tabellenseiten, auf die ein wahlfreier Zugriff erfolgt, zwischen sequenziellen Seitenzugriffen beim Abrufen über den Index
NUMRIDS	BIGINT	Nein	Nein	Gesamtanzahl von Satz-IDs (RIDs) im Index
NUMRIDS_DELETED	BIGINT	Nein	Nein	Gesamtanzahl quasi-gelöschter Satz-IDs (RIDs) im Index
NUM_EMPTY_LEAFS	BIGINT	Nein	Nein	Gesamtanzahl leerer Blattseiten im Index
ACTIVE_BLOCKS	BIGINT	Nein	Nein	Gesamtanzahl aktiver MDC-Blöcke in der Tabelle
NUM_DATA_PART	INTEGER	Nein	Nein	Anzahl Datenpartitionen für eine partitionierte Tabelle. Hat den Wert 1, wenn die Tabelle nicht partitioniert ist.

Tabelle 84. Mögliche Werte für OBJECT_TYPE

Wert	Beschreibung
IX	Index
NK	Kurzname
RX	RCT-Index (Bereichsclustertabellenindex)
DP_TABLE	Partitionierte Datentabelle
TA	Tabelle
TF	Tabellenfunktion
+A	Vom Compiler referenzierter Aliasname
+C	Vom Compiler referenzierte Integritätsbedingung
+F	Vom Compiler referenzierte Funktion
+G	Vom Compiler referenzierter Trigger
+N	Vom Compiler referenzierter Kurzname
+T	Vom Compiler referenzierte Tabelle
+V	Vom Compiler referenzierte Sicht

Die Tabelle EXPLAIN_OPERATOR

Die Tabelle EXPLAIN_OPERATOR enthält alle Operatoren, die zur Erfüllung der Abfrageanweisung durch den Abfragecompiler erforderlich sind.

Tabelle 85. Die Tabelle EXPLAIN_OPERATOR. PK bedeutet, dass die Spalte Teil eines Primärschlüssels ist. FK bedeutet, dass die Spalte Teil eines Fremdschlüssels ist.

Spaltenname	Datentyp	Nullwert?	Schlüssel?	Beschreibung
EXPLAIN_REQUESTER	VARCHAR(128)	Nein	FK	Berechtigungs-ID des Initiators dieser EXPLAIN-Anforderung
EXPLAIN_TIME	TIMESTAMP	Nein	FK	Initialisierungszeitpunkt für die EXPLAIN-Anforderung
SOURCE_NAME	VARCHAR(128)	Nein	FK	Name des Pakets, das ausgeführt wurde, als die dynamische Anweisung mit EXPLAIN bearbeitet wurde, oder der Name der Quelldatei, als das statische SQL mit EXPLAIN bearbeitet wurde.
SOURCE_SCHEMA	VARCHAR(128)	Nein	FK	Schema oder Qualifikationsmerkmal der Quelle der EXPLAIN-Anforderung
SOURCE_VERSION	VARCHAR(64)	Nein	FK	Version der Quelle der EXPLAIN-Anforderung
EXPLAIN_LEVEL	CHAR(1)	Nein	FK	Ebene der EXPLAIN-Informationen, für die diese Zeile relevant ist
STMTNO	INTEGER	Nein	FK	Anweisungsnummer im Paket, auf die sich diese EXPLAIN-Informationen beziehen
SECTNO	INTEGER	Nein	FK	Abschnittsnummer im Paket, auf die sich diese EXPLAIN-Informationen beziehen
OPERATOR_ID	INTEGER	Nein	Nein	Eindeutige ID für diesen Operator in dieser Abfrage
OPERATOR_TYPE	CHAR(6)	Nein	Nein	Beschreibender Kennsatz für den Operatortyp
TOTAL_COST	DOUBLE	Nein	Nein	Geschätzter kumulativer Gesamtaufwand (in Timerons) für die Ausführung des ausgewählten Zugriffsplans bis zu diesem Operator einschließlich
IO_COST	DOUBLE	Nein	Nein	Geschätzter kumulativer E/A-Aufwand (in Datenseitenein-/ausgaben) für die Ausführung des ausgewählten Zugriffsplans bis zu diesem Operator einschließlich
CPU_COST	DOUBLE	Nein	Nein	Geschätzter kumulativer CPU-Aufwand (in Instruktionen) für die Ausführung des ausgewählten Zugriffsplans bis zu diesem Operator einschließlich
FIRST_ROW_COST	DOUBLE	Nein	Nein	Geschätzter kumulativer Aufwand (in Timerons) für das Abrufen der ersten Zeile für den ausgewählten Zugriffsplan bis zu diesem Operator einschließlich. Dieser Wert enthält den gesamten erforderlichen Anfangsaufwand.
RE_TOTAL_COST	DOUBLE	Nein	Nein	Geschätzter kumulativer Aufwand (in Timerons) für das Abrufen der nächsten Zeile für den ausgewählten Zugriffsplan bis zu diesem Operator einschließlich
RE_IO_COST	DOUBLE	Nein	Nein	Geschätzter kumulativer E/A-Aufwand (in Datenseitenein-/ausgaben) für das Abrufen der nächsten Zeile für den ausgewählten Zugriffsplan bis zu diesem Operator einschließlich
RE_CPU_COST	DOUBLE	Nein	Nein	Geschätzter kumulativer CPU-Aufwand (in Instruktionen) für das Abrufen der nächsten Zeile für den ausgewählten Zugriffsplan bis zu diesem Operator einschließlich
COMM_COST	DOUBLE	Nein	Nein	Geschätzter kumulativer Übertragungsaufwand (in TCP/IP-Rahmen) für die Ausführung des ausgewählten Zugriffsplans bis zu diesem Operator einschließlich

Tabelle EXPLAIN_OPERATOR

Tabelle 85. Die Tabelle EXPLAIN_OPERATOR (Forts.). PK bedeutet, dass die Spalte Teil eines Primärschlüssels ist. FK bedeutet, dass die Spalte Teil eines Fremdschlüssels ist.

Spaltenname	Datentyp	Nullwert?	Schlüssel?	Beschreibung
FIRST_COMM_COST	DOUBLE	Nein	Nein	Geschätzter kumulativer Übertragungsaufwand (in TCP/IP-Rahmen) für das Abrufen der nächsten Zeile für den ausgewählten Zugriffsplan bis zu diesem Operator einschließlich. Dieser Wert enthält den gesamten erforderlichen Anfangsaufwand.
BUFFERS	DOUBLE	Nein	Nein	Geschätzte Puffervoraussetzungen für diesen Operator und seine Eingaben
REMOTE_TOTAL_COST	DOUBLE	Nein	Nein	Geschätzter kumulativer Gesamtaufwand (in Timerons) für die Ausführung von Operationen für die ferne(n) Datenbank(en)
REMOTE_COMM_COST	DOUBLE	Nein	Nein	Geschätzter kumulativer Übertragungsaufwand für die Ausführung des ausgewählten fernen Zugriffsplans bis zu diesem Operator einschließlich

Tabelle 86. Werte für OPERATOR_TYPE

Wert	Beschreibung
DELETE	Delete (Löschung)
EISCAN	Durchsuchen des erweiterten Index
FETCH	Abrufen
FILTER	Filterzeilen
GENROW	Zeile generieren
GRPBY	Gruppieren nach
HSJOIN	Hash-Join
INSERT	Insert (Einfügung)
IXAND	Dynamischer logischer AND-Join von Indizes über Bitzuordnungen (Index ANDing)
IXSCAN	Durchsuchen des relationalen Index
MSJOIN	Mischjoin
NLJOIN	Join mit Verschachtelungsschleifen (Nested Loop Join)
RETURN	Ergebnis
RIDSCAN	Durchsuchen von Satz-IDs (RIDs)
SHIP	Senden der Abfrage an fernes System
SORT	Sortierung
TBSCAN	Tabellensuche
TEMP	Erstellen einer temporären Tabelle
TQ	Tabellenwarteschlange
UNION	Union-Verknüpfung
UNIQUE	Eliminierung doppelter Werte
UPDATE	Update (Aktualisierung)
XISCAN	Durchsuchen des Index für XML-Daten
XSCAN	Navigationssuche im XML-Dokument
XANDOR	Logisches Verknüpfen von Indizes für XML-Daten über AND und OR
XITERATE	XML-Sequenziterator
XUNNEST	Auflösung der XML-Sequenzverschachtelung

Die Tabelle EXPLAIN_PREDICATE

Die Tabelle EXPLAIN_PREDICATE gibt die Vergleichselemente an, die von einem bestimmten Operator angewandt werden.

Tabelle 87. Die Tabelle EXPLAIN_PREDICATE. PK bedeutet, dass die Spalte Teil eines Primärschlüssels ist. FK bedeutet, dass die Spalte Teil eines Fremdschlüssels ist.

Spaltenname	Datentyp	Nullwert?	Schlüssel?	Beschreibung
EXPLAIN_REQUESTER	VARCHAR(128)	Nein	FK	Berechtigungs-ID des Initiators dieser EXPLAIN-Anforderung
EXPLAIN_TIME	TIMESTAMP	Nein	FK	Initialisierungszeitpunkt für die EXPLAIN-Anforderung
SOURCE_NAME	VARCHAR(128)	Nein	FK	Name des Pakets, das ausgeführt wurde, als die dynamische Anweisung mit EXPLAIN bearbeitet wurde, oder der Name der Quelldatei, als das statische SQL mit EXPLAIN bearbeitet wurde.
SOURCE_SCHEMA	VARCHAR(128)	Nein	FK	Schema oder Qualifikationsmerkmal der Quelle der EXPLAIN-Anforderung
SOURCE_VERSION	VARCHAR(64)	Nein	FK	Version der Quelle der EXPLAIN-Anforderung
EXPLAIN_LEVEL	CHAR(1)	Nein	FK	Ebene der EXPLAIN-Informationen, für die diese Zeile relevant ist
STMTNO	INTEGER	Nein	FK	Anweisungsnummer im Paket, auf die sich diese EXPLAIN-Informationen beziehen
SECTNO	INTEGER	Nein	FK	Abschnittsnummer im Paket, auf die sich diese EXPLAIN-Informationen beziehen
OPERATOR_ID	INTEGER	Nein	Nein	Eindeutige ID für diesen Operator in dieser Abfrage
PREDICATE_ID	INTEGER	Nein	Nein	Eindeutige ID für dieses Vergleichselement für den angegebenen Operator
				Der Wert "-1" wird für durch das EXPLAIN-Tool konstruierte Operatorvergleichselemente angezeigt, die keine Objekte des Optimierungsprogramms sind und im Plan des Optimierungsprogramms nicht vorhanden sind.
HOW_APPLIED	CHAR(10)	Nein	Nein	Verwendungsart des angegebenen Operators für dieses Vergleichselement
WHEN_EVALUATED	CHAR(3)	Nein	Nein	Kennzeichnet, wann die in diesem Vergleichselement verwendete Unterabfrage ausgewertet wird.

Die folgenden Werte sind möglich:

Leer	Dieses Vergleichselement enthält keine Unterabfrage.
EAA	Die in diesem Vergleichselement verwendete Unterabfrage wird bei der Anwendung des Vergleichselement (EAA) ausgewertet. D. h., sie wird bei der Anwendung des Vergleichselements für jede vom angegebenen Operator verarbeiteten Zeile erneut ausgewertet.
EAO	Die in diesem Vergleichselement verwendete Unterabfrage wird beim Öffnen des Vergleichselements (EAO) ausgewertet. D. h., sie wird für den angegebenen Operator nur einmal erneut ausgewertet, und ihre Ergebnisse werden bei der Anwendung des Vergleichselements für jede Zeile erneut verwendet.
MUL	Es gibt mehrere Arten von Unterabfragen in diesem Vergleichselement.

Tabelle EXPLAIN_PREDICATE

Tabelle 87. Die Tabelle EXPLAIN_PREDICATE (Forts.). PK bedeutet, dass die Spalte Teil eines Primärschlüssels ist. FK bedeutet, dass die Spalte Teil eines Fremdschlüssels ist.

Spaltenname	Datentyp	Nullwert?	Schlüssel?	Beschreibung
RELOP_TYPE	CHAR(2)	Nein	Nein	Der Vergleichsoperatortyp, der in diesem Vergleichselement verwendet wird
SUBQUERY	CHAR(1)	Nein	Nein	Gibt an, ob ein Datenstrom aus einer Unterabfrage für dieses Vergleichselement erforderlich ist. Es können mehrere Datenströme aus Unterabfragen erforderlich sein. Die folgenden Werte sind möglich: N Es ist kein Datenstrom aus einer Unterabfrage erforderlich. Y Ein oder mehrere Datenströme sind aus einer Unterabfrage erforderlich.
FILTER_FACTOR	DOUBLE	Nein	Nein	Der geschätzte Anteil der Zeilen, die durch dieses Vergleichselement qualifiziert wird Der Wert "-1" wird angezeigt, wenn FILTER_FACTOR nicht zutreffend ist. FILTER_FACTOR ist für durch das EXPLAIN-Tool konstruierte Operatorvergleichselemente nicht zutreffend, die keine Objekte des Optimierungsprogramms sind und im Plan des Optimierungsprogramm nicht vorhanden sind.
PREDICATE_TEXT	CLOB(2M)	Ja	Nein	Der Text des Vergleichselements, wie er aus der internen Darstellung der SQL- bzw. XQuery-Anweisung reproduziert wurde. Wenn bei der Kompilierung der Anweisung der Wert einer Hostvariablen, eines Sonderregisters oder einer Parametermarke verwendet wird, wird dieser Wert am Ende des Vergleichselementtexts in Klammern angegeben. Der Wert wird in der Tabelle EXPLAIN_PREDICATE nur gespeichert, wenn die Anweisung von einem Benutzer mit der Berechtigung DBADM ausgeführt wird oder wenn die DB2-Registrierdatenbankvariable DB2_VIEW_REOPT_VALUES auf den Wert YES gesetzt ist. Ansonsten werden am Ende des Vergleichselementtexts leere Klammern angezeigt. Null, wenn nicht vorhanden.
RANGE_NUM	INTEGER	Ja	Nein	Bereich von Ausschlussvergleichselementen für Datenpartitionen, der eine Gruppierung von Vergleichselementen ermöglicht, die für den Ausschluss von Datenpartitionen nach Bereich verwendet werden. Ein Nullwert für alle anderen Typen von Vergleichselementen.

Tabelle 88. Mögliche Werte für HOW_APPLIED

Wert	Beschreibung
BSARG	Wird für jeden Block als ein als Suchargument verwendbares Vergleichselement ausgewertet.
DPSTART	Startschlüsselvergleichselement, das beim Ausschluss von Datenpartitionen verwendet wird
DPSTOP	Stoppschlüsselvergleichselement, das beim Ausschluss von Datenpartitionen verwendet wird
JOIN	Für den Join von Tabellen verwendet
RESID	Als Restvergleichselement ausgewertet
SARG	Wird ausgewertet als ein als Suchargument verwendbares Vergleichselement für Index oder Daten-seite.
START	Als Startbedingung verwendet

Tabelle 88. Mögliche Werte für HOW_APPLIED (Forts.)

Wert	Beschreibung
STOP	Als Stoppbedingung verwendet

Tabelle 89. Gültige Werte für RELOP_TYPE

Wert	Beschreibung
Leerzeichen	Nicht zutreffend
EQ	Gleich
GE	Größer-gleich
GT	Größer als
IN	IN-Liste
LE	Kleiner-gleich
LK	Ähnlich
LT	Kleiner als
NE	Ungleich
NL	Gleich null
NN	Ungleich null

Die Tabelle EXPLAIN_STATEMENT

Die Tabelle EXPLAIN_STATEMENT enthält den Text der SQL-Anweisung, wie sie für die verschiedenen Stufen der EXPLAIN-Informationen existiert. Die ursprüngliche SQL-Anweisung, wie sie vom Benutzer eingegeben wurde, wird in dieser Tabelle zusammen mit der (vom Optimierungsprogramm) zum Auswählen eines Zugriffsplans für die Ausführung der SQL-Anweisung verwendeten Version gespeichert. Die letztere Version hat oft sehr wenig Ähnlichkeit mit dem Original, da sie möglicherweise vom SQL-Compiler umgeschrieben und/oder mit zusätzlichen Vergleichselementen versehen wurde.

Tabelle 90. Tabelle EXPLAIN_STATEMENT. PK bedeutet, dass die Spalte Teil eines Primärschlüssels ist. FK bedeutet, dass die Spalte Teil eines Fremdschlüssels ist.

Spaltenname	Datentyp	Nullwert?	Schlüssel?	Beschreibung
EXPLAIN_REQUESTER	VARCHAR(128)	Nein	PK, FK	Berechtigungs-ID des Initiators dieser EXPLAIN-Anforderung
EXPLAIN_TIME	TIMESTAMP	Nein	PK, FK	Initialisierungszeitpunkt für die EXPLAIN-Anforderung
SOURCE_NAME	VARCHAR(128)	Nein	PK, FK	Name des Pakets, das ausgeführt wurde, als die dynamische Anweisung mit EXPLAIN bearbeitet wurde, oder der Name der Quelldatei, als das statische SQL mit EXPLAIN bearbeitet wurde.
SOURCE_SCHEMA	VARCHAR(128)	Nein	PK, FK	Schema oder Qualifikationsmerkmal der Quelle der EXPLAIN-Anforderung
SOURCE_VERSION	VARCHAR(64)	Nein	FK	Version der Quelle der EXPLAIN-Anforderung
EXPLAIN_LEVEL	CHAR(1)	Nein	PK	Ebene der EXPLAIN-Informationen, für die diese Zeile relevant ist Gültige Werte: O Originaltext (wie vom Benutzer eingegeben) P PLAN AUSWAHL
STMTNO	INTEGER	Nein	PK	Anweisungsnummer im Paket, auf die sich diese EXPLAIN-Informationen beziehen. Bei dynamischen EXPLAIN-SQL-Anweisungen wird dieser Wert auf 1 gesetzt. Bei statischen SQL-Anweisungen gleicht dieser Wert dem für die Katalogsicht SYSCAT.STATEMENTS verwendeten Wert.
SECTNO	INTEGER	Nein	PK	Abschnittsnummer in dem Paket, die diese SQL-Anweisung enthält. Bei dynamischen SQL-Anweisungen EXPLAIN ist dies die Abschnittsnummer, die zum Speichern des Abschnitts für diese Anweisung zur Laufzeit verwendet wird. Bei statischen SQL-Anweisungen gleicht dieser Wert dem für die Katalogsicht SYSCAT.STATEMENTS verwendeten Wert.
QUERYNO	INTEGER	Nein	Nein	Numerische Kennung für eine mit EXPLAIN bearbeitete SQL-Anweisung. Für dynamische SQL-Anweisungen (außer der SQL-Anweisung EXPLAIN), die mit Hilfe von CLP oder CLI angegeben wurden, ist der Standardwert ein sequenziell inkrementierter Wert. Andernfalls ist der Standardwert für statische SQL-Anweisungen der Wert STMTNO und für dynamische SQL-Anweisungen der Wert 1.
QUERYTAG	CHAR(20)	Nein	Nein	Identifizierendes Kennzeichen für jede mit EXPLAIN bearbeitete SQL-Anweisung. Für dynamische SQL-Anweisungen, die mit CLP angegeben wurden (außer der SQL-Anweisung EXPLAIN), ist der Standardwert 'CLP'. Für dynamische SQL-Anweisungen, die mit CLI angegeben wurden (außer der SQL-Anweisung EXPLAIN), ist der Standardwert 'CLI'. Andernfalls werden Leerzeichen als Standardwert verwendet.

Tabelle 90. Tabelle EXPLAIN_STATEMENT (Forts.). PK bedeutet, dass die Spalte Teil eines Primärschlüssels ist. FK bedeutet, dass die Spalte Teil eines Fremdschlüssels ist.

Spaltenname	Datentyp	Nullwert?	Schlüssel?	Beschreibung
STATEMENT_TYPE	CHAR(2)	Nein	Nein	Beschreibender Kennsatz für den mit EXPLAIN bearbeiteten Abfragetyp. Die folgenden Werte sind möglich: S Auswählen D Delete (Löschung) DC Löschen bei aktueller Cursorposition I Insert (Einfügung) U Aktualisieren UC Aktualisieren bei aktueller Cursorposition
UPDATABLE	CHAR(1)	Nein	Nein	Zeigt an, ob diese Anweisung als aktualisierbar angesehen wird. Dies ist besonders wichtig für Anweisungen SELECT, die als potenziell aktualisierbar bestimmt werden können. Die folgenden Werte sind möglich: ' ' Nicht zutreffend (leer) N Nein Y Ja
DELETABLE	CHAR(1)	Nein	Nein	Zeigt an, ob diese Anweisung als löschar angesehen wird. Dies ist besonders wichtig für Anweisungen SELECT, die als potenziell löschar bestimmt werden können. Die folgenden Werte sind möglich: ' ' Nicht zutreffend (leer) N Nein Y Ja
TOTAL_COST	DOUBLE	Nein	Nein	Der geschätzte Gesamtaufwand (in Timerons) der Ausführung des ausgewählten Zugriffsplans für diese Anweisung; der Wert wird auf 0 (null) gesetzt, wenn EXPLAIN_LEVEL 0 ist (ursprünglicher Text), da zu diesem Zeitpunkt kein Zugriffsplan ausgewählt ist.
STATEMENT_TEXT	CLOB(2M)	Nein	Nein	Text oder Teil des Textes der mit EXPLAIN bearbeiteten SQL-Anweisung. Der Text, der für die Planauswahlstufe von EXPLAIN gezeigt wird, wurde aus der internen Darstellung rekonstruiert und ist SQL-ähnlich; das bedeutet, dass nicht garantiert wird, dass die rekonstruierte Anweisung der gültigen SQL-Syntax entspricht.
SNAPSHOT	BLOB(10M)	Ja	Nein	Momentaufnahme der internen Darstellung für diese SQL-Anweisung beim angezeigten EXPLAIN_LEVEL. Diese Spalte ist für die Verwendung mit DB2 Visual Explain konzipiert. Die Spalte wird auf null gesetzt, wenn der Wert für EXPLAIN_LEVEL gleich 0 ist (ursprüngliche Anweisung), da zu dem Zeitpunkt, zu dem diese spezifische Version der Anweisung erfasst wird, kein Zugriffsplan ausgewählt ist.
QUERY_DEGREE	INTEGER	Nein	Nein	Gibt den Grad der partitionsinternen Parallelität zum Zeitpunkt des Aufrufs von EXPLAIN an. Für die Originalanweisung enthält diese Angabe den gesteuerten Grad der partitionsinternen Parallelität. Bei PLAN-AUSWAHL enthält sie den Grad der partitionsinternen Parallelität, der für den zu verwendenden Zugriffsplan generiert wurde.

Die Tabelle EXPLAIN_STREAM

Die Tabelle EXPLAIN_STREAM zeigt den Eingabe- und Ausgabedatenstrom zwischen einzelnen Operatoren und Datenobjekten. Die Datenobjekte selbst werden in der Tabelle EXPLAIN_OBJECT beschrieben. Die an einem Datenstrom beteiligten Operatoren befinden sich in der Tabelle EXPLAIN_OPERATOR.

Tabelle 91. Die Tabelle EXPLAIN_STREAM. PK bedeutet, dass die Spalte Teil eines Primärschlüssels ist. FK bedeutet, dass die Spalte Teil eines Fremdschlüssels ist.

Spaltenname	Datentyp	Nullwert?	Schlüssel?	Beschreibung
EXPLAIN_REQUESTER	VARCHAR(128)	Nein	FK	Berechtigungs-ID des Initiators dieser EXPLAIN-Anforderung
EXPLAIN_TIME	TIMESTAMP	Nein	FK	Initialisierungszeitpunkt für die EXPLAIN-Anforderung
SOURCE_NAME	VARCHAR(128)	Nein	FK	Name des Pakets, das ausgeführt wurde, als die dynamische Anweisung mit EXPLAIN bearbeitet wurde, oder der Name der Quelldatei, als das statische SQL mit EXPLAIN bearbeitet wurde.
SOURCE_SCHEMA	VARCHAR(128)	Nein	FK	Schema oder Qualifikationsmerkmal der Quelle der EXPLAIN-Anforderung
SOURCE_VERSION	VARCHAR(64)	Nein	FK	Version der Quelle der EXPLAIN-Anforderung
EXPLAIN_LEVEL	CHAR(1)	Nein	FK	Ebene der EXPLAIN-Informationen, für die diese Zeile relevant ist
STMTNO	INTEGER	Nein	FK	Anweisungsnummer im Paket, auf die sich diese EXPLAIN-Informationen beziehen
SECTNO	INTEGER	Nein	FK	Abschnittsnummer im Paket, auf die sich diese EXPLAIN-Informationen beziehen
STREAM_ID	INTEGER	Nein	Nein	Eindeutige ID für diesen Datenstrom im angegebenen Operator
SOURCE_TYPE	CHAR(1)	Nein	Nein	Kennzeichnet die Quelle dieses Datenstroms: O Operator D Datenobjekt
SOURCE_ID	SMALLINT	Nein	Nein	Eindeutige ID für den Operator in dieser Abfrage, die die Quelle dieses Datenstroms ist. Wird auf -1 gesetzt, wenn SOURCE_TYPE 'D' ist.
TARGET_TYPE	CHAR(1)	Nein	Nein	Kennzeichnet das Ziel dieses Datenstroms: O Operator D Datenobjekt
TARGET_ID	SMALLINT	Nein	Nein	Eindeutige ID für den Operator in dieser Abfrage, die das Ziel dieses Datenstroms ist. Wird auf -1 gesetzt, wenn TARGET_TYPE 'D' ist.
OBJECT_SCHEMA	VARCHAR(128)	Ja	Nein	Schema, zu dem das betreffende Datenobjekt gehört. Wird auf null gesetzt, wenn SOURCE_TYPE und TARGET_TYPE gleich 'O' sind.
OBJECT_NAME	VARCHAR(128)	Ja	Nein	Name des Objekts, das das Subjekt des Datenstroms ist. Wird auf null gesetzt, wenn SOURCE_TYPE und TARGET_TYPE gleich 'O' sind.
STREAM_COUNT	DOUBLE	Nein	Nein	Geschätzte Kardinalität des Datenstroms
COLUMN_COUNT	SMALLINT	Nein	Nein	Anzahl der Spalten im Datenstrom
PREDICATE_ID	INTEGER	Nein	Nein	Wenn dieser Datenstrom Teil einer Unterabfrage für ein Vergleichselement ist, wird hier die Vergleichselement-ID wiedergegeben; andernfalls wird die Spalte auf -1 gesetzt.

Tabelle 91. Die Tabelle EXPLAIN_STREAM (Forts.). PK bedeutet, dass die Spalte Teil eines Primärschlüssels ist. FK bedeutet, dass die Spalte Teil eines Fremdschlüssels ist.

Spaltenname	Datentyp	Nullwert?	Schlüssel?	Beschreibung
COLUMN_NAMES	CLOB(2M)	Ja	Nein	Diese Spalte enthält die Namen und die Informationen zur Reihenfolge der von diesem Datenstrom betroffenen Spalten. Diese Namen haben folgendes Format: NAME1 (A)+NAME2 (D)+NAME3+NAME4 Dabei gibt (A) eine aufsteigend sortierte Spalte an, (D) gibt eine absteigend sortierte Spalte an, und fehlende Sortierangaben bedeuten, dass die Spalte entweder nicht sortiert oder die Sortierfolge nicht relevant ist.
PMID	SMALLINT	Nein	Nein	Verteilungszuordnungs-ID
SINGLE_NODE	CHAR(5)	Ja	Nein	Gibt an, ob sich dieser Datenstrom in einer einzigen oder in mehreren Datenbankpartitionen befindet: MULT In mehreren Datenbankpartitionen COOR Auf Koordinatorknoten HASH Weiterleitung durch Hashverfahren RID Weiterleitung durch Zeilen-ID FUNC Weiterleitung durch Funktion (HASHEDVALUE() oder DBPARTITIONNUM()) CORR Weiterleitung durch Korrelationswert Numerisch Auf vorbestimmten einzigen Knoten gerichtet
PARTITION_COLUMNS	CLOB(2M)	Ja	Nein	Liste der Spalten, nach denen dieser Datenstrom verteilt wird.
SEQUENCE_SIZES	CLOB(2M)	Ja	Nein	Listet die erwartete Sequenzgröße für XML-Spalten auf oder zeigt den Wert "NA" (nicht zutreffend) für alle Nicht-XML-Spalten im Datenstrom an. Auf null gesetzt, wenn sich nicht mindestens eine XML-Spalte im Datenstrom befindet.

Die Tabelle ADVISE_INDEX

Die Tabelle ADVISE_INDEX zeigt die empfohlenen Indizes.

Tabelle 92. Die Tabelle ADVISE_INDEX. PK bedeutet, dass die Spalte Teil eines Primärschlüssels ist. FK bedeutet, dass die Spalte Teil eines Fremdschlüssels ist.

Spaltenname	Datentyp	Nullwert?	Schlüssel?	Beschreibung
EXPLAIN_REQUESTER	VARCHAR(128)	Nein	Nein	Berechtigungs-ID des Initiators dieser EXPLAIN-Anforderung
EXPLAIN_TIME	TIMESTAMP	Nein	Nein	Initialisierungszeitpunkt für die EXPLAIN-Anforderung
SOURCE_NAME	VARCHAR(128)	Nein	Nein	Name des Pakets, das ausgeführt wurde, als die dynamische Anweisung mit EXPLAIN bearbeitet wurde, oder der Name der Quelldatei, als das statische SQL mit EXPLAIN bearbeitet wurde.
SOURCE_SCHEMA	VARCHAR(128)	Nein	Nein	Schema oder Qualifikationsmerkmal der Quelle der EXPLAIN-Anforderung
SOURCE_VERSION	VARCHAR(64)	Nein	Nein	Version der Quelle der EXPLAIN-Anforderung
EXPLAIN_LEVEL	CHAR(1)	Nein	Nein	Ebene der EXPLAIN-Informationen, für die diese Zeile relevant ist
STMTNO	INTEGER	Nein	Nein	Anweisungsnummer im Paket, auf die sich diese EXPLAIN-Informationen beziehen
SECTNO	INTEGER	Nein	Nein	Abschnittsnummer im Paket, auf die sich diese EXPLAIN-Informationen beziehen
QUERYNO	INTEGER	Nein	Nein	Numerische Kennung für eine mit EXPLAIN bearbeitete SQL-Anweisung. Für dynamische SQL-Anweisungen (außer der SQL-Anweisung EXPLAIN), die mit Hilfe von CLP oder CLI angegeben wurden, ist der Standardwert ein sequenziell inkrementierter Wert. Andernfalls ist der Standardwert für statische SQL-Anweisungen der Wert STMTNO und für dynamische SQL-Anweisungen der Wert 1.
QUERYTAG	CHAR(20)	Nein	Nein	Identifizierendes Kennzeichen für jede mit EXPLAIN bearbeitete SQL-Anweisung. Für dynamische SQL-Anweisungen, die mit CLP angegeben wurden (außer der EXPLAIN-SQL-Anweisung), ist der Standardwert 'CLP'. Für dynamische SQL-Anweisungen, die mit CLI angegeben wurden (außer der EXPLAIN-SQL-Anweisung), ist der Standardwert 'CLI'. Andernfalls werden Leerzeichen als Standardwert verwendet.
NAME	VARCHAR(128)	Nein	Nein	Name des Index.
CREATOR	VARCHAR(128)	Nein	Nein	Qualifikationsmerkmal des Indexnamens.
TBNAME	VARCHAR(128)	Nein	Nein	Name oder Kurzname der Tabelle, für die der Index definiert ist
TBCREATOR	VARCHAR(128)	Nein	Nein	Qualifikationsmerkmal des Tabellennamens
COLNAMES	CLOB(2M)	Nein	Nein	Liste mit Spaltennamen
UNIQUERULE	CHAR(1)	Nein	Nein	Regel zur Eindeutigkeit: D=Gleiche Werte zulässig P=Primärindex U=Nur eindeutige Einträge zulässig
COLCOUNT	SMALLINT	Nein	Nein	Anzahl der Spalten im Schlüssel plus Anzahl der INCLUDE-Spalten, wenn vorhanden
IID	SMALLINT	Nein	Nein	Interne Index-ID
NLEAF	BIGINT	Nein	Nein	Anzahl der Blattseiten (Leaf pages); -1, wenn keine statistischen Daten erfasst wurden.

Tabelle 92. Die Tabelle ADVISE_INDEX (Forts.). PK bedeutet, dass die Spalte Teil eines Primärschlüssels ist. FK bedeutet, dass die Spalte Teil eines Fremdschlüssels ist.

Spaltenname	Datentyp	Nullwert?	Schlüssel?	Beschreibung
NLEVELS	SMALLINT	Nein	Nein	Anzahl der Indexstufen; -1, wenn keine statistischen Daten erfasst wurden.
FIRSTKEYCARD	BIGINT	Nein	Nein	Anzahl der eindeutigen ersten Schlüsselwerte; -1, wenn keine statistischen Daten erfasst wurden.
FULLKEYCARD	BIGINT	Nein	Nein	Anzahl der eindeutigen vollständigen Schlüsselwerte; -1, wenn keine statistischen Daten erfasst wurden.
CLUSTERRATIO	SMALLINT	Nein	Nein	Grad der Datenclusterbildung bezüglich des Index; -1, wenn keine statistischen Daten erfasst wurden oder wenn detaillierte statistische Daten zum Index erfasst wurden (in diesem Fall wird CLUSTERFACTOR verwendet).
AVGPARTITION_ CLUSTERRATIO	SMALLINT	Nein	Nein	Der Grad der Datenclusterbildung in einer einzelnen Datenpartition. -1, wenn die Tabelle nicht partitioniert ist, wenn keine statistischen Daten erfasst wurden oder wenn detaillierte statistische Daten zum Index erfasst wurden (in diesem Fall wird AVGPARTITION_CLUSTERFACTOR verwendet).
AVGPARTITION_ CLUSTERFACTOR	DOUBLE	Nein	Nein	Feinerer Messwert für den Grad der Clusterbildung in einer einzelnen Datenpartition. -1, wenn die Tabelle nicht partitioniert ist, wenn keine statistischen Daten erfasst wurden oder wenn der Index für einen Kurznamen definiert ist.
AVGPARTITION_PAGE_ FETCH_PAIRS	VARCHAR(520)	Nein	Nein	Eine Liste von Paaren ganzer Zahlen in Zeichenform. Jedes Paar stellt eine potenzielle Pufferpoolgröße und die entsprechenden Seitenabrufe dar, die für den Zugriff auf eine einzelne Datenpartition aus der Tabelle benötigt werden. Eine Zeichenfolge der Länge null, wenn keine Daten verfügbar sind oder wenn die Tabelle nicht partitioniert ist.
DATAPARTITION_ CLUSTERFACTOR	DOUBLE	Nein	Nein	Ein statistischer Messwert für den Grad der Clusterbildung (Clustering) der Indexschlüssel in Bezug auf Datenpartitionen. Dieses Feld enthält einen Wert zwischen 0 und 1, wobei 1 eine perfekte Clusterbildung und 0 keine Clusterbildung bedeutet.
CLUSTERFACTOR	DOUBLE	Nein	Nein	Feinere Erfassung des Grades der Clusterbildung oder -1, wenn keine detaillierte Indexstatistik gesammelt wurde oder wenn der Index für einen Kurznamen definiert ist.
USERDEFINED	SMALLINT	Nein	Nein	Vom Benutzer definiert.

Tabelle ADVISE_INDEX

Tabelle 92. Die Tabelle ADVISE_INDEX (Forts.). PK bedeutet, dass die Spalte Teil eines Primärschlüssels ist. FK bedeutet, dass die Spalte Teil eines Fremdschlüssels ist.

Spaltenname	Datentyp	Nullwert?	Schlüssel?	Beschreibung
SYSTEM_REQUIRED	SMALLINT	Nein	Nein	1, wenn die eine oder die andere der folgenden Bedingungen gilt: – Dieser Index ist für eine Integritätsbedingung durch einen Primärschlüssel oder eindeutigen Schlüssel erforderlich oder dieser Index ist ein Dimensionsblockindex bzw. zusammengesetzter Blockindex für eine MDC-Tabelle (Multi-Dimensional Clustering). – Dies ist ein Index für die Spalte (OID) einer typisierten Tabelle. 2, falls beide der folgenden Bedingungen gelten: – Dieser Index ist für eine Integritätsbedingung durch einen Primärschlüssel oder eindeutigen Schlüssel erforderlich oder dieser Index ist ein Dimensionsblockindex bzw. zusammengesetzter Blockindex für eine MDC-Tabelle. – Dies ist ein Index für die Spalte (OID) einer typisierten Tabelle. Ansonsten 0.
CREATE_TIME	TIMESTAMP	Nein	Nein	Zeitmarke für den Zeitpunkt der Indexerstellung
STATS_TIME	TIMESTAMP	Ja	Nein	Zeitmarke für den Zeitpunkt der letzten Änderung an den für diesen Index aufgezeichneten Statistikdaten. Null, wenn keine Statistikdaten verfügbar sind.
PAGE_FETCH_PAIRS	VARCHAR(520)	Nein	Nein	Eine Liste von Paaren ganzer Zahlen, die in Zeichenform dargestellt sind. Jedes Paar zeigt die Anzahl der Seiten in einem angenommenen Puffer, und die Anzahl der Abrufoperationen für Seiten (Page fetches), die erforderlich wäre, um die Tabelle mit diesem Index unter Verwendung des angenommenen Puffers zu durchsuchen. (Eine Zeichenfolge der Länge 0, wenn keine Daten verfügbar sind.)
REMARKS	VARCHAR(254)	Ja	Nein	Ein Kommentar des Benutzers oder null
DEFINER	VARCHAR(128)	Nein	Nein	Der Benutzer, der den Index erstellt hat
CONVERTED	CHAR(1)	Nein	Nein	Zur zukünftigen Verwendung reserviert.
SEQUENTIAL_PAGES	BIGINT	Nein	Nein	Anzahl der Blattseiten (Leaf pages), die sich mit wenigen oder keinen größeren Abständen nach Indexschlüsseln sortiert auf der Platte befinden. (–1, wenn keine statistischen Daten verfügbar sind.)
DENSITY	INTEGER	Nein	Nein	Verhältnis der SEQUENTIAL_PAGES zur Anzahl der Indexseiten ausgedrückt als Prozentwert (ganze Zahl zwischen 0 und 100, –1, wenn keine statistischen Daten verfügbar sind).
FIRST2KEYCARD	BIGINT	Nein	Nein	Anzahl der eindeutigen Schlüssel in den ersten beiden Spalten des Index (–1, wenn keine statistischen Daten gesammelt werden oder wenn dies nicht zutrifft)
FIRST3KEYCARD	BIGINT	Nein	Nein	Anzahl der eindeutigen Schlüssel in den ersten drei Spalten des Index (–1, wenn keine statistischen Daten gesammelt werden oder wenn dies nicht zutrifft)
FIRST4KEYCARD	BIGINT	Nein	Nein	Anzahl der eindeutigen Schlüssel in den ersten vier Spalten des Index (–1, wenn keine statistischen Daten gesammelt werden oder wenn dies nicht zutrifft)
PCTFREE	SMALLINT	Nein	Nein	Prozentsatz jeder äußeren Seite (Blattseite) des Index, der beim ersten Erstellen des Index reserviert wird. Dieser Speicherbereich ist für zukünftige Einfügungen nach der Erstellung des Index verfügbar.

Tabelle 92. Die Tabelle ADVISE_INDEX (Forts.). PK bedeutet, dass die Spalte Teil eines Primärschlüssels ist. FK bedeutet, dass die Spalte Teil eines Fremdschlüssels ist.

Spaltenname	Datentyp	Nullwert?	Schlüssel?	Beschreibung
UNIQUE_COLCOUNT	SMALLINT	Nein	Nein	Anzahl der für einen eindeutigen Schlüssel erforderlichen Spalten. Immer <=COLCOUNT. < COLCOUNT nur, wenn INCLUDE-Spalten vorhanden sind. -1, wenn der Index keinen eindeutigen Schlüssel hat (gleiche Werte möglich).
MINPCTUSED	SMALLINT	Nein	Nein	Wenn nicht Null, ist die Onlinedefragmentierung des Index aktiviert, und der Wert ist der Schwellenwert des minimal belegten Speicherbereichs vor dem Zusammenfügen der Seiten.
REVERSE_SCANS	CHAR(1)	Nein	Nein	Y=Index unterstützt umgekehrtes Durchsuchen N=Index unterstützt nicht umgekehrtes Durchsuchen
USE_INDEX	CHAR(1)	Ja	Nein	Y=Index empfohlen oder ausgewertet N=Index wird nicht empfohlen R=Ein vorhandener RID-Clusterindex wurde (von Designadvisor) als Nichtclusterindex empfohlen. Dies ist der Fall, wenn ein neuer RID-Clusterindex für die Tabelle empfohlen wird.
CREATION_TEXT	CLOB(2M)	Nein	Nein	Die zum Erstellen des Index verwendete SQL-Anweisung.
PACKED_DESC	BLOB(1M)	Ja	Nein	Interne Beschreibung der Tabelle
RUN_ID	TIMESTAMP	Ja	FK	Ein Wert, der dem Wert für START_TIME einer Zeile in der Tabelle ADVISE_INSTANCE entspricht, so dass diese Zeile der gleichen Ausführung des Designadvisors zugeordnet wird.
INDEXTYPE	VARCHAR(4)	Nein	Nein	Typ des Index. CLUS = Clusterindex REG = regulärer DIM = Dimensionsblockindex BLOK = Blockindex
EXISTS	CHAR(1)	Nein	Nein	Auf 'Y' gesetzt, wenn der Index im Datenbankkatalog vorhanden ist.
RIDTOBLOCK	CHAR(1)	Nein	Nein	Auf 'Y' gesetzt, wenn der RID-Index im Designadvisor zur Erstellung eines Blockindex verwendet wurde.

Die Tabelle ADVISE_INSTANCE

Die Tabelle ADVISE_INSTANCE enthält Informationen zur Ausführung von 'db2advis'. Diese Informationen beinhalten auch die Startzeit. Für jede Ausführung von 'db2advis' wird eine Zeile eingefügt. Andere ADVISE-Tabellen besitzen einen Fremdschlüssel (RUN_ID), der mit der Spalte START_TIME der Tabelle ADVISE_INSTANCE verbunden ist, und enthalten Zeilen, die während der gleichen Ausführung des Designadvisors erstellt wurden.

Tabelle 93. Tabelle ADVISE_INSTANCE. PK bedeutet, dass die Spalte Teil eines Primärschlüssels ist. FK bedeutet, dass die Spalte Teil eines Fremdschlüssels ist.

Spaltenname	Datentyp	Nullwert?	Schlüssel?	Beschreibung
START_TIME	TIMESTAMP	Nein	PK	Der Zeitpunkt, zu dem die Ausführung von 'db2advis' beginnt.
END_TIME	TIMESTAMP	Nein	Nein	Der Zeitpunkt, zu dem die Ausführung von 'db2advis' endet.
MODE	VARCHAR(4)	Nein	Nein	Der Wert, der mit der Option -m für den Designadvisor angegeben wurde, zum Beispiel 'MC' zur Angabe von MQT und MDC.
WKLD_COMPRESSION	CHAR(4)	Nein	Nein	Die Auslastungskomprimierung, unter der der Designadvisor ausgeführt wurde.
STATUS	CHAR(9)	Nein	Nein	Der Status einer Ausführung des Designadvisors. Der Status kann die Werte 'STARTED' oder 'COMPLETED' (wenn erfolgreich) annehmen bzw. eine Fehlernummer mit dem Präfix 'EI' für internen Fehler oder 'EX' für externen Fehler sein, wobei die Fehlernummer den SQLCODE darstellt.

Die Tabelle ADVISE_MQT

Die Tabelle ADVISE_MQT enthält Informationen zu gespeicherten Abfragetabellen (MQTs), die vom Designadvisor empfohlen werden.

Tabelle 94. Tabelle ADVISE_MQT. PK bedeutet, dass die Spalte Teil eines Primärschlüssels ist. FK bedeutet, dass die Spalte Teil eines Fremdschlüssels ist.

Spaltenname	Datentyp	Nullwert?	Schlüssel?	Beschreibung
EXPLAIN_REQUESTER	VARCHAR(128)	Nein	Nein	Berechtigungs-ID des Initiators dieser EXPLAIN-Anforderung
EXPLAIN_TIME	TIMESTAMP	Nein	Nein	Initialisierungszeitpunkt für die EXPLAIN-Anforderung
SOURCE_NAME	VARCHAR(128)	Nein	Nein	Name des Pakets, das ausgeführt wurde, als die dynamische Anweisung mit EXPLAIN bearbeitet wurde, oder der Name der Quelldatei, als das statische SQL mit EXPLAIN bearbeitet wurde.
SOURCE_SCHEMA	VARCHAR(128)	Nein	Nein	Schema oder Qualifikationsmerkmal der Quelle der EXPLAIN-Anforderung
SOURCE_VERSION	VARCHAR(64)	Nein	Nein	Version der Quelle der EXPLAIN-Anforderung
EXPLAIN_LEVEL	CHAR(1)	Nein	Nein	Ebene der EXPLAIN-Informationen, für die diese Zeile relevant ist
STMTNO	INTEGER	Nein	Nein	Anweisungsnummer im Paket, auf die sich diese EXPLAIN-Informationen beziehen
SECTNO	INTEGER	Nein	Nein	Anweisungsnummer im Paket, auf die sich diese EXPLAIN-Informationen beziehen
NAME	VARCHAR(128)	Nein	Nein	MQT-Name
CREATOR	VARCHAR(128)	Nein	Nein	Name des Erstellers der MQT
IID	SMALLINT	Nein	Nein	Interne Kennung
CREATE_TIME	TIMESTAMP	Nein	Nein	Zeitmarke für den Zeitpunkt der Erstellung der MQT
STATS_TIME	TIMESTAMP	Ja	Nein	Zeitmarke für den Zeitpunkt der Statistikerfassung
NUMROWS	DOUBLE	Nein	Nein	Die Anzahl der geschätzten Zeilen in der MQT
NUMCOLS	SMALLINT	Nein	Nein	Anzahl der in der MQT definierten Spalten
ROWSIZE	DOUBLE	Nein	Nein	Durchschnittliche Länge (in Byte) einer Zeile in der MQT
BENEFIT	FLOAT	Nein	Nein	Zur zukünftigen Verwendung reserviert.
USE_MQT	CHAR(1)	Ja	Nein	Auf 'Y' gesetzt, wenn die MQT empfohlen wird.
MQT_SOURCE	CHAR(1)	Ja	Nein	Gibt an, wo der MQT-Kandidat generiert wurde. Auf 'I' gesetzt, wenn der MQT-Kandidat eine MQT mit REFRESH IMMEDIATE ist, oder auf 'D' gesetzt, wenn die MQT nur mit REFRESH DEFERRED erstellt werden kann.
QUERY_TEXT	CLOB(2M)	Nein	Nein	Enthält die Abfrage, die die MQT definiert.
CREATION_TEXT	CLOB(2M)	Nein	Nein	Enthält die DDL-Anweisungen CREATE TABLE für die MQT.
SAMPLE_TEXT	CLOB(2M)	Nein	Nein	Enthält die Stichprobenabfrage, die zum Abrufen detaillierter Statistiken für die MQT verwendet wird. Wird nur verwendet, wenn für den Designadvisor detaillierte Statistiken erforderlich sind. Die resultierenden Stichprobenstatistiken werden in dieser Tabelle gezeigt. Wenn null, wurde keine Stichprobenabfrage für diese MQT erstellt.
COLSTATS	CLOB(2M)	Nein	Nein	Enthält die Spaltenstatistiken für die MQT (falls nicht null). Diese Statistiken liegen im XML-Format vor und enthalten den Spaltennamen, die Spaltenkardinalität und optional die Werte für HIGH2KEY und LOW2KEY.

Tabelle ADVISE_MQT

Tabelle 94. Tabelle ADVISE_MQT (Forts.). PK bedeutet, dass die Spalte Teil eines Primärschlüssels ist. FK bedeutet, dass die Spalte Teil eines Fremdschlüssels ist.

Spaltenname	Datentyp	Nullwert?	Schlüssel?	Beschreibung
EXTRA_INFO	BLOB(2M)	Nein	Nein	Reserviert für verschiedene Ausgaben.
TBSPACE	VARCHAR(128)	Nein	Nein	Der für die MQT empfohlene Tabellenbereich
RUN_ID	TIMESTAMP	Ja	FK	Ein Wert, der dem Wert für START_TIME einer Zeile in der Tabelle ADVISE_INSTANCE entspricht, so dass diese Zeile der gleichen Ausführung des Designadvisors zugeordnet wird.
REFRESH_TYPE	CHAR(1)	Nein	Nein	Entweder 'I' für IMMEDIATE oder 'D' für DEFERRED.
EXISTS	CHAR(1)	Nein	Nein	Auf 'Y' gesetzt, wenn die MQT im Datenbankkatalog vorhanden ist.

Die Tabelle ADVISE_PARTITION

Die Tabelle ADVISE_PARTITION enthält Informationen zu Datenbankpartitionen, die vom Designadvisor empfohlen werden. Sie kann nur in einer partitionierten Datenbankumgebung mit Daten gefüllt werden.

Tabelle 95. Tabelle ADVISE_PARTITION. PK bedeutet, dass die Spalte Teil eines Primärschlüssels ist. FK bedeutet, dass die Spalte Teil eines Fremdschlüssels ist.

Spaltenname	Datentyp	Nullwert?	Schlüssel?	Beschreibung
EXPLAIN_REQUESTER	VARCHAR(128)	Nein	Nein	Berechtigungs-ID des Initiators dieser EXPLAIN-Anforderung
EXPLAIN_TIME	TIMESTAMP	Nein	Nein	Initialisierungszeitpunkt für die EXPLAIN-Anforderung
SOURCE_NAME	VARCHAR(128)	Nein	Nein	Name des Pakets, das ausgeführt wurde, als die dynamische Anweisung mit EXPLAIN bearbeitet wurde, oder der Name der Quelldatei, als das statische SQL mit EXPLAIN bearbeitet wurde.
SOURCE_SCHEMA	VARCHAR(128)	Nein	Nein	Schema oder Qualifikationsmerkmal der Quelle der EXPLAIN-Anforderung
SOURCE_VERSION	VARCHAR(64)	Nein	Nein	Version der Quelle der EXPLAIN-Anforderung
EXPLAIN_LEVEL	CHAR(1)	Nein	Nein	Ebene der EXPLAIN-Informationen, für die diese Zeile relevant ist
STMTNO	INTEGER	Nein	Nein	Anweisungsnummer im Paket, auf die sich diese EXPLAIN-Informationen beziehen
SECTNO	INTEGER	Nein	Nein	Anweisungsnummer im Paket, auf die sich diese EXPLAIN-Informationen beziehen
QUERYNO	INTEGER	Nein	Nein	Numerische Kennung für eine mit EXPLAIN bearbeitete SQL-Anweisung. Für dynamische SQL-Anweisungen (außer der SQL-Anweisung EXPLAIN), die mit Hilfe von CLP oder CLI angegeben wurden, ist der Standardwert ein sequenziell inkrementierter Wert. Andernfalls ist der Standardwert für statische SQL-Anweisungen der Wert STMTNO und für dynamische SQL-Anweisungen der Wert 1.
QUERYTAG	CHAR(20)	Nein	Nein	Identifizierendes Kennzeichen für jede mit EXPLAIN bearbeitete SQL-Anweisung. Für dynamische SQL-Anweisungen, die mit CLP angegeben wurden (außer der EXPLAIN-SQL-Anweisung), ist der Standardwert 'CLP'. Für dynamische SQL-Anweisungen, die mit CLI angegeben wurden (außer der EXPLAIN-SQL-Anweisung), ist der Standardwert 'CLI'. Andernfalls werden Leerzeichen als Standardwert verwendet.
TBNAME	VARCHAR(128)	Ja	Nein	Gibt den Tabellennamen an.
TBCREATOR	VARCHAR(128)	Ja	Nein	Gibt den Namen des Tabellenerstellers an.
PMID	SMALLINT	Ja	Nein	Gibt die Verteilungszuordnungs-ID an.
TBSPACE	VARCHAR(128)	Ja	Nein	Gibt den Tabellenbereich an, in dem sich die Tabelle befindet.
COLNAMES	CLOB(2M)	Ja	Nein	Gibt die Datenbankpartitionsspaltennamen durch Komma getrennt an.
COLCOUNT	SMALLINT	Ja	Nein	Gibt die Anzahl von Datenbankpartitionierungsspalten an.
REPLICATE	CHAR(1)	Ja	Nein	Gibt an, ob die Datenbankpartition repliziert wird.
COST	DOUBLE	Ja	Nein	Gibt den Aufwand für die Verwendung der Datenbankpartition an.

Tabelle ADVISE_PARTITION

Tabelle 95. Tabelle ADVISE_PARTITION (Forts.). PK bedeutet, dass die Spalte Teil eines Primärschlüssels ist. FK bedeutet, dass die Spalte Teil eines Fremdschlüssels ist.

Spaltenname	Datentyp	Nullwert?	Schlüssel?	Beschreibung
USEIT	CHAR(1)	Ja	Nein	Gibt an, ob die Datenbankpartition im Modus EVALUATE PARTITION verwendet wird. Eine Datenbankpartition wird verwendet, wenn USEIT auf 'Y' oder 'y' gesetzt ist.
RUN_ID	TIMESTAMP	Ja	FK	Ein Wert, der dem Wert für START_TIME einer Zeile in der Tabelle ADVISE_INSTANCE entspricht, so dass diese Zeile der gleichen Ausführung des Designadvisors zugeordnet wird.

Die Tabelle ADVISE_TABLE

Die Tabelle ADVISE_TABLE speichert die DDL-Anweisungen (Data Definition Language) für die Erstellung von Tabellen unter Verwendung der endgültigen Empfehlungen des Designadvisors für MQTs (Materialized Query Tables), MDC-Tabellen und Datenbankpartitionierung.

Tabelle 96. Tabelle ADVISE_TABLE. PK bedeutet, dass die Spalte Teil eines Primärschlüssels ist. FK bedeutet, dass die Spalte Teil eines Fremdschlüssels ist.

Spaltenname	Datentyp	Nullwert?	Schlüssel?	Beschreibung
RUN_ID	TIMESTAMP	Ja	FK	Ein Wert, der dem Wert für START_TIME einer Zeile in der Tabelle ADVISE_INSTANCE entspricht, so dass diese Zeile der gleichen Ausführung des Designadvisors zugeordnet wird.
TABLE_NAME	VARCHAR(128)	Nein	Nein	Name der Tabelle
TABLE_SCHEMA	VARCHAR(128)	Nein	Nein	Name des Tabellenerstellers
TABLESPACE	VARCHAR(128)	Nein	Nein	Der Tabellenbereich, in dem die Tabelle zu erstellen ist
SELECTION_FLAG	VARCHAR(4)	Nein	Nein	Gibt den Empfehlungstyp an. Gültige Werte sind 'M' für MQT, 'P' für Datenbankpartitionierung und 'C' für MDC. Dieses Feld kann eine beliebige Untermenge dieser Werte enthalten. Zum Beispiel gibt 'MC' an, dass die Tabelle als MQT und als MDC-Tabelle empfohlen wird.
TABLE_EXISTS	CHAR(1)	Nein	Nein	Auf 'Y' gesetzt, wenn die Tabelle im Datenbankkatalog vorhanden ist.
USE_TABLE	CHAR(1)	Nein	Nein	Auf 'Y' gesetzt, wenn die Tabelle Empfehlungen vom Designadvisor hat.
GEN_COLUMNS	CLOB(2M)	Nein	Nein	Enthält eine Zeichenfolge für generierte Spalten, wenn diese Zeile eine MDC-Empfehlung enthält, die generierte Spalten in der DDL-Anweisung CREATE TABLE fordert.
ORGANIZE_BY	CLOB(2M)	Nein	Nein	Enthält für MDC-Empfehlungen die Klausel ORGANIZE BY für die DDL-Anweisung CREATE TABLE.
CREATION_TEXT	CLOB(2M)	Nein	Nein	Enthält die DDL-Anweisung CREATE TABLE.
ALTER_COMMAND	CLOB(2M)	Nein	Nein	Enthält eine Anweisung ALTER TABLE für die Tabelle.

Die Tabelle ADVISE_WORKLOAD

Die Tabelle ADVISE_WORKLOAD zeigt die Anweisung, die die Auslastung bildet.

Tabelle 97. Die Tabelle ADVISE_WORKLOAD. PK bedeutet, dass die Spalte Teil eines Primärschlüssels ist. FK bedeutet, dass die Spalte Teil eines Fremdschlüssels ist.

Spaltenname	Datentyp	Nullwert?	Schlüssel?	Beschreibung
WORKLOAD_NAME	CHAR(128)	Nein	Nein	Name der Objektgruppe von SQL-Anweisungen (Auslastung), zu der diese Anweisung gehört.
STATEMENT_NO	INTEGER	Nein	Nein	Anweisungsnummer in der Auslastung, auf die sich diese EXPLAIN-Informationen beziehen.
STATEMENT_TEXT	CLOB(1M)	Nein	Nein	Inhalt der SQL-Anweisung
STATEMENT_TAG	VARCHAR(256)	Nein	Nein	Identifizierendes Kennzeichen für jede mit EXPLAIN bearbeitete SQL-Anweisung
FREQUENCY	INTEGER	Nein	Nein	Gibt an, wie oft diese Anweisung in der Auslastung erscheint.
IMPORTANCE	DOUBLE	Nein	Nein	Wichtigkeit der Anweisung
WEIGHT	DOUBLE	Nein	Nein	Priorität der Anweisung
COST_BEFORE	DOUBLE	Ja	Nein	Der Aufwand der Abfrage (in Timerons), wenn die Empfehlungen nicht erstellt werden.
COST_AFTER	DOUBLE	Ja	Nein	Der Aufwand der Abfrage (in Timerons), wenn die Empfehlungen erstellt werden. COST_AFTER spiegelt alle Empfehlungen wider, mit Ausnahme derjenigen, die sich auf Clusterindizes und mehrdimensionales Clustering (MDC) beziehen.
COMPILABLE	CHAR(17)	Ja	Nein	Gibt Abfragekompilierfehler an, die beim Versuch, die Anweisung vorzubereiten, aufgetreten sind. Wenn diese Spalte den Wert NULL hat oder der Wert nicht mit SQLCA beginnt, konnte die SQL-Abfrage durch 'db2advis' kompiliert werden. Wenn ein Kompilierfehler von 'db2advis' oder dem Designadvisor gefunden wird, besteht der Wert der Spalte COMPILABLE aus einem acht Byte langen SQLCA.sqlcaid-Feld, gefolgt von einem Doppelpunkt (:) sowie einem acht Byte langen SQLCA.sqlstate-Feld, das den Rückkehrcode für die SQL-Anweisung darstellt.

Anhang D. EXPLAIN-Operatoren

Operator CMPEXP

Operatorname: CMPEXP

Aufgabe: Stellt die Berechnung von Ausdrücken dar, die für Zwischen- und Endergebnisse erforderlich sind.

(Dieser Operator wird nur für den Debugmodus verwendet.)

Zugehörige Konzepte:

- „Operator“ in *Systemverwaltung: Implementierung*

Operator DELETE

Operatorname: DELETE

Aufgabe: Stellt das Löschen von Zeilen aus einer Tabelle dar.

Dieser Operator stellt eine erforderliche Operation dar. Wenn Sie den Aufwand eines Zugriffsplans verbessern möchten, sollten Sie sich auf andere Operatoren konzentrieren (wie Durchsuchungsvorgänge und Joins), mit denen die Gruppe der zu löschenden Zeilen definiert wird.

Vorschlag zur Leistungsverbesserung:

- Wenn Sie alle Zeilen aus einer Tabelle löschen, sollten Sie vorzugsweise die Anweisung **DROP TABLE** oder den Befehl **LOAD REPLACE** verwenden.

Zugehörige Konzepte:

- „Operator“ in *Systemverwaltung: Implementierung*

Operator EISCAN

Operatorname: EISCAN

Aufgabe: Dieser Operator durchsucht einen benutzerdefinierten Index, um einen verkleinerten Datenstrom an Zeilen zu generieren. Beim Durchsuchen werden die verschiedenen Start-Stopp-Bedingungen der Bereichsgenerierungsfunktion verwendet, die vom Benutzer bereitgestellt wird.

Mit dieser Operation wird die Menge der qualifizierten Zeilen eingeschränkt, bevor auf die Basistabelle (auf Vergleichselementen basierend) zugegriffen wird.

Vorschlag zur Leistungsverbesserung:

- Im Laufe der Zeit wird ein Index möglicherweise durch Datenbankaktualisierungen fragmentiert und benötigt mehr Indexseiten als erforderlich. Dies kann durch Löschen und erneutes Erstellen des Index oder durch Reorganisieren des Index korrigiert werden.

- Wenn die Statistikdaten nicht auf dem aktuellen Stand sind, aktualisieren Sie sie mit dem Befehl RUNSTATS.

Zugehörige Konzepte:

- „Operator“ in *Systemverwaltung: Implementierung*

Operator FETCH

Operatorname: FETCH

Aufgabe: Das Abrufen von Spalten aus einer Tabelle mit Hilfe einer bestimmten Zeilenkennung (bzw. Satz-ID, RID).

Vorschläge zur Leistungsverbesserung:

- Erweitern Sie die Indexschlüssel, um die abgerufenen Spalten einzuschließen, so dass auf die Datenseiten nicht zugegriffen werden muss.
- Suchen Sie den Index, der zu dem Abruf gehört, und klicken Sie seinen Knoten doppelt an, um das Fenster mit den Statistikdaten anzuzeigen. Stellen Sie sicher, dass der Grad der Clusterbildung für den Index hoch ist.
- Erhöhen Sie die Puffergröße, wenn die Eingabe/Ausgabe (E/A), die aus dem Abruf resultiert, größer als die Anzahl der Seiten in der Tabelle ist.
- Aktualisieren Sie die Statistikdaten, wenn sie nicht auf dem aktuellen Stand sind.

Die Statistikdaten zu Quantilen und häufigen Werten geben Informationen zur Selektivität der Vergleichselemente an, wodurch bestimmt wird, wann eine Indexsuche vor einer Tabellensuche ausgewählt wird. Wenn Sie diese Statistikdaten aktualisieren möchten, verwenden Sie den Befehl **runstats** für eine Tabelle mit der Klausel WITH DISTRIBUTION.

Zugehörige Konzepte:

- „Operator“ in *Systemverwaltung: Implementierung*

Operator FILTER

Operatorname: FILTER

Aufgabe: Die Anwendung verbleibender Vergleichselemente, so dass die Daten anhand der Bedingungen der Vergleichselemente gefiltert werden.

Vorschläge zur Leistungsverbesserung:

- Stellen Sie sicher, dass Sie Vergleichselemente verwendet haben, die ausschließlich die von Ihnen benötigten Daten abrufen. Stellen Sie beispielsweise sicher, dass der Wert der Selektivität für die Vergleichselemente den Teil der Tabelle darstellt, der zurückgegeben werden soll.
- Stellen Sie sicher, dass die Optimierungsklasse auf mindestens 3 gesetzt ist, so dass das Optimierungsprogramm einen Join anstelle einer Unterabfrage verwendet. Wenn dies nicht möglich ist, sollten Sie die SQL-Abfrage manuell umschreiben, um die Unterabfrage zu vermeiden.

Zugehörige Konzepte:

- „Operator“ in *Systemverwaltung: Implementierung*

Operator GENROW

Operatorname: GENROW

Aufgabe: Stellt eine integrierte Funktion dar, die eine Tabelle mit Zeilen generiert, wobei keine Eingaben aus Tabellen, Indizes oder Operatoren verwendet werden.

GENROW kann vom Optimierungsprogramm verwendet werden, um Datenzeilen zu generieren (z. B. für eine INSERT-Anweisung oder für bestimmte IN-Listen, die in Joins umgesetzt werden).

Klicken Sie den Knoten der Funktion GENROW doppelt an, um die geschätzten Statistikdaten für die Tabellen anzuzeigen, die mit der Funktion GENROW generiert wurden.

Zugehörige Konzepte:

- „Operator“ in *Systemverwaltung: Implementierung*

Operator GRPBY

Operatorname: GRPBY

Aufgabe: Stellt die Gruppierung von Zeilen nach gemeinsamen Werten der bezeichneten Spalten oder Funktionen dar. Diese Operation ist zur Generierung von Wertegruppen oder zur Auswertung von Gruppenfunktionen erforderlich.

Wenn keine GROUP BY-Spalten angegeben werden, kann der Operator GRPBY dennoch verwendet werden, wenn es in der SELECT-Liste Funktionen zur Spaltenberechnung gibt, die darauf hinweisen, dass die gesamte Tabelle bei einer Spaltenberechnung wie eine einzige Gruppe behandelt wird.

Vorschläge zur Leistungsverbesserung:

- Dieser Operator stellt eine erforderliche Operation dar. Wenn Sie den Aufwand eines Zugriffsplans verbessern möchten, sollten Sie sich auf andere Operatoren konzentrieren (wie Durchsuchungsvorgänge und Joins), mit denen die zu gruppierende Menge von Zeilen definiert wird.
- Versuchen Sie die folgenden Methoden, um die Leistung einer SELECT-Anweisung zu erhöhen, die eine einzige Spaltenfunktion, jedoch keine GROUP BY-Klausel enthält:
 - Erstellen Sie für die Spaltenfunktion MIN(C) einen aufsteigenden Index für C.
 - Erstellen Sie für die Spaltenfunktion MAX(C) einen absteigenden Index für C.

Zugehörige Konzepte:

- „Operator“ in *Systemverwaltung: Implementierung*

Operator HSJOIN

Operatorname: HSJOIN

Aufgabe: Stellt einen Hash-Join dar, für den die qualifizierten Zeilen aus Tabellen mit einem Hashverfahren verarbeitet werden, um eine direkte Verknüpfung ohne vorherige Sortierung des Inhalts der Tabellen zu ermöglichen.

Ein Join ist erforderlich, wenn in einer FROM-Klausel mehrere Tabellen angegeben werden. Ein Hash-Join ist möglich, wenn ein Joinvergleichselement vorhanden ist, das Spalten aus zwei verschiedenen Tabellen gleichsetzt. Die Joinvergleichselemente müssen exakt den gleichen Datentyp haben. Hash-Joins können auch aus einer umgeschriebenen Unterabfrage entstehen, wie dies bei NLJOIN der Fall ist.

Bei einem Hash-Join müssen die Eingabetabellen nicht geordnet sein. Der Join wird durchgeführt, indem die innere Tabelle des Hash-Joins durchsucht und eine Referenztable generiert wird, indem die Werte der Joinspalten in einen Hashwert umgerechnet werden. Anschließend wird die äußere Tabelle gelesen, wobei die Werte der Joinspalten in Hashwerte umgerechnet und in der Referenztable überprüft werden, die für die innere Tabelle generiert wurde.

Vorschläge zur Leistungsverbesserung:

- Verwenden Sie lokale Vergleichselemente (d. h. Vergleichselemente, die nur auf eine Tabelle verweisen), um die Anzahl der zu verknüpfenden Zeilen zu reduzieren.
- Erhöhen Sie die Größe des Sortierspeichers, so dass er die Hashreferenztable aufnehmen kann.
- Wenn die Statistikdaten nicht auf dem aktuellen Stand sind, aktualisieren Sie sie mit dem Befehl **RUNSTATS**.

Zugehörige Konzepte:

- „Operator“ in *Systemverwaltung: Implementierung*

Operator INSERT

Operatorname: INSERT

Aufgabe: Stellt die Einfügung von Zeilen in eine Tabelle dar.

Dieser Operator stellt eine erforderliche Operation dar. Wenn Sie den Aufwand eines Zugriffsplans verbessern möchten, sollten Sie sich auf andere Operatoren konzentrieren (wie Durchsuchungsvorgänge und Joins), mit denen die einzufügende Gruppe von Zeilen definiert wird.

Zugehörige Konzepte:

- „Operator“ in *Systemverwaltung: Implementierung*

Operator IXAND

Operatorname: IXAND

Aufgabe: Stellt die logische AND-Verknüpfung der Ergebnisse mehrerer Indexsuchvorgänge mit Hilfe von Techniken zur dynamischen Bitzuordnung dar. Der Operator ermöglicht die Anwendung von Vergleichselementen, die logisch über AND verknüpft sind, auf mehrere Indizes, um die Zugriffe auf die zugrunde liegende Tabelle auf ein Minimum zu reduzieren.

Dieser Operator wird zu folgenden Zwecken angewendet:

- Einschränken der Menge von Zeilen vor dem Zugriff auf die Basistabelle
- Logisches Verknüpfen über AND von Vergleichselementen, die auf mehrere Indizes angewendet werden

- Logischen Verknüpfen über AND der Ergebnisse von einfachen Gleichheitsjoins, die in Sternjoins verwendet werden

Vorschläge zur Leistungsverbesserung:

- Im Laufe der Zeit wird ein Index möglicherweise durch Datenbankaktualisierungen fragmentiert und benötigt mehr Indexseiten als erforderlich. Dies kann durch Löschen und erneutes Erstellen des Index oder durch Reorganisieren des Index korrigiert werden.
- Wenn die Statistikdaten nicht auf dem aktuellen Stand sind, aktualisieren Sie sie mit dem Befehl **RUNSTATS**.
- Allgemein ist eine Indexsuche am effektivsten, wenn sich nur wenige Zeilen qualifizieren. Zur Schätzung der Anzahl der qualifizierten Zeilen verwendet das Optimierungsprogramm die Statistikdaten, die für die Spalten verfügbar sind, auf die in den Vergleichselementen verwiesen wird. Wenn einige Werte häufiger als andere auftreten, ist es wichtig, eine Verteilungsstatistik mit Hilfe der Klausel **WITH DISTRIBUTION** im Befehl **RUNSTATS** anzufordern. Wenn die Statistikdaten zu einer nicht einheitlichen Verteilung verwendet werden, kann das Optimierungsprogramm zwischen häufig und weniger häufig auftretenden Werten unterscheiden.
- **IXAND** kann am besten einzelne Spaltenindizes nutzen, da Start- und Stopp-schlüssel bei der Verwendung von **IXAND** kritisch sind.
- Erstellen Sie für Sternjoins einspaltige Indizes für die jeweils selektivsten Spalten in der Faktabelle und den dazugehörigen Dimensionstabellen.

Zugehörige Konzepte:

- „Operator“ in *Systemverwaltung: Implementierung*

Operator IXSCAN

Operatorname: IXSCAN

Aufgabe: Stellt das Durchsuchen eines Indexes dar, um einen kleineren Datenstrom an Zeilen-IDs zu generieren. Beim Durchsuchen können optionale Start-Stopp-Bedingungen verwendet werden. Der Durchsuchungsvorgang kann sich auch auf indexierbare Vergleichselemente beziehen, die auf Spalten des Indexes verweisen.

Mit dieser Operation wird die Menge der qualifizierten Zeilen-IDs eingeschränkt, bevor (auf der Basis von Vergleichselementen) auf die Basistabelle zugegriffen wird.

Vorschläge zur Leistungsverbesserung:

- Im Laufe der Zeit wird ein Index möglicherweise durch Datenbankaktualisierungen fragmentiert und benötigt mehr Indexseiten als erforderlich. Dies kann durch Löschen und erneutes Erstellen des Indexes oder durch Reorganisieren des Indexes korrigiert werden.
- Wenn auf zwei oder mehr Tabellen zugegriffen wird, kann der Zugriff auf die innere Tabelle über einen Index effizienter werden, wenn ein Index für die Joinspalte der äußeren Tabelle erstellt wird.
Weitere Richtlinien zu Indizes finden Sie in der Onlinehilfefunktion von Visual Explain.
- Wenn die Statistikdaten nicht auf dem aktuellen Stand sind, aktualisieren Sie sie mit dem Befehl **RUNSTATS**.

- Allgemein ist eine Indexsuche am effektivsten, wenn nur wenige Zeilen-IDs die Qualifikationsmerkmale erfüllen. Zur Schätzung der Anzahl der qualifizierten Zeilen-IDs verwendet das Optimierungsprogramm die Statistikdaten, die für die Spalten verfügbar sind, auf die in den Vergleichselementen verwiesen wird. Wenn einige Werte häufiger als andere auftreten, ist es wichtig, eine Verteilungsstatistik mit Hilfe der Klausel WITH DISTRIBUTION im Befehl **runstats** anzufordern. Wenn die Statistikdaten zu einer nicht einheitlichen Verteilung verwendet werden, kann das Optimierungsprogramm zwischen häufig und weniger häufig auftretenden Werten unterscheiden.

Zugehörige Konzepte:

- „Operator“ in *Systemverwaltung: Implementierung*

Operator MSJOIN

Operatorname: MSJOIN

Aufgabe: Ein Mischjoin, bei dem die qualifizierten Zeilen sowohl der äußeren als auch der inneren Tabelle die vom Joinvergleichselement vorgegebene Reihenfolge haben müssen. Ein Mischjoin wird auch als *Mischsuchjoin* oder *Mischjoin nach Sortieren* bezeichnet.

Ein Join ist erforderlich, wenn in einer FROM-Klausel mehrere Tabellen angegeben werden. Ein Mischjoin ist dann möglich, wenn es ein Joinvergleichselement gibt, das Spalten aus zwei verschiedenen Tabellen gleichsetzt. Er kann auch aus einer umgeschriebenen Unterabfrage resultieren.

Für einen Mischjoin ist eine geordnete Eingabe von Joinspalten erforderlich, da die Tabellen in der Regel nur einmal durchsucht werden. Diese geordnete Eingabe wird durch Zugriff auf einen Index oder eine sortierte Tabelle erreicht.

Vorschläge zur Leistungsverbesserung:

- Verwenden Sie lokale Vergleichselemente (d. h. Vergleichselemente, die nur auf eine Tabelle verweisen), um die Anzahl der zu verknüpfenden Zeilen zu reduzieren.
- Wenn die Statistikdaten nicht auf dem aktuellen Stand sind, aktualisieren Sie sie mit dem Befehl **RUNSTATS**.

Zugehörige Konzepte:

- „Operator“ in *Systemverwaltung: Implementierung*

Operator NLJOIN

Operatorname: NLJOIN

Aufgabe: Stellt einen Join mit Verschachtelungsschleife dar, bei dem (gewöhnlich mit Hilfe einer Indexsuche) die innere Tabelle einmal für jede Zeile der äußeren Tabelle durchsucht wird.

Ein Join ist erforderlich, wenn in einer FROM-Klausel mehrere Tabellen angegeben werden. Für einen Join mit Verschachtelungsschleife ist kein Joinvergleichselement erforderlich, die Leistung lässt sich jedoch meist durch die Verwendung eines Joinvergleichselements erhöhen.

Ein Join Verschachtelungsschleife wird durch eine der beiden folgenden Methoden ausgeführt:

- Durch Durchsuchen der inneren Tabelle für alle Zeilen der äußeren Tabelle, auf die zugegriffen wird.
- Durch Ausführen einer Indexsuchfunktion für die innere Tabelle für jede Zeile der äußeren Tabelle, auf die zugegriffen wird.

Vorschläge zur Leistungsverbesserung:

- Ein Join mit Verschachtelungsschleife ist wahrscheinlich effizienter, wenn es einen Index für die Spalten mit dem Joinvergleichselement der inneren Tabelle (die Tabelle, die rechts neben dem Operator NLJOIN angezeigt wird) gibt. Überprüfen Sie, ob für die innere Tabelle eine Tabellensuche (TBSCAN) an Stelle einer Indexsuche (IXSCAN) ausgeführt wird. Wenn eine Tabellensuche ausgeführt wird, sollten Sie vielleicht einen Index für die Joinspalten hinzufügen.
Eine andere (weniger bedeutsame) Möglichkeit zur Verbesserung des Joins ist das Erstellen eines Index für die Joinspalten der äußeren Tabelle, so dass die äußere Tabelle geordnet wird.
- Wenn die Statistikdaten nicht auf dem aktuellen Stand sind, aktualisieren Sie sie mit dem Befehl **RUNSTATS**.

Zugehörige Konzepte:

- „Operator“ in *Systemverwaltung: Implementierung*

Operator PIPE

Operatorname: PIPE

Aufgabe: Stellt die Übertragung von Zeilen zu anderen Operatoren ohne Änderungen an den Zeilen dar.

(Dieser Operator wird nur für den Debugmodus verwendet.)

Zugehörige Konzepte:

- „Operator“ in *Systemverwaltung: Implementierung*

Operator RETURN

Operatorname: RETURN

Aufgabe: Stellt die Datenrückgabe aus einer Abfrage an den Benutzer dar. Dies ist der letzte Operator im Zugriffsplandiagramm. Er zeigt die Gesamtendwerte und den Gesamtaufwand des Zugriffsplans.

Dieser Operator stellt eine erforderliche Operation dar.

Vorschlag zur Leistungsverbesserung:

- Stellen Sie sicher, dass Sie Vergleichselemente verwendet haben, die ausschließlich die von Ihnen benötigten Daten abrufen. Stellen Sie beispielsweise sicher, dass der Wert der Selektivität für die Vergleichselemente den Teil der Tabelle darstellt, der zurückgegeben werden soll.

Zugehörige Konzepte:

- „Operator“ in *Systemverwaltung: Implementierung*

Operator RIDSCN

Operatorname: RIDSCN

Aufgabe: Stellt das Durchsuchen einer Liste von Zeilenkennungen (Satz-IDs, RIDs) dar, die aus mindestens einem Index gewonnen wurden.

Dieser Operator wird in den folgenden Fällen vom Optimierungsprogramm in Betracht gezogen:

- Vergleichselemente werden durch Schlüsselwörter OR verknüpft, oder es ist ein IN-Vergleichselement vorhanden. Mit der Technik des logischen Verknüpfens von Indizes über OR (Index ORing) können Ergebnisse aus mehreren Indexzugriffen auf dieselbe Tabelle verbunden werden.
- Es sollte ein Vorablesezugriff über Listen für einen einzigen Indexzugriff verwendet werden, da durch das Sortieren der Zeilenkennungen vor dem Zugriff auf die Basiszeilen die Eingabe/Ausgabe effizienter wird.

Zugehörige Konzepte:

- „Operator“ in *Systemverwaltung: Implementierung*

Operator RPD

Operatorname: RPD

Aufgabe: Ein Operator, der in einem föderierten Datenbanksystem verwendet wird, um Daten von einer fernen Datenquelle über einen nicht relationalen Wrapper abzurufen.

Dieser Operator wird vom Optimierungsprogramm berücksichtigt, wenn er einen fernen Zugriffsplan enthält, der vom Optimierungsprogramm nicht untersucht wird. Ein RPD-Operator sendet eine Anforderung an eine ferne nicht relationale Datenquelle, um das Abfrageergebnis abzurufen. Die Anforderung wird vom nicht relationalen Wrapper mit Hilfe der API generiert, die von der Datenquelle unterstützt wird.

Zugehörige Konzepte:

- „Operator“ in *Systemverwaltung: Implementierung*

Operator SHIP

Operatorname: SHIP

Aufgabe: Ein Operator, der in einem föderierten Datenbanksystem verwendet wird, um Daten von einer fernen Datenquelle abzurufen. Dieser Operator wird vom Optimierungsprogramm berücksichtigt, wenn er einen fernen Zugriffsplan enthält, der vom Optimierungsprogramm nicht untersucht wird. Der Operator SHIP sendet eine SQL- oder XQuery-Anweisung SELECT an eine ferne Datenquelle, um das Abfrageergebnis abzurufen. Die SELECT-Anweisung wird mit Hilfe des SQL- oder XQuery-Dialekts generiert, der von der Datenquelle unterstützt wird, und kann jede gültige Abfrage enthalten, die von der Datenquelle zugelassen wird.

Zugehörige Konzepte:

- „Operator“ in *Systemverwaltung: Implementierung*

Operator SORT

Operatorname: SORT

Aufgabe: Stellt das Sortieren von Zeilen in einer Tabelle in der Reihenfolge einer oder mehrerer der Spalten dar, wobei doppelte Einträge optional entfernt werden können.

Das Sortieren ist erforderlich, wenn kein Index vorhanden ist, der die angeforderte Reihenfolge liefert, oder wenn das Sortieren weniger aufwendig als eine Indexsuche ist. Das Sortieren erfolgt in der Regel als letzte Operation, wenn die erforderlichen Zeilen abgerufen wurden oder um die Daten vor einem Join oder einer Gruppierung zu sortieren.

Wenn die Anzahl der Zeilen hoch ist oder die sortierten Daten nicht über eine Pipe geleitet werden können, ist für die Operation die aufwendige Generierung temporärer Tabellen erforderlich.

Vorschläge zur Leistungsverbesserung:

- Ziehen Sie für die Sortierspalten das Hinzufügen eines Indexes in Betracht.
- Stellen Sie sicher, dass Sie Vergleichselemente verwendet haben, die ausschließlich die von Ihnen benötigten Daten abrufen. Stellen Sie beispielsweise sicher, dass der Wert der Selektivität für die Vergleichselemente den Teil der Tabelle darstellt, der zurückgegeben werden soll.
- Stellen Sie sicher, dass die Größe des temporären Systemtabellenbereichs für den Vorabesezugriff ausreichend ist, so dass der Vorabesezugriff nicht an die Eingabe/Ausgabe gebunden ist. (Wählen Sie **Anweisung->Statistikdaten anzeigen->Tabellenbereiche** aus, um dies zu überprüfen.)
- Wenn häufig große Sortierungen erforderlich sind, sollten Sie die Werte der folgenden Konfigurationsparameter erhöhen:
 - Zwischenspeicher für Sortierlisten (sortheap). Klicken Sie mit der rechten Maustaste die Datenbank in der Steuerzentrale an, und wählen Sie anschließend **Konfigurieren** im Kontextmenü aus, um diesen Parameter zu ändern. Wählen Sie im daraufhin angezeigten Notizbuch die Indexzunge 'Leistung' aus.
 - Schwellenwert für Sortierspeicher (sheapthres). Klicken Sie mit der rechten Maustaste die Datenbankinstanz in der Steuerzentrale an, und wählen Sie anschließend **Konfigurieren** im Kontextmenü aus, um diesen Parameter zu ändern. Wählen Sie im daraufhin angezeigten Notizbuch die Indexzunge 'Leistung' aus.
- Wenn die Statistikdaten nicht auf dem aktuellen Stand sind, aktualisieren Sie sie mit dem Befehl **RUNSTATS**.

Zugehörige Konzepte:

- „Operator“ in *Systemverwaltung: Implementierung*

Operator TBSCAN

Operatorname: TBSCAN

Aufgabe: Stellt eine Tabellensuche dar, die Zeilen abrufen, indem alle erforderlichen Daten direkt aus den Datenseiten gelesen werden.

Dieser Typ von Suche wird vom Optimierungsprogramm in den folgenden Fällen einer Indexsuche vorgezogen:

- Der durchsuchte Wertebereich tritt häufig auf (d. h. es muss auf den größten Teil der Tabelle zugegriffen werden).
- Die Tabelle ist klein.
- Die Clusterbildung des Indexes ist gering.
- Es ist kein Index vorhanden.

Vorschläge zur Leistungsverbesserung:

- Eine Indexsuche ist effizienter als eine Tabellensuche, wenn die Tabelle groß ist und auf die meisten Zeilen der Tabelle nicht zugegriffen wird. Wenn Sie die Wahrscheinlichkeit erhöhen möchten, dass das Optimierungsprogramm in diesem Fall eine Indexsuche verwendet, sollten Sie Indizes den Spalten hinzufügen, für die es selektive Vergleichselemente gibt.
- Wenn ein Index bereits vorhanden ist, jedoch nicht verwendet wurde, sollten Sie überprüfen, ob selektive Vergleichselemente für alle führenden Spalten vorhanden sind. Wenn diese Vergleichselemente vorhanden sind, sollten Sie als Nächstes überprüfen, ob der Grad der Clusterbildung für den Index hoch ist. (Wenn Sie diese Statistik anzeigen möchten, öffnen Sie das Fenster **Tabellenstatistik** für die zugrunde liegende Tabelle der Sortierung, und wählen Sie den Druckknopf *Indizes* aus, um das Fenster **Indexstatistik** anzuzeigen.)
- Überprüfen Sie, ob die Größe des Tabellenbereichs für den Vorabesezugriff ausreichend ist, so dass der Vorabesezugriff nicht an die Eingabe/Ausgabe gebunden ist. (Wählen Sie **Anweisung->Statistikdaten anzeigen->Tabellenbereiche** aus, um dies zu überprüfen.)
- Wenn die Statistikdaten nicht auf dem aktuellen Stand sind, aktualisieren Sie sie mit dem Befehl **RUNSTATS**.

Die Statistikdaten zu Quantilen und häufigen Werten stellen Informationen zur Selektivität von Vergleichselementen bereit. Diese Statistikdaten werden z. B. verwendet, um zu bestimmen, wann eine Indexsuche einer Tabellensuche vorgezogen wird. Wenn Sie diese Werte aktualisieren möchten, verwenden Sie den Befehl **RUNSTATS** für eine Tabelle mit der Klausel **WITH DISTRIBUTION**.

Zugehörige Konzepte:

- „Operator“ in *Systemverwaltung: Implementierung*

Operator TEMP

Operatorname: TEMP

Aufgabe: Stellt das Speichern von Daten in einer temporären Tabelle dar, die von einem anderen Operator (möglicherweise mehrfach) wieder ausgelesen werden sollen. Die Tabelle wird erst entfernt, nachdem die SQL- bzw. XQuery-Anweisung verarbeitet wurde.

Dieser Operator ist erforderlich, um Unterabfragen auszuwerten oder Zwischenergebnisse zu speichern. In einigen Fällen (z. B. wenn die Anweisung aktualisiert werden kann) ist dieser Operator möglicherweise obligatorisch.

Zugehörige Konzepte:

- „Operator“ in *Systemverwaltung: Implementierung*

Operator TQUEUE

Operatorname: TQUEUE

Aufgabe: Stellt eine Tabellenwarteschlange dar, die verwendet wird, um die Tabellendaten von einem Datenbankagenten zu einem anderen Datenbankagenten weiterzuleiten, wenn mehrere Datenbankagenten eine Abfrage verarbeiten. Mehrere Datenbankagenten werden verwendet, um eine Abfrage parallel zu verarbeiten.

Es gibt folgende Typen von Tabellenwarteschlangen:

- **Lokal:** Die Tabellenwarteschlange wird verwendet, um Daten zwischen Datenbankagenten innerhalb eines Knotens weiterzuleiten. Eine lokale Tabellenwarteschlange wird für die partitionsinterne Parallelität verwendet.
- **Nicht lokal:** Die Tabellenwarteschlange wird verwendet, um Daten zwischen Datenbankagenten auf verschiedenen Knoten weiterzuleiten.

Zugehörige Konzepte:

- „Operator“ in *Systemverwaltung: Implementierung*

Operator UNION

Operatorname: UNION

Aufgabe: Stellt die Verkettung von Datenströmen von Zeilen aus mehreren Tabellen dar.

Dieser Operator stellt eine erforderliche Operation dar. Wenn Sie den Aufwand eines Zugriffsplans verbessern möchten, sollten Sie sich auf andere Operatoren konzentrieren (wie Durchsuchungsvorgänge und Joins), mit denen die Gruppe der zu verkettenden Zeilen definiert wird.

Zugehörige Konzepte:

- „Operator“ in *Systemverwaltung: Implementierung*

Operator UNIQUE

Operatorname: UNIQUE

Aufgabe: Stellt das Ausschließen von Zeilen mit doppelten Werten für angegebene Spalten dar.

Vorschlag zur Leistungsverbesserung:

- Dieser Operator ist nicht notwendig, wenn ein eindeutiger Index für die entsprechenden Spalten vorhanden ist.
Richtlinien zu Indizes finden Sie unter **Erstellen geeigneter Indizes** in der Onlinehilfefunktion von Visual Explain.

Zugehörige Konzepte:

- „Operator“ in *Systemverwaltung: Implementierung*

Operator UPDATE

Operatorname: UPDATE

Aufgabe: Stellt die Aktualisierung von Daten in den Zeilen einer Tabelle dar.

Dieser Operator stellt eine erforderliche Operation dar. Wenn Sie den Aufwand eines Zugriffsplans verbessern möchten, sollten Sie sich auf andere Operatoren konzentrieren (wie Durchsuchungsvorgänge und Joins), mit denen die Gruppe der zu aktualisierenden Zeilen definiert wird.

Zugehörige Konzepte:

- „Operator“ in *Systemverwaltung: Implementierung*

Anhang E. SQL- und XQuery-EXPLAIN-Tools

Mit Hilfe der Tool 'db2expln' und 'dynexpln' können Sie den für eine bestimmte Abfrageanweisung ausgewählten Zugriffsplan untersuchen, um ihn zu verstehen. Sie können auch die in die Steuerzentrale integrierte EXPLAIN-Einrichtung in Verbindung mit Visual Explain verwenden, um sich einen Einblick in den für eine bestimmte Abfrageanweisung ausgewählten Zugriffsplan zu verschaffen. Sowohl dynamische als auch statische Abfrageanweisungen können mit der EXPLAIN-Einrichtung untersucht werden. Ein Unterschied zu den EXPLAIN-Tools besteht darin, dass mit Visual Explain die EXPLAIN-Informationen in einem grafischen Format dargestellt werden. Ansonsten ist die Detaillierungsebene beider Methoden gleichwertig.

Damit Sie die Ausgabe von db2expln und dynexpln optimal verwenden können, müssen Sie mit folgenden Punkten vertraut sein:

- Die verschiedenen unterstützten Abfrageanweisungen und die im Zusammenhang mit diesen Anweisungen verwendete Terminologie (wie zum Beispiel Vergleichselemente in einer SELECT-Anweisung)
- Der Zweck eines Pakets (Zugriffsplans)
- Der Zweck und Inhalt der Systemkatalogtabellen
- Allgemeine Konzepte zur Anwendungsoptimierung

Die Themen dieses Abschnitts enthalten Informationen zu db2expln und dynexpln.

SQL- und XQuery-EXPLAIN-Tools

Das Tool db2expln beschreibt den Zugriffsplan, der für SQL- und XQuery-Anweisungen ausgewählt wird. Mit diesem Tool kann eine schnelle Erläuterung des ausgewählten Zugriffsplans abgerufen werden, wenn keine EXPLAIN-Daten erfasst wurden. Für statische SQL- und XQuery-Anweisungen überprüft db2expln die Pakete, die in den Systemkatalogtabellen gespeichert sind. Für dynamische SQL- und XQuery-Anweisungen überprüft db2expln die Abschnitte im Abfragecache.

Das Tool dynexpln kann ebenfalls zur Beschreibung des für dynamische Anweisungen ausgewählten Zugriffsplans verwendet werden. Es erstellt ein statisches Paket für die Anweisungen und verwendet dann das Tool db2expln, um sie zu beschreiben. Da dynamische SQL- und XQuery-Anweisungen jedoch auch mit db2expln überprüft werden können, wird dieses Dienstprogramm nur aus Gründen der Abwärtskompatibilität beibehalten.

Die EXPLAIN-Tools (db2expln und dynexpln) befinden sich im Unterverzeichnis bin im Verzeichnis sqllib Ihrer Instanz. Wenn sich db2expln und dynexpln nicht in Ihrem aktuellen Verzeichnis befinden, müssen sie sich in einem Verzeichnis befinden, das in Ihrer Umgebungsvariablen PATH definiert ist.

Das Programm db2expln stellt eine Verbindung zu einer Datenbank her und bindet sich mit Hilfe der Dateien db2expln.bnd, db2exsrv.bnd und db2exdyn.bnd selbst an diese Datenbank, wenn auf die Datenbank zum ersten Mal zugegriffen wird.

Zum Ausführen von db2expln müssen Sie über das Zugriffsrecht SELECT für die Systemkatalogsichten sowie über das Zugriffsrecht EXECUTE für die Pakete

db2expln - SQL- und XQuery-EXPLAIN

db2expln, db2exsrv und db2exdyn verfügen. Zum Ausführen von dynexpln müssen Sie über die Berechtigung BINDADD für die Datenbank verfügen, und das SQL-Schema, das Sie zur Herstellung einer Verbindung zur Datenbank verwenden, muss vorhanden sein, oder Sie müssen über die Berechtigung IMPLICIT_SCHEMA für die Datenbank verfügen. Zur EXPLAIN-Bearbeitung dynamischer SQL- und XQuery-Anweisungen mit db2expln oder dynexpln müssen Sie außerdem über alle Zugriffsrechte verfügen, die für die mit EXPLAIN bearbeiteten Abfrageanweisungen erforderlich sind. (Wenn Sie über SYSADM- oder DBADM-Berechtigung verfügen, haben Sie automatisch alle erforderlichen Berechtigungsstufen.)

Zugehörige Konzepte:

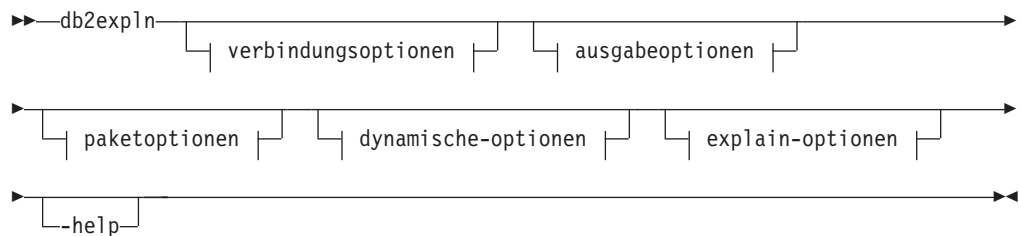
- „Richtlinien zur Analyse von EXPLAIN-Informationen“ auf Seite 271
- „Richtlinien zur Erfassung von EXPLAIN-Informationen“ auf Seite 269
- „EXPLAIN-Einrichtung“ auf Seite 257

db2expln

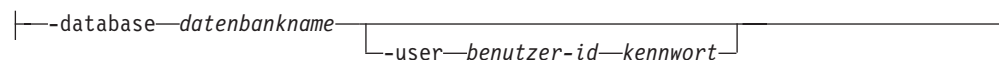
Die folgenden Abschnitte beschreiben die Syntax und die Parameter für *db2expln* und geben Hinweise zur Verwendung.

db2expln - SQL- und XQuery-EXPLAIN

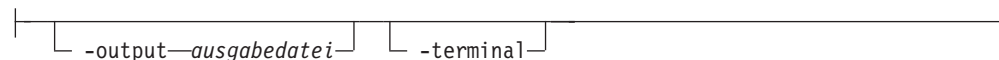
Befehlssyntax:



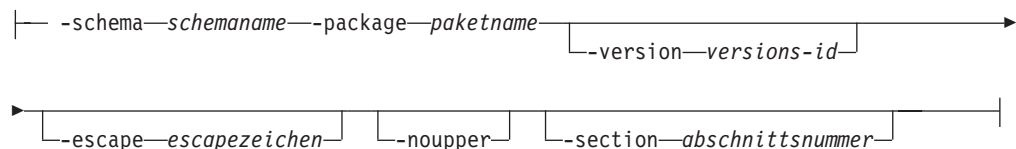
verbindungsoptionen:

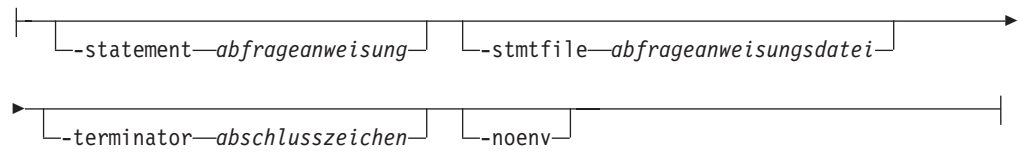
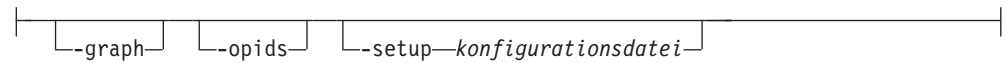


ausgabeoptionen:



paketoptionen:



dynamische-optionen:**explain-optionen:****Befehlsparameter:**

Die Optionen können in beliebiger Reihenfolge angegeben werden.

verbindungsoptionen:

Diese Optionen geben die Datenbank an, zu der eine Verbindung herzustellen ist, sowie alle für die Verbindung erforderlichen Optionen. Die Verbindungsoptionen sind außer bei Angabe der Option **-help** immer erforderlich.

-database *datenbankname*

Der Name der Datenbank, die die mit EXPLAIN zu bearbeitenden Pakete enthält.

Aus Gründen der Abwärtskompatibilität können Sie **-d** anstelle von **-database** verwenden.

-user *benutzer-id kennwort*

Die Berechtigungs-ID und das Kennwort zur Einrichtung der Datenbankverbindung. Sowohl *benutzer-id* als auch *kennwort* müssen entsprechend den DB2-Namenskonventionen gültig sein und von der Datenbank erkannt werden.

Aus Gründen der Abwärtskompatibilität können Sie **-u** anstelle von **-user** verwenden.

ausgabeoptionen:

Diese Optionen geben an, wohin die Ausgabe von db2expln geleitet werden soll. Sofern nicht die Option **-help** angegeben wird, müssen Sie mindestens eine Ausgabeoption angeben. Wenn Sie beide Optionen angeben, wird die Ausgabe in eine Datei und an das Terminal geleitet.

-output *ausgabedatei*

Die Ausgabe von db2expln wird in die von Ihnen angegebene Datei geschrieben.

Aus Gründen der Abwärtskompatibilität können Sie **-o** anstelle von **-output** verwenden.

-terminal

Die Ausgabe von db2expln wird an das Terminal geleitet.

Aus Gründen der Abwärtskompatibilität können Sie **-t** anstelle von **-terminal** verwenden.

paketoptionen:

Diese Optionen geben ein oder mehrere Pakete und Abschnitte (Sections) an, die mit EXPLAIN zu bearbeiten sind. Nur statische Abfragen in den Paketen und Abschnitten werden mit EXPLAIN bearbeitet.

Wie bei einem LIKE-Vergleichselement können Sie Platzhalterzeichen, d. h. das Prozentzeichen (%) und das Unterstreichungszeichen (_), verwenden, um den *schemanamen*, den *paketnamen* und die *versions-id* anzugeben.

-schema *schemaname*

Das SQL-Schema des Pakets bzw. der Pakete, die mit EXPLAIN zu bearbeiten sind.

Aus Gründen der Abwärtskompatibilität können Sie **-c** anstelle von **-schema** verwenden.

-package *paketname*

Der Name des Pakets oder der Pakete, die mit EXPLAIN zu bearbeiten sind.

Aus Gründen der Abwärtskompatibilität können Sie **-p** anstelle von **-package** verwenden.

-version *versions-id*

Die Versionskennung (ID) des Pakets oder der Pakete, die mit EXPLAIN zu bearbeiten sind. Der Standardwert für die Version ist eine leere Zeichenfolge.

-escape *escapezeichen*

Das Zeichen, das als Escapezeichen beim Musterabgleich in den Angaben *schemaname*, *paketname* und *versions-id* zu verwenden ist.

Zum Beispiel sieht der db2expln-Befehl zur Bearbeitung des Pakets TESTID.CALC% wie folgt aus:

```
db2expln -schema TESTID -package CALC% ....
```

Mit diesem Befehl würden jedoch auch alle anderen Pläne, die mit CALC beginnen, von EXPLAIN bearbeitet. Wenn jedoch nur das Paket TESTID.CALC% bearbeitet werden soll, muss das Escapezeichen verwendet werden. Wenn Sie das Ausrufungszeichen (!) als Escapezeichen angeben, können Sie den Befehl in folgende Form umwandeln: db2expln -schema TESTID -escape ! -package CALC!% Nun wird das Ausrufungszeichen (!) als Escapezeichen verwendet und daher die Zeichenfolge !% als das Prozentzeichen (%) interpretiert und nicht als Platzhalterzeichen für eine beliebige Zeichenfolge. Es gibt kein Standardescapezeichen.

Aus Gründen der Abwärtskompatibilität können Sie **-e** anstelle von **-escape** verwenden.

Geben Sie zur Vermeidung von Problemen nicht das Escapezeichen des Betriebssystems als Escapezeichen für **db2expln** an.

-noupper

Gibt an, dass *schemaname*, *paketname* und *versions-id* nicht in Großbuchstaben umzusetzen sind, bevor nach entsprechenden Paketen gesucht wird.

Standardmäßig werden die Werte dieser Variablen in Großbuchstaben umgesetzt, bevor nach Paketen gesucht wird. Diese Option ermöglicht eine Verwendung der Werte in exakt der eingegebenen Form.

Aus Gründen der Abwärtskompatibilität können Sie **-l** (kleines L, nicht die Ziffer 1) anstelle von **-noupper** verwenden.

-section *abschnittsnummer*

Die Nummer des mit EXPLAIN zu bearbeitenden Abschnitts im ausgewählten Paket bzw. in den ausgewählten Paketen.

Zur Bearbeitung aller Abschnitte in jedem Paket geben Sie die Ziffer 0 an. Dies ist der Standardwert. Wenn Sie diese Option nicht angeben oder wenn *schemaname*, *paketname* oder *versions-id* ein Platzhalterzeichen enthalten, werden alle Abschnitte angezeigt.

Zur Ermittlung von Abschnittsnummern fragen Sie die Systemkatalogsicht SYSCAT.STATEMENTS ab. Im Handbuch *SQL Reference* finden Sie eine Beschreibung der Systemkatalogsichten.

Aus Gründen der Abwärtskompatibilität können Sie **-s** anstelle von **-section** verwenden.

dynamische-optionen:

Diese Optionen geben eine oder auch mehrere dynamische Abfrageanweisungen an, die mit EXPLAIN zu bearbeiten sind.

-statement *abfrageanweisung*

Eine SQL- oder XQuery-Abfrageanweisung, die dynamisch vorzubereiten und mit EXPLAIN zu bearbeiten ist. Wenn Sie mehrere Anweisungen mit EXPLAIN bearbeiten wollen, geben Sie entweder mit der Option **-stmtfile** eine Datei an, in der die Abfrageanweisungen enthalten sind, die mit EXPLAIN bearbeitet werden sollen, oder Sie definieren mit Hilfe der Option **-terminator** ein Abschlusszeichen, das zur Trennung von Anweisungen in der Option **-statement** verwendet werden kann.

Aus Gründen der Kompatibilität mit `dynexpln` können Sie **-q** anstelle von **-statement** verwenden.

-stmtfile *abfrageanweisungsdatei*

Eine Datei, die eine oder mehrere Abfrageanweisungen enthält, die dynamisch vorzubereiten und mit EXPLAIN zu bearbeiten sind. Standardmäßig wird jede Zeile der Datei als separate Abfrageanweisung interpretiert. Wenn sich Anweisungen über mehrere Zeilen erstrecken, geben Sie mit der Option **-terminator** das Zeichen an, das das Ende einer Abfrageanweisung markiert.

Aus Gründen der Kompatibilität mit `dynexpln` können Sie **-f** anstelle von **-stmtfile** verwenden.

-terminator *abschlusszeichen*

Das Zeichen, das das Ende dynamischer Abfrageanweisungen angibt. Standardmäßig wird mit der Option **-statement** eine einzelne Abfrageanweisung angegeben. In der mit der Option **-stmtfile** angegebenen Datei wird jede Zeile als separate Abfrageanweisung behandelt. Das Abschlusszeichen, das Sie angeben, kann dazu verwendet werden, mehrere Abfrageanweisungen mit der Option **-statement** anzugeben oder Anweisungen in die durch die Option **-stmtfile** angegebene Datei zu schreiben, die sich über mehrere Zeilen erstrecken.

Aus Gründen der Kompatibilität mit `dynexpln` können Sie **-z** anstelle von **-terminator** verwenden.

-noenv

Gibt an, dass dynamische Anweisungen, die die Kompilierungsumgebung ändern, nach der Bearbeitung durch EXPLAIN nicht ausgeführt werden sollen.

Standardmäßig führt db2expln jede der folgenden Anweisungen nach der Bearbeitung durch EXPLAIN aus:

```
SET CURRENT DEFAULT TRANSFORM GROUP
SET CURRENT DEGREE
SET CURRENT MAINTAINED TABLE TYPES FOR OPTIMIZATION
SET CURRENT QUERY OPTIMIZATION
SET CURRENT REFRESH AGE
SET PATH
SET SCHEMA
```

Mit diesen Anweisungen ist es möglich, den Plan zu ändern, der für nachfolgende dynamische Abfrageanweisungen ausgewählt wird, die durch db2expln verarbeitet werden.

Wenn Sie **-noenv** angeben, werden diese Anweisungen mit EXPLAIN bearbeitet, jedoch nicht ausgeführt.

Zur Bearbeitung einer dynamische Abfragen mit EXPLAIN muss entweder **-statement** oder **-stmtfile** angegeben werden. Beide Optionen können in einem einzigen Aufruf von db2expln angegeben werden.

explain-optionen:

Diese Optionen legen fest, welche zusätzlichen Informationen in den EXPLAIN-Plänen bereitgestellt werden.

-graph Zeigt die Plandiagramme des Optimierungsprogramms. Jeder Abschnitt wird untersucht, und das ursprüngliche Plandiagramm des Optimierungsprogramms wird, wie von Visual Explain dargestellt, konstruiert.

Das generierte Diagramm stimmt eventuell nicht exakt mit dem von Visual Explain generierten Diagramm überein. Es ist möglich, dass das Diagramm des Optimierungsprogramms je nach den im Abschnittsplan enthaltenen Informationen einige Diskrepanzen aufweist.

Aus Gründen der Abwärtskompatibilität können Sie **-g** anstelle von **-graph** angeben.

-opids Zeigt die Operator-IDs in dem mit EXPLAIN bearbeiteten Plan an.

Die Operator-IDs ermöglichen das Abgleichen der Ausgabe von db2expln mit der Ausgabe der EXPLAIN-Einrichtung. Nicht alle Operatoren haben eine ID-Nummer, und einige ID-Nummern, die in der Ausgabe der EXPLAIN-Einrichtung angezeigt werden, kommen in der Ausgabe von db2expln nicht vor.

Aus Gründen der Abwärtskompatibilität können Sie **-i** anstelle von **-opids** angeben.

-help Zeigt den Hilfetext zu db2expln an. Wenn diese Option angegeben wird, findet keine Bearbeitung von Paketen durch EXPLAIN statt.

Der größte Teil der Befehlszeile wird in der gespeicherten Prozedur *db2exsrv* ausgeführt. Um Hilfe für alle verfügbaren Optionen zu erhalten, ist es erforderlich, **verbindungsoptionen** zusammen mit der Option **-help** anzugeben. Geben Sie zum Beispiel Folgendes ein:

```
db2expln -help -database SAMPLE
```

Aus Gründen der Abwärtskompatibilität können Sie auch **-h** oder **-?** angeben.

-setup *konfigurationsdatei*

Eine Datei, die mindestens eine Anweisung enthält, die zur Konfiguration der Umgebung für dynamische Anweisungen oder für statische Anweisungen, die erneut kompiliert werden müssen (zum Beispiel eine statische Anweisung, die auf eine deklarierte globale temporäre Tabelle verweist), benötigt wird. Jede Anweisung in der Datei wird ausgeführt, wobei etwaige Fehler oder Warnungen zurückgemeldet werden. Die Anweisungen in der Datei werden nicht mit EXPLAIN bearbeitet.

Hinweise zur Verwendung:

Wenn Sie nicht die Option **-help** angeben, müssen Sie entweder Paketooptionen oder dynamische Optionen angeben. Sie können in einem Aufruf von db2expln sowohl Pakete als auch dynamisches SQL bearbeiten.

Einige der oben erläuterten Optionsangaben haben für Ihr Betriebssystem vielleicht eine spezielle Bedeutung, so dass sie möglicherweise in der Befehlszeile für den Befehl db2expln nicht korrekt interpretiert werden. Sie können diese Zeichen möglicherweise jedoch eingeben, indem Sie ihnen ein Escapezeichen des Betriebssystems voranstellen. Weitere Informationen zu dieser Möglichkeit finden Sie in der Dokumentation zu Ihrem Betriebssystem. Stellen Sie sicher, dass Sie nicht versehentlich das Escapezeichen des Betriebssystems als Escapezeichen für db2expln angeben.

Hilfenachrichten und Nachrichten zum Anfangsstatus, die von db2expln erstellt werden, werden an die Standardausgabeeinheit geleitet. Alle Dialognachrichten und anderen Statusnachrichten, die durch das EXPLAIN-Tool generiert werden, werden an die Standardausgabeeinheit für Fehler geleitet. EXPLAIN-Text wird je nach der gewählten Ausgabeoption an die Standardausgabeeinheit oder in eine Datei geleitet.

Beispiele:

Zur Bearbeitung mehrerer Pläne in einem Aufruf von db2expln verwenden Sie die Optionen **-package**, **-schema** und **-version** und geben Zeichenfolgekonstanten für Pakete und Ersteller mit LIKE-Mustern an. Dabei können das Unterstreichungszeichen () zur Darstellung eines einzelnen Zeichens und das Prozentzeichen (%) zur Darstellung von null oder mehr Zeichen verwendet werden.

Wenn alle Abschnitte für alle Pakete in der Datenbank SAMPLE bearbeitet und die Ergebnisse in der Datei **my.exp** gespeichert werden sollen, geben Sie Folgendes ein:

```
db2expln -database SAMPLE -schema % -package % -output my.exp
```

Nehmen Sie als weiteres Beispiel an, dass ein Benutzer eine CLP-Scriptdatei mit dem Namen "statements.db2" hat und die Anweisungen in der Datei mit EXPLAIN bearbeiten will. Die Datei enthält die folgenden Anweisungen:

```
SET PATH=SYSIBM, SYSFUN, DEPT01, DEPT93@
SELECT EMPNO, TITLE(JOBID) FROM EMPLOYEE@
```

Zur Bearbeitung dieser Anweisungen geben Sie den folgenden Befehl ein:

```
db2expln -database DEPTDATA -stmtfile statements.db2 -terminator @ -terminal
```

Zugehörige Konzepte:

- „Beschreibung der Ausgabe von db2expln und dynexpln“ auf Seite 719
- „Beispiele für die Ausgabe von db2expln und dynexpln“ auf Seite 739

- „SQL- und XQuery-EXPLAIN-Tools“ auf Seite 711

Hinweise zur Verwendung von db2expln

Es folgen einige allgemeine Nachrichten, die db2expln ausgeben kann:

- No packages found for database package pattern: "<creator>".<package> with version "<version>"

Diese Nachricht wird in der Ausgabe angezeigt, wenn keine Pakete in der Datenbank gefunden wurden, die dem angegebenen Muster entsprachen.

- Bind messages can be found in db2expln.msg

Diese Nachricht wird in der Ausgabe angezeigt, wenn das Binden von db2expln.bnd nicht erfolgreich war. Weitere Informationen über die aufgetretenen Fehler finden Sie in der Datei db2expln.msg im aktuellen Verzeichnis.

- Section number overridden to 0 (all sections) for potential multiple packages.

Die Nachricht, dass die Abschnittsnummer mit 0 überschrieben wurde, wird in der Ausgabe angezeigt, wenn von db2expln mehrere Pakete bearbeitet werden können. Diese Aktion wird ausgeführt, wenn eines der Platzhalterzeichen in den Eingabeargumenten für Pakete oder Ersteller verwendet wird.

- Bind messages for <bind file> can be found in <message file>

Diese Nachricht wird angezeigt, wenn das Binden der angegebenen Bindedatei nicht erfolgreich war. Weitere Informationen über die aufgetretenen Fehler finden Sie in der angegebenen Nachrichtendatei auf dem Datenbankserver.

- No static sections qualify from package.

Diese Nachricht wird in der Ausgabe angezeigt, wenn das angegebene Paket nur dynamische Abfrageanweisungen enthält, was bedeutet, dass keine statischen Abschnitte existieren.

- Package "<creator>".<package>", "<version>", is not valid. Rebind the package and then rerun db2expln.

Die Nachricht wird in der Ausgabe angezeigt, wenn das angegebene Paket momentan ungültig ist. Führen Sie, wie in der Nachricht angegeben, den Befehl BIND oder REBIND für den Plan neu aus, um erneut ein gültiges Paket in der Datenbank zu erstellen, und wiederholen Sie anschließend db2expln.

Ausgeschlossene SQL-Anweisungen: Die folgenden Anweisungen werden von EXPLAIN nicht bearbeitet:

- BEGIN/END DECLARE SECTION
- BEGIN/END COMPOUND
- INCLUDE
- WHENEVER
- COMMIT und ROLLBACK
- CONNECT
- OPEN Cursor
- FETCH
- CLOSE Cursor
- PREPARE
- EXECUTE
- EXECUTE IMMEDIATE
- DESCRIBE

- Dynamisches DECLARE CURSOR
- SQL-Steueranweisungen

Jede Unteranweisung innerhalb einer Compound-SQL-Anweisung kann über einen eigenen Abschnitt verfügen, der von **db2expln** bearbeitet werden kann.

Anmerkung: db2expln schließt keine XQuery-Anweisungen aus.

Zugehörige Konzepte:

- „Beschreibung der Ausgabe von db2expln und dynexpln“ auf Seite 719
- „Beispiele für die Ausgabe von db2expln und dynexpln“ auf Seite 739

Zugehörige Referenzen:

- „db2expln - SQL- und XQuery-EXPLAIN“ auf Seite 712

Informationen der EXPLAIN-Ausgabe

Die folgenden Abschnitte beschreiben die Art der Informationen, die von *db2expln* und *dynexpln* bereitgestellt werden können.

Beschreibung der Ausgabe von db2expln und dynexpln

In der Ausgabe werden die EXPLAIN-Informationen für jedes Paket in den beiden folgenden Teilen dargestellt:

- Paketinformationen wie das Datum des Bindevorgangs und relevante Bindeoptionen
- Abschnittsinformationen, wie z. B. die Abschnittsnummer, gefolgt von der mit EXPLAIN bearbeiteten SQL- oder XQuery-Anweisung. Unter den Abschnittsinformationen befindet sich die EXPLAIN-Ausgabe des für die SQL- oder XQuery-Anweisung ausgewählten Zugriffsplans.

Die Schritte eines Zugriffsplans oder Abschnitts werden in der Reihenfolge aufgeführt, in der sie vom Datenbankmanager ausgeführt werden. Jeder Hauptschritt wird als linksbündig ausgerichtete Überschrift angezeigt. Informationen zum Schritt werden darunter eingerückt angezeigt. Am linken Rand der EXPLAIN-Ausgabe für den Zugriffsplan befinden sich Einrückungsbalken. Diese Balken markieren gleichzeitig den Geltungsbereich der Operation. Operationen niedrigerer Einrückungsstufe in derselben Operation, das heißt solche, die weiter rechts angezeigt werden, werden verarbeitet, bevor zur vorherigen Einrückungsstufe zurückgekehrt wird.

Beachten Sie, dass der ausgewählte Zugriffsplan auf einer erweiterten Version der ursprünglichen SQL- oder XQuery-Anweisung basiert, die in der Ausgabe gezeigt wird. Beispielsweise kann die ursprüngliche Anweisung die Aktivierung von Triggern und Integritätsbedingungen bewirken. Darüber hinaus kann die Komponente des Abfragecompilers zum Umschreiben von Abfragen die SQL- oder XQuery-Anweisung in ein äquivalentes, jedoch effizienteres Format umschreiben. Alle diese Faktoren sind in den Informationen enthalten, die das Optimierungsprogramm verwendet, wenn es den effizientesten Plan zum Ausführen der Anweisung ermittelt. Daher kann sich der in der EXPLAIN-Ausgabe angezeigte Zugriffsplan erheblich von dem unterscheiden, den Sie möglicherweise für die ursprüngliche SQL- oder XQuery-Anweisung erwarten. Die EXPLAIN-Einrichtung, zu der die EXPLAIN-Tabellen, der Modus SET CURRENT EXPLAIN und Visual Explain gehören, zeigt die tatsächlich für die Optimierung verwendete SQL- oder XQuery-Anweisung in

Form einer SQL- oder XQuery-ähnlichen Anweisung, die durch Rückübersetzung der internen Darstellung der Abfrage erstellt wird.

Beim Vergleichen der Ausgabe von db2expln oder dynexpln mit der Ausgabe der EXPLAIN-Einrichtung kann die Option für die Operator-ID (**-opids**) sehr nützlich sein. Jedesmal, wenn db2expln bzw. dynexpln die Verarbeitung eines neuen Operators aus der EXPLAIN-Einrichtung beginnt, wird die Operator-ID links neben dem von EXPLAIN gezeigten Plan ausgegeben. Die Operator-ID kann zum Abgleichen der Schritte in den verschiedenen Darstellungen des Zugriffsplans verwendet werden. Beachten Sie, dass es nicht immer eine Eins-zu-eins-Entsprechung gibt zwischen den Operatoren in der Ausgabe der EXPLAIN-Einrichtung und den Operationen, die in der Ausgabe von db2expln und dynexpln angezeigt werden.

Zugehörige Konzepte:

- „Informationen zu Spaltenberechnungen (Aggregation)“ auf Seite 732
- „Informationen zur Vorbereitung von Block- und Zeilen-IDs“ auf Seite 731
- „Datenstrominformationen“ auf Seite 730
- „EXPLAIN-Einrichtung“ auf Seite 257
- „Informationen zu Abfragen auf föderierte Datenbanken“ auf Seite 736
- „Informationen zu INSERT, UPDATE und DELETE“ auf Seite 731
- „Joininformationen“ auf Seite 728
- „Informationen zur Parallelverarbeitung“ auf Seite 733
- „Tabellenzugriffsinformationen“ auf Seite 720
- „Übersicht über Visual Explain“ in *Systemverwaltung: Implementierung*
- „Informationen zu temporären Tabellen“ auf Seite 725

Zugehörige Referenzen:

- „db2expln - SQL- und XQuery-EXPLAIN“ auf Seite 712

Tabellenzugriffsinformationen

Diese Angabe zeigt den Namen und den Typ der Tabelle an, auf die zugegriffen wird. Es werden zwei Formate verwendet:

1. Drei Typen von regulären Tabellen:

- Zugriff auf Tabellennamen (Access Table Name):

Access Table Name = schema.name ID = ts,n

Dabei gilt Folgendes:

- *schema.name* ist der vollständig qualifizierte Name der Tabelle, auf die zugegriffen wird.
- *ID* ist die entsprechende TABLESPACEID und TABLEID aus dem Katalog SYSCAT.TABLES für die Tabelle.

- Zugriff auf Hierarchietabellennamen (Access Hierarchy Table Name):

Access Hierarchy Table Name = schema.name ID = ts,n

Dabei gilt Folgendes:

- *schema.name* ist der vollständig qualifizierte Name der Tabelle, auf die zugegriffen wird.
- *ID* ist die entsprechende TABLESPACEID und TABLEID aus dem Katalog SYSCAT.TABLES für die Tabelle.

- Zugriff auf den Namen einer MQT (Materialized Query Table, gespeicherten Abfragetabelle) (Access Materialized Query Table Name):

Access Materialized Query Table Name = schema.name ID = ts,n

Dabei gilt Folgendes:

- *schema.name* ist der vollständig qualifizierte Name der Tabelle, auf die zugegriffen wird.
- *ID* ist die entsprechende TABLESPACEID und TABLEID aus dem Katalog SYSCAT.TABLES für die Tabelle.

2. Zwei Typen von temporären Tabellen:

- Zugriff auf ID für temporäre Tabellen (Access Temporary Table ID):

Access Temp Table ID = tn

Dabei gilt Folgendes:

- *ID* ist die entsprechende Kennung, die von db2expln zugeordnet wurde.

- Zugriff auf ID für deklarierte temporäre Tabellen (Access Declared Global Temporary Table ID):

Access Global Temp Table ID = ts,tn

Dabei gilt Folgendes:

- *ID* ist die entsprechende TABLESPACEID aus dem Katalog SYSCAT.TABLES für die Tabelle (ts) und die entsprechende von db2expln zugeordnete Kennung (tn)

Nach der Angabe über den Zugriff auf die Tabelle folgen weitere Angaben, die den Zugriff weiter beschreiben. Diese Angaben werden unter der Angabe über den Zugriff auf die Tabelle eingerückt. Folgende Angaben sind möglich:

- Anzahl von Spalten
- Blockzugriff
- Parallelsuche
- Suchrichtung
- Zeilenzugriffsmethode
- Sperrabsichten
- Vergleichselemente
- Verschiedene Angaben

Anzahl von Spalten

Die folgende Angabe zeigt die Anzahl von Spalten an, die aus jeder Zeile der Tabelle verwendet werden:

#Columns = n

Blockzugriff

Die folgende Angabe zeigt an, dass für die Tabelle ein oder mehrere Dimensionsblockindizes definiert sind:

Clustered by Dimension for Block Index Access

Wenn dieser Text nicht angezeigt wird, wurde die Tabelle ohne die Klausel DIMENSION erstellt.

Parallelsuche

Die folgende Angabe zeigt an, dass der Datenbankmanager mehrere Subagenten zum parallelen Lesen aus der Tabelle verwendet:

```
Parallel Scan
```

Wenn diese Angabe fehlt, wird die Tabelle nur von einem Agenten (bzw. Subagenten) gelesen.

Suchrichtung

Die folgende Angabe zeigt an, dass der Datenbankmanager Zeilen in umgekehrter Reihenfolge liest:

```
Scan Direction = Reverse
```

Wenn dieser Text nicht angezeigt wird, wird in Vorwärtsrichtung gesucht (dies ist der Standardwert).

Zeilenzugriffsmethode

Eine der folgenden Angaben zur Art und Weise des Zugriffs auf die Zeilen in der Tabelle, die den angegebenen Bedingungen entsprechen, wird angezeigt:

- Die Angabe Relation Scan bedeutet, dass die Tabelle sequenziell durchsucht wird, um die den angegebenen Bedingungen entsprechenden Zeilen zu ermitteln.
 - Die folgende Angabe bedeutet, dass kein Vorablesezugriff auf Daten erfolgt:

```
Relation Scan
| Prefetch: None
```
 - Die folgende Angabe bedeutet, dass das Optimierungsprogramm die Anzahl der Seiten, die vorabgelesen werden, vorbestimmt hat:

```
Relation Scan
| Prefetch: n Pages
```
 - Die folgende Angabe weist darauf hin, dass die Daten wahrscheinlich vorabgelesen werden:

```
Relation Scan
| Prefetch: Eligible
```
 - Die folgende Angabe bedeutet, dass die Identifizierung der Zeilen, die den angegebenen Kriterien entsprechen, und der Zugriff auf sie über einen Index erfolgen:

```
Index Scan: Name = schema.name ID = xx
| Index typ
| Index Columns:
```

Dabei gilt Folgendes:

- *schema.name* ist der vollständig qualifizierte Name des Index, der durchsucht wird.
- *ID* ist die entsprechende Spalte IID in der Katalogsicht SYSCAT.INDEXES.
- Der Indextyp ist einer der folgenden:

```
Regular Index (Not Clustered)
Regular Index (Clustered)
Dimension Block Index
Composite Dimension Block Index
Index für XML-Daten
```

Diesen Angaben folgen jeweils eine Zeile für jede Spalte im Index. Jede Spalte im Index wird in einer der folgenden Formen aufgeführt:

```
n: spaltenname (Ascending)
n: spaltenname (Descending)
n: spaltenname (Include Column)
```

Die folgenden Angaben präzisieren die Art der Indexsuche:

- Die bereichsbegrenzenden Vergleichselemente für den Index werden folgendermaßen angegeben:

```
#Key Columns = n
| Start Key: xxxxx
| Stop Key: xxxxx
```

Dabei ist xxxxx eine der folgenden Angaben:

- Start of Index
- End of Index
- Inclusive Value: oder Exclusive Value:

Ein inklusiver Schlüsselwert wird in die Indexsuche mit einbezogen. Ein exklusiver Schlüsselwert wird in der Suchoperation nicht berücksichtigt. Der Wert für den Schlüssel wird durch eine der folgenden Zeilen für jeden Teil des Schlüssels angegeben:

```
n: 'zeichenfolge'
n: nnn
n: jjjj-mm-tt
n: ss:mm:ss
n: jjjj-mm-tt ss:mm:ss.uuuuuu
n: NULL
n: ?
```

Wenn eine Literalzeichenfolge angezeigt wird, werden nur die ersten 20 Zeichen angegeben. Ist die Zeichenfolge länger als 20 Zeichen, wird dieses durch ... am Ende der Zeichenfolge angezeigt. Einige Schlüssel können erst bestimmt werden, wenn der Abschnitt (section) ausgeführt wird. Dies wird durch den Wert ? angezeigt.

- Index-Only Access

Wenn alle benötigten Spalten aus dem Indexschlüssel abgerufen werden können, wird diese Angabe für reinen Indexzugriff angezeigt, und es wird nicht auf Tabellendaten zugegriffen.

- Die folgende Angabe bedeutet, dass kein Vorabesezugriff auf die Indexseiten erfolgt:

```
Index Prefetch: None
```

- Die folgende Angabe bedeutet, dass Indexseiten wahrscheinlich vorabgelesen werden:

```
Index Prefetch: Eligible
```

- Die folgende Angabe bedeutet, dass kein Vorabesezugriff auf Datenseiten erfolgt:

```
Data Prefetch: None
```

- Die folgende Angabe weist darauf hin, dass Datenseiten wahrscheinlich vorabgelesen werden:

```
Data Prefetch: Eligible
```

- Wenn Vergleichselemente existieren, die an den Indexmanager übermittelt werden können, um bei der Qualifizierung von Indexeinträgen zu helfen, zeigt die folgende Angabe die Anzahl der Vergleichselemente:

```
Sargable Index Predicate(s)
| #Predicates = n
```

- Wenn auf die den Bedingung entsprechenden Zeilen über die Zeilen-IDs (RIDs) zugegriffen wird, die zuvor im Zugriffsplan vorbereitet wurden, wird dies durch die folgende Angabe gezeigt:

Fetch Direct Using Row IDs

Wenn für die Tabelle ein oder mehrere Blockindizes definiert sind, kann der Zugriff auf die Zeilen entweder über Block-IDs oder über Zeilen-IDs erfolgen. Dies wird wie folgt angegeben:

Fetch Direct Using Block or Row IOs

Sperrabsicht

Für jeden Tabellenzugriff wird die Art der Sperre, die auf Tabellen- und Zeilen-ebene aktiviert werden soll, mit der folgenden Angabe angezeigt:

```
Lock Intents
| Table: xxxx
| Row : xxxx
```

Folgende Werte sind für eine Tabellensperre möglich:

- Exclusive
- Intent Exclusive
- Intent None
- Intent Share
- Share
- Share Intent Exclusive
- Super Exclusive
- Update

Folgende Werte sind für eine Zeilensperre möglich:

- Exclusive
- Next Key Exclusive (wird nicht in der Ausgabe von db2expln angezeigt)
- None
- Share
- Next Key Share
- Update
- Next Key Weak Exclusive
- Weak Exclusive

Vergleichselemente

Es gibt zwei Angaben, die Informationen über die in einem Zugriffsplan verwendeten Vergleichselemente enthalten:

1. Die folgende Angabe zeigt die Anzahl von Vergleichselementen, die für jeden Datenblock ausgewertet werden, der aus einem Blockindex abgerufen wird.

```
Block Predicates(s)
| #Predicates = n
```

2. Die folgende Angabe zeigt die Anzahl von Vergleichselementen, die ausgewertet werden, während auf die Daten zugegriffen wird. Die Anzahl der Vergleichselemente lässt verschobene (Pushdown) Operationen wie Spaltenberechnung oder Sortierung unberücksichtigt.

```
Sargable Predicate(s)
| #Predicates = n
```

3. Die folgende Angabe zeigt die Anzahl von Vergleichselementen, die ausgewertet werden, nachdem die Daten zurückgegeben wurden (d. h. Restvergleichselemente):

```
Residual Predicate(s)
| #Predicates = n
```

Die Anzahl der in den oben stehenden Anweisungen angezeigten Vergleichselemente spiegelt möglicherweise die Anzahl der Vergleichselemente der Abfrageanweisung aus folgenden Gründen nicht genau wider:

- Vergleichselemente können mehrmals in derselben Abfrage verwendet werden.
- Vergleichselemente können durch Hinzufügung impliziter Vergleichselemente während der Optimierung der Abfrage umgewandelt und erweitert worden sein.
- Vergleichselemente können während der Optimierung der Abfrage in weniger Vergleichselemente umgewandelt und komprimiert worden sein.

Verschiedene Tabellenangaben

- Die folgende Angabe weist darauf hin, dass nur auf eine Zeile zugegriffen wird:

```
Single Record
```

- Die folgende Angabe wird angezeigt, wenn die für diesen Tabellenzugriff verwendete Isolationsstufe nicht der Isolationsstufe der Anweisung entspricht:

```
Isolation Level: xxxx
```

Eine andere Isolationsstufe kann aus einer Reihe von Gründen verwendet werden, zum Beispiel:

- Ein Paket wurde mit wiederholtem Lesen (RR) gebunden und hat Auswirkungen auf referenzielle Integritätsbedingungen. Der Zugriff der übergeordneten Tabelle zum Prüfen referenzieller Integritätsbedingungen wird auf die Isolationsstufe Cursorstabilität (CS) herabgestuft, um unnötige Sperren für diese Tabelle zu vermeiden.
- Ein mit der Isolationsstufe für nicht festgeschriebenen Lesevorgang (UR) gebundenes Paket gibt eine Anweisung DELETE oder UPDATE aus. Der Tabellenzugriff für den tatsächlichen Löschvorgang wird auf Cursorstabilität (CS) hochgestuft.
- Die folgende Angabe weist darauf hin, dass einige oder alle Zeilen, die aus der temporären Tabelle gelesen wurden, außerhalb des Pufferpools zwischengespeichert werden, wenn genügend Sortierspeicher verfügbar ist:

```
Keep Rows In Private Memory
```

- Wenn für die Tabelle das Attribut für flüchtige Kardinalität definiert wurde, wird dies folgendermaßen angezeigt:

```
Volatile Cardinality
```

Zugehörige Konzepte:

- „Beschreibung der Ausgabe von db2expln und dynexpln“ auf Seite 719
- „Beispiele für die Ausgabe von db2expln und dynexpln“ auf Seite 739

Informationen zu temporären Tabellen

Eine temporäre Tabelle wird von einem Zugriffsplan während dessen Ausführung zum Speichern von Daten in einer temporären Arbeitstabelle oder Übergangstabelle verwendet. Die Tabelle existiert nur, solange der Zugriffsplan ausgeführt

wird. Allgemein werden temporäre Tabellen verwendet, wenn Unterabfragen frühzeitig im Zugriffsplan ausgewertet werden müssen oder wenn Zwischenergebnisse nicht in den vorhandenen Speicher passen.

Wenn eine temporäre Tabelle erstellt werden muss, kann eine von zwei möglichen Angaben angezeigt werden. Diese Angaben weisen darauf hin, dass eine temporäre Tabelle erstellt und Zeilen in sie eingefügt werden. Die ID ist eine Kennung, die aus praktischen Gründen von db2expln zugeordnet wird, wenn auf die temporäre Tabelle Bezug genommen wird. Dieser ID wird als Präfix der Buchstabe 't' vorangestellt, um anzuzeigen, dass es sich um eine temporäre Tabelle handelt.

- Die folgende Angabe bedeutet, dass eine normale temporäre Tabelle erstellt wird:

```
Insert Into Temp Table ID = tn
```

- Die folgende Angabe bedeutet, dass eine normale temporäre Tabelle von mehreren Subagenten parallel erstellt wird:

```
Insert Into Shared Temp Table ID = tn
```

- Die folgende Angabe bedeutet, dass eine sortierte temporäre Tabelle erstellt wird:

```
Insert Into Sorted Temp Table ID = tn
```

- Die folgende Angabe bedeutet, dass eine sortierte temporäre Tabelle von mehreren Subagenten parallel erstellt wird:

```
Insert Into Sorted Shared Temp Table ID = tn
```

- Die folgende Angabe bedeutet, dass eine deklarierte globale temporäre Tabelle erstellt wird:

```
Insert Into Global Temp Table ID = ts,tn
```

- Die folgende Angabe bedeutet, dass eine deklarierte globale temporäre Tabelle von mehreren Subagenten parallel erstellt wird:

```
Insert Into Shared Global Temp Table ID = ts,tn
```

- Die folgende Angabe bedeutet, dass eine sortierte deklarierte globale temporäre Tabelle erstellt wird:

```
Insert Into Sorted Global Temp Table ID = ts,tn
```

- Die folgende Angabe bedeutet, dass eine sortierte deklarierte globale temporäre Tabelle von mehreren Subagenten parallel erstellt wird:

```
Insert Into Sorted Shared Global Temp Table ID = ts,tn
```

Nach jeder der oben gezeigten Angaben folgt die Angabe:

```
#Columns = n
```

Diese Angabe zeigt die Anzahl der Spalten für jede Zeile, die in die temporäre Tabelle eingefügt wird.

Speicherbedarf für temporäre Tabellen

Sortierte temporäre Tabellen treten z. B. als Ergebnis folgender Operationen auf:

- ORDER BY
- DISTINCT
- GROUP BY
- Merge Join
- '= ANY' Unterabfrage
- '<> ALL' Unterabfrage
- INTERSECT oder EXCEPT
- UNION (ohne das Schlüsselwort ALL)

Eine Reihe zusätzlicher Angaben kann auf die ursprüngliche Angabe über die Erstellung für eine sortierte temporäre Tabelle folgen:

- Die folgende Angabe zeigt die Anzahl der bei der Sortierung verwendeten Spalten:

#Sort Key Columns = n

Für jede Spalte im Sortierschlüssel wird eine der folgenden Zeilen angezeigt:

Key n: spaltenname (Ascending)
 Key n: spaltenname (Descending)
 Key n: (Ascending)
 Key n: (Descending)

- Die folgenden Angaben enthalten Schätzwerte für die Anzahl und die Größe der Zeilen, so dass die optimale Größe für den Sortierspeicher zur Laufzeit zugeordnet werden kann.

Sortheap Allocation Parameters:

| #Rows = n
 | Row Width = n

- Wenn lediglich die ersten Zeilen des sortierten Ergebnisses benötigt werden, wird Folgendes angezeigt:

Sort Limited To Estimated Row Count

- Für Sortiervorgänge in einer symmetrischen Mehrprozessorumgebung (SMP-Umgebung) wird die Art der durchzuführenden Sortierung durch eine der folgenden Angaben angezeigt:

Use Partitioned Sort
 Use Shared Sort
 Use Replicated Sort
 Use Round-Robin Sort

- Die folgenden Angaben zeigen an, ob das Ergebnis der Sortierung im Sortierspeicher verbleibt:

Piped

und

Not Piped

Wenn eine über Pipe geleitete Sortierung angegeben wird, behält der Datenbankmanager die sortierte Ausgabe im Speicher, anstatt das sortierte Ergebnis in eine andere temporäre Tabelle zu schreiben.

- Die folgende Angabe bedeutet, dass doppelte Werte während der Sortierung entfernt werden:

Eliminierung doppelter Werte

- Wenn Spaltenberechnungen in der Sortierung durchgeführt werden, wird dies durch eine der folgenden Angaben angezeigt:

Partial Aggregation
 Intermediate Aggregation
 Buffered Partial Aggregation
 Buffered Intermediate Aggregation

Abschließen der temporären Tabelle: Nach einem Tabellenzugriff, der eine verschobene (Pushdown) Operation zum Erstellen einer temporären Tabelle enthält (d. h. eine Operation zum Erstellen einer temporären Tabelle, die im Rahmen eines Tabellenzugriffs auftritt), folgt eine Abschlussangabe (Completion), die das Dateieinde verarbeitet, indem sie die temporäre Tabelle darauf vorbereitet, Zeilen für den nachfolgenden Zugriff auf die temporäre Tabelle zur Verfügung zu stellen. Es wird eine der folgenden Zeilen angezeigt:


```
Temp Table Completion ID = tn
Shared Temp Table Completion ID = tn
Sorted Temp Table Completion ID = tn
Sorted Shared Temp Table Completion ID = tn
```

Tabellenfunktionen

Tabellenfunktionen sind benutzerdefinierte Funktionen (UDFs), die Daten in Form einer Tabelle an die Anweisung zurückgeben. Tabellenfunktionen werden durch folgende Angaben angezeigt:

```
Access User Defined Table Function
| Name = schema.funkname
| Specific Name = spezifischername
| SQL Access Level = zugriffsebene
| Language = sprache
| Parameter Style = parmstil
| Fenced                               Not Deterministic
| Called on NULL Input                 Disallow Parallel
| Not Federated                       Not Threadsafe
```

Der spezifische Name identifiziert die aufgerufene Tabellenfunktion eindeutig. Die übrigen Zeilen geben Details über die Attribute der Funktion an.

Zugehörige Konzepte:

- „Beschreibung der Ausgabe von db2expln und dynexpln“ auf Seite 719
- „Beispiele für die Ausgabe von db2expln und dynexpln“ auf Seite 739

Joininformationen

Es gibt drei Arten von Joins:

- Hash-Join
- Mischjoin
- Join mit Verschachtelungsschleife (Nested Loop Join)

Wenn bei der Ausführung eines Abschnitts der Zeitpunkt kommt, an dem ein Join ausgeführt wird, wird eine der folgenden Angaben angezeigt:

```
Hash Join
Merge Join
Nested Loop Join
```

Es ist möglich, dass ein linker Outer Join durchgeführt wird. Ein linker Outer Join wird durch eine der folgenden Angaben angezeigt:

```
Left Outer Hash Join
Left Outer Merge Join
Left Outer Nested Loop Join
```

Bei Mischjoins und Joins mit Verschachtelungsschleife ist die äußere Tabelle des Joins die Tabelle, auf die in der vorherigen Zugriffsangabe, die in der Ausgabe angezeigt wird, verwiesen wird. Die innere Tabelle bei diesem Join ist die Tabelle, auf die in der Zugriffsangabe verwiesen wird, die sich im Bereich der Joinangabe befindet. Bei Hash-Joins sind die Zugriffsangaben umgekehrt, d. h. die äußere Tabelle ist im Joinbereich enthalten, und die innere Tabelle wird vor dem Join angezeigt.

Bei einem Hash- oder Mischjoin können folgende weitere Angaben auftreten:

- In einigen Fällen muss bei einem Join lediglich festgestellt werden, ob eine Zeile der inneren Tabelle mit der aktuellen Zeile in der äußeren Tabelle übereinstimmt. Dies wird durch folgende Angabe angezeigt:

Early Out: Single Match Per Outer Row

- Es ist möglich, nach Abschluss des Joins Vergleichselemente anzuwenden. Die Anzahl der Vergleichselemente, die angewandt werden, wird folgendermaßen angezeigt:

```
Residual Predicate(s)
| #Predicates = n
```

Bei einem Hash-Join können folgende weitere Angaben auftreten:

- Die Hashtabelle wird aus der inneren Tabelle erstellt. Wenn die Erstellung der Hashtabelle in ein Vergleichselement im Zugriff auf die innere Tabelle verschoben wurde, wird darauf durch die folgende Angabe im Zugriff der inneren Tabelle hingewiesen:

Process Hash Table For Join

- Beim Zugriff der äußeren Tabelle kann eine Prüftabelle erstellt werden, um die Leistung des Joins zu steigern. Auf die Erstellung der Prüftabelle wird durch die folgende Angabe im Zugriff der äußeren Tabelle hingewiesen:

Process Probe Table For Hash Join

- Die geschätzte erforderliche Anzahl Byte zum Erstellen der Hashtabelle wird durch folgende Angabe dargestellt:

Estimated Build Size: n

- Die geschätzte erforderliche Anzahl Byte für die Prüftabelle wird durch folgende Angabe dargestellt:

Estimated Probe Size: n

Bei einem Join mit Verschachtelungsschleife kann die folgende zusätzliche Angabe direkt nach der Joinangabe angezeigt werden:

Piped Inner

Diese Angabe bedeutet, dass die innere Tabelle des Joins das Ergebnis einer anderen Reihe von Operationen ist. Dies wird auch als eine *zusammengesetzte innere (composite inner)* Tabelle bezeichnet.

Wenn ein Join mehr als zwei Tabellen umfasst, müssen die EXPLAIN-Schritte von oben nach unten gelesen werden. Angenommen, die EXPLAIN-Ausgabe hat folgende Struktur:

```
Access ..... W
Join
| Access ..... X
Join
| Access ..... Y
Join
| Access ..... Z
```

Die Ausführung würde in diesem Fall in folgenden Schritten ablaufen:

1. Abrufen einer den Bedingungen entsprechenden Zeile aus W
2. Join der Zeile aus W mit der (nächsten) Zeile aus X und Benennung des Ergebnisses als P1 (für Jointeilergesamt Nr. 1)
3. Join von P1 mit der (nächsten) Zeile aus Y zum Erstellen von P2
4. Join von P2 mit der (nächsten) Zeile aus Z zum Erstellen einer vollständigen Ergebniszeile

5. Wenn weitere Zeilen in Z sind, weiter mit Schritt 4
6. Wenn weitere Zeilen in Y sind, weiter mit Schritt 3
7. Wenn weitere Zeilen in X sind, weiter mit Schritt 2
8. Wenn weitere Zeilen in W sind, weiter mit Schritt 1

Zugehörige Konzepte:

- „Beschreibung der Ausgabe von db2expln und dynexpln“ auf Seite 719
- „Beispiele für die Ausgabe von db2expln und dynexpln“ auf Seite 739

Datenstrominformationen

In einem Zugriffsplan ist es oft erforderlich, die Erstellung und den Fluss von Daten von einer Reihe von Operationen zur anderen zu steuern. Das Konzept des Datenstroms erlaubt es, eine Gruppe von Operationen innerhalb eines Zugriffsplans als Einheit zu steuern. Der Beginn eines Datenstroms wird durch folgende Anweisung gekennzeichnet:

Data Stream n

Hierbei ist n eine eindeutige Kennung, die zur leichteren Bezugnahme von db2expln zugeordnet wird. Das Ende des Datenstroms wird durch folgende Angabe gekennzeichnet:

End of Data Stream n

Alle Operationen zwischen diesen Angaben werden als Teil desselben Datenstroms angesehen.

Ein Datenstrom hat eine Anzahl von Merkmalen, und auf die einleitende Datenstromangabe können eine oder mehrere Angaben folgen, um diese Merkmale zu beschreiben:

- Wenn die Verarbeitung des Datenstroms von einem Wert abhängt, der früher im Zugriffsplan generiert wurde, wird der Datenstrom mit folgender Angabe markiert:

Correlated

- Ähnlich wie bei einer sortierten temporären Tabelle zeigen die folgenden Angaben, ob die Ergebnisse des Datenstroms im Speicher behalten werden:

Piped

und

Not Piped

Wie bei temporären Tabellen kann ein über eine Pipe geleiteter Datenstrom auf die Platte geschrieben werden, wenn zur Ausführungszeit zu wenig Speicher vorhanden ist. Der Zugriffsplan sieht beide Möglichkeiten vor.

- Die folgende Angabe bedeutet, dass nur ein einziger Satz aus diesem Datenstrom benötigt wird:

Single Record

Wenn auf einen Datenstrom zugegriffen wird, wird die folgende Angabe in der Ausgabe angezeigt:

Access Data Stream n

Zugehörige Konzepte:

- „Beschreibung der Ausgabe von db2expln und dynexpln“ auf Seite 719

- „Beispiele für die Ausgabe von db2expln und dynexpln“ auf Seite 739

Informationen zu INSERT, UPDATE und DELETE

Der EXPLAIN-Text für diese SQL-Anweisungen ist selbsterklärend. Nachfolgend sind mögliche Texte für diese SQL-Operationen dargestellt:

```
Insert: Table Name = schema.name ID = ts,n
Update: Table Name = schema.name ID = ts,n
Delete: Table Name = schema.name ID = ts,n
Insert: Hierarchy Table Name = schema.name ID = ts,n
Update: Hierarchy Table Name = schema.name ID = ts,n
Delete: Hierarchy Table Name = schema.name ID = ts,n
Insert: Materialized Query Table = schema.name ID = ts,n
Update: Materialized Query Table = schema.name ID = ts,n
Delete: Materialized Query Table = schema.name ID = ts,n
Insert: Global Temporary Table ID = ts, tn
Update: Global Temporary Table ID = ts, tn
Delete: Global Temporary Table ID = ts, tn
```

Zugehörige Konzepte:

- „Beschreibung der Ausgabe von db2expln und dynexpln“ auf Seite 719
- „Beispiele für die Ausgabe von db2expln und dynexpln“ auf Seite 739

Informationen zur Vorbereitung von Block- und Zeilen-IDs

Für einige Zugriffspläne ist es effizienter, wenn die den Bedingungen entsprechenden Satz-IDs (RIDs) und Block-IDs (BIDs) sortiert und mehrfach auftretende Werte entfernt werden (bei OR-Joins von Indizes) bzw. wenn eine Technik verwendet wird, mit der IDs, die in allen verwendeten Indizes auftreten (bei AND-Joins von Indizes), identifiziert werden, bevor der tatsächliche Zugriff auf die Tabelle erfolgt. Es gibt drei Hauptgründe für die Vorbereitung von IDs, auf die in den EXPLAIN-Angaben hingewiesen wird:

- Die beiden folgenden Angaben bedeuten, dass ein OR-Join von Indizes (Index ORing) zur Vorbereitung der Liste der qualifizierten IDs verwendet wird:

```
Index ORing Preparation
Block Index ORing Preparation
```

Index ORing (OR-Join von Indizes) bezeichnet ein Verfahren, bei dem mehrere Indexzugriffe vorgenommen und die Ergebnisse kombiniert werden, so dass die unterschiedlichen IDs, die mindestens in einem der Indizes auftreten, auf die zugegriffen wird, zusammengefasst werden. Das Optimierungsprogramm erwägt die Verwendung dieses Verfahrens, wenn Vergleichselemente durch Schlüsselwörter OR verknüpft werden oder ein Vergleichselement IN existiert. Die Zugriffe können auf den gleichen Index oder auf verschiedene Indizes erfolgen.

- Ein weiterer Grund für die Vorbereitung von IDs ist die Vorbereitung der Eingabedaten, die während des Vorabesezugriffs über Listen verwendet werden sollen, wie dies durch folgende Angaben angezeigt wird:

```
List Prefetch Preparation
Block List Prefetch RID Preparation
```

- *Index ANDing* (AND-Joins von Indizes) bezeichnet ein Verfahren, beim dem mehrere Indexzugriffe vorgenommen und die Ergebnisse kombiniert werden, so dass nur die IDs zusammengefasst werden, die in allen Indizes auftreten, auf die zugegriffen wird. Die folgenden Angaben zeigen an, dass die Verarbeitung eines AND-Joins von Indizes gestartet wird:

Index ANDing
Block Index ANDing

Wenn das Optimierungsprogramm die Größe der Ergebnismenge abgeschätzt hat, wird der Schätzwert mit der folgenden Angabe angezeigt:

Optimizer Estimate of Set Size: n

Die Filteroperationen bei AND-Joins von Indizes verarbeiten die IDs und verwenden Bit-Filtermethoden, um die IDs zu bestimmen, die in allen Indizes vorkommen, auf die zugegriffen wird. Die folgenden Angaben weisen darauf hin, dass IDs für AND-Joins von Indizes verarbeitet werden:

Index ANDing Bitmap Build Using Row IDs
Index ANDing Bitmap Probe Using Row IDs
Index ANDing Bitmap Build and Probe Using Row IDs
Block Index ANDing Bitmap Build Using Block IDs
Block Index ANDing Bitmap Build and Probe Using Block IDs
Block Index ANDing Bitmap Build and Probe Using Row IDs
Block Index ANDing Bitmap Probe Using Block IDs and Build Using Row IDs
Block Index ANDing Bitmap Probe Using Block IDs
Block Index ANDing Bitmap Probe Using Row IDs

Wenn das Optimierungsprogramm die Größe der Ergebnismenge für eine Bitmaske (Bitmap) abgeschätzt hat, wird der Schätzwert mit der folgenden Angabe angezeigt:

Optimizer Estimate of Set Size: n

Bei jeder Art von ID-Vorbereitung wird die Möglichkeit, dass ein Vorabesezugriff über Listen erfolgen kann, mit folgender Angabe angezeigt:

Prefetch: Enabled

Zugehörige Konzepte:

- „Beschreibung der Ausgabe von db2expln und dynexpln“ auf Seite 719
- „Beispiele für die Ausgabe von db2expln und dynexpln“ auf Seite 739

Informationen zu Spaltenberechnungen (Aggregation)

Spaltenberechnungen (Aggregation) werden für die Zeilen vorgenommen, die den etwaigen, in den Vergleichselementen der SQL-Anweisungen festgelegten Bedingungen entsprechen. Wenn eine Art von Spaltenfunktion auszuführen ist, wird eine der folgenden Angaben angezeigt:

Aggregation
Predicate Aggregation
Partial Aggregation
Partial Predicate Aggregation
Intermediate Aggregation
Intermediate Predicate Aggregation
Final Aggregation
Final Predicate Aggregation

Die Angabe *Predicate aggregation* besagt, dass die Spaltenberechnungsoperation zur Verarbeitung als Vergleichselement auf den Zeitpunkt, zu dem auf die Daten wirklich zugegriffen wird, verschoben wurde.

Unter jeder der oben aufgeführten Angaben über Spaltenberechnungen befindet sich eine Angabe, die die Art der durchgeführten Spaltenfunktion anzeigt:

```

Group By
Column Function(s)
Single Record

```

Die spezifische Spaltenfunktion kann von der ursprünglichen SQL-Anweisung abgeleitet werden. Ein einzelner Satz (Single Record) wird aus einem Index abgerufen, um eine Operation MIN oder MAX auszuführen.

Wenn eine Spaltenberechnung über Vergleichselemente verwendet wird, folgt auf die Angabe über den Tabellenzugriff, in der die Spaltenberechnung auftrat, eine Angabe über den Abschluss der Spaltenberechnung, die alle erforderlichen Verarbeitungsschritte bei Gruppenende bzw. bei Dateiende vornimmt. Dazu wird eine der folgenden Zeilen in der Ausgabe angezeigt:

```

Aggregation Completion
  Partial Aggregation Completion
  Intermediate Aggregation Completion
  Final Aggregation Completion

```

Zugehörige Konzepte:

- „Beschreibung der Ausgabe von db2expln und dynexpln“ auf Seite 719
- „Beispiele für die Ausgabe von db2expln und dynexpln“ auf Seite 739

Informationen zur Parallelverarbeitung

Für die parallele Ausführung einer SQL-Anweisung (entweder mit partitionsinterner oder partitionsübergreifender Parallelität) sind einige besondere Operationen erforderlich. Die Operationen für Parallelpläne werden im Folgenden beschrieben.

- Bei der Ausführung eines partitionsinternen Parallelplans werden Abschnitte des Plans gleichzeitig mit Hilfe verschiedener Subagenten ausgeführt. Die Erstellung der Subagenten wird durch folgende Angabe angezeigt:

```
Process Using n Subagents
```

- Bei der Ausführung eines partitionsübergreifenden parallelen Zugriffsplans wird der Abschnitt (Section) in mehrere Teilbereiche (Subsections) geteilt. Jeder Teilbereich wird zur Ausführung an einen oder mehrere Knoten gesendet. Ein wichtiger Teilbereich ist der *Koordinator Teilbereich*. Der Koordinator Teilbereich ist der erste Teilbereich in jedem Plan. Er erhält zunächst die Steuerung und ist dann für die Verteilung der anderen Teilbereiche und die Rückgabe von Ergebnissen an die aufrufende Anwendung zuständig.

Die Verteilung von Teilbereichen wird durch folgende Angabe angezeigt:

```
Distribute Subsection #n
```

Die Knoten, die einen Teilbereich empfangen, können durch eine von acht Methoden bestimmt werden:

- Die folgende Angabe bedeutet, dass der Teilbereich abhängig vom Wert der Spalten zu einem Knoten innerhalb der Datenbankpartitionsgruppe gesendet wird.

```

Directed by Hash
| #Columns = n
| Partition Map ID = n, Nodegroup = ngname, #Nodes = n

```

- Die folgende Angabe bedeutet, dass der Teilbereich an einen vorbestimmten Knoten gesendet wird. (Dies tritt häufig auf, wenn die Anweisung mit der Funktion NODENUMBER() arbeitet.)

```
Directed by Node Number
```

- Die folgende Angabe weist darauf hin, dass der Teilbereich an den Knoten gesendet wird, der einer vorbestimmten Datenbankpartitionsnummer in der angegebenen Datenbankpartitionsgruppe entspricht. (Dies tritt häufig auf, wenn die Anweisung mit der Funktion PARTITION() arbeitet.)
Directed by Partition Number
| Partition Map ID = n, Nodegroup = ngname, #Nodes = n
- Die folgende Angabe bedeutet, dass der Teilbereich an den Knoten gesendet wird, der die aktuelle Zeile für den Cursor der Anwendung verfügbar gemacht hat.
Directed by Position
- Die folgende Angabe bedeutet, dass nur ein einziger Knoten, der bei der Kompilierung der Anweisung festgelegt wird, den Teilbereich empfängt.
Directed to Single Node
| Node Number = n
- Die folgenden Angaben bedeuten, dass der Teilbereich auf dem Koordinator-knoten ausgeführt wird.
Directed to Application Coordinator Node
Directed to Local Coordinator Node
- Die folgende Angabe bedeutet, dass der Teilbereich an alle aufgeführten Knoten gesendet wird.
Broadcast to Node List
| Nodes = n1, n2, n3, ...
- Die folgende Angabe bedeutet, dass nur ein einziger Knoten, der bei der Ausführung der Anweisung festgelegt wird, den Teilbereich empfängt.
Directed to Any Node
- Tabellenwarteschlangen werden zum Versetzen von Daten zwischen Teilbereichen in einer Umgebung mit partitionierten Datenbanken oder zwischen Subagenten in einer symmetrischen Mehrprozessorumgebung (SMP) verwendet. Tabellenwarteschlangen werden folgendermaßen beschrieben:
 - Die folgenden Angaben zeigen an, dass Daten in eine Tabellenwarteschlange eingefügt werden:
Insert Into Synchronous Table Queue ID = qn
Insert Into Asynchronous Table Queue ID = qn
Insert Into Synchronous Local Table Queue ID = qn
Insert Into Asynchronous Local Table Queue ID = qn
 - Bei Tabellenwarteschlangen einer Datenbankpartition wird das Ziel für Zeilen, die in die Tabellenwarteschlange eingefügt werden, durch eine der folgenden Angaben angezeigt:
Alle Zeilen werden an den Koordinatorknoten gesendet:
Broadcast to Coordinator Node

Alle Zeilen werden an jede Datenbankpartition gesendet, auf der der angegebene Teilbereich ausgeführt wird:
Broadcast to All Nodes of Subsection n

Jede Zeile wird abhängig von den Werten in der Zeile an eine Datenbankpartition gesendet:
Hash to Specific Node

Jede Zeile wird an eine Datenbankpartition gesendet, die während der Ausführung der Anweisung bestimmt wird:
Send to Specific Node

Jede Zeile wird an einen zufällig bestimmten Knoten gesendet:

Send to Random Node

- In einigen Fällen ist es möglich, dass die Tabellenwarteschlange der Datenbankpartition einige Zeilen wegen Kapazitätsüberschreitung in eine temporäre Tabelle versetzen muss. Diese Möglichkeit wird durch folgende Angabe angezeigt:

Rows Can Overflow to Temporary Table

- Auf einen Tabellenzugriff, der eine verschobene Operation (Pushdown) zum Einfügen von Zeilen in eine Tabellenwarteschlange enthält, folgt eine Abschlussangabe (Completion), die die Zeilen handhabt, die nicht unmittelbar gesendet werden konnten. Dazu wird eine der folgenden Zeilen in der Ausgabe angezeigt:

```
Insert Into Synchronous Table Queue Completion ID = qn
Insert Into Asynchronous Table Queue Completion ID = qn
Insert Into Synchronous Local Table Queue Completion ID = qn
Insert Into Asynchronous Local Table Queue Completion ID = qn
```

- Die folgenden Angaben zeigen an, dass Daten aus einer Tabellenwarteschlange abgerufen werden:

```
Access Table Queue ID = qn
Access Local Table Queue ID = qn
```

Diesen Nachrichten folgt stets eine Angabe der Anzahl der Zeilen, die abgerufen werden.

#Columns = n

- Wenn die Tabellenwarteschlange die Zeilen auf der Empfängerseite sortiert, wird der Angabe über den Tabellenwarteschlangenzugriff außerdem eine der folgenden Nachrichten beigelegt:

```
Output Sorted
Output Sorted and Unique
```

Diesen Nachrichten folgt eine Angabe der Anzahl der Schlüsselspalten, die für die Sortierung verwendet wurden.

#Key Columns = n

Für jede Spalte im Sortierschlüssel wird eine der folgenden Angaben angezeigt:

```
Key n: (Ascending)
Key n: (Descending)
```

- Wenn Vergleichselemente auf der Empfängerseite der Tabellenwarteschlange auf die Zeilen angewendet werden, wird folgende Nachricht in der Ausgabe angezeigt:

```
Residual Predicate(s)
| #Predicates = n
```

- Einige Teilbereiche in Umgebungen mit partitionierten Datenbanken springen explizit zurück an den Start des Teilbereichs. Dies wird durch folgende Angabe angezeigt:

Jump Back to Start of Subsection

Zugehörige Konzepte:

- „Beschreibung der Ausgabe von db2expln und dynexpln“ auf Seite 719
- „Beispiele für die Ausgabe von db2expln und dynexpln“ auf Seite 739

Informationen zu Abfragen auf föderierte Datenbanken

Für die Ausführung einer SQL-Anweisung in einer föderierten Datenbank ist die Fähigkeit erforderlich, Teile der betreffenden Anweisung an anderen Datenquellen auszuführen.

Die folgende Angabe weist darauf hin, dass eine Datenquelle gelesen wird:

```
Ship Distributed Subquery #n
| #Columns = n
```

Es ist möglich, dass Vergleichselemente auf die Daten angewandt werden, die von der verteilten Unterabfrage (Distributed Subquery) zurückgegeben werden. Die Anzahl der Vergleichselemente, die angewandt werden, wird folgendermaßen angezeigt:

```
Residual Predicate(s)
| #Predicates = n
```

Eine INSERT-, UPDATE- oder DELETE-Operation, die an einer Datenquelle stattfindet, wird durch die entsprechende Angabe gezeigt:

```
Ship Distributed Insert #n
Ship Distributed Update #n
Ship Distributed Delete #n
```

Wenn eine Tabelle an einer Datenquelle explizit gesperrt wird, wird dies durch folgende Angabe gezeigt:

```
Ship Distributed Lock Table #n
```

DDL-Anweisungen für eine Datenquelle werden in zwei Teile aufgeteilt. Der Teil, der an der Datenquelle aufgerufen wird, wird wie folgt gezeigt:

```
Ship Distributed DDL Statement #n
```

Wenn der Server der föderierten Datenbanken eine partitionierte Datenbank ist, muss ein Teil der DDL-Anweisung auf dem Katalogknoten ausgeführt werden. Dies wird wie folgt angegeben:

```
Distributed DDL Statement #n Completion
```

Die Einzelheiten für die verteilten Unteranweisungen werden getrennt bereitgestellt. Die Optionen für verteilte Unteranweisungen werden nachfolgend beschrieben:

- Die Datenquelle für die Unterabfrage wird durch eine der folgenden Angaben gezeigt:

```
Server: servername (typ, version)
Server: servername (typ)
Server: servername
```

- Wenn die Datenquelle relational ist, wird das SQL für die Unteranweisung wie folgt dargestellt:

```
SQL Statement:
anweisung
```

Nicht relationale Datenquellen werden wie folgt angegeben:

```
Non-Relational Data Source
```

- Die Kurznamen (Nicknames), auf die in der Unteranweisung verwiesen wird, werden folgendermaßen aufgelistet:

```
Nicknames Referenced:
schema.kurzname ID = n
```

Wenn die Datenquelle relational ist, wird die Basistabelle für den Kurznamen wie folgt angegeben:

Base = basisschema.basistabelle

Wenn die Datenquelle nicht relational ist, wird die Quellendatei für den Kurznamen wie folgt angezeigt:

Source File = dateiname

- Wenn die Werte vom Server mit föderierten Datenbanken an die Datenquelle übergeben werden, bevor die Unteranweisung ausgeführt wird, wird die Zahl der Werte folgendermaßen gezeigt:

#Input Columns: n

- Wenn die Werte von der Datenquelle an den Server mit föderierten Datenbanken übergeben werden, nachdem die Unteranweisung ausgeführt wird, wird die Zahl der Werte folgendermaßen gezeigt:

#Output Columns: n

Zugehörige Konzepte:

- „Beschreibung der Ausgabe von db2expln und dynexpln“ auf Seite 719
- „Richtlinien zur Analyse, wo eine Abfrage für föderierte Datenbanken ausgewertet wird“ auf Seite 249

Verschiedene EXPLAIN-Informationen

- Abschnitte für Anweisungen der Datendefinitionssprache (DDL - Data Definition Language) werden in der Ausgabe folgendermaßen gekennzeichnet:

DDL Statement

Für DDL-Anweisungen wird keine weitere EXPLAIN-Ausgabe bereitgestellt.

- Abschnitte für SET-Anweisungen für die aktualisierbaren Sonderregister wie **CURRENT EXPLAIN SNAPSHOT** werden in der Ausgabe folgendermaßen gekennzeichnet:

SET Statement

Für SET-Anweisungen wird keine weitere EXPLAIN-Ausgabe bereitgestellt.

- Wenn die SQL-Anweisung die Klausel DISTINCT enthält, kann der folgende Text in der Ausgabe angezeigt werden:

Distinct Filter #Columns = n

Dabei ist n die Anzahl von Spalten, die beim Abrufen eindeutiger Zeilen verwendet wird. Zum Abrufen eindeutiger Zeilenwerte müssen die Zeilen geordnet sein, so dass gleiche Werte übersprungen werden können. Diese Angabe wird nicht angezeigt, wenn der Datenbankmanager gleiche Werte nicht explizit entfernen muss wie in folgenden Fällen:

- Es existiert ein eindeutiger Index, und alle Spalten im Indexschlüssel sind Teil der Operation für die Klausel DISTINCT.
- Die mehrfach auftretenden Werte können beim Sortieren entfernt werden.
- Die folgende Angabe wird angezeigt, wenn die nächste Operation von einer bestimmten Satz-ID (RID) abhängig ist:

Positioned Operation

Wenn die Positionierungsoperation an einer Datenquelle in einer föderierten Datenbankumgebung stattfindet, sieht die Angabe wie folgt aus:

Distributed Positioned Operation

Diese Angabe wird für jede SQL-Anweisung angezeigt, die die Syntax WHERE CURRENT OF verwendet.

- Die folgende Angabe wird angezeigt, wenn Vergleichselemente vorhanden sind, die auf das Ergebnis angewendet werden müssen, aber die nicht als Teil einer anderen Operation angewendet werden konnten (Restvergleichselemente):

```
Residual Predicate Application
| #Predicates = n
```

- Die folgende Angabe wird angezeigt, wenn ein Operator UNION in der SQL-Anweisung enthalten ist:

```
UNION
```

- Die folgende Angabe wird angezeigt, wenn sich im Zugriffsplan eine Operation befindet, deren einziger Zweck das Erstellen von Zeilenwerten zur Verwendung durch nachfolgende Operationen ist:

```
Table Constructor
| n-Row(s)
```

Table Constructors können verwendet werden, um Werte in einer Menge in eine Reihe von Zeilen umzuwandeln, die anschließend an nachfolgende Operationen übergeben werden. Wenn die nächste Zeile von einem Table Constructor angefordert wird, wird die folgende Angabe angezeigt:

```
Access Table Constructor
```

- Die folgende Angabe wird angezeigt, wenn eine Operation existiert, die nur unter bestimmten Bedingungen verarbeitet wird:

```
Conditional Evaluation
| Condition #n:
| #Predicates = n
| Action #n:
```

Durch bedingte Auswertung werden Aktivitäten wie die SQL-Anweisung CASE oder interne Mechanismen wie referenzielle Integritätsbedingungen oder Trigger implementiert. Wenn keine Aktion angezeigt wird, werden nur Datenbearbeitungsoperationen verarbeitet, wenn die Bedingung wahr ist.

- Eine der folgenden Angaben wird angezeigt, wenn eine Unterabfrage nach ALL, ANY oder EXISTS im Zugriffsplan verarbeitet wird:

- ANY/ALL Subquery
- EXISTS Subquery
- EXISTS SINGLE Subquery

- Vor bestimmten UPDATE- und DELETE-Operationen muss die Position einer speziellen Zeile in der Tabelle bestimmt werden. Dies wird durch folgende Angabe gekennzeichnet:

```
Establish Row Position
```

- Die folgende Angabe wird angezeigt, wenn Zeilen an die Anwendung zurückgegeben werden:

```
Return Data to Application
| #Columns = n
```

Wenn die Operation in einen Tabellenzugriff verschoben (Pushdown) wurde, wird eine Abschlussphase (Completion) erforderlich. Diese Phase wird wie folgt angezeigt:

```
Return Data Completion
```

- Die folgenden Informationen werden angezeigt, wenn eine gespeicherte Prozedur aufgerufen wird:

```

Call Stored Procedure
  Name = schema.funkname
  Specific Name = spezifischername
  SQL Access Level = zugriffsebene
  Language = sprache
  Parameter Style = parmstil
  Expected Result Sets = n
  Fenced                                Not Deterministic
  Called on NULL Input                  Disallow Parallel
  Not Federated                        Not Threadsafe

```

- Die folgenden Informationen werden angezeigt, wenn mindestens ein LOB-Querverweis freigegeben wird:
Free LOB Locators

Zugehörige Konzepte:

- „Beschreibung der Ausgabe von db2expln und dynexpln“ auf Seite 719
- „Beispiele für die Ausgabe von db2expln und dynexpln“ auf Seite 739

Beispiele für die Ausgabe von db2expln und dynexpln

Die folgenden Ausgabebeispiele zeigen die EXPLAIN-Informationen, die für bestimmte Abfragen in bestimmten Umgebungen erfasst werden.

Beispiele für die Ausgabe von db2expln und dynexpln

Zum besseren Verständnis finden Sie im Folgenden Beispiele, die den Aufbau und das Format der Ausgabe von db2expln und dynexpln verdeutlichen. Diese Beispiele wurden für die Beispieldatenbank SAMPLE ausgeführt, die mit DB2 ausgeliefert wird. Jedes Beispiel wird kurz erläutert. Die signifikanten Unterschiede von einem Beispiel zum nächsten wurden **fett** hervorgehoben.

Zugehörige Konzepte:

- „dynexpln“ auf Seite 258
- „Beispiel 8: Operator XISCAN“ auf Seite 752
- „Beispiel 5: Zugriffsplan für eine föderierte Datenbank“ auf Seite 747
- „Beispiel 4: Zugriffsplan für mehrere Partitionen mit partitionsübergreifender und partitionsinterner Parallelität“ auf Seite 745
- „Beispiel 1: keine Parallelität“ auf Seite 739
- „Beispiel 11: Operator XSCAN“ auf Seite 751
- „Beispiel 6: Operatoren XANDOR und XISCAN“ auf Seite 749
- „Beispiel 3: Zugriffsplan für mehrere Partitionen mit partitionsübergreifender Parallelität“ auf Seite 743
- „Beispiel 2: Zugriffsplan für Einzelpartition mit partitionsinterner Parallelität“ auf Seite 741

Zugehörige Referenzen:

- „db2expln - SQL- und XQuery-EXPLAIN“ auf Seite 712

Beispiel 1: keine Parallelität

In diesem Beispiel wird einfach eine Liste aller Namen von Mitarbeitern (employee), ihrer Aufgaben (job), ihrer Abteilungen (department) und Standorte (location) sowie der Namen der Projekte, an denen sie arbeiten, abgerufen. Das wesentliche Merkmal dieses Zugriffsplans besteht darin, dass die relevanten Daten aus allen

db2expln - SQL- und XQuery-EXPLAIN

angegebenen Tabellen durch Hash-Joins verknüpft werden. Da keine Indizes verfügbar sind, führt der Zugriffsplan beim Join einer Tabelle eine Tabellensuche in der jeweiligen Tabelle aus.

***** PACKAGE *****

Package Name = "DOOLE"."EXAMPLE" Version = ""

Prep Date = 2002/01/04

Prep Time = 14:05:00

Bind Timestamp = 2002-01-04-14.05.00.415403

Isolation Level = Cursor Stability

Blocking = Block Unambiguous Cursors

Query Optimization Class = 5

Partition Parallel = No

Intra-Partition Parallel = No

SQL Path = "SYSIBM", "SYSFUN", "SYSPROC", "DOOLE"

----- SECTION -----
Section = 1

SQL Statement:

DECLARE EMPCUR CURSOR

FOR

SELECT e.lastname, e.job, d.deptname, d.location, p.projname

FROM employee AS e, department AS d, project AS p

WHERE e.workdept = d.deptno AND e.workdept = p.deptno

Estimated Cost = 120.518692

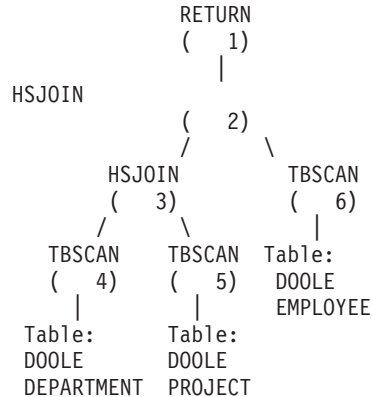
Estimated Cardinality = 221.535980

```
( 6) Access Table Name = DOOLE.EMPLOYEE ID = 2,5
    | #Columns = 3
    | Relation Scan
    | | Prefetch: Eligible
    | Lock Intents
    | | Table: Intent Share
    | | Row : Next Key Share
( 6) | Process Build Table for Hash Join
( 2) | Hash Join
    | Estimated Build Size: 7111
    | Estimated Probe Size: 9457
( 5) | Access Table Name = DOOLE.PROJECT ID = 2,7
    | #Columns = 2
    | Relation Scan
    | | Prefetch: Eligible
    | Lock Intents
    | | Table: Intent Share
    | | Row : Next Key Share
( 5) | Process Build Table for Hash Join
( 3) | Hash Join
    | Estimated Build Size: 5737
    | Estimated Probe Size: 6421
( 4) | Access Table Name = DOOLE.DEPARTMENT ID = 2,4
    | #Columns = 3
    | Relation Scan
    | | Prefetch: Eligible
    | Lock Intents
    | | Table: Intent Share
    | | Row : Next Key Share
( 4) | Process Probe Table for Hash Join
```

```
( 1) Return Data to Application
| #Columns = 5
```

End of section

Optimizer Plan:



Im ersten Teil des Plans wird auf die Tabellen DEPARTMENT und PROJECT zugegriffen, die über einen Hash-Join verknüpft werden. Das Ergebnis dieses Joins wird mit der Tabelle EMPLOYEE verknüpft. Die Ergebniszeilen werden an die Anwendung zurückgegeben.

Beispiel 2: Zugriffsplan für Einzelpartition mit partitions-interner Parallelität

In diesem Beispiel wird dieselbe SQL-Anweisung wie im ersten Beispiel gezeigt, jedoch wurde diese Abfrage für eine 4-Wege-SMP-Maschine kompiliert.

***** PACKAGE *****

Package Name = "DOOLE"."EXAMPLE" Version = ""

Prep Date = 2002/01/04

Prep Time = 14:12:38

Bind Timestamp = 2002-01-04-14.12.38.732627

Isolation Level = Cursor Stability

Blocking = Block Unambiguous Cursors

Query Optimization Class = 5

Partition Parallel = No

Intra-Partition Parallel = **Yes (Bind Degree = 4)**

SQL Path = "SYSIBM", "SYSFUN", "SYSPROC", "DOOLE"

----- SECTION -----
Section = 1

SQL Statement:

DECLARE EMPCUR CURSOR

FOR

```

SELECT e.lastname, e.job, d.deptname, d.location, p.projname
FROM employee AS e, department AS d, project AS p
WHERE e.workdept = d.deptno AND e.workdept = p.deptno

```

Intra-Partition Parallelism Degree = 4

db2expln - SQL- und XQuery-EXPLAIN

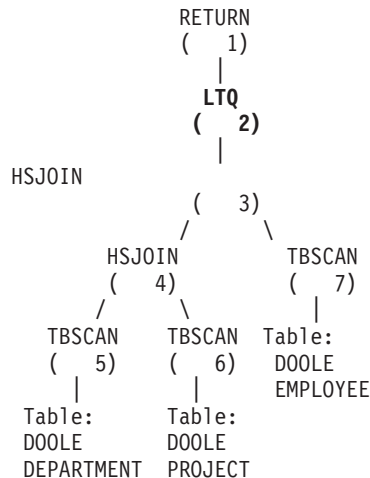
Estimated Cost = 133.934692

Estimated Cardinality = 221.535980

```
( 2) Process Using 4 Subagents
( 7) Access Table Name = DOOLE.EMPLOYEE ID = 2,5
      #Columns = 3
      Parallel Scan
      Relation Scan
      | Prefetch: Eligible
      Lock Intents
      | Table: Intent Share
      Row : Next Key Share
( 7) Process Build Table for Hash Join
( 3) Hash Join
      Estimated Build Size: 7111
      Estimated Probe Size: 9457
( 6) Access Table Name = DOOLE.PROJECT ID = 2,7
      #Columns = 2
      Parallel Scan
      Relation Scan
      | Prefetch: Eligible
      Lock Intents
      | Table: Intent Share
      Row : Next Key Share
( 6) Process Build Table for Hash Join
( 4) Hash Join
      Estimated Build Size: 5737
      Estimated Probe Size: 6421
( 5) Access Table Name = DOOLE.DEPARTMENT ID = 2,4
      #Columns = 3
      Parallel Scan
      Relation Scan
      | Prefetch: Eligible
      Lock Intents
      | Table: Intent Share
      Row : Next Key Share
( 5) Process Probe Table for Hash Join
( 2) Insert Into Asynchronous Local Table Queue ID = q1
( 2) Access Local Table Queue ID = q1 #Columns = 5
( 1) Return Data to Application
      | #Columns = 5
```

End of section

Optimizer Plan:



Dieser Plan stimmt mit dem Plan des ersten Beispiels weitgehend überein. Der Unterschied besteht in der Erstellung von vier Subagenten beim ersten Starten des Plans und der Tabellenwarteschlange am Ende des Plans, um die Ergebnisse der Arbeit aller Subagenten aufzunehmen, bevor sie an die Anwendung zurückgegeben werden.

Beispiel 3: Zugriffsplan für mehrere Partitionen mit partitionsübergreifender Parallelität

Dieses Beispiel zeigt wiederum dieselbe SQL-Anweisung wie das erste Beispiel, jedoch wurde hier die Abfrage auf einer partitionierten Datenbank kompiliert, die aus drei Datenbankpartitionen besteht.

***** PACKAGE *****

Package Name = "DOOLE"."EXAMPLE" Version = ""

Prep Date = 2002/01/04

Prep Time = 14:54:57

Bind Timestamp = 2002-01-04-14.54.57.033666

Isolation Level = Cursor Stability

Blocking = Block Unambiguous Cursors

Query Optimization Class = 5

Partition Parallel = Yes

Intra-Partition Parallel = No

SQL Path = "SYSIBM", "SYSFUN", "SYSPROC", "DOOLE"

----- SECTION -----

Section = 1

SQL Statement:

DECLARE EMPCUR CURSOR

FOR

SELECT e.lastname, e.job, d.deptname, d.location, p.projname

FROM employee AS e, department AS d, project AS p

WHERE e.workdept = d.deptno AND e.workdept = p.deptno

Estimated Cost = 118.483406

Estimated Cardinality = 474.720032

Coordinator Subsection:(-----) Distribute Subsection #2

| Broadcast to Node List

| Nodes = 10, 33, 55

(-----) Distribute Subsection #3

| Broadcast to Node List

| Nodes = 10, 33, 55

(-----) Distribute Subsection #1

| Broadcast to Node List

| Nodes = 10, 33, 55

(2) Access Table Queue ID = q1 #Columns = 5

(1) Return Data to Application

| #Columns = 5

Subsection #1:(8) Access Table Queue ID = q2 #Columns = 2

(3) Hash Join

| Estimated Build Size: 5737

| Estimated Probe Size: 8015

(6) Access Table Queue ID = q3 #Columns = 3

(4) Hash Join

db2expln - SQL- und XQuery-EXPLAIN

```

Estimated Build Size: 5333
Estimated Probe Size: 6421
( 5) Access Table Name = DOOLE.DEPARTMENT ID = 2,4
      #Columns = 3
      Relation Scan
      | Prefetch: Eligible
      Lock Intents
      | Table: Intent Share
      Row : Next Key Share
( 5) Process Probe Table for Hash Join
( 2) Insert Into Asynchronous Table Queue ID = q1
      Broadcast to Coordinator Node
      Rows Can Overflow to Temporary Table

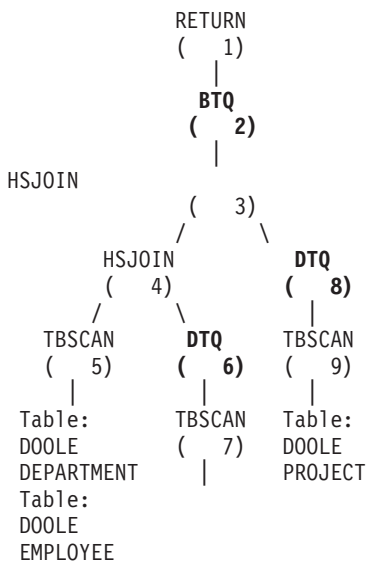
Subsection #2:
( 9) Access Table Name = DOOLE.PROJECT ID = 2,7
      #Columns = 2
      Relation Scan
      | Prefetch: Eligible
      Lock Intents
      | Table: Intent Share
      Row : Next Key Share
( 9) Insert Into Asynchronous Table Queue ID = q2
      Hash to Specific Node
      Rows Can Overflow to Temporary Tables
( 8) Insert Into Asynchronous Table Queue Completion ID = q2

Subsection #3:
( 7) Access Table Name = DOOLE.EMPLOYEE ID = 2,5
      #Columns = 3
      Relation Scan
      | Prefetch: Eligible
      Lock Intents
      | Table: Intent Share
      Row : Next Key Share
( 7) Insert Into Asynchronous Table Queue ID = q3
      Hash to Specific Node
      Rows Can Overflow to Temporary Tables
( 6) Insert Into Asynchronous Table Queue Completion ID = q3

```

End of section

Optimizer Plan:



Dieser Plan enthält dieselben Bestandteile wie der Plan im ersten Beispiel, aber der Bereich wurde in vier Teilbereiche (Subsection) unterteilt. Die Teilbereiche haben folgende Aufgaben:

- **Coordinator Subsection (Koordinator Teilbereich).** Dieser Teilbereich koordiniert die anderen Teilbereiche. In diesem Plan sorgt er dafür, dass die anderen Teilbereiche verteilt werden, und er verwendet anschließend eine Tabellenwarteschlange, um die Ergebnisse zu sammeln, die an die Anwendung zurückgegeben werden sollen.
- **Subsection #1.** Dieser Teilbereich durchsucht die Tabellenwarteschlange q2 und verwendet einen Hash-Join, um sie mit den Daten aus Tabellenwarteschlange q3 zu verknüpfen. Ein zweiter Hash-Join fügt anschließend die Daten aus der Tabelle DEPARTMENT hinzu. Die verknüpften Zeilen werden dann über Tabellenwarteschlange q1 an den Koordinator Teilbereich gesendet.
- **Subsection #2.** Dieser Teilbereich durchsucht die Tabelle PROJECT und verteilt sie per Hashverfahren mit den Ergebnissen an einen bestimmten Knoten. Diese Ergebnisse werden von Subsection #1 gelesen.
- **Subsection #3.** Dieser Teilbereich durchsucht die Tabelle EMPLOYEE und verteilt sie per Hashverfahren mit den Ergebnissen an einen bestimmten Knoten. Diese Ergebnisse werden von Subsection #1 gelesen.

Beispiel 4: Zugriffsplan für mehrere Partitionen mit partitionsübergreifender und partitionsinterner Parallelität

In diesem Beispiel wird dieselbe SQL-Anweisung wie im ersten Beispiel gezeigt, jedoch wurde die Abfrage auf einer partitionierten Datenbank mit drei Datenbankpartitionen kompiliert, die sich jeweils auf einer 4-Wege-SMP-Maschine befinden.

***** PACKAGE *****

Package Name = "DOOLE"."EXAMPLE" Version = ""

Prep Date = 2002/01/04

Prep Time = 14:58:35

Bind Timestamp = 2002-01-04-14.58.35.169555

Isolation Level = Cursor Stability
Blocking = Block Unambiguous Cursors
Query Optimization Class = 5

Partition Parallel = Yes
Intra-Partition Parallel = Yes (Bind Degree = 4)

SQL Path = "SYSIBM", "SYSFUN", "SYSPROC", "DOOLE"

----- SECTION -----
Section = 1

SQL Statement:

DECLARE EMPCUR CURSOR

FOR

SELECT e.lastname, e.job, d.deptname, d.location, p.projname
FROM employee AS e, department AS d, project AS p
WHERE e.workdept = d.deptno AND e.workdept = p.deptno

Intra-Partition Parallelism Degree = 4

Estimated Cost = 145.198898

Estimated Cardinality = 474.720032

db2expln - SQL- und XQuery-EXPLAIN

```
Coordinator Subsection:(-----) Distribute Subsection #2
| Broadcast to Node List
| | Nodes = 10, 33, 55
(-----) Distribute Subsection #3
| Broadcast to Node List
| | Nodes = 10, 33, 55
(-----) Distribute Subsection #1
| Broadcast to Node List
| | Nodes = 10, 33, 55
( 2) Access Table Queue ID = q1 #Columns = 5
( 1) Return Data to Application
| #Columns = 5

Subsection #1:( 3) Process Using 4 Subagents
( 10) Access Table Queue ID = q3 #Columns = 2
( 4) Hash Join
| Estimated Build Size: 5737
| Estimated Probe Size: 8015
( 7) Access Table Queue ID = q5 #Columns = 3
( 5) Hash Join
| Estimated Build Size: 5333
| Estimated Probe Size: 6421
( 6) Access Table Name = DOOLE.DEPARTMENT ID = 2,4
| #Columns = 3
| Parallel Scan
| Relation Scan
| | Prefetch: Eligible
| Lock Intents
| | Table: Intent Share
| | Row : Next Key Share
( 6) | Process Probe Table for Hash Join
( 3) | Insert Into Asynchronous Local Table Queue ID = q2
( 3) Access Local Table Queue ID = q2 #Columns = 5
( 2) Insert Into Asynchronous Table Queue ID = q1
| Broadcast to Coordinator Node
| Rows Can Overflow to Temporary Table

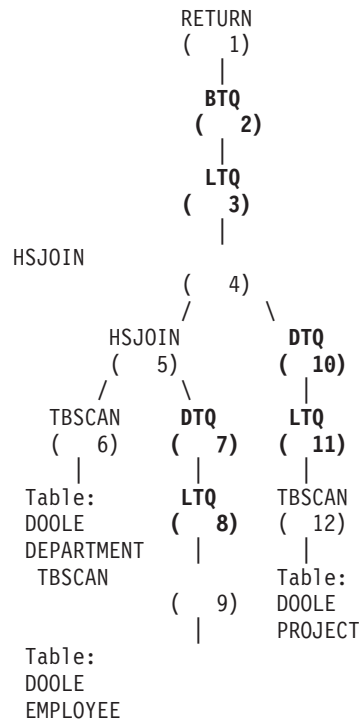
Subsection #2:
( 11) Process Using 4 Subagents
( 12) Access Table Name = DOOLE.PROJECT ID = 2,7
| #Columns = 2
| Parallel Scan
| Relation Scan
| | Prefetch: Eligible
| Lock Intents
| | Table: Intent Share
| | Row : Next Key Share
( 11) | Insert Into Asynchronous Local Table Queue ID = q4
( 11) Access Local Table Queue ID = q4 #Columns = 2
( 10) Insert Into Asynchronous Table Queue ID = q3
| Hash to Specific Node
| Rows Can Overflow to Temporary Tables

Subsection #3:
( 8) Process Using 4 Subagents
( 9) Access Table Name = DOOLE.EMPLOYEE ID = 2,5
| #Columns = 3
| Parallel Scan
| Relation Scan
| | Prefetch: Eligible
| Lock Intents
| | Table: Intent Share
| | Row : Next Key Share
( 8) | Insert Into Asynchronous Local Table Queue ID = q6
( 8) Access Local Table Queue ID = q6 #Columns = 3
( 7) Insert Into Asynchronous Table Queue ID = q5
| Hash to Specific Node
```

| Rows Can Overflow to Temporary Tables

End of section

Optimizer Plan:



Dieser Plan ist ähnlich wie der im dritten Beispiel, nur dass mehrere Subagenten jeden Teilbereich ausführen. Außerdem sammelt am Ende eines jeden Teilbereichs eine lokale Tabelle die Ergebnisse von allen Subagenten, bevor die entsprechenden Zeilen in die zweite Tabellenwarteschlange eingefügt werden, die per Hashverfahren an einen bestimmten Knoten gesendet wird.

Beispiel 5: Zugriffsplan für eine föderierte Datenbank

In diesem Beispiel wird dieselbe SQL-Anweisung wie im ersten Beispiel gezeigt. Die Abfrage wurde jedoch auf einer föderierten Datenbank kompiliert, bei der sich die Tabellen DEPARTMENT und PROJECT auf einer Datenquelle befinden und die Tabelle EMPLOYEE auf dem Server für die föderierten Datenbanken ist.

***** PACKAGE *****

Package Name = "DOOLE"."EXAMPLE" Version = ""

Prep Date = 2002/01/11

Prep Time = 13:52:48

Bind Timestamp = 2002-01-11-13.52.48.325413

Isolation Level = Cursor Stability

Blocking = Block Unambiguous Cursors

Query Optimization Class = 5

Partition Parallel = No

Intra-Partition Parallel = No

SQL Path = "SYSIBM", "SYSFUN", "SYSPROC", "DOOLE"

db2expln - SQL- und XQuery-EXPLAIN

----- SECTION -----
Section = 1

SQL Statement:

```
DECLARE EMPCUR CURSOR
FOR
  SELECT e.lastname, e.job, d.deptname, d.location, p.projname
  FROM employee AS e, department AS d, project AS p
  WHERE e.workdept = d.deptno AND e.workdept = p.deptno
```

Estimated Cost = 1804.625000
Estimated Cardinality = 112000.000000

```
( 7) Ship Distributed Subquery #2
| #Columns = 2
( 2) Hash Join
| Estimated Build Size: 48444
| Estimated Probe Size: 232571
( 6) Access Table Name = DOOLE.EMPLOYEE ID = 2,5
| #Columns = 3
| Relation Scan
| | Prefetch: Eligible
| Lock Intents
| | Table: Intent Share
| | Row : Next Key Share
( 6) Process Build Table for Hash Join
( 3) Hash Join
| Estimated Build Size: 7111
| Estimated Probe Size: 64606
( 4) Ship Distributed Subquery #1
| #Columns = 3
( 1) Return Data to Application
| #Columns = 5
```

Distributed Substatement #1:

(4) Server: REMOTE (DB2/UDB 8.1)

SQL Statement:

```
SELECT A0."DEPTNO", A0."DEPTNAME", A0."LOCATION"
FROM "DOOLE"."DEPARTMENT" A0
```

Nicknames Referenced:

```
DOOLE.DEPARTMENT ID = 32768
Base = DOOLE.DEPARTMENT
#Output Columns = 3
```

Distributed Substatement #2:

(7) Server: REMOTE (DB2/UDB 8.1)

SQL Statement:

```
SELECT A0."DEPTNO", A0."PROJNAME"
FROM "DOOLE"."PROJECT" A0
```

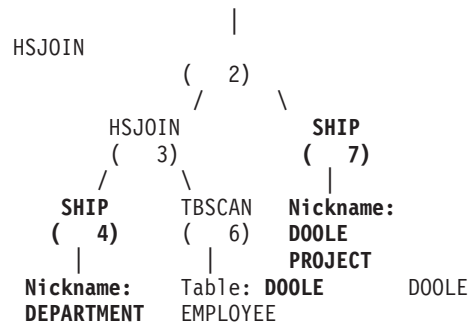
Nicknames Referenced:

```
DOOLE.PROJECT ID = 32769
Base = DOOLE.PROJECT
#Output Columns = 2
```

End of section

Optimizer Plan:

```
RETURN
( 1)
```

Dieser Plan enthält dieselben Bestandteile wie der Plan im ersten Beispiel. Ausgenommen hiervon sind die Daten für zwei der Tabellen, die von Datenquellen kommen. Auf diese beiden Tabellen wird über verteilte Unterabfragen zugegriffen, die in diesem Fall einfach alle Zeilen der betreffenden Tabellen auswählen. Wenn die Daten zum Server mit föderierten Datenbanken zurückgegeben werden, werden sie mit den Daten aus der lokalen Tabelle verknüpft.

Beispiel 6: Operatoren XANDOR und XISCAN

Dieses Beispiel veranschaulicht, wie der Operator XANDOR die XISCAN-Suchoperationen von zwei einzelnen Indizes für XML-Daten (*IDX1* und *IDX2*) kombiniert, die für dieselbe Tabelle *XISCANTABLE* definiert wurden.

IBM DB2 Database SQL Explain Tool

***** DYNAMIC *****

===== STATEMENT =====

Isolation Level = Cursor Stability
 Blocking = Block Unambiguous Cursors
 Query Optimization Class = 5

Partition Parallel = No
 Intra-Partition Parallel = No

SQL Path = "SYSIBM", "SYSFUN", "SYSPROC", "ATTALURI"

Query Statement:

```
xquery
for $c in db2-fn:xmlcolumn("XISCANTABLE.XMLCOL ")/a[@x="1" ]/b[@y=
"2" ] return $c
```

Section Code Page = 819

Estimated Cost = 192.266113

Estimated Cardinality = 1.800000

```
( 6) Index ANDing and ORing over XML
   Xpath is
   /child::element(a)[./child::element(b)
   /attribute::attribute(y)(:Index Search over XML 1:)
   and ./attribute::attribute(x)(:Index Search over XML 2:)]
   ]
   Index Search over XML 1
   Access Table Name = ATTALURI.XISCANTABLE
   Index Scan over XML: Name = ATTALURI.IDX1 ID = 6
   Physical Index over XML
   Index Columns:
   | 1: XMLCOL (Ascending)
   #Key Columns = 4
   Start Key: Inclusive Value
   1: ?
   2: ?
   3: ?
   4: ?
```

db2expln - SQL- und XQuery-EXPLAIN

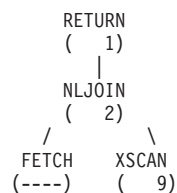
```

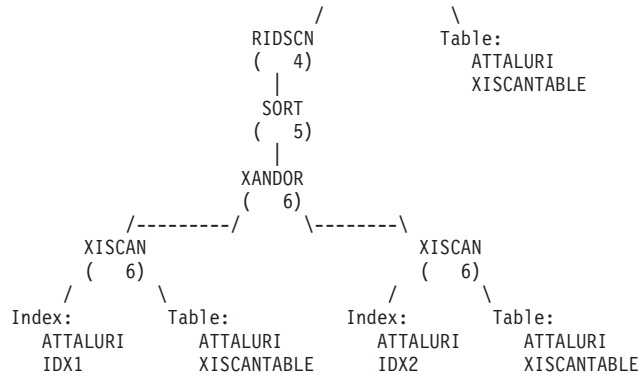
| | | | Stop Key: Inclusive Value
| | | | 1: ?
| | | | 2: ?
| | | | Index-Only Access
| | | | Index Prefetch: None
| | | | Isolation Level: Uncommitted Read
| | | | Lock Intents
| | | | Table: Intent None
| | | | Row : None
| | | | StopKey = StartKey
| | | | Value Start Key = ?
| | | | Index Search over XML 2
| | | | Access Table Name = ATTALURI.XISCANTABLE
| | | | Index Scan over XML: Name = ATTALURI.IDX2 ID = 4
| | | | Physical Index over XML
| | | | Index Columns:
| | | | 1: XMLCOL (Ascending)
| | | | #Key Columns = 4
| | | | Start Key: Inclusive Value
| | | | 1: ?
| | | | 2: ?
| | | | 3: ?
| | | | 4: ?
| | | | Stop Key: Inclusive Value
| | | | 1: ?
| | | | 2: ?
| | | | Index-Only Access
| | | | Index Prefetch: None
| | | | Isolation Level: Uncommitted Read
| | | | Lock Intents
| | | | Table: Intent None
| | | | Row : None
| | | | StopKey = StartKey
| | | | Value Start Key = ?
( 5) Insert Into Sorted Temp Table ID = t1
| #Columns = 1
| #Sort Key Columns = 1
| Key 1: (Ascending)
| Sortheap Allocation Parameters:
| #Rows = 2
| Row Width = 16
| Piped
| Duplicate Elimination
( 4) List Prefetch Preparation
( 4) Access Table Name = ATTALURI.XISCANTABLE ID = 2,16
| #Columns = 1
| Fetch Using Prefetched List
| Prefetch: Eligible
| Lock Intents
| Table: Intent Share
| Row : Next Key Share
( 2) Nested Loop Join
| Piped Inner
( 9) XML Doc Navigation
| Navigator is
| /fn:root($CONTEXT_NODE$())/child::element(a)(:#Xpath Predicates = 1:)
| [./child::element(b)(:Output nodeSeqRef :)]
| (::#Xpath Predicates = 1:)
| /attribute::attribute(y) and
| ./attribute::attribute(x)]
( 1) Iterate over XML sequence for Xquery bindout
( 1) Return Data to Application
| #Columns = 1

```

End of section

Optimizer Plan:





Beispiel 11: Operator XSCAN

In diesen Beispiel wird gezeigt, wie der Operator XSCAN im Zugriffsplan dargestellt werden kann. Dieser Operator verarbeitet Knotenverweise, die von einem Joinoperator (NLJOIN) mit verschachtelter Schleife übergeben wurden. Im Zugriffsplan wird dieses Element nicht mit einer direkten Eingabe dargestellt.

IBM DB2 Database SQL Explain Tool

***** DYNAMIC *****

===== STATEMENT =====

```

Isolation Level      = Cursor Stability
Blocking             = Block Unambiguous Cursors
Query Optimization Class = 5

Partition Parallel   = No
Intra-Partition Parallel = No

SQL Path              = "SYSIBM", "SYSFUN", "SYSPROC", "ATTALURI"

```

Query Statement:

```

xquery
for $b in db2-fn:xmlcolumn("XISCANTABLE.XMLCOL" )//book[position()<=
2] return $b

```

Section Code Page = 819

Estimated Cost = 779592.625000

Estimated Cardinality = 540000000.000000

```

( 4) Access Table Name = ATTALURI.XISCANTABLE ID = 2,16
    #Columns = 1
    Relation Scan
    | Prefetch: Eligible
    Lock Intents
    | Table: Intent Share
    Row : Next Key Share
( 3) Nested Loop Join
    Piped Inner
( 6) XML Doc Navigation
    | Navigator is
    | /fn:root($CONTEXT_NODE$())/descendant-or-self::node():Output nodeSeqRef :)
( 5) Nested Loop Join
    Piped Inner
( 11) XML Doc Navigation
    | Navigator is
    | /fn:root($CONTEXT_NODE$())/child::element(book):Output nodeSeqRef :)
( 10) Aggregation
    | Column Function(s)
( 9) Nested Loop Join
    Piped Inner
( 12) Unnest input XML sequence into stream of items with item number

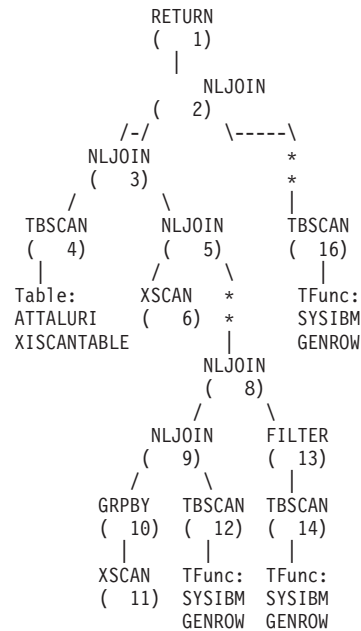
```

db2expln - SQL- und XQuery-EXPLAIN

```
( 8) | | Nested Loop Join
      | | Piped Inner
( 14) | | Table Constructor
      | | 1-Row(s)
( 2) | Nested Loop Join
      | Piped Inner
( 16) | Unnest input XML sequence into stream of items
( 1) | Iterate over XML sequence for Xquery bindout
( 1) | Return Data to Application
      | #Columns = 1
```

End of section

Optimizer Plan:



Beispiel 8: Operator XISCAN

Im vorliegenden Beispiel wird gezeigt, wie der Operator XISCAN den Index für XML-Daten *IDX1* durchsucht, der für die Tabelle *XISCANTABLE* definiert wurde.

IBM DB2 Database SQL Explain Tool

***** DYNAMIC *****

===== STATEMENT =====

```
Isolation Level           = Cursor Stability
Blocking                   = Block Unambiguous Cursors
Query Optimization Class = 5

Partition Parallel        = No
Intra-Partition Parallel = No

SQL Path                   = "SYSIBM", "SYSFUN", "SYSPROC", "ATTALURI"
```

Query Statement:

```
xquery
for $c in db2-fn:xmlcolumn("XISCANTABLE.XMLCOL ")/a[@x="1" ] return
$c
```

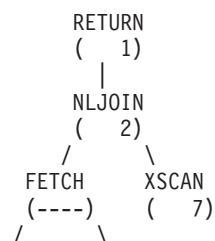
Section Code Page = 819

Estimated Cost = 1666.833862
 Estimated Cardinality = 18.000000

```
( 6) Access Table Name = ATTALURI.XISCANTABLE
    | Index Scan over XML: Name = ATTALURI.IDX1 ID = 4
    |   | Physical Index over XML
    |   | Index Columns:
    |   |   | 1: XMLCOL (Ascending)
    |   | #Key Columns = 2
    |   |   | Start Key: Inclusive Value
    |   |   |   | 1: ?
    |   |   |   | 2: ?
    |   |   | Stop Key: Inclusive Value
    |   |   |   | 1: ?
    |   |   |   | 2: ?
    |   | Index-Only Access
    |   | Index Prefetch: None
    |   | Isolation Level: Uncommitted Read
    |   | Lock Intents
    |   |   | Table: Intent None
    |   |   | Row : None
    |   | StopKey = StartKey
    |   | Value Start Key = ?
    |   | Xpath is
    |   |   | /child::element(a)/attribute::attribute(x)
( 5) Insert Into Sorted Temp Table ID = t1
    | #Columns = 1
    | #Sort Key Columns = 1
    |   | Key 1: (Ascending)
    | Sorthheap Allocation Parameters:
    |   | #Rows = 18
    |   | Row Width = 16
    | Piped
    | Duplicate Elimination
( 4) List Prefetch Preparation
( 4) Access Table Name = ATTALURI.XISCANTABLE ID = 2,16
    | #Columns = 1
    | Fetch Using Prefetched List
    |   | Prefetch: Eligible
    | Lock Intents
    |   | Table: Intent Share
    |   | Row : Next Key Share
( 2) Nested Loop Join
    | Piped Inner
( 7) XML Doc Navigation
    | Navigator is
    |   | /fn:root($CONTEXT_NODE$())/child::element(a)(:Output nodeSeqRef :)
    |   | (:#Xpath Predicates = 1:)
    |   | /attribute::attribute(x)
( 1) Iterate over XML sequence for Xquery bindout
( 1) Return Data to Application
    | #Columns = 1
```

End of section

Optimizer Plan:



db2expln - SQL- und XQuery-EXPLAIN

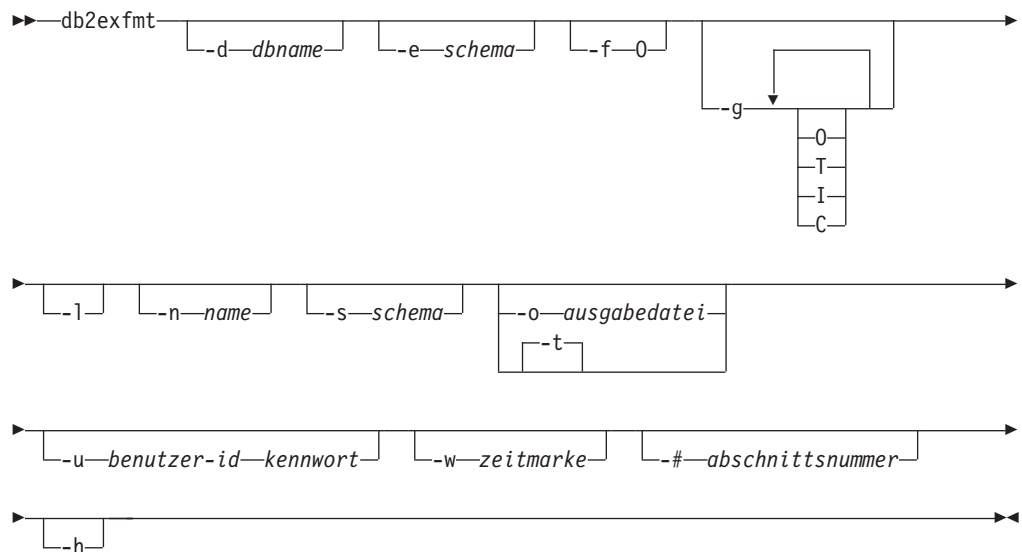
```
      RIDSCN      Table:
      (    4)     ATTALURI
      |           XISCANTABLE
      |
      SORT
      (    5)
      |
      XISCAN
      (    6)
    /      \
Index:      Table:
ATTALURI    ATTALURI
IDX1        XISCANTABLE
```

Anhang F. db2exfmt - EXPLAIN-Tabellen formatieren

Mit dem Tool db2exfmt können Sie den Inhalt der EXPLAIN-Tabellen formatieren. Dieses Tool befindet sich im Unterverzeichnis misc des Verzeichnisses sqllib für Ihre Instanz.

Sie müssen über Lesezugriff auf die zu formatierenden EXPLAIN-Tabellen verfügen, um dieses Tool verwenden zu können.

Befehlssyntax:



Befehlsparameter:

- d dbname**
Name der Datenbank mit den Paketen
- e schema**
SQL-Schema der EXPLAIN-Tabelle
- f**
Formatierungsmarkierungen. In diesem Release wird nur der Wert 0 (Operatorzusammenfassung) unterstützt.
- g**
Diagrammzugriffsplan. Wenn nur -g angegeben wird, wird ein Diagramm gefolgt von den Formatierungsangaben für alle Tabellen generiert. Ansonsten kann eine beliebige Kombination der folgenden gültigen Werte angegeben werden:
 - O** Nur ein Diagramm wird generiert. Der Tabelleninhalt wird nicht formatiert.
 - T** In das Diagramm wird der Gesamtaufwand unter jedem Operator aufgenommen.
 - I** In das Diagramm wird der Ein-/Ausgabebefwand unter jedem Operator aufgenommen.
 - C** In das Diagramm wird die erwartete Ausgabekardinalität (Anzahl Tupel) für jeden Operator aufgenommen.
- l**
Die Groß-/Kleinschreibung wird während der Verarbeitung der Paketnamen beachtet.

- n** *name*
Name für die Quelle der EXPLAIN-Anforderung (SOURCE_NAME)
- s** *schema*
SQL-Schema oder Qualifikationsmerkmal der Quelle der EXPLAIN-Anforderung (SOURCE_SCHEMA)
- o** *ausgabedatei*
Name der Ausgabedatei
- t**
Leiten der Ausgabe an die Workstation
- u** *benutzer-id kennwort*
Mit dieser Option werden eine Benutzer-ID und ein Kennwort für die Verbindung zu einer Datenbank angegeben.

Sowohl die Benutzer-ID als auch das Kennwort müssen gemäß den Namenskonventionen gültig sein und von der Datenbank erkannt werden.
- w** *zeitmarke*
EXPLAIN-Zeitmarke. Geben Sie -1 an, um die letzte EXPLAIN-Anforderung abzurufen.
- #** *abschnittsnummer*
Abschnittsnummer in der Quelle. Geben Sie null an, um alle Abschnitte anzufordern.
- h**
Anzeigen des Hilfetexts. Wenn diese Option angegeben wird, werden alle anderen Optionen ignoriert, und lediglich der Hilfetext wird angezeigt.

Hinweise:

Sie werden aufgefordert, die nicht angegebenen Parameterwerte anzugeben, oder Ihnen wird mitgeteilt, dass nicht alle angegeben wurden, ausgenommen im Fall der Optionen -h und -l.

Wenn kein SQL-Schema für die EXPLAIN-Tabelle angegeben wird, wird standardmäßig der Wert der Umgebungsvariablen **USER** verwendet. Wenn diese Variable nicht gefunden wird, wird der Benutzer zur Eingabe eines SQL-Schemas für die EXPLAIN-Tabelle aufgefordert.

Quellenname, SQL-Schema der Quelle und die EXPLAIN-Zeitmarke können in Form eines LIKE-Vergleichselements angegeben werden, so dass das Prozentzeichen (%) und das Unterstreichungszeichen (_) als Platzhalterzeichen verwendet werden können, um mehrere Quellen mit einem Aufruf auszuwählen. Für die letzte mit EXPLAIN bearbeitete Anweisung kann die EXPLAIN-Zeit als -1 angegeben werden.

Wenn -o ohne Dateiname und -t nicht angegeben wird, wird der Benutzer aufgefordert, einen Dateinamen anzugeben (der Standardname ist db2exfmt.out). Wenn weder -o noch -t angegeben wird, wird der Benutzer aufgefordert, einen Dateinamen anzugeben (die Standardoption ist Terminalausgabe). Wenn sowohl -o als auch -t angegeben werden, wird die Ausgabe an das Terminal geleitet.

Zugehörige Konzepte:

- „EXPLAIN-Tools“ auf Seite 258
- „Richtlinien zur Erfassung von EXPLAIN-Informationen“ auf Seite 269
- „Richtlinien zur Verwendung von EXPLAIN-Informationen“ auf Seite 260

Anhang G. Verwenden des Windows-Systemmonitors

Windows-Systemmonitor - Einführung

Wenn Sie mit dem DB2-Datenbankmanager für Windows arbeiten, stehen Ihnen Tools zur Überwachung der Leistung zur Verfügung:

- **DB2 Performance Expert**

DB2 Performance Expert for Multiplatforms Version 1.1 dient zur Konsolidierung, Dokumentation, Analyse und Empfehlung von Änderungen zur Selbstverwaltung und zur Ressourcenoptimierung auf der Grundlage leistungsbezogener DB2-Datenbankinformationen.

- **DB2-Diagnosezentrale**

Die Funktionen der Diagnosezentrale stellen Ihnen verschiedene Methoden zur Arbeit mit leistungsbezogenen Informationen zur Verfügung. Diese Funktionen ersetzen in gewissen Rahmen die Funktionen des Performance Monitor der Steuerzentrale.

- **Windows-Systemmonitor**

Mit dem Windows-Systemmonitor können Sie die Datenbank- und Systemleistung überwachen und dabei Informationen von allen Leistungsdatenquellen abrufen, die im System registriert sind. Windows bietet auch Leistungsinformationen zu allen Aspekten des Computerbetriebs, darunter:

- CPU-Auslastung
- Speicherauslastung
- Plattenaktivität
- Netzwerkaktivität

Zugehörige Tasks:

- „Zugreifen auf Leistungsinformationen ferner DB2-Datenbanken“ auf Seite 760
- „Anzeigen von Leistungswerten der DB2-Datenbank und von DB2 Connect“ auf Seite 759
- „Aktivieren des Fernzugriffs auf DB2-Leistungsinformationen“ auf Seite 758
- „Registrieren von DB2 im Windows-Systemmonitor“ auf Seite 757
- „Zurücksetzen von DB2-Leistungswerten“ auf Seite 761

Zugehörige Referenzen:

- „Windows-Leistungsobjekte“ auf Seite 759

Registrieren von DB2 im Windows-Systemmonitor

Vorgehensweise:

Das Konfigurationsprogramm registriert DB2 automatisch im Windows-Systemmonitor.

Um die Leistungsdaten der DB2-Datenbank und von DB2 Connect für den Windows-Systemmonitor verfügbar zu machen, müssen Sie die DLL für die Leistungszähler von DB2 für Windows registrieren. Dadurch können andere Windows-Anwendungen mit den Win32-Leistungs-APIs Leistungsdaten abrufen.

Geben Sie zum Installieren und Registrieren der Leistungszähler-DLL von DB2 für Windows (DB2Perf.DLL) im Windows-Systemmonitor folgenden Befehl ein:

```
db2perfi -i
```

Durch das Registrieren der DLL wird ein neuer Schlüssel in der Option **Services** der Registrierdatenbank erstellt. Ein Eintrag gibt den Namen der DLL an, die die Zählerunterstützung bietet. Drei weitere Einträge geben Namen von Funktionen an, die in der DLL enthalten sind. Dies sind folgende Funktionen:

- **Open**
Wird aufgerufen, wenn die DLL zuerst vom System in einem Prozess geladen wird.
- **Collect**
Wird aufgerufen, um Leistungsinformationen von der DLL anzufordern.
- **Close**
Wird aufgerufen, wenn die DLL entladen wird.

Zugehörige Referenzen:

- „db2perfi - Performance counters registration utility command“ in *Command Reference*

Aktivieren des Fernzugriffs auf DB2-Leistungsinformationen

Vorgehensweise:

Wenn Ihre Workstation unter DB2 für Windows mit anderen Windows-Computern vernetzt ist, können Sie die in diesem Abschnitt beschriebene Funktion verwenden.

Damit Windows-Leistungsobjekte von einem anderen DB2 für Windows-Computer angezeigt werden können, müssen Sie einen Administratorbenutzernamen und ein Administratorkennwort im DB2-Datenbankmanager registrieren. (Der Standardbenutzername des Windows-Systemmonitors, **SYSTEM**, ist in der DB2-Datenbank ein reserviertes Wort und darf nicht verwendet werden.) Geben Sie folgenden Befehl ein, um den Namen zu registrieren:

```
db2perfr -r benutzername kennwort
```

Anmerkung: Der verwendete benutzername muss den Namenskonventionen für DB2-Datenbanken entsprechen.

Die Angaben benutzername und kennwort sind in einem Schlüssel in der Registrierdatenbank enthalten. Der Zugriff auf diese ist nur Administratoren und dem Konto SYSTEM gestattet. Die Daten sind codiert, um Sicherheitsprobleme beim Speichern eines Administratorkennworts in der Registrierdatenbank zu vermeiden.

Anmerkungen:

1. Sobald eine Kombination aus Benutzername und Kennwort im DB2-Datenbanksystem registriert wurde, melden sich auch lokale Instanzen des Performance Monitor explizit mit diesem Benutzernamen und Kennwort an. Dies bedeutet, dass bei lokalen Sitzungen des Performance Monitor keine DB2-Datenbankleistungsdaten angezeigt werden, wenn die beim DB2-Datenbanksystem registrierten Informationen zum Benutzernamen nicht übereinstimmen.
2. Die Kombination aus Benutzername und Kennwort muss so gepflegt werden, dass die Werte für den Benutzernamen und das Kennwort den in der Sicherheitsdatenbank von Windows gespeicherten Werten entsprechen.

Wenn der Benutzername oder das Kennwort in der Sicherheitsdatenbank von Windows geändert wird, muss auch die Kombination aus Benutzername und Kennwort für die ferne Leistungsüberwachung neu definiert werden.

3. Geben Sie den folgenden Befehl ein, um die Registrierung zurückzunehmen:

```
db2perfr -u <benutzername> <kennwort>
```

Zugehörige Konzepte:

- „Allgemeine Namenskonventionen“ in *Systemverwaltung: Implementierung*

Zugehörige Referenzen:

- „db2perfr - Performance monitor registration tool command“ in *Command Reference*

Anzeigen von Leistungswerten der DB2-Datenbank und von DB2 Connect

Vorgehensweise:

Wenn Sie Leistungswerte für die DB2-Datenbank und DB2 Connect mit dem Performance Monitor anzeigen möchten, wählen Sie einfach die Leistungszähler, deren Werte angezeigt werden sollen, im Feld **Hinzufügen zu** aus. Dieses Fenster zeigt eine Liste von Leistungsobjekten für die Leistungsdaten an. Wählen Sie ein Objekt aus, um eine Liste der Datenquellen anzuzeigen, die es bereitstellt.

Ein Leistungsobjekt kann auch mehrere Instanzen haben. Beispielsweise bietet das Objekt **Logische Platte** Zähler wie „% Lesezeit“ und „Bytes gelesen/s“. Es hat zudem eine Instanz für jedes logische Laufwerk des Computers, einschließlich „C:“ und „D:“.

Zugehörige Konzepte:

- „Windows-Systemmonitor - Einführung“ auf Seite 757

Zugehörige Referenzen:

- „Windows-Leistungsobjekte“ auf Seite 759

Windows-Leistungsobjekte

Windows stellt die folgenden Leistungsobjekte bereit:

- **DB2-Datenbankmanager (Database Manager)**

Dieses Objekt bietet allgemeine Informationen für eine einzige Windows-Instanz. Die überwachte DB2-Datenbankinstanz wird als Objektinstanz angezeigt.

Aus praktischen Gründen sowie aus Leistungsgründen können Sie Leistungsdaten jeweils nur von einer DB2-Datenbankinstanz abrufen. Die DB2-Datenbankinstanz, die im Performance Monitor angezeigt wird, wird von der Registrierdatenbankvariablen `db2instance` im Prozess des Performance Monitor gesteuert. Wenn mehrere DB2-Datenbankinstanzen gleichzeitig ausgeführt werden und Sie zu mehreren Instanzen die Leistungsdaten anzeigen möchten, müssen Sie für jede zu überwachende Instanz eine separate Sitzung des Performance Monitor starten. Dabei muss `db2instance` auf den entsprechenden Wert für die zu überwachende DB2-Datenbankinstanz gesetzt sein.

Wenn Sie mit einer Umgebung mit partitionierten Datenbanken arbeiten, können Leistungsdaten immer nur von einem Datenbankpartitionsserver abgerufen werden. Standardmäßig werden die Leistungsdaten für die Standarddatenbankpartition (die Datenbankpartition, der der logische Port 0 zugeordnet ist) angezeigt. Um die Leistungsdaten einer anderen Datenbankpartition anzuzeigen, müssen Sie eine separate Sitzung des Performance Monitor starten und für die Umgebungsvariable DB2NODE die Datenbankpartitionsnummer der zu überwachenden Datenbankpartition definieren.

- **DB2-Datenbanken (DB2 Databases)**

Dieses Objekt bietet Informationen zu einer bestimmten Datenbank. Informationen sind für jede momentan aktive Datenbank verfügbar.

- **DB2-Anwendungen (DB2 Applications)**

Dieses Objekt bietet Informationen zu einer bestimmten DB2-Datenbankanwendung. Informationen sind für jede derzeit aktive DB2-Datenbankanwendung verfügbar.

- **DB2-DCS-Datenbanken (DB2 DCS Databases)**

Dieses Objekt bietet Informationen zu einer bestimmten DCS-Datenbank. Informationen sind für jede momentan aktive Datenbank verfügbar.

- **DB2-DCS-Anwendungen (DB2 DCS Applications)**

Dieses Objekt bietet Informationen zu einer bestimmten DB2-DCS-Anwendung. Informationen sind für jede derzeit aktive DB2-DCS-Anwendung verfügbar.

Welche dieser Objekte im Windows-Systemmonitor aufgelistet werden, ist davon abhängig, was auf Ihrem Windows-Computer installiert ist und welche Anwendungen aktiv sind. Wenn zum Beispiel der DB2-Datenbankmanager installiert ist und gestartet wurde, wird das Objekt für den DB2-Datenbankmanager aufgeführt. Wenn auch einige DB2-Datenbanken und -Anwendungen auf diesem Computer momentan aktiv sind, werden auch die Objekte für DB2-Datenbanken und DB2-Anwendungen aufgeführt. Wenn Sie Ihr Windows-System als DB2 Connect-Gateway verwenden und einige DCS-Datenbanken und -Anwendungen momentan aktiv sind, werden die Objekte für DB2-DCS-Datenbanken und DB2-DCS-Anwendungen aufgeführt.

Zugreifen auf Leistungsinformationen ferner DB2-Datenbanken

Vorgehensweise:

Die Aktivierung des Fernzugriffs auf DB2-Leistungsinformationen wurde bereits in einem früheren Abschnitt behandelt. Wählen Sie im Fenster **Diagramm erweitern** einen anderen Computer zur Überwachung aus. Dadurch wird eine Liste aller verfügbaren Leistungsobjekte auf diesem Computer angezeigt.

Damit Sie ein DB2-Leistungsobjekt auf einem fernen Computer überwachen können, muss die Stufe des auf diesem Computer installierten DB2-Datenbank- oder DB2 Connect-Codes Version 6 oder höher sein.

Zugehörige Konzepte:

- „Windows-Systemmonitor - Einführung“ auf Seite 757

Zurücksetzen von DB2-Leistungswerten

Vorgehensweise:

Wenn eine Anwendung die Überwachungs-APIs von DB2 aufruft, werden normalerweise kumulierte Werte seit dem Start des DB2-Datenbankservers zurückgegeben. Oft ist es jedoch sinnvoll, folgende Aktionen auszuführen:

- Zurücksetzen von Leistungswerten
- Ausführen eines Tests
- Erneutes Zurücksetzen der Werte
- Erneutes Ausführen des Tests

Verwenden Sie zum Zurücksetzen der Datenbankleistungswerte das Programm **db2perf**. Geben Sie folgenden Befehl ein:

```
db2perf
```

Standardmäßig werden dadurch die Leistungswerte für alle aktiven DB2-Datenbanken zurückgesetzt. Sie können jedoch auch eine Liste der Datenbanken angeben, die zurückgesetzt werden sollen. Sie können auch den Parameter `-d` verwenden, um anzugeben, dass Leistungswerte für DCS-Datenbanken zurückgesetzt werden sollen. Beispiel:

```
db2perf
db2perf dbalias1 dbalias2 ... dbaliasn

db2perf -d
db2perf -d dbalias1 dbalias2 ... dbaliasn
```

Im ersten Beispiel werden die Leistungswerte für alle aktiven DB2-Datenbanken zurückgesetzt. Im zweiten Beispiel werden die Leistungswerte für bestimmte DB2-Datenbanken zurückgesetzt. Im dritten Beispiel werden die Leistungswerte für alle aktiven DB2-DCS-Datenbanken zurückgesetzt. Im letzten Beispiel werden die Leistungswerte für bestimmte DB2-DCS-Datenbanken zurückgesetzt.

Das Programm **db2perf** setzt die Werte für ALLE Programme zurück, die derzeit auf Datenbankleistungsdaten für die entsprechende DB2-Datenbankserverinstanz zugreifen (d. h., die in DB2INSTANCE definierte Instanz in der Sitzung, in der Sie **db2perf** ausführen).

Durch den Aufruf von **db2perf** werden auch die Werte zurückgesetzt, die bei jedem Fernzugriff auf DB2-Datenbankleistungsdaten mit dem Befehl **db2perf** angezeigt werden.

Anmerkung: Es gibt eine DB2-Datenbank-API, `sqlmrset`, mit der eine Anwendung nur die lokal (nicht global) angezeigten Werte für bestimmte Datenbanken zurücksetzen kann.

Zugehörige Referenzen:

- „db2perf - Reset database performance values command“ in *Command Reference*
- „db2ResetMonitor API - Reset the database system monitor data“ in *Administrative API Reference*

Anhang H. Technische Informationen zu DB2-Datenbanken

Übersicht über technische Informationen zu DB2

Die technischen Informationen zu DB2 stehen über die folgenden Tools und Methoden zur Verfügung:

- DB2-Informationszentrale
 - Themen
 - Hilfe für DB2-Tools
 - Beispielprogramme
 - Lernprogramme
- DB2-Bücher
 - PDF-Dateien (für den Download verfügbar)
 - PDF-Dateien (auf der DB2-PDF-CD)
 - Gedruckte Bücher
- Befehlszeilenhilfe
 - Hilfe für Befehle
 - Hilfe für Nachrichten
- Beispielprogramme

IBM stellt in regelmäßigen Abständen Dokumentationsaktualisierungen zur Verfügung. Wenn Sie auf die Onlineversion der DB2-Informationszentrale unter ibm.com zugreifen, müssen Sie die Dokumentationsaktualisierungen nicht installieren, da diese Version von IBM auf dem neuesten Stand gehalten wird. Wenn Sie die DB2-Informationszentrale installiert haben, sollten Sie die Dokumentationsaktualisierungen installieren. Diese Dokumentationsaktualisierungen ermöglichen Ihnen, die Informationen, die Sie von der CD mit der *DB2-Informationszentrale* installiert oder von Passport Advantage heruntergeladen haben, auf den neuesten Stand zu bringen, sobald neue Informationen verfügbar sind.

Anmerkung: Die Themen der DB2-Informationszentrale werden häufiger aktualisiert als die PDF- und Hardcopybücher. Um stets die neuesten Informationen zur Verfügung zu haben, sollten Sie die Dokumentationsaktualisierungen installieren, sobald diese verfügbar sind, oder die DB2-Informationszentrale unter ibm.com aufrufen.

Darüber hinaus können Sie auf zusätzliche technische Informationen zu DB2, wie beispielsweise technische Hinweise (Technotes), White Papers und Redbooks online über ibm.com zugreifen. Rufen Sie die Website 'DB2 Information Management - Software - Library' unter <http://www.ibm.com/software/data/sw-library/> auf.

Feedback zur Dokumentation

Senden Sie uns Ihr Feedback zur DB2-Dokumentation! Wenn Sie Anregungen zur Verbesserung der DB2-Dokumentation haben, senden Sie eine E-Mail an db2docs@ca.ibm.com. Das DB2-Dokumentationsteam bearbeitete das gesamte Feedback, kann jedoch nicht im Einzelnen auf Ihre E-Mails antworten. Nennen Sie uns, wenn möglich, konkrete Beispiele, sodass wir die Problemstellung besser beurteilen können. Wenn Sie uns Feedback zu einem bestimmten Thema oder einer bestimmten Hilfedatei senden, geben Sie den entsprechenden Titel sowie die URL an.

Verwenden Sie diese E-Mail-Adresse nicht, wenn Sie sich an die DB2-Kundenunterstützung wenden möchten. Wenn ein technisches Problem bei DB2 vorliegt, das Sie mit Hilfe der Dokumentation nicht beheben können, fordern Sie beim zuständigen IBM Service-Center Unterstützung an.

Zugehörige Konzepte:

- „Funktionen der DB2-Informationszentrale“ in der *DB2-Informationszentrale (Online)*
- „Sample files“ in den *Themen mit Beispielen*

Zugehörige Tasks:

- „Aufrufen der Hilfe für Befehle über den Befehlszeilenprozessor“ in *Command Reference*
- „Aufrufen der Hilfe für Nachrichten über den Befehlszeilenprozessor“ in *Command Reference*
- „Aktualisieren der auf Ihrem Computer oder Intranet-Server installierten DB2-Informationszentrale“ auf Seite 769

Zugehörige Referenzen:

- „Bibliothek mit technischen Informationen zu DB2 im PDF-Format“ auf Seite 764

Bibliothek mit technischen Informationen zu DB2 im PDF-Format

Die folgenden Tabellen enthalten eine Beschreibung der DB2-Bibliothek, die im IBM Publications Center unter www.ibm.com/shop/publications/order zur Verfügung steht.

In den Tabellen sind die Bücher, die in gedrucktem Format zur Verfügung stehen, gekennzeichnet; möglicherweise sind diese in Ihrem Land oder Ihrer Region jedoch nicht verfügbar.

Diese Bücher enthalten grundlegende Informationen für alle DB2-Benutzer. Diese Informationen sind sowohl für Programmierer als auch für Datenbankadministratoren geeignet und unterstützen Sie bei der Arbeit mit DB2 Connect oder anderen DB2-Produkten.

Tabelle 98. Technische Informationen zu DB2

Name	IBM Form	In gedrucktem Format verfügbar
<i>Systemverwaltung: Implementierung</i>	SC12-3628	Ja
<i>Systemverwaltung: Konzept</i>	SC12-3630	Ja
<i>Administrative API Reference</i>	SC10-4231	Ja
<i>Administrative SQL Routines and Views</i>	SC10-4293	Nein
<i>Call Level Interface Guide and Reference, Volume 1</i>	SC10-4224	Ja
<i>Call Level Interface Guide and Reference, Volume 2</i>	SC10-4225	Ja
<i>Command Reference</i>	SC10-4226	Nein

Tabelle 98. Technische Informationen zu DB2 (Forts.)

Name	IBM Form	In gedrucktem Format verfügbar
<i>Dienstprogramme für das Versetzen von Daten Handbuch und Referenz</i>	SC12-3634	Ja
<i>Datenrecovery und hohe Verfügbarkeit Handbuch und Referenz</i>	SC12-3631	Ja
<i>Developing ADO.NET and OLE DB Applications</i>	SC10-4230	Ja
<i>Developing Embedded SQL Applications</i>	SC10-4232	Ja
<i>Developing SQL and External Routines</i>	SC10-4373	Nein
<i>Developing Java Applications</i>	SC10-4233	Ja
<i>Developing Perl and PHP Applications</i>	SC10-4234	Nein
<i>Getting Started with Database Application Development</i>	SC10-4252	Ja
<i>Installation und Verwaltung von DB2 unter Linux und Windows - Erste Schritte</i>	GC12-3638	Ja
<i>Fehlernachrichten, Band 1</i>	SC12-3643	Nein
<i>Fehlernachrichten, Band 2</i>	SC12-3644	Nein
<i>Migration</i>	GC12-3639	Ja
<i>Net Search Extender Verwaltung und Benutzerhandbuch</i>	SH12-3054	Ja
Anmerkung: Die HTML-Version dieses Dokuments wird nicht von der HTML-Dokumentations-CD installiert.		
<i>Systemverwaltung: Optimierung</i>	SC12-3629	Ja
<i>Query Patroller Verwaltung und Benutzerhandbuch</i>	GC12-3636	Ja
<i>DB2-Clients - Einstieg</i>	GC12-3640	Nein
<i>DB2-Server - Einstieg</i>	GC12-3637	Ja
<i>Spatial Extender und Geodetic Data Management Feature Benutzer- und Referenzhandbuch</i>	SC12-3722	Ja
<i>SQL Guide</i>	SC10-4248	Ja
<i>SQL Reference, Volume 1</i>	SC10-4249	Ja
<i>SQL Reference, Volume 2</i>	SC10-4250	Ja
<i>Systemmonitor Handbuch und Referenz</i>	SC12-3633	Ja
<i>Fehlerbehebung</i>	GC12-3635	Nein
<i>Lernprogramm für Visual Explain</i>	SC12-3754	Nein
<i>Neue Funktionen</i>	SC12-3645	Ja

Tabelle 98. Technische Informationen zu DB2 (Forts.)

Name	IBM Form	In gedrucktem Format verfügbar
<i>XML Extender Verwaltung und Programmierung</i>	SC12-3723	Ja
<i>XML-Handbuch</i>	SC12-3632	Ja
<i>XQuery Reference</i>	SC18-9796	Ja

Tabelle 99. Technische Informationen zu DB2 Connect

Name	IBM Form	In gedrucktem Format verfügbar
<i>DB2 Connect Benutzerhandbuch</i>	SC12-3642	Ja
<i>DB2 Connect Personal Edition - Einstieg</i>	GC12-3725	Ja
<i>DB2 Connect-Server - Einstieg</i>	GC12-3641	Ja

Tabelle 100. Technische Informationen zu WebSphere Information Integration

Name	IBM Form	In gedrucktem Format verfügbar
<i>WebSphere Information Integration: Administration Guide for Federated Systems</i>	SC19-1020	Ja
<i>WebSphere Information Integration: ASNCLP Program Reference for Replication and Event Publishing</i>	SC19-1018	Ja
<i>WebSphere Information Integration: Konfiguration föderierter Datenquellen</i>	SC12-3777	Nein
<i>WebSphere Information Integration: SQL Replication Handbuch und Referenz</i>	SC12-3782	Ja

Anmerkung: Die DB2-Release-Informationen enthalten zusätzliche Informationen für das verwendete Produktrelease und die verwendete Fixpackstufe. Weitere Informationen hierzu finden Sie mit Hilfe der zugehörigen Links.

Zugehörige Konzepte:

- „Übersicht über technische Informationen zu DB2“ auf Seite 763
- „Anmerkungen zu den Release-Informationen“ in den *Release-Informationen*

Zugehörige Tasks:

- „Bestellen gedruckter DB2-Bücher“ auf Seite 767

Bestellen gedruckter DB2-Bücher

Gedruckte DB2-Bücher können Sie in den meisten Ländern oder Regionen online bestellen. Das Bestellen gedruckter DB2-Bücher ist stets über den zuständigen IBM Ansprechpartner möglich. Beachten Sie hierbei bitte, dass einige Softcopybücher auf der CD mit der *DB2-PDF-Dokumentation* nicht in gedruckter Form verfügbar sind. So sind beispielsweise die beiden Bände des Handbuchs *DB2 Fehlernachrichten* nicht in gedruckter Form erhältlich.

Gedruckte Versionen vieler DB2-Bücher, die auf der CD mit der DB2-PDF-Dokumentation verfügbar sind, können gegen eine Gebühr bei IBM bestellt werden. Abhängig vom jeweiligen Land bzw. der jeweiligen Region können Sie Bücher möglicherweise online über das IBM Publications Center bestellen. Ist im jeweiligen Land bzw. der jeweiligen Region keine Onlinebestellung möglich, können Sie gedruckte DB2-Bücher stets über den zuständigen IBM Ansprechpartner bestellen. Nicht alle Bücher, die auf der CD mit der DB2-PDF-Dokumentation verfügbar sind, können in gedruckter Form bestellt werden.

Anmerkung: Über <http://publib.boulder.ibm.com/infocenter/db2help/> haben Sie Zugriff auf die DB2-Informationszentrale, wo Sie die neueste und umfassendste DB2-Dokumentation finden.

Vorgehensweise:

Gehen Sie wie folgt vor, um gedruckte DB2-Bücher zu bestellen:

- Informationen dazu, ob in Ihrem Land oder Ihrer Region die Bestellung von gedruckten DB2-Büchern möglich ist, finden Sie auf der Website mit dem IBM Publications Center unter <http://www.ibm.com/shop/publications/order>. Wählen Sie ein Land, eine Region oder eine Sprache aus, um die Bestellinformationen für Veröffentlichungen aufzurufen, und führen Sie dann die entsprechenden Schritte des Bestellverfahrens für Ihr Land bzw. Ihre Region aus.
- Gehen Sie wie folgt vor, um gedruckte DB2-Bücher beim zuständigen IBM Ansprechpartner zu bestellen:
 - Kontaktinformationen zum zuständigen Ansprechpartner finden Sie auf einer der folgenden Websites:
 - IBM Verzeichnis weltweiter Kontakte unter www.ibm.com/planetwide.
 - Website mit IBM Veröffentlichungen unter <http://www.ibm.com/shop/publications/order>. Wählen Sie das gewünschte Land, die gewünschte Region oder die gewünschte Sprache aus, um auf die entsprechende Homepage mit Veröffentlichungen Ihres Landes bzw. Ihrer Region zuzugreifen. Folgen Sie auf dieser Seite dem Link für Informationen zu dieser Site ("About this Site").
 - Geben Sie bei Ihrem Anruf an, dass Sie eine DB2-Veröffentlichung bestellen möchten.
 - Teilen Sie dem zuständigen Ansprechpartner die Titel und Formelarnummern der Bücher mit, die Sie bestellen möchten .

Zugehörige Konzepte:

- „Übersicht über technische Informationen zu DB2“ auf Seite 763

Zugehörige Referenzen:

- „Bibliothek mit technischen Informationen zu DB2 im PDF-Format“ auf Seite 764

Anzeigen der Hilfe für den SQL-Status über den Befehlszeilenprozessor

DB2 gibt für Bedingungen, die auf Grund einer SQL-Anweisung generiert werden können, einen SQLSTATE-Wert zurück. Die SQLSTATE-Hilfe erläutert die Bedeutung der SQL-Statuswerte und der SQL-Statusklassencodes.

Vorgehensweise:

Zum Aufrufen der Hilfe für SQL-Statuswerte müssen Sie den Befehlszeilenprozessor öffnen und Folgendes eingeben:

`? sqlstate` oder `? klassencode`

Hierbei steht *sqlstate* für einen gültigen fünfstelligen SQL-Statuswert und *klassencode* für die ersten beiden Ziffern dieses Statuswertes.

So kann beispielsweise durch die Eingabe von `? 08003` Hilfe für den SQL-Statuswert 08003 angezeigt werden, durch die Eingabe von `? 08` Hilfe für den Klassencode 08.

Zugehörige Tasks:

- „Aufrufen der Hilfe für Befehle über den Befehlszeilenprozessor“ in *Command Reference*
- „Aufrufen der Hilfe für Nachrichten über den Befehlszeilenprozessor“ in *Command Reference*

Zugriff auf verschiedene Versionen der DB2-Informationszentrale

Für Themen aus DB2 Version 9 lautet die URL der DB2-Informationszentrale <http://publib.boulder.ibm.com/infocenter/db2luw/v9/>.

Für Themen aus DB2 Version 8 lautet die URL der Informationszentrale (Version 8, 'Information - Unterstützung') <http://publib.boulder.ibm.com/infocenter/db2luw/v8/>.

Zugehörige Tasks:

- „Aktualisieren der auf Ihrem Computer oder Intranet-Server installierten DB2-Informationszentrale“ auf Seite 769

Anzeigen von Themen in der gewünschten Sprache in der DB2-Informationszentrale

In der DB2-Informationszentrale werden Themen, wenn möglich, in der Sprache angezeigt, die in den Vorgaben Ihres Browsers angegeben ist. Falls ein Thema nicht in die gewünschte Sprache übersetzt wurde, wird es in der DB2-Informationszentrale in Englisch angezeigt.

Vorgehensweise:

Um Themen in der gewünschten Sprache im Browser 'Internet Explorer' anzuzeigen, gehen Sie wie folgt vor:

1. Klicken Sie im Internet Explorer **Extras** —> **Internetoptionen...** —> **Sprachen...** an. Das Fenster **Spracheinstellung** wird geöffnet.
2. Stellen Sie sicher, dass die gewünschte Sprache als erster Eintrag in der Liste angegeben ist.
 - Klicken Sie den Knopf **Hinzufügen...** an, um eine neue Sprache zur Liste hinzuzufügen.

Anmerkung: Das Hinzufügen einer Sprache bedeutet nicht zwangsläufig, dass der Computer über die erforderlichen Schriftarten verfügt, um die Themen in der gewünschten Sprache anzuzeigen.

- Um eine Sprache an den Anfang der Liste zu verschieben, wählen Sie zunächst die gewünschte Sprache und anschließend den Knopf **Nach oben** aus, bis die Sprache an erster Stelle in der Liste steht.
3. Löschen Sie den Inhalt des Browser-Cache, und aktualisieren Sie anschließend die Seite, um die DB2-Informationszentrale in der gewünschten Sprache anzuzeigen.

Um Themen in der gewünschten Sprache in einem Firefox- oder Mozilla-Browser anzuzeigen, gehen Sie wie folgt vor:

1. Wählen Sie **Tools** —> **Options** —> **Languages** aus. Die Anzeige für die Auswahl der Sprache wird im Fenster mit den Einstellungen aufgerufen.
2. Stellen Sie sicher, dass die gewünschte Sprache als erster Eintrag in der Liste angegeben ist.
 - Wenn Sie eine neue Sprache zur Liste hinzufügen möchten, klicken Sie den Knopf **Add...** an, um eine Sprache im entsprechenden Fenster auszuwählen.
 - Um eine Sprache an den Anfang der Liste zu verschieben, wählen Sie zunächst die gewünschte Sprache und anschließend den Knopf **Move Up** aus, bis die Sprache an erster Stelle in der Liste steht.
3. Löschen Sie den Inhalt des Browser-Cache, und aktualisieren Sie anschließend die Seite, um die DB2-Informationszentrale in der gewünschten Sprache anzuzeigen.

Bei einigen Kombinationen aus Browser und Betriebssystem müssen Sie möglicherweise auch die Ländereinstellungen des Betriebssystems in die gewünschte Locale und Sprache ändern.

Zugehörige Konzepte:

- „Übersicht über technische Informationen zu DB2“ auf Seite 763

Aktualisieren der auf Ihrem Computer oder Intranet-Server installierten DB2-Informationszentrale

Wenn Sie eine lokal installierte DB2-Informationszentrale verwenden, stehen ggf. Aktualisierungen zum Download bereit. Der Wert für die 'letzte Aktualisierung' am Ende der meisten Themen gibt den aktuellen Stand für das jeweilige Thema an.

Um festzustellen, ob für die gesamte DB2-Informationszentrale eine Aktualisierung verfügbar ist, suchen Sie nach dem Wert für die 'letzte Aktualisierung' auf der Homepage der Informationszentrale. Vergleichen Sie den Wert auf Ihrer lokal installierten Homepage mit dem Datum der neuesten für den Download verfügbaren Aktualisierung unter <http://www.ibm.com/software/data/db2/udb/support/>

icupdate.html. Sie können dann die lokal installierte Informationszentrale aktualisieren, falls eine neuere für den Download verfügbare Aktualisierung zur Verfügung steht.

Zur Aktualisierung der lokal installierten DB2-Informationszentrale sind die folgenden Schritte erforderlich:

1. Stoppen Sie die DB2-Informationszentrale auf Ihrem Computer, und starten Sie die Informationszentrale im Standalone-Modus erneut. Durch die Ausführung der Informationszentrale im Standalone-Modus wird verhindert, dass andere Benutzer in Ihrem Netz auf die Informationszentrale zugreifen, und ermöglicht Ihnen das Herunterladen und Anwenden von Aktualisierungen.
2. Verwenden Sie die Aktualisierungsfunktion, um festzustellen, ob Aktualisierungspakete von IBM verfügbar sind.

Anmerkung: Darüber hinaus sind Aktualisierungen auf CD verfügbar. Einzelheiten zur Konfiguration Ihrer Informationszentrale für die Installation von Aktualisierungen von einer CD finden Sie unter den zugehörigen Links.

Ist dies der Fall, verwenden Sie die Aktualisierungsfunktion, um die Pakete herunterzuladen. (Die Aktualisierungsfunktion ist nur im Standalone-Modus verfügbar.)

3. Stoppen Sie die im Standalone-Modus gestartete Informationszentrale, und starten Sie den Service der DB2-Informationszentrale auf Ihrem Computer erneut.

Vorgehensweise:

Gehen Sie wie folgt vor, um die auf Ihrem Computer bzw. Intranet-Server installierte DB2-Informationszentrale zu aktualisieren:

1. Stoppen Sie den Service der DB2-Informationszentrale.
 - Unter Windows: Klicken Sie **Start → Einstellungen → Systemsteuerung → Verwaltung → Dienste** an. Klicken Sie mit der rechten Maustaste die **DB2-Informationszentrale** an, und wählen Sie **Stoppen** aus.
 - Unter Linux: Geben Sie den folgenden Befehl ein:
`/etc/init.d/db2icdv9 stop`
2. Starten Sie die Informationszentrale im Standalone-Modus.
 - Unter Windows:
 - a. Öffnen Sie ein Befehlsfenster.
 - b. Navigieren Sie zu dem Pfad, in dem die Informationszentrale installiert ist. Standardmäßig ist die DB2-Informationszentrale im Verzeichnis `C:\Programme\IBM\DB2 Information Center\Version 9` installiert.
 - c. Führen Sie die Datei `help_start.bat` aus, und verwenden Sie dabei den vollständig qualifizierten Pfad für die DB2-Informationszentrale:
`<verzeichnis_der_db2_informationszentrale>\doc\bin\help_start.bat`
 - Unter Linux:
 - a. Navigieren Sie zu dem Pfad, in dem die Informationszentrale installiert ist. Standardmäßig ist die DB2-Informationszentrale im Verzeichnis `/opt/ibm/db2ic/V9` installiert.
 - b. Führen Sie das Script `help_start` aus, und verwenden Sie dabei den vollständig qualifizierten Pfad für die DB2-Informationszentrale:
`<verzeichnis_der_db2_informationszentrale>/doc/bin/help_start`

Der standardmäßig auf dem System verwendete Web-Browser wird aufgerufen und zeigt die Standalone-Informationszentrale an.

3. Klicken Sie den Aktualisierungsknopf (🔄) an. Klicken Sie im rechten Fenster der Informationszentrale den Knopf für die Suche nach Aktualisierungen an. Eine Liste der Aktualisierungen für die vorhandene Dokumentation wird angezeigt.
4. Wählen Sie zum Initiieren des Downloadprozesses die gewünschten Aktualisierungen aus, und klicken Sie anschließend den Knopf für die Installation der Aktualisierungen an.
5. Klicken Sie nach Abschluss des Download- und Installationsprozesses **Fertig stellen** an.
6. Stoppen Sie die im Standalone-Modus gestartete Informationszentrale.
 - Unter Windows: Führen Sie die Datei `help_end.bat` aus, und verwenden Sie dabei den vollständig qualifizierten Pfad für die DB2-Informationszentrale:
`<verzeichnis_der_db2_informationszentrale>\doc\bin\help_end.bat`

Anmerkung: Die Stapeldatei `help_end` enthält die Befehle, die erforderlich sind, um die Prozesse, die mit der Stapeldatei `help_start` gestartet wurden, ordnungsgemäß zu beenden. Verwenden Sie nicht die Tastenkombination `Strg+C` oder eine andere Methode, um `help_start.bat` zu beenden.

- Unter Linux: Führen Sie das Script `help_end` aus, und verwenden Sie dabei den vollständig qualifizierten Pfad für die DB2-Informationszentrale:
`<verzeichnis_der_db2_informationszentrale>/doc/bin/help_end`

Anmerkung: Das Script `help_end` enthält die Befehle, die erforderlich sind, um die Prozesse, die mit dem Script `help_start` gestartet wurden, ordnungsgemäß zu beenden. Verwenden Sie keine andere Methode, um das Script `help_start` zu beenden.

7. Starten Sie den Service der DB2-Informationszentrale erneut.
 - Unter Windows: Klicken Sie **Start** → **Einstellungen** → **Systemsteuerung** → **Verwaltung** → **Dienste** an. Klicken Sie mit der rechten Maustaste die **DB2-Informationszentrale** an, und wählen Sie **Starten** aus.
 - Unter Linux: Geben Sie den folgenden Befehl ein:
`/etc/init.d/db2icdv9 start`

In der aktualisierten DB2-Informationszentrale werden die neuen und aktualisierten Themen angezeigt.

Zugehörige Konzepte:

- „Optionen für die Installation der DB2-Informationszentrale “ in *DB2-Server - Einstieg*

Zugehörige Tasks:

- „Installation der DB2-Informationszentrale mit dem DB2-Installationsassistenten (Linux)“ in *DB2-Server - Einstieg*
- „Installation der DB2-Informationszentrale mit dem DB2-Installationsassistenten (Windows)“ in *DB2-Server - Einstieg*

DB2-Lernprogramme

Die DB2-Lernprogramme unterstützen Sie dabei, sich mit den unterschiedlichen Aspekten der DB2-Produkte vertraut zu machen. Die Lerneinheiten bieten eine in einzelne Schritte unterteilte Anleitung.

Vorbereitungen:

Die XHTML-Version des Lernprogramms kann über die Informationszentrale unter <http://publib.boulder.ibm.com/infocenter/db2help/> angezeigt werden.

In einigen der Lerneinheiten werden Beispieldaten und Codebeispiele verwendet. Informationen zu bestimmten Voraussetzungen für die Ausführung der Tasks finden Sie in der Beschreibung des Lernprogramms.

DB2-Lernprogramme:

Klicken Sie zum Anzeigen des Lernprogramms den Titel an.

Nativer XML-Datenspeicher

Einrichten einer DB2-Datenbank, um XML-Daten zu speichern und Basisoperationen mit dem nativen XML-Datenspeicher auszuführen.

Lernprogramm für Visual Explain

Analysieren, Optimieren und Anpassen von SQL-Anweisungen zur Leistungsverbesserung mit Hilfe von Visual Explain.

Zugehörige Konzepte:

- „Übersicht über Visual Explain “ in *Systemverwaltung: Implementierung*

Informationen zur Fehlerbehebung in DB2

Eine breite Palette verschiedener Informationen zur Fehlerbestimmung und Fehlerbehebung steht zur Verfügung, um Sie bei der Verwendung von DB2-Produkten zu unterstützen.

DB2-Dokumentation

Informationen zur Fehlerbehebung stehen im Handbuch DB2 Fehlerbehebung oder im Abschnitt zur Unterstützung und Fehlerbehebung der DB2-Informationszentrale zur Verfügung. Dort finden Sie Informationen dazu, wie Sie Probleme mit Hilfe der DB2-Diagnosetools und -Dienstprogramme eingrenzen und identifizieren können, Lösungen für einige der häufigsten Probleme sowie weitere Hinweise zur Behebung von Fehlern und Problemen, die bei der Verwendung der DB2-Produkte auftreten können.

DB2-Website mit technischer Unterstützung

Auf der DB2-Website mit technischer Unterstützung finden Sie Informationen zu Problemen und den möglichen Ursachen und Fehlerbehebungsmaßnahmen. Die Website mit technischer Unterstützung enthält Links zu den neuesten DB2-Veröffentlichungen, technischen Hinweisen (TechNotes), APARs (Authorized Program Analysis Reports) und Fehlerkorrekturen, Fixpacks sowie weiteren Ressourcen. Sie können diese Wissensbasis nach möglichen Lösungen für aufgetretene Probleme durchsuchen.

Rufen Sie die DB2-Website mit technischer Unterstützung unter <http://www.ibm.com/software/data/db2/udb/support.html> auf.

Zugehörige Konzepte:

- „Einführung in die Fehlerbestimmung“ in *Fehlerbehebung*
- „Übersicht über technische Informationen zu DB2“ auf Seite 763

Bedingungen

Die Berechtigungen zur Nutzung dieser Veröffentlichungen werden Ihnen auf der Basis der folgenden Bedingungen gewährt.

Persönliche Nutzung: Sie dürfen diese Veröffentlichungen für Ihre persönliche, nicht kommerzielle Nutzung unter der Voraussetzung vervielfältigen, dass alle Eigentumsvermerke erhalten bleiben. Sie dürfen diese Veröffentlichungen oder Teile der Veröffentlichungen ohne ausdrückliche Genehmigung von IBM nicht weitergeben, anzeigen oder abgeleitete Werke davon erstellen.

Kommerzielle Nutzung: Sie dürfen diese Veröffentlichungen nur innerhalb Ihres Unternehmens und unter der Voraussetzung, dass alle Eigentumsvermerke erhalten bleiben, vervielfältigen, weitergeben und anzeigen. Sie dürfen diese Veröffentlichungen oder Teile der Veröffentlichungen ohne ausdrückliche Genehmigung von IBM außerhalb Ihres Unternehmens nicht vervielfältigen, weitergeben, anzeigen oder abgeleitete Werke davon erstellen.

Abgesehen von den hier gewährten Berechtigungen erhalten Sie keine weiteren Berechtigungen, Lizenzen oder Rechte (veröffentlicht oder stillschweigend) in Bezug auf die Veröffentlichungen oder darin enthaltene Informationen, Daten, Software oder geistiges Eigentum.

IBM behält sich das Recht vor, die in diesem Dokument gewährten Berechtigungen nach eigenem Ermessen zurückzuziehen, wenn sich die Nutzung der Veröffentlichungen für IBM als nachteilig erweist oder wenn die obigen Nutzungsbestimmungen nicht genau befolgt werden.

Sie dürfen diese Informationen nur in Übereinstimmung mit allen anwendbaren Gesetzen und Vorschriften, einschließlich aller US-amerikanischen Exportgesetze und Verordnungen, herunterladen und exportieren.

IBM übernimmt keine Gewährleistung für den Inhalt dieser Informationen. Diese Veröffentlichungen werden auf der Grundlage des gegenwärtigen Zustands (auf "as-is"-Basis) und ohne eine ausdrückliche oder stillschweigende Gewährleistung für die Handelsüblichkeit, die Verwendungsfähigkeit oder die Freiheit der Rechte Dritter zur Verfügung gestellt.

Anhang I. Bemerkungen

Möglicherweise bietet IBM die in dieser Dokumentation beschriebenen Produkte, Services oder Funktionen nicht in allen Ländern an. Informationen über die gegenwärtig im jeweiligen Land verfügbaren Produkte und Services sind beim IBM Ansprechpartner erhältlich. Hinweise auf IBM Lizenzprogramme oder andere IBM Produkte bedeuten nicht, dass nur Programme, Produkte oder Services von IBM verwendet werden können. An Stelle der Produkte, Programme oder Services können auch andere ihnen äquivalente Produkte, Programme oder Services verwendet werden, solange diese keine gewerblichen oder andere Schutzrechte der IBM verletzen. Die Verantwortung für den Betrieb der Produkte, Programme oder Dienstleistungen in Verbindung mit Fremdprodukten und Fremddienstleistungen liegt beim Kunden, soweit nicht ausdrücklich solche Verbindungen erwähnt sind.

Für in diesem Handbuch beschriebene Erzeugnisse und Verfahren kann es IBM Patente oder Patentanmeldungen geben. Mit der Auslieferung dieses Handbuchs ist keine Lizenzierung dieser Patente verbunden. Lizenzanforderungen sind schriftlich an folgende Adresse zu richten (Anfragen an diese Adresse müssen auf Englisch formuliert werden):

IBM Director of Licensing
IBM Europe, Middle East & Africa
Tour Descartes 2, avenue Gambetta
92066 Paris La Defense
France

Trotz sorgfältiger Bearbeitung können technische Ungenauigkeiten oder Druckfehler in dieser Veröffentlichung nicht ausgeschlossen werden. Die Angaben in diesem Handbuch werden in regelmäßigen Zeitabständen aktualisiert. Die Änderungen werden in Überarbeitungen bekanntgegeben. IBM kann ohne weitere Mitteilung jederzeit Verbesserungen und/oder Änderungen an den in dieser Veröffentlichung beschriebenen Produkten und/oder Programmen vornehmen.

Verweise in diesen Informationen auf Websites anderer Anbieter dienen lediglich als Benutzerinformationen und stellen keinerlei Billigung des Inhalts dieser Websites dar. Das über diese Websites verfügbare Material ist nicht Bestandteil des Materials für dieses IBM Produkt; die Verwendung dieser Websites geschieht auf eigene Verantwortung.

Werden an IBM Informationen eingesandt, können diese beliebig verwendet werden, ohne dass eine Verpflichtung gegenüber dem Einsender entsteht.

Lizenznehmer des Programms, die Informationen zu diesem Produkt wünschen mit der Zielsetzung: (i) den Austausch von Informationen zwischen unabhängigen, erstellten Programmen und anderen Programmen (einschließlich des vorliegenden Programms) sowie (ii) die gemeinsame Nutzung der ausgetauschten Informationen zu ermöglichen, wenden sich an folgende Adresse:

IBM Canada Limited
Office of the Lab Director
8200 Warden Avenue
Markham, Ontario
L6G 1C7
CANADA

Die Bereitstellung dieser Informationen kann unter Umständen von bestimmten Bedingungen - in einigen Fällen auch von der Zahlung einer Gebühr - abhängig sein.

Die Lieferung des im Dokument aufgeführten Lizenzprogramms sowie des zugehörigen Lizenzmaterials erfolgt auf der Basis der IBM Rahmenvereinbarung sowie der Allgemeinen Geschäftsbedingungen von IBM, der Internationalen Nutzungsbedingungen der IBM für Programmpakete oder einer äquivalenten Vereinbarung.

Alle in diesem Dokument enthaltenen Leistungsdaten stammen aus einer gesteuerten Umgebung. Die Ergebnisse, die in anderen Betriebsumgebungen erzielt werden, können daher erheblich von den hier erzielten Ergebnissen abweichen. Einige Daten stammen möglicherweise von Systemen, deren Entwicklung noch nicht abgeschlossen ist. Eine Garantie, dass diese Daten auch in allgemein verfügbaren Systemen erzielt werden, kann nicht gegeben werden. Darüber hinaus wurden einige Daten unter Umständen durch Extrapolation berechnet. Die tatsächlichen Ergebnisse können abweichen. Benutzer dieses Dokuments sollten die entsprechenden Daten in ihrer spezifischen Umgebung prüfen.

Informationen über Produkte anderer Hersteller als IBM wurden von den Herstellern dieser Produkte zur Verfügung gestellt, bzw. aus von ihnen veröffentlichten Ankündigungen oder anderen öffentlich zugänglichen Quellen entnommen. IBM hat diese Produkte nicht getestet und kann daher keine Aussagen zu Leistung, Kompatibilität oder anderen Merkmalen machen. Fragen zu den Leistungsmerkmalen von Produkten anderer Anbieter sind an den jeweiligen Anbieter zu richten.

Aussagen über Pläne und Absichten der IBM unterliegen Änderungen oder können zurückgenommen werden und repräsentieren nur die Ziele der IBM.

Diese Veröffentlichung enthält Beispiele für Daten und Berichte des alltäglichen Geschäftsablaufes. Sie sollen nur die Funktionen des Lizenzprogrammes illustrieren; sie können Namen von Personen, Firmen, Marken oder Produkten enthalten. Alle diese Namen sind frei erfunden, Ähnlichkeiten mit tatsächlichen Namen und Adressen sind rein zufällig.

COPYRIGHTLIZENZ:

Diese Veröffentlichung enthält Musteranwendungsprogramme, die in Quellsprache geschrieben sind. Sie dürfen diese Musterprogramme kostenlos kopieren, ändern und verteilen, wenn dies zu dem Zweck geschieht, Anwendungsprogramme zu entwickeln, verwenden, vermarkten oder zu verteilen, die mit der Anwendungsprogrammierschnittstelle konform sind, für die diese Musterprogramme geschrieben werden. Diese Beispiele wurden nicht unter allen denkbaren Bedingungen getestet. Daher kann IBM die Zuverlässigkeit, Wartungsfreundlichkeit oder Funktion dieser Programme weder zusagen noch gewährleisten.

Kopien oder Teile der Musterprogramme bzw. daraus abgeleiteter Code müssen folgenden Copyrightvermerk beinhalten:

© (*Name Ihrer Firma*) (*Jahr*). Teile des vorliegenden Codes wurden aus Musterprogrammen der IBM Corp. abgeleitet. © Copyright IBM Corp. *_Jahr/Jahre angeben_*. Alle Rechte vorbehalten.

Marken

Namen von Unternehmen, Produkten oder Services in den Dokumenten der Dokumentationsbibliothek von DB2 Version 9 können Marken oder Servicemarken der International Business Machines Corporation oder anderer Unternehmen sein. Informationen zu den Marken der IBM Corporation in den jeweiligen Ländern finden Sie unter <http://www.ibm.com/legal/copytrade.shtml>.

Folgende Namen sind in gewissen Ländern Marken oder eingetragene Marken anderer Unternehmen und wurden in mindestens einem der Dokumente in der DB2-Dokumentationsbibliothek verwendet.

Microsoft, Windows, Windows NT und das Windows-Logo sind in gewissen Ländern Marken der Microsoft Corporation.

Intel, Itanium, Pentium und Xeon sind in gewissen Ländern Marken der Intel Corporation.

Java und alle Java-basierten Marken sind in gewissen Ländern Marken von Sun Microsystems, Inc.

UNIX ist in gewissen Ländern eine eingetragene Marke von The Open Group.

Linux ist in gewissen Ländern eine Marke von Linus Torvalds.

Andere Namen von Unternehmen, Produkten oder Dienstleistungen können Marken anderer Unternehmen sein.

Index

A

- Abfragen
 - Größe des Anweisungszwischenspeichers, Konfigurationsparameter 460
 - optimieren
 - SELECT-Anweisungen 126
- Abfrageoptimierung
 - Auswirkung von Datenbankpartitionsgruppen 189
 - Konfigurationsparameter 186
- Abfrageoptimierungsklassen
 - auswählen 179
 - Beschreibung 181
- ADVISE_INDEX, Tabelle 688
- ADVISE_INSTANCE, Tabelle 692
- ADVISE_MQT, Tabelle 693
- ADVISE_PARTITION, Tabelle 695
- ADVISE_TABLE, Tabelle 697
- ADVISE_WORKLOAD, Tabelle 698
- Advisorfunktionen
 - Designadvisor 51
- agent_stack_sz, Konfigurationsparameter 452
- Agenten
 - Arbeitsagententypen 153
 - Beschreibung 153
 - Clientverbindungen 171
 - für Datenseitenverwaltung 138
 - in einer partitionierten Datenbank 163
 - Konfigurationsparameter mit Auswirkung auf Anzahl 165
 - verwalten 163
- Agentenpoolgröße, Konfigurationsparameter 492
- Agentenprozess
 - Agentenpriorität, Konfigurationsparameter 482
 - applheapsz, Konfigurationsparameter 454
 - aslheapsz, Konfigurationsparameter 461
 - maximale Anzahl gleichzeitig aktiver Agenten 489
 - maximale Anzahl von Agenten 487
- agentpri, Konfigurationsparameter 482
- Aktivieren, partitionsinterne Parallelität, Konfigurationsparameter 559
- Aktualisierende Recovery
 - Definition 151
- Aktualisierungen
 - DB2-Informationszentrale 769
 - Informationszentrale 769
 - Prozessmodell 139
 - verlorene
 - Steuerung des gemeinsamen Zugriffs 61
- Als Suchargument verwendbare Vergleichselemente
 - Definition 200
- alt_collate, Konfigurationsparameter 535
- ALTER TABLE, Anweisung
 - sperrenbezogene Leistungsprobleme verhindern 82
- ALTER TABLESPACE, Anweisung
 - Beispiel 189
- Anfangsanzahl Agenten im Pool, Konfigurationsparameter 491
- Anfangswert für die Anzahl abgeschirmter Prozesse, Konfigurationsparameter 496
- Anfügemodus
 - Prozess einfügen 150
- Antwortzeit für Verbindung, Konfigurationsparameter 554
- Anweisungen
 - Isolationsstufe angeben 66
- Anwendungen
 - maximale Anzahl koordinierender Agenten am Knoten 485
 - Zwischenspeicher für Steuerung, Einstellung 449
- Anwendungsprozess
 - Auswirkung auf Sperren 73
- Anzahl aktiver Protokolle, Konfigurationsparameter 505
- Anzahl der Datenbank-Backups, Konfigurationsparameter 526
- Anzahl der Gruppencommits, Konfigurationsparameter 512
- app_ctl_heap_sz 449
- appgroup_mem_sz, Konfigurationsparameter 450
- APPLHEAPSZ, Konfigurationsparameter
 - Verwendung 454
- Architektur
 - Übersicht 155
- archretrydelay, Konfigurationsparameter 507
- aslheapsz, Konfigurationsparameter 461
- Assistenten
 - Designadvisor 51
- audit_buf_sz, Konfigurationsparameter 466
- Aufbewahrungszeitraum für Recoveryprotokoll, Konfigurationsparameter 526
- Auslastungen
 - Designadvisor
 - definieren 56
 - optimieren 51
- authentication, DAS-Konfigurationsparameter 589
- authentication, Konfigurationsparameter 575
- Authentifizierung
 - alle Clients akzeptieren, Konfigurationsparameter 587
 - gesicherter Client, Konfigurationsparameter 588
- Authentifizierung bei Servern mit föderierten Datenbanken umgehen, Konfigurationsparameter 579
- AUTO_PROF_UPD
 - verwenden 361
- AUTO_STATS_PROF
 - verwenden 361
- AUTOCONFIGURE, Befehl 284
- Automatisch aktualisierte Übersichtstabellen
 - Beschreibung 11
- Automatische Konfiguration, API 284
- Automatische Reorganisation
 - aktivieren 374
- Automatische Speicheroptimierung 299
- Automatische Statistikerfassung 359
- Automatischer Neustart aktiviert, Konfigurationsparameter 518
- autonomic_switches, Konfigurationsparameter 547
- avg_appls, Konfigurationsparameter 484

B

- Backup anstehend, Anzeiger, Konfigurationsparameter 538
- backup_pending, Konfigurationsparameter 538
- Backups
 - geänderte Seiten protokollieren 527
- Bedingungen
 - Verwendung der Veröffentlichungen 773

- Befehle
 - db2expln 712
 - db2gov
 - verwenden 109
- Beispielausgabe
 - Analyse mit Vergleichstests 426
- Bemerkungen 775
- Benutzerdefinierte Funktionen (UDFs)
 - Statistikdaten eingeben für 341
- Benutzerexit aktivieren, Konfigurationsparameter 516
- Benutzerexitstatusanzeiger, Konfigurationsparameter 540
- Berechtigungen
 - Gruppennamen definieren
 - Systempflegeberechtigung, Gruppenname, Konfigurationsparameter 585
 - Systemsteuerungsberechtigung, Gruppenname, Konfigurationsparameter 584
 - Systemverwaltungsberechtigung, Gruppenname, Konfigurationsparameter 583
- Bereich
 - Sperrgranularität 74
- Bestellen von DB2-Büchern 767
- Binden
 - Isolationsstufe angeben 66
 - Konfigurationsparameter ändern 277
- blk_log_dsk_ful, Konfigurationsparameter 508
- Block-IDs
 - vor Tabellenzugriff vorbereiten 731
- Blockieren, bei voller Protokollplatte (blk_log_dsk_ful), Konfigurationsparameter 508
- BLOCKINSERT
 - LOCKSIZE-Ursachenwert
 - Vorteile 74
- Blockorientierte Pufferpools 144

C

- catalog_noauth, Konfigurationsparameter 576
- catalogcache_sz, Konfigurationsparameter 432
- chnpggs_thresh, Konfigurationsparameter 476
- CLI (Call Level Interface)
 - Isolationsstufe angeben 66
- Clientunterstützung
 - E/A-Blockgröße für Clients, Konfigurationsparameter 464
 - TCP/IP-Servicename, Konfigurationsparameter 551
- clnt_krb_plugin, Konfigurationsparameter 577
- clnt_pw_plugin, Konfigurationsparameter 578
- Clustering-Indizes 384
 - bei partitionierten Tabellen 45
 - Vorteile bei partitionierten Tabellen 45
- CMPEXP, Operator
 - Beschreibung 699
- codepage, Konfigurationsparameter 535
- Codepages
 - Datenbank, Konfigurationsparameter 535
- codeset, Konfigurationsparameter 535
- collate_info, Konfigurationsparameter 536
- comm_bandwidth, Konfigurationsparameter
 - Auswirkung auf die Abfrageoptimierung 186
 - Beschreibung 566
- Commit
 - Anzahl der Gruppencommits (mincommit) 512
- COMMIT, Anweisung
 - sperrenbezogene Leistungsprobleme verhindern 82
- Compiler
 - Informationen erfassen
 - EXPLAIN-Einrichtung 257

- Compiler (*Forts.*)
 - Umschreibungen
 - implizierte Vergleichselemente hinzufügen 242
 - korrelierte Unterabfragen 240
 - Mischsicht 238
- conn_elapse, Konfigurationsparameter 554
- contact_host, Konfigurationsparameter 590
- cpuspeed, Konfigurationsparameter
 - Auswirkung auf die Abfrageoptimierung 186
 - Beschreibung 567
- CREATE SERVER, Anweisung
 - Optionen für föderierte Datenbanken 256
- CURRENT EXPLAIN MODE, Sonderregister
 - EXPLAIN-Daten erfassen 269
- CURRENT EXPLAIN SNAPSHOT, Sonderregister
 - EXPLAIN-Daten erfassen 269
- CURRENT LOCK TIMEOUT, Sonderregister
 - Strategie für Wartestatus für Sperre 81
- Cursor
 - schließen
 - sperrenbezogene Leistungsprobleme verhindern 82

D

- Dämonen
 - Governor-Tool 108
- das_codepage, Konfigurationsparameter 591
- DAS-Konfigurationsparameter
 - authentication 589
 - contact_host 590
 - das_codepage 591
 - das_territory 591
 - dasadm_group 591
 - db2system 592
 - exec_exp_task 593
 - jdk_64_path 594
 - jdk_path 595
 - sched_enable 595
 - sched_userid 596
 - smtp_server 596
 - toolscat_db 597
 - toolscat_inst 597
 - toolscat_schema 598
- das_territory, Konfigurationsparameter 591
- dasadm_group, Konfigurationsparameter 591
- database_consistent, Konfigurationsparameter 538
- database_level, Konfigurationsparameter 537
- database_memory, Konfigurationsparameter 434
 - selbstoptimierend 295
- Datenbankaktivität
 - Statistikprofil
 - automatisch generieren 360
- Datenbanken
 - autorestart, Konfigurationsparameter 518
 - backup_pending, Konfigurationsparameter 538
 - codepage, Konfigurationsparameter 535
 - codeset, Konfigurationsparameter 535
 - Gebietscode, Konfigurationsparameter 537
 - Informationen zur Sortierfolge 536
 - Konfigurationsparameter, Übersicht 305
 - maximale Anzahl gleichzeitig aktiver Datenbanken 570
 - Prozesse 158
 - Releasestand, Konfigurationsparameter 537
 - territory, Konfigurationsparameter 538
- Datenbankgebietscode, Konfigurationsparameter 537
- Datenbankkonfigurationsparameter
 - empfohlene Werte generieren 284

- Datenbankmanager
 - Knotenart des Systems, Konfigurationsparameter 570
 - Konfigurationsparameter
 - empfohlene Werte generieren 284
 - Konfigurationsparameter, Übersicht 305
 - Nutzung des gemeinsamen Speichers 289
 - Zeitlimit für DB2START 557
 - Zeitlimit für DB2STOP 557
- Datenbankmonitor
 - verwenden 405
- Datenbankobjekte
 - EXPLAIN-Informationen 264
- Datenbankpartitionsgruppen
 - Auswirkung auf die Abfrageoptimierung 189
- Datenbankpartitionsserver
 - Mehrpartitionsverarbeitung 166
- Datenbankprotokollpfad ändern, Konfigurationsparameter 504
- Datenbanksystemmonitor
 - Standardschalter für Datenbanksystemmonitor, Konfigurationsparameter 565
- Datenpartitionseliminierung
 - Beschreibung 21
- Datenquellen
 - E/A-Geschwindigkeit und Leistung 253
- Datenseite in Standardtabellen 384
- Datenstichproben
 - mit TABLESAMPLE 128
 - Statistikerfassung 329
- Datenstrominformationen
 - von db2expln angezeigt 730
- Datenumverteilung
 - Anweisungen 397
 - Fehlerbehebung 403
 - Notwendigkeit ermitteln 397
 - Protokollspeicherbedarf 400
 - Prozessbeschreibung 395
 - Richtlinien 395
- db_mem_thresh
 - Konfigurationsparameter 436
- DB2_ALLOCATION_SIZE 630
- DB2_ALTERNATE_GROUP_LOOKUP 607
- DB2_ANTIJOIN 624
- DB2_APM_PERFORMANCE 630
- DB2_AVOID_PREFETCH 630
- DB2_AWE 630
- DB2_CLPPROMPT 621
- DB2_COMMIT_ON_EXIT 649
- DB2_CONNRETRIES_INTERVAL 602
- DB2_CREATE_DB_ON_PATH 649
- DB2_DISABLE_FLUSH_LOG 649
- DB2_DISCONNECT_ON_INTERRUPT 649
- DB2_DISPATCHER_PEEKTIMEOUT 649
- DB2_DJ_COMM 649
- DB2_DJ_INI 649
- DB2_DOCHOST 649
- DB2_DOCPORT 649
- DB2_ENABLE_AUTOCONFIG_DEFAULT 649
- DB2_ENABLE_BUFDP 630
- DB2_ENABLE_LDAP 649
- DB2_EVALUNCOMMITTED 630
 - Verschiebung von Zeilensperren 92
- DB2_EXTENDED_IO_FEATURES 630
- DB2_EXTENDED_OPTIMIZATION 630
- DB2_EXTSECURITY 649
- DB2_FALLBACK 649
- DB2_FMP_COMM_HEAPSZ 649
- DB2_FORCE_NLS_CACHE 615
- DB2 für Windows-Leistungszähler 757
- DB2_GRP_LOOKUP 649
- DB2_HADR_BUF_SIZE 649
- DB2_INDEX_TYPE2 602
- DB2-Informationszentrale
 - aktualisieren 769
 - in unterschiedlichen Sprachen anzeigen 768
 - Versionen 768
- DB2_INLIST_TO_NLJN 624
- DB2_IO_PRIORITY_SETTING 630
- DB2_KEEPTABLELOCK 630
- DB2_LGPAGE_BP 630
- DB2_LIC_STAT_SIZE 602
- DB2_LIKE_VARCHAR 624
- DB2_LOAD_COPY_NO_OVERRIDE 649
- DB2_MAP_XML_AS_CLOB_FOR_DLC 649
- DB2_MAX_CLIENT_CONNRETRIES 602
- DB2_MAX_LOB_BLOCK_SIZE 649
- DB2_MEM_TUNING_RANGE 630
- DB2_MINIMIZE_LISTPREFETCH 624
- DB2_MMAP_READ 630
- DB2_MMAP_WRITE 630
- DB2_NEW_CORR_SQ_FF 624
- DB2_NEWLOGPATH2 649
- DB2_NO_FORK_CHECK 630
- DB2_NO_MPFA_FOR_NEW_DB 630
- DB2_NUM_CKPW_DAEMONS 649
- DB2_NUM_FAILOVER_NODES 623
- DB2_OBJECT_TABLE_ENTRIES 630
- DB2_OPT_MAX_TEMP_SIZE 624
- DB2_OVERRIDE_BPF 630
- DB2_PARALLEL_IO 607
- DB2_PARTITIONEDLOAD_DEFAULT 623
- DB2_PINNED_BP 630
- DB2_PRED_FACTORIZE 624
- DB2_REDUCED_OPTIMIZATION 624
- DB2_RESOLVE_CALL_CONFLICT 649
- DB2_RESOURCE_POLICY 630
- DB2_SELECTIVITY 624
- DB2_SERVER_CONTIMEOUT 649
- DB2_SMS_TRUNC_TMPTABLE_THRESH 630
- DB2_SORT_AFTER_TQ 630
- DB2_SQLROUTINE_PREOPTS 624
- DB2_TRUNCATE_REUSESTORAGE 649
- DB2_TRUSTED_BINDIN 630
- DB2_USE_ALTERNATE_PAGE_CLEANING 630
 - Verwendung 140
- DB2_USE_DB2CCT2_JROUTINE 649
- DB2_UTIL_MSGPATH 649
- DB2_VENDOR_INI 649
- DB2_VIEW_REOPT_VALUES 602
- DB2_XBSA_LIBRARY 649
- DB2ACCOUNT 602
- DB2ADMINSERVER 649
- db2advis 7, 33
- db2advis, Befehl
 - verwenden 55
- DB2ASSUMEUPDATE 630
- db2batch, Vergleichstest-Tool
 - Beispiele 414
 - Tests erstellen 412
- DB2BIDI 602
- DB2BPVARS 630
- DB2BQTIME 621
- DB2BQTRY 621
- DB2CHECKCLIENTINTERVAL 615

DB2CHGPWD_ESE 623
 DB2CHKPTR 630
 DB2CHKSQLDA 630
 DB2CLIINIPATH 649
 DB2CODEPAGE 602
 DB2COMM 615
 DB2CONNECT_IN_APP_PROCESS 607
 DB2DBDFT 602
 DB2DBMSADDR 602
 DB2DEFPREP 649
 DB2DISCOVERYTIME 602
 DB2DMNBCKCTL 649
 DB2DOMAINLIST 607
 DB2ENVLIST 607
 db2exfmt, Tool 755
 Ausgabebeispiele
 Beschreibung 739
 db2expln, Befehl 712
 db2expln, Tool
 Ausgabebeispiele
 aktualisieren 731
 Beschreibung 739
 einfügen 731
 keine Parallelität 739
 löschen 731
 XANDOR, Operator 749
 XISCAN, Operator 752
 XSCAN, Operator 749, 751
 Zugriffsplan für Einzelpartition mit partitionsinterner Parallelität 741
 Zugriffsplan für föderierte Datenbanken 747
 Zugriffsplan für mehrere Partitionen mit partitionsübergreifender Parallelität 743
 Zugriffsplan für mehrere Partitionen mit voller Parallelität 745
 Ausgabebeschreibung 719
 föderierte Datenbanken 736
 Fehlernachrichten 718
 Informationen, angezeigte
 Datenstrom 730
 Join 728
 Parallelverarbeitung 733
 Spaltenberechnung 732
 Tabellenzugriff 720
 temporäre Tabelle 725
 verschiedene 737
 Verwendungshinweise 718
 Vorbereitung von Block-IDs 731
 Vorbereitung von Zeilenkennungen 731
 db2gov, Befehl
 verwenden 109
 DB2GRAPHICUNICODESERVER 602
 DB2INCLUDE 602
 DB2INSTANCE 607
 DB2INSTDEF 602
 DB2INSTOWNER 602
 DB2INSTPROF 607
 DB2IQTIME 621
 DB2JD_PORT_NUMBER 615
 DB2LDAP_BASEDN 649
 DB2LDAP_CLIENT_PROVIDER 649
 DB2LDAP_KEEP_CONNECTION 649
 DB2LDAP_SEARCH_SCOPE 649
 DB2LDAPCACHE 649
 DB2LDAPHOST 649
 DB2LIBPATH 607
 DB2LOADREC 649
 DB2LOCALE 602
 DB2LOCK_TO_RB 649
 DB2MAXFSCRSEARCH 630
 DB2MEMDISCLAIM 630
 DB2MEMMAXFREE 630
 db2mtrk, Befehl
 Beispielausgabe 131
 DB2NBADAPTERS 615
 DB2NBCHECKUPTIME 615
 DB2NBDISCOVERRCVBUFS 602
 DB2NBINTRLISTENS 615
 DB2NBRECVBUFSIZE 615
 DB2NBRECVNCBS 615
 DB2NBRESOURCES 615
 DB2NBSENDNCBS 615
 DB2NBSESSIONS 615
 DB2NBXTRANCBS 615
 DB2NODE 607
 DB2NOEXITLIST 649
 DB2NTMEMSIZE 630
 DB2NTNOCACHE 630
 DB2NTPRICLASS 630
 DB2NETWORKSET 630
 DB2OPTIONS 607
 DB2PATH 607
 db2perfc, Befehl 761
 db2perfi, Befehl 757
 db2perfr, Befehl 758
 DB2PORTRANGE 623
 DB2PRIORITIES 630
 DB2RCMD_LEGACY_MODE 607
 DB2REMOTEPREG 649
 DB2RETRY 615
 DB2RETRYTIME 615
 DB2ROUTINE_DEBUG 649
 DB2RQTIME 621
 DB2RSHCMD 615
 DB2RSHTIMEOUT 615
 DB2SATELLITEID 649
 DB2SERVICETPINSTANCE 615
 DB2SORCVBUF 615
 DB2SORT 649
 DB2SOSNDBUF 615
 DB2SYSPLEX_SERVER 615
 db2system, Konfigurationsparameter 592
 DB2TCP_CLIENT_CONTIMEOUT 615
 DB2TCP_CLIENT_RCVTIMEOUT 615
 DB2TCPCONNMGERS 615
 DB2TERRITORY 602
 DBHEAP, Konfigurationsparameter 437
 Deadlocks
 Beschreibung 78
 Detektor 78
 dlchktime, Konfigurationsparameter 471
 prüfen auf 471
 Defragmentierung
 Index 392
 Dekorrelierung einer Abfrage
 Compiler, Umschreibungen für 240
 DELETE, Operator
 Definition 699
 Designadvisor 7, 33
 Auslastung definieren 56
 Einrichtungen 53
 Einschränkungen 58
 Migration auf partitionierte Datenbanken durchführen 57
 Übersicht 51

Designadvisor (*Forts.*)
 verwenden 55

Dezimalrechnung
 drei Kommastellen bei Dezimaldivision, Konfigurationsparameter 463

dft_account_str, Konfigurationsparameter 568

dft_degree, Konfigurationsparameter 541
 Auswirkung auf die Abfrageoptimierung 186

dft_extent_sz, Konfigurationsparameter 477

dft_loadrec_ses, Konfigurationsparameter 518

dft_mon_bufpool, Konfigurationsparameter 565

dft_mon_lock, Konfigurationsparameter 565

dft_mon_sort, Konfigurationsparameter 565

dft_mon_stmt, Konfigurationsparameter 565

dft_mon_table, Konfigurationsparameter 565

dft_mon_timestamp, Konfigurationsparameter 565

dft_mon_uow, Konfigurationsparameter 565

dft_monswitches, Konfigurationsparameter 565

dft_mttb_types, Konfigurationsparameter 541

dft_prefetch_sz, Konfigurationsparameter 477

dft_queryopt, Konfigurationsparameter 542

dft_refresh_age, Konfigurationsparameter 542

dft_sqlmathwarn, Konfigurationsparameter 543

dftdbpath, Konfigurationsparameter 578

diaglevel, Konfigurationsparameter 561

diagpath, Konfigurationsparameter 562

dir_cache, Konfigurationsparameter 467

discover (DAS), Konfigurationsparameter 593

discover, Konfigurationsparameter 552

discover_db, Konfigurationsparameter 552

discover_inst, Konfigurationsparameter 553

Discover-Serverinstanz, Konfigurationsparameter 553

Discovery-Modus, Konfigurationsparameter 552

dlchktime, Konfigurationsparameter 471

DLFM_ASNCOPYD_PORT 647

DLFM_BACKUP_DIR_NAME 647

DLFM_BACKUP_TARGET 647

DLFM_BACKUP_TARGET_LIBRARY 647

DLFM_GC_MODE 647

DLFM_INSTALL_PATH 647

DLFM_PORT 647

DLFM_START_ASNCOPYD 647

DLFM_TSM_MGMTCLASS 647

Dokumentation 763, 764
 Bedingungen für die Verwendung 773

Durchführen von Vergleichstests
 Beispielbericht 426

db2batch, Tool 412

SQL-Anweisungen 410

Testbeispiele 414

Testmethoden 409

Testprozess 424

Übersicht 409

Vorbereitung 410

Zusammenfassung 424

dyn_query_mgmt, Konfigurationsparameter
 für Query Patroller 534

Dynamische Abfragen
 Optimierungsklasse einstellen 184

Dynamische Konfiguration 281

Dynamisches SQL
 Isolationsstufe angeben 66

dynexpln, Tool
 Ausgabebeschreibung 719

Fehlernachrichten 273

Parameter 258

Syntax 258

dynexpln, Tool (*Forts.*)
 Verwendungshinweise 273

DYNEXPLN_OPTIONS, Umgebungsvariable
 Beschreibung 273

DYNEXPLN_PACKAGE, Umgebungsvariable
 Beschreibung 273

E

E/A-Blockgröße für Clients, Konfigurationsparameter 464

EDU (Engine Dispatchable Unit, zuteilbare Einheit der Steuerkomponente)
 Agenten 153

Ein-/Ausgabeparallelität
 verwalten 148

Vorablesezugriff 147

Einfügen von Daten
 Prozess 150

wenn Tabelle nach Index angeordnet 150

Einfügungen
 nicht festgeschriebene überspringen 95

EISCAN, Operator
 Definition 699

Entwurf
 Designadvisor 51

Ereignismomentaufnahmen 405

Erste aktive Protokolldatei, Konfigurationsparameter 498

exec_exp_task, Konfigurationsparameter 593

EXPLAIN_ARGUMENT, Tabelle 665

EXPLAIN_DIAGNOSTIC, Tabelle 670

EXPLAIN_DIAGNOSTIC_DATA, Tabelle 671

EXPLAIN-Einrichtung
 Abfragen auf föderierte Datenbanken auswerten 249

Beschreibung 257

Daten erfassen 269

db2exfmt 258

db2expln 258

dynexpln 258

erfasste Informationen verwenden 260

föderierte Datenbanken 250

Informationen, angezeigte
 Datenobjekte 264

Datenoperatoren 268

Instanzen 264

verschiedene 737

Informationen analysieren 271

Momentaufnahmen, Erstellung 269

Übersicht 258

verwenden 711

Visual Explain 258

EXPLAIN_INSTANCE, Tabelle 673

EXPLAIN-Instanz 261

EXPLAIN_OBJECT, Tabelle 676

EXPLAIN_OPERATOR, Tabelle 679

EXPLAIN_PREDICATE, Tabelle 681

EXPLAIN_STATEMENT, Tabelle 684

EXPLAIN_STREAM, Tabelle 686

EXPLAIN-Tabellen
 Formatierungstool für Daten von 755

Organisation 261

Übersicht 663

F

failarchpath, Konfigurationsparameter 509

FCM-Kanäle 555

- fcf_num_buffers, Konfigurationsparameter 554
- fcf_num_channel
 - Konfigurationsparameter 555
- FCF-Pufferpool
 - Illustration 291
 - Speicheranforderungen 291
- fed_noauth, Konfigurationsparameter 579
- federated, Konfigurationsparameter 569
- Fehlerbehebung
 - Lernprogramme 772
 - Onlineinformationen 772
- Fehlerbestimmung
 - Lernprogramme 772
 - Onlineinformationen 772
- fenced_pool, Konfigurationsparameter 493
- Fern
 - Leistung 760
- FETCH, Operator
 - Definition 700
- FILTER, Operator
 - Definition 700
- Föderierte Datenbanken
 - Abfrageinformationen 736
 - Auswertungsort einer Abfrage analysieren 249
 - db2expln-Ausgabe für Abfrage auf 747
 - globale Analyse von Abfragen 250
 - globale Optimierung 253
 - Pushdown-Analyse 244
 - Serveroptionen 256
 - Steuerung des gemeinsamen Zugriffs 61
 - Systemunterstützung, Konfigurationsparameter 569
- Fragmenteliminierung
 - siehe Datenpartitionseliminierung 21
- Free Space Control Record (FSCR, Steuersatz für freien Speicherbereich)
 - in MDC-Tabellen 387
 - in Standardtabellen 384

G

- Geänderte Seiten protokollieren, Konfigurationsparameter 527
- Gedruckte Bücher
 - bestellen 767
- GENROW, Operator
 - Definition 701
- Governor-Tool
 - Beispielkonfigurationsdatei 119
 - Beschreibung 107
 - Dämonen
 - Beschreibung 108
 - Konfigurationsdatei
 - Regelbeschreibungen 111
 - konfigurieren 110
 - Protokolldateiabfragen 124
 - Protokolldateien, erstellte 120
 - Regelelemente 114
 - starten 109
 - stoppen 109
- Granularität
 - Sperre
 - Übersicht 74
- Größe des Anweisungszwischenspeichers, Konfigurationsparameter 460
- Größe des gemeinsam benutzten Datenbankspeichers, Konfigurationsparameter 434
- group_plugin, Konfigurationsparameter 580

- groupheap_ratio, Konfigurationsparameter 451
- GRPBY, Operator
 - Definition 701
- Gruppierung, Auswirkung auf Zugriffsplan 202

H

- hadr_db_role, Konfigurationsparameter 519
- hadr_local_host, Konfigurationsparameter 520
- hadr_local_svc, Konfigurationsparameter 520
- hadr_remote_host, Konfigurationsparameter 521
- hadr_remote_inst, Konfigurationsparameter 521
- hadr_remote_svc, Konfigurationsparameter 522
- hadr_syncmode, Konfigurationsparameter 522
- hadr_timeout, Konfigurationsparameter 523
- Hash-Join
 - Beschreibung 221
 - Optimierung der Leistung 221
- Hauptspeicher
 - applheapsz, Konfigurationsparameter 454
 - aslheapsz, Konfigurationsparameter 461
 - dbheap, Konfigurationsparameter 437
 - Größe des Anweisungszwischenspeichers, Konfigurationsparameter 460
 - Größe des Paketcache, Konfigurationsparameter 442
 - Instanzspeicher, Konfigurationsparameter 468
 - Organisation der Nutzung 286
 - Pufferpoolzuordnung bei Start 134
 - Schwellenwert für Sortierspeicher, Konfigurationsparameter 456
 - Sortierzwischenspeichergöße, Konfigurationsparameter 458
 - Zeitpunkt der Zuordnung 286
- Hauptspeicherzuordnung
 - Anpassungsparameter 293
- health_mon, Konfigurationsparameter 563
- Hilfe
 - anzeigen 768
 - für SQL-Anweisungen 768
- HSJOIN, Operator
 - Definition 701

I

- IN-Modus (Intent None)
 - Sperrmodusbeschreibung 71
- Inaktivieren
 - selbstoptimierender Speicher 297
- INCLUDE, Klausel
 - Auswirkung auf erforderlichen Speicherplatz für Indizes 384
- Index
 - über XML-Daten
 - Übersicht 39
- indexrec, Konfigurationsparameter 523
- Indexsuchen
 - Sperrmodi
 - Standardtabellen 87
 - Suchvorgänge 31
 - Verwendung 31
 - Zeiger auf vorherige Blattseiten 31
 - Zugriff auf Daten 193
- Indizes
 - Assistenten zum Entwerfen 51
 - asynchrone Bereinigung 48

Indizes (Forts.)

- Auswirkung des Typs auf Sperren der nächsten Schlüssel 91
 - bei partitionierten Tabellen 41
 - bereinigen 48
 - Block
 - Suchsperrmodus 101
 - Clusterbildung 45, 384
 - Clusterverhältnis 199
 - Datenzugriffsmethode 196
 - Defragmentierung, online 392
 - EXPLAIN-Informationen zur Analyse der Verwendung 271
 - Katalogstatistiken sammeln 325
 - Pflege 390
 - planen 33
 - reorganisieren
 - automatisch 374
 - Beschreibung 366
 - Statistiken
 - erfasste detaillierte Daten 338
 - Regeln für die manuelle Aktualisierung 357
 - Struktur 31
 - Suchoperationen 31
 - Tipps zur Leistung 35
 - Typ 2, Beschreibung 390
 - Verhalten bei partitionierten Tabellen 41
 - verwalten
 - für MDC-Tabellen 387
 - für Standardtabellen 384
 - Vorteile 29
 - Vorteile und Nachteile 29
 - Zeitpunkt für Indexneuerstellung, Konfigurationsparameter 523
- ## Informationszentrale
- aktualisieren 769
 - in unterschiedlichen Sprachen anzeigen 768
 - Versionen 768
- ## INSERT, Operator
- Definition 702
- ## instance_memory, Konfigurationsparameter
- 468
- ## Instanzen
- EXPLAIN-Informationen 264
- ## Instanzspeicher, Konfigurationsparameter
- 468
- ## Integritätsbedingungen
- EXPLAIN-Tabellen 663
- ## intra_parallel, Konfigurationsparameter
- 559
- ## IS-Modus (Intent Share)
- Sperrrmodusbeschreibung 71
- ## Isolationsstufen
- angeben 66
 - Auswirkung auf Leistung 62
 - Auswirkung auf Sperrgranularität 69
 - sperrenbezogene Leistungsprobleme verhindern 82
- ## IX-Modus (Intent Exclusive)
- Sperrrmodusbeschreibung 71
- ## IXAND, Operator
- Definition 702
- ## IXSCAN, Operator
- Definition 703

J

- Java Development Kit, Installationspfad (DAS), Konfigurationsparameter 595
- Java Development Kit-Installationspfad, Konfigurationsparameter 569

- java_heap_sz, Konfigurationsparameter 469
- JDBC (Java Database Connectivity)
 - Isolationsstufe angeben 66
- jdk_64_path, Konfigurationsparameter 594
- jdk_path, DAS-Konfigurationsparameter 595
- jdk_path, Konfigurationsparameter 569
- Joins
 - db2expln-Informationen 728
 - Definition 220
 - Eliminierung von Redundanzen 238
 - EXPLAIN-Informationen zur Analyse der Methoden 271
 - gemeinsam benutzte Spaltenberechnung 238
 - Hashverfahren, Beschreibung 221
 - in partitionierten Datenbanken 231
 - Methoden, Liste 221
 - Mischverfahren, Beschreibung 221
 - mit Verschachtelungsschleife, Beschreibung 221
 - rundgesendete äußere Tabelle 231
 - rundgesendete innere Tabelle 231
 - Strategien des Optimierungsprogramms für optimale 224
 - Tabellenwarteschlangenstrategie in partitionierten Datenbanken 229
- Typen
 - gezielt übertragene äußere Tabelle 231
 - gezielt übertragene innere Tabelle 231
- Unterabfragenumsetzung durch Optimierungsprogramm 238
- zusammengefasste Tabellen 231

K

- Kardinalitätsschätzungen
 - mit Statistiksichten 214
- Katalogcachegröße, Konfigurationsparameter 432
- Katalogstatistiken
 - erfassen
 - Anforderungen und Methode 322
 - Indexstatistiken 325
 - Verteilungsstatistiken zu bestimmten Spalten 324
 - erfasste detaillierte Indexdaten 338
 - für benutzerdefinierte Funktionen 341
 - für Unterelemente in Spalten 340
 - Indexclusterverhältnis 199
 - Katalogtabellen, Beschreibungen 330
 - manuelle Aktualisierung, Regeln
 - Indexstatistiken 357
 - Kurznamen 356
 - Spaltenstatistiken 354
 - Tabellen 356
 - Verteilung 355
 - manuelle Aktualisierung, Richtlinien 353
 - manuelle Anpassung zur Modellierung 349
 - Richtlinien aktualisieren 326
 - Richtlinien erfassen 326
 - Verteilungsstatistik
 - erweitertes Verwendungsbeispiel 345
 - Häufigkeit 335
 - Quantil 335
 - Zeitpunkt der Erfassung 335
 - verwenden 319
 - Zeitpunkt der Erfassung 319
 - zur Modellierung der Produktionsdatenbanken verwenden 350
- Katalogtabellen
 - Beschreibung 330
- keepfenced, Konfigurationsparameter 494

Knoten

- Antwortzeit für Verbindung 554
- koordinierende Agenten, maximale Anzahl 485
- maximale Zeitdifferenz zwischen 557

Knotenname für ferne Datenservices, Konfigurationsparameter 550

Kommunikation

- Antwortzeit für Verbindung 554

Konfiguration

- Datenbankparameter ändern 277
- dynamische 281
- Parameterzusammenfassung
 - Datenbank 305
 - Datenbankmanager 305

Konfigurationsadvisor

- empfohlene Werte generieren 284

Konfigurationsdateien

- Beschreibung 275
- Governor-Tool 110
 - Beispiel 119
 - Regelbeschreibungen 111
 - Regelemente 114
- Position 275

Konfigurationsparameter

- agent_stack_sz 452
- agentpri 482
- alt_collate 535
- app_ctl_heap_sz 449
- appgroup_mem_sz 450
- applheapsz 454
- archretrydelay 507
- aslheapsz 461
- audit_buf_sz 466
- authentication 575
- authentication (DAS) 589
- autonomic_switches 547
- autorestart 518
- avg_appls 484
- backup_pending 538
- Beschreibung 275
- blk_log_dsk_ful 508
- catalog_noauth 576
- catalogcache_sz 432
- chnpgps_thresh 476
- clnt_krb_plugin 577
- clnt_pw_plugin 578
- codepage 535
- codeset 535
- collate_info 536
- comm_bandwidth 566
- conn_elapse 554
- contact_host 590
- cpuspeed 567
- das_codepage 591
- das_territory 591
- dasadm_group 591
- database_consistent 538
- database_level 537
- database_memory 434
- db_mem_thresh 436
- db2system 592
- dbheap 437
- dft_account_str 568
- dft_degree 541
- dft_extent_sz 477
- dft_loadrec_ses 518
- dft_monswitches 565

Konfigurationsparameter (Forts.)

- dft_mttb_types 541
- dft_prefetch_sz 477
- dft_queryopt 542
- dft_refresh_age 542
- dft_sqlmathwarn 543
- dftdbpath 578
- diaglevel 561
- diagpath 562
- dir_cache 467
- discover 552
- discover (DAS) 593
- discover_db 552
- discover_inst 553
- dlchktime 471
- dyn_query_mgmt 534
- exec_exp_task 593
- failarchpath 509
- fcm_num_buffers 554
- fcm_num_channels 555
- fed_noauth 579
- federated 569
- fenced_pool 493
- group_plugin 580
- groupheap_ratio 451
- hadr_db_role 519
- hadr_local_host 520
- hadr_local_svc 520
- hadr_remote_host 521
- hadr_remote_inst 521
- hadr_remote_svc 522
- hadr_syncmode 522
- hadr_timeout 523
- health_mon 563
- indexrec 523
- instance_memory 468
- intra_parallel 559
- java_heap_sz 469
- jdk_64_path 594
- jdk_path 569
- jdk_path (DAS) 595
- keepfenced 166, 494
- local_gssplugin 580
- locklist 438
- locktimeout 472
- log_retain_status 539
- logarchmeth1 509
- logarchmeth2 510
- logarchopt1 510
- logarchopt2 511
- logbufsz 441
- logfilsiz 497
- loghead 498
- logindexbuild 511
- logpath 498
- logprimary 499
- logretain 511
- logsecond 501
- max_connections 485
- max_connretries 556
- max_coordagents 485
- max_querydegree 559
- max_time_diff 557
- maxagents 487
- maxappls 487
- maxcagents 489
- maxfilop 490

Konfigurationsparameter (Forts.)

- maxlocks 474
- maxlog 502
- maxtotfilop 491
- min_dec_div_3 463
- mincommit 512
- mirrorlogpath 502
- mit Auswirkung auf Anzahl von Agenten 165
- mit Auswirkung auf die Abfrageoptimierung 186
- mon_heap_sz 470
- multipage_alloc 539
- newlogpath 504
- nname 550
- nodetype 570
- notifylevel 563
- num_db_backups 526
- num_freqvalues 544
- num_initagents 491
- num_initfenced 496
- num_iocleaners 478
- num_ioservers 480
- num_poolagents 492
- num_quantiles 546
- numarchretry 514
- numdb 570
- numlogspan 505
- numsegs 481
- overflowlogpath 506
- pagesize 537
- pckcachesz 442
- query_heap_sz 455
- rec_his_retentn 526
- release 537
- restore_pending 540
- restrict_access 589
- resync_interval 530
- rollfwd_pending 540
- rqrioblk 464
- sched_enable 595
- sched_userid 596
- self_tuning_mem 444
- seqdetect 481
- sheapthres 456
- sheapthres_shr 446
- smtp_server 596
- softmax 514
- sortheap 458
- spm_log_file_sz 530
- spm_log_path 531
- spm_max_resync 532
- spm_name 532
- srv_plugin_mode 583
- srvcon_auth 581
- srvcon_gssplugin_list 581
- srvcon_pw_plugin 582
- start_stop_time 557
- stat_heap_sz 459
- stmtheap 460
- svcname 551
- sysadm_group 583
- sysctrl_group 584
- sysmaint_group 585
- sysmon_group 586
- territory 538
- tm_database 533
- toolscat_db 597
- toolscat_inst 597

Konfigurationsparameter (Forts.)

- toolscat_schema 598
- tp_mon_name 571
- trackmod 527
- trust_allclnts 587
- trust_clntauth 588
- tsm_mgmtclass 527
- tsm_nodename 528
- tsm_owner 528
- tsm_password 529
- user_exit_status 540
- userexit 516
- util_heap_sz 448
- util_impact_lim 573
- vendoropt 517
- Kontaktaufnahme mit IBM 781
- Koordinatoragent
 - Beschreibung 158, 166
 - Verbindungskonzentratornutzung 171
- Kosten-Nutzen-Analyse
 - Designadvisor 53
- Kurznamen
 - Statistiken manuell aktualisieren 356

L

Leistung

- db2batch, Vergleichstest-Tool 412
- Elemente 3
- Fernzugriff auf Informationen aktivieren 758
- Grenzen der Optimierung 6
- Informationen anzeigen 759
- Isolationsstufen 62
- optimieren
 - Benutzereingaben 6
 - Tipps zum Einstieg 7
- Optimierungsklasse anpassen 184
- Optimierungsprozess entwickeln 5
- Plattenspeicherfaktoren 8
- Werte zurücksetzen 761
- Windows 759
- Zugriff auf ferne Informationen 760

Leistungsoptimierung

- EXPLAIN-Informationen verwenden 260
- Fehlerbehebung 6
- Richtlinien 3
- Sortiervorgänge 129
- Sperren 82
- Übersicht 3

Leistungssteigerung

- Bindeoption REOPT 205
- relationale Indizes 35

Lernprogramme

- Fehlerbestimmung und Fehlerbehebung 772
- Visual Explain 772

- local_gssplugin, Konfigurationsparameter 580

LOCK TABLE, Anweisung

- sperrenbezogene Leistungsprobleme verhindern 82
- Sperreneskalationen minimieren 76

locklist, Konfigurationsparameter

- Auswirkung auf die Abfrageoptimierung 186
- Auswirkung auf Sperrgranularität 69
- Beschreibung 438

LOCKSIZE, Klausel

- Auswirkung auf Sperrgranularität 69
- Sperrgranularität angeben 74

- locktimeout, Konfigurationsparameter 472

- log_retain_status, Konfigurationsparameter 539
- logarchmeth1, Konfigurationsparameter 509
- logarchmeth2, Konfigurationsparameter 510
- logarchopt1, Konfigurationsparameter 510
- logarchopt2, Konfigurationsparameter 511
- LOGBUFSZ, Konfigurationsparameter 441
- logfilsiz, Konfigurationsparameter 497
- loghead, Konfigurationsparameter 498
- logindexbuild, Konfigurationsparameter 511
- Logische Knoten, siehe Datenbankpartitionsserver 166
- Logische Partitionen
 - mehrere 166
- logpath, Konfigurationsparameter 498
- logprimary, Konfigurationsparameter 499
- logretain, Konfigurationsparameter 511
- logsecond, Konfigurationsparameter 501

M

- Materialized Query Tables (MQTs, gespeicherte Abfrage-
tabellen)
 - automatisch aktualisierte Übersichtstabellen 11
 - repliziert, in partitionierten Datenbanken 206
- max_connections, Konfigurationsparameter
 - für Agentenverwaltung 163
- max_connretries, Konfigurationsparameter 556
- max_coordagents, Konfigurationsparameter 485
 - für Agentenverwaltung 163
- max_logicagents, Konfigurationsparameter 485
- max_querydegree, Konfigurationsparameter 559
- max_time_diff, Konfigurationsparameter 557
- maxagents, Konfigurationsparameter 487
 - Auswirkung auf Speichernutzung 286
- maxappls, Konfigurationsparameter 487
 - Auswirkung auf Speichernutzung 286
- maxcagents, Konfigurationsparameter 489
- maxcoordagents, Konfigurationsparameter 286
- MAXDARI, Konfigurationsparameter
 - umbenannt in fenced_pool, Konfigurationsparameter 493
- maxfilop, Konfigurationsparameter 490
- Maximale Anzahl abgeschirmter Prozesse, Konfigurations-
parameter 493
- Maximale Anzahl aktiver Anwendungen, Konfigurations-
parameter 487
- Maximale Anzahl gleichzeitig aktiver Agenten, Konfigurations-
parameter 489
- Maximale Anzahl gleichzeitig aktiver Datenbanken,
Konfigurationsparameter 570
- Maximale Anzahl koordinierender Agenten, Konfigurations-
parameter 485
- Maximale Anzahl offener Datenbankdateien pro Anwendung,
Konfigurationsparameter 490
- Maximale Anzahl von Agenten, Konfigurationsparameter 487
- Maximale Anzahl von Sperren pro Anwendung,
Konfigurationsparameter 474
- Maximale Speichergröße für Anwendungsgruppe,
Konfigurationsparameter 450
- Maximale Zeitdifferenz zwischen Knoten, Konfigurations-
parameter 557
- Maximale Zwischenspeichergröße für Java-Interpreter,
Konfigurationsparameter 469
- Maximaler Grad der Parallelität bei Abfragen, Konfigurations-
parameter 559
 - Auswirkung auf die Abfrageoptimierung 186
- Maximaler Protokollspeicher pro Transaktion, Konfigurations-
parameter 502

- Maximaler Speicher für Sperrenliste, Konfigurations-
parameter 438
- maxlocks, Konfigurationsparameter 474
 - Zeitpunkt für Auslösen der Sperreneskulation angeben 76
- maxlog, Konfigurationsparameter 502
- maxtotfilop, Konfigurationsparameter 491
- MDC-Tabellen (MDC = mehrdimensionales Clustering)
 - bei partitionierten Tabellen 13
 - Optimierungsstrategien 19
 - Sperren auf Blockebene 69
 - Sperrr Modi
 - Tabellen- und RID-Indexsuchen 96
 - Verwaltung von Tabellen und Indizes 387
- Mehrpartitionsdatenbanken
 - Migration von Einzelpartitionsdatenbanken durchführen
Designadvisor 57
- Methoden
 - Join mit Verschachtelungsschleife 221
- min_dec_div_3, Konfigurationsparameter 463
- mincommit, Konfigurationsparameter 512
- MINPCTUSED, Klausel
 - für Onlineindexdefragmentierung 384
- mirrorlogpath, Konfigurationsparameter 502
- Mischjoin 221
- Modellieren von Anwendungsleistung
 - Katalogstatistik verwenden 350
 - manuell angepasste Katalogstatistik verwenden 349
- Momentaufnahmen
 - Überwachung zu einem Zeitpunkt 405
- Momentaufnahmen, Überwachung
 - Ausgabe für Datenpartitionen interpretieren 375
 - für Datenpartitionen 375
- mon_heap_sz, Konfigurationsparameter 470
- Monitorschalter
 - aktualisieren 405
- MQTs (Materialized Query Tables, gespeicherte Abfrage-
tabellen)
 - automatisch aktualisierte Übersichtstabellen 11
 - repliziert, in partitionierten Datenbanken 206
- MSJOIN, Operator
 - Definition 704
- multipage_alloc, Konfigurationsparameter 539

N

- NetBIOS
 - Workstationname, Konfigurationsparameter 550
- Neuverteilen von Daten
 - Notwendigkeit 397
 - schrittweise Umverteilungsprozeduren 401
- newlogpath, Konfigurationsparameter 504
- Nicht festgeschriebene Daten
 - Steuerung des gemeinsamen Zugriffs 61
- Nicht wiederholbare Lesevorgänge
 - Steuerung des gemeinsamen Zugriffs 61
- NLJOIN, Operator
 - Definition 704
- nname, Konfigurationsparameter 550
- NO_SORT_MGJOIN 624
- NO_SORT_NLJOIN 624
- nodetype, Konfigurationsparameter 570
- notifylevel, Konfigurationsparameter 563
 - für Fehlerbehebung bei Sperreneskulation angeben 76
- NS-Modus (Next Key Share)
 - Sperrrmodusbeschreibung 71
- num_db_backups, Konfigurationsparameter 526
- num_freqvalues, Konfigurationsparameter 544

- num_initagents, Konfigurationsparameter 491
- num_initfenced, Konfigurationsparameter 496
- num_iocleaners, Konfigurationsparameter 478
- num_ioservers, Konfigurationsparameter 480
- num_poolagents, Konfigurationsparameter 492
- num_quantiles, Konfigurationsparameter 546
- numarchretry, Konfigurationsparameter 514
- numdb, Konfigurationsparameter 570
 - Auswirkung auf Speichernutzung 286
- numlogspan, Konfigurationsparameter 505
- numsegs, Konfigurationsparameter 481
- NW-Modus (Next Key Weak Exclusive)
 - Sperrmodusbeschreibung 71

O

- Objekte
 - Leistung unter Windows 759
- ODBC (Open Database Connectivity)
 - Isolationsstufe angeben 66
- Operationen
 - zusammengefügt oder versetzt durch Optimierungsprogramm 236
- Operatoren
 - CMPEXP 699
 - DELETE 699
 - EISCAN 699
 - EXPLAIN-Informationen 268
 - FETCH 700
 - FILTER 700
 - GENROW 701
 - GRPBY 701
 - HSJOIN 701
 - INSERT 702
 - IXAND 702
 - IXSCAN 703
 - MSJOIN 704
 - NLJOIN 704
 - PIPE 705
 - RETURN 705
 - RIDSCN 706
 - RPD 706
 - SHIP 706
 - SORT 707
 - TBSCAN 707
 - TEMP 708
 - TQUEUE 709
 - UNION 709
 - UNIQUE 709
 - UPDATE 710
 - XANDOR
 - Beispielausgabe 749
 - XISCAN
 - Beispielausgabe 749, 752
 - XSCAN
 - Beispielausgabe 751
- Optimieren
 - Einschränkungen 6
 - Fehlerbehebung 6
 - Leistung
 - Sortiervorgänge 129
 - Übersicht 3
 - Leistungsoptimierungsprozess
 - Übersicht 5
 - Parameter für Hauptspeicherzuordnung 293
 - Richtlinien 3

- Optimierung
 - Abfrageumschreibung, Methoden 236
 - Joins
 - Definition 220
 - in partitionierter Datenbank 231
 - Strategien für optimale 224
 - partitionierte Tabellen 21
 - partitionsinterne Parallelität 208
 - relevante Statistikdaten anzeigen 213
 - Strategien für MDC-Tabellen 19
 - Verteilungsstatistik 343
 - Zugriffspläne 192
 - Auswirkungen von Sortierungen und Gruppierungen 202
 - Indexzugriffsmethoden 196
 - mit Index 193
 - Spaltenkorrelation 227
- Optimierung, Partition
 - feststellen 302
- Optimierungsklassen
 - auswählen 179
 - Beschreibung 181
 - einstellen 184
- Optimierungsprogramm
 - Statistiksichten
 - erstellen 212
 - Übersicht 211
- overflowlogpath, Konfigurationsparameter 506
- OVERHEAD, Einstellung
 - durch Zeilenblockung verringern 125

P

- pagesize
 - Konfigurationsparameter 537
- Parallelität
 - Ein-/Ausgabe
 - Serverkonfiguration 146
 - verwalten 148
 - maximaler Grad der Parallelität bei Abfragen, Konfigurationsparameter 559
 - partitionsintern
 - Optimierungsstrategien 208
 - partitionsinterne Parallelität aktivieren, Konfigurationsparameter 559
- Parallelverarbeitung
 - db2expln, Tool
 - Informationen, angezeigte 733
- Partitionierte Datenbanken
 - Datenumverteilung, Fehlerbehebung 403
 - Dekorrelierung einer Abfrage 240
 - Joinmethoden 231
 - Joinstrategien 229
 - Neuverteilen von Daten 397
 - replizierte MQT 206
- Partitionierte Datenbanken, Umgebungen
 - selbstoptimierender Speicher 300, 302
- Partitionierte Tabellen
 - Indexverhalten 41
 - mit MDC-Tabellen (MDC = mehrdimensionales Clustering) 13
 - optimieren 21
 - reorganisieren 375
 - sperrern 104
- Partitionsinterne Parallelität
 - db2expln, Tool
 - Ausgabebeispiel 741

- Partitionsinterne Parallelität (*Forts.*)
 - db2expln, Tool (*Forts.*)
 - Ausgabebeispiele 745
 - Optimierungsstrategien 208
- Partitionsübergreifende Parallelität
 - db2expln, Tool
 - Ausgabebeispiel 743
 - Ausgabebeispiele 745
- pckcachesz, Konfigurationsparameter 442
- PCTFREE, Klausel
 - Speicherplatz für Clustering reservieren 384
- Pfad für Protokollspiegelung, Konfigurationsparameter 502
- Phantomzeilen
 - Steuerung des gemeinsamen Zugriffs 61
- PIPE, Operator
 - Definition 705
- Platten
 - Speicherleistungsfaktoren 8
- Poolgröße für Agenten
 - steuern 492
- Protokolldateien
 - Governor-Tool 124
- Protokolldateispeicher
 - Anforderungen für Datenumverteilung 400
- Protokolle
 - Anzahl primärer Protokolldateien, Konfigurationsparameter 499
 - Anzahl sekundärer Protokolldateien, Konfigurationsparameter 501
 - bei voller Protokollplatte blockieren, Konfigurationsparameter 508
 - Beibehalten von Protokollen aktivieren, Konfigurationsparameter 511
 - Benutzerexit aktivieren, Konfigurationsparameter 516
 - erste aktive Protokolldatei, Konfigurationsparameter 498
 - erstellt von Governor-Tool 120
 - Größe der Protokolldateien, Konfigurationsparameter 497
 - NetBIOS-Workstationname, Konfigurationsparameter 550
 - newlogpath, Konfigurationsparameter 504
 - Pfad für Protokollspiegelung, Konfigurationsparameter 502
 - Pfad zu Protokolldateien, Konfigurationsparameter 498
 - Protokollpuffergröße, Konfigurationsparameter 441
 - Statusanzeiger für Beibehalten der Protokolle, Konfigurationsparameter 539
 - TCP/IP-Servicename, Konfigurationsparameter 551
 - Überlaufprotokollpfad, Konfigurationsparameter 506
 - vor bedingtem Prüfpunkt zu schreibende Protokollsätze, Konfigurationsparameter 514
- Protokollieren
 - Protokollsätze aufbewahren
 - Definition 151
 - Umlaufprotokollierung
 - Definition 151
- Protokollpuffer 151
- Prozeduren
 - STEPWISE_REDISTRIBUTE_DBPG 401
- Prozesse
 - Übersicht 155
- Prozessmodell
 - Aktualisierungen 139
 - für DB2 158
 - für SQL- und XQuery-Compiler 175
 - Übersicht 166
- Pufferpools
 - Auswirkung auf die Abfrageoptimierung 186

- Pufferpools (*Forts.*)
 - blockbasiert
 - Leistung beim Vorabesezugriff 144
 - Datenseitenverwaltung 138
 - große, Vorteile 134
 - mehrere
 - Seitengrößen 134
 - verwalten 134
 - Vorteile 134
 - Seitenbereinigung, Methoden 140
 - Seitenlöschfunktionen optimieren 136
 - Speicherzuordnung bei Start 134
 - Übersicht 131
- Pushdown-Analyse
 - für Abfragen auf föderierte Datenbanken 244

Q

- Quantilverteilungsstatistik 335
- query_heap_sz, Konfigurationsparameter 455

R

- rec_his_retentn, Konfigurationsparameter 526
- Recovery
 - aktualisierende Recovery anstehend, Anzeiger, Konfigurationsparameter 540
 - Anzahl der Datenbank-Backups, Konfigurationsparameter 526
 - automatischer Neustart aktiviert, Konfigurationsparameter 518
 - Backup anstehend, Anzeiger, Konfigurationsparameter 538
 - Benutzerexitstatusanzeiger, Konfigurationsparameter 540
 - restore_pending, Konfigurationsparameter 540
 - Standardanzahl Recoverysitzungen, Konfigurationsparameter 518
 - Statusanzeiger für Beibehalten der Protokolle, Konfigurationsparameter 539
 - Zeitpunkt für Indexneuerstellung, Konfigurationsparameter 523
- Registrierdatenbankvariablen
 - DB2_ALLOCATION_SIZE 630
 - DB2_ALTERNATE_GROUP_LOOKUP 607
 - DB2_ANTIJOIN 624
 - DB2_APM_PERFORMANCE 630
 - DB2_AVOID_PREFETCH 630
 - DB2_AWE 630
 - DB2_CLPPROMPT 621
 - DB2_COMMIT_ON_EXIT 649
 - DB2_CONNRETRIES_INTERVAL 602
 - DB2_CREATE_DB_ON_PATH 649
 - DB2_DISABLE_FLUSH_LOG 649
 - DB2_DISCONNECT_ON_INTERRUPT 649
 - DB2_DISPATCHER_PEEKTIMEOUT 649
 - DB2_DJ_COMM 649
 - DB2_DJ_INI 649
 - DB2_DOCHOST 649
 - DB2_DOCPORT 649
 - DB2_ENABLE_AUTOCONFIG_DEFAULT 649
 - DB2_ENABLE_BUFDPD 630
 - DB2_ENABLE_LDAP 649
 - DB2_EVALUNCOMMITTED 630
 - DB2_EXTENDED_IO_FEATURES 630
 - DB2_EXTENDED_OPTIMIZATION 630
 - DB2_EXTSECURITY 649

Registrierdatenbankvariablen (Forts.)

DB2_FALLBACK 649
 DB2_FMP_COMM_HEAPSZ 649
 DB2_FORCE-NLS_CACHE 615
 DB2_GRP_LOOKUP 649
 DB2_HADR_BUF_SIZE 649
 DB2_INDEX_TYPE2 602
 DB2_INLIST_TO_NLJN 624
 DB2_IO_PRIORITY_SETTING 630
 DB2_KEEPTABLELOCK 630
 DB2_LGPAGE_BP 630
 DB2_LIC_STAT_SIZE 602
 DB2_LIKE_VARCHAR 624
 DB2_LOAD_COPY_NO_OVERRIDE 649
 DB2_MAP_XML_AS_CLOB_FOR_DLC 649
 DB2_MAX_CLIENT_CONNRTRIES 602
 DB2_MAX_LOB_BLOCK_SIZE 649
 DB2_MEM_TUNING_RANGE 630
 DB2_MINIMIZE_LISTPREFETCH 624
 DB2_MMAP_READ 630
 DB2_MMAP_WRITE 630
 DB2_NEW_CORR_SQ_FF 624
 DB2_NEWLOGPATH2 649
 DB2_NO_FORK_CHECK 630
 DB2_NO_MPFA_FOR_NEW_DB 630
 DB2_NUM_CKPW_DAEMONS 649
 DB2_NUM_FAILOVER_NODES 623
 DB2_OBJECT_TABLE_ENTRIES 630
 DB2_OPT_MAX_TEMP_SIZE 624
 DB2_OVERRIDE_BPF 630
 DB2_PARALLEL_IO 607
 DB2_PARTITIONEDLOAD_DEFAULT 623
 DB2_PINNED_BP 630
 DB2_PRED_FACTORIZE 624
 DB2_REDUCED_OPTIMIZATION 624
 DB2_RESOLVE_CALL_CONFLICT 649
 DB2_RESOURCE_POLICY 630
 DB2_SELECTIVITY 624
 DB2_SERVER_CONTIMEOUT 649
 DB2_SKIPDELETED 630
 DB2_SKIPINSERTED 95
 DB2_SMS_TRUNC_TMPTABLE_THRESH 630
 DB2_SORT_AFTER_TQ 630
 DB2_SQLROUTINE_PREOPTS 624
 DB2_TRUNCATE_REUSESTORAGE 649
 DB2_TRUSTED_BINDIN 630
 DB2_USE_ALTERNATE_PAGE_CLEANING 630
 DB2_USE_DB2CCT2_JROUTINE 649
 DB2_USE_PAGE_CONTAINER_TAG 607
 DB2_UTIL_MSGPATH 649
 DB2_VENDOR_INI 649
 DB2_VIEW_REOPT_VALUES 602
 DB2_XBSA_LIBRARY 649
 DB2ACCOUNT 602
 DB2ADMINSERVER 649
 DB2ASSUMEUPDATE 630
 DB2BIDI 602
 DB2BPVARS 630
 DB2BQTIME 621
 DB2BQTRY 621
 DB2CHECKCLIENTINTERVAL 615
 DB2CHGPWD_ESE 623
 DB2CHKPTR 630
 DB2CHKSQlda 630
 DB2CLIINIPATH 649
 DB2CODEPAGE 602
 DB2COMM 615

Registrierdatenbankvariablen (Forts.)

DB2CONNECT_IN_APP_PROCESS 607
 DB2DBDFT 602
 DB2DBMSADDR 602
 DB2DEFPREP 649
 DB2DISCOVERYTIME 602
 DB2DMNBCKCTLR 649
 DB2DOMAINLIST 607
 DB2ENVLIST 607
 DB2GRAPHICUNICODESERVER 602
 DB2INCLUDE 602
 DB2INSTANCE 607
 DB2INSTDEF 602
 DB2INSTOWNER 602
 DB2INSTPROF 607
 DB2IQTIME 621
 DB2JD_PORT_NUMBER 615
 DB2LDAP_BASEDN 649
 DB2LDAP_CLIENT_PROVIDER 649
 DB2LDAP_KEEP_CONNECTION 649
 DB2LDAP_SEARCH_SCOPE 649
 DB2LDAPCACHE 649
 DB2LDAPHOST 649
 DB2LIBPATH 607
 DB2LOADREC 649
 DB2LOCALE 602
 DB2LOCK_TO_RB 649
 DB2MAXFSCSEARCH 630
 DB2MEMDISCLAIM 630
 DB2MEMMAXFREE 630
 DB2NBADAPTERS 615
 DB2NBCHECKUPTIME 615
 DB2NBDISCOVERRCVBUFS 602
 DB2NBINTRLISTENS 615
 DB2NBRECVBUFFSIZE 615
 DB2NBRECVNCBS 615
 DB2NBRESOURCES 615
 DB2NBSENDNCBS 615
 DB2NBSESSIONS 615
 DB2NBXTRANCBS 615
 DB2NODE 607
 DB2NOEXITLIST 649
 DB2NTMEMSIZE 630
 DB2NTNOCACHE 630
 DB2NTPRICLASS 630
 DB2NETWORKSET 630
 DB2OPTIONS 607
 DB2PATH 607
 DB2PORTRANCE 623
 DB2PRIORITIES 630
 DB2RCMD_LEGACY_MODE 607
 DB2REMOTEPREG 649
 DB2RETRY 615
 DB2RETRYTIME 615
 DB2ROUTINE_DEBUG 649
 DB2RQTIME 621
 DB2RSHCMD 615
 DB2RSHTIMEOUT 615
 DB2SATELLITEID 649
 DB2SERVICETPINSTANCE 615
 DB2SLOGON 602
 DB2SORCVBUF 615
 DB2SORT 649
 DB2SOSNDBUF 615
 DB2SYSPLEX_SERVER 615
 DB2TCP_CLIENT_CONTIMEOUT 615
 DB2TCP_CLIENT_RCVTIMEOUT 615

- Registrierdatenbankvariablen (*Forts.*)
 - DB2TCPCONNMGRS 615
 - DB2TERRITORY 602
 - DB2TRACEFLUSH 602
 - DB2TRACENAME 602
 - DB2TRACEON 602
 - DB2TRCSYSERR 602
 - DB2YIELD 602
 - DLFM_ASNCOPYD_PORT 647
 - DLFM_BACKUP_DIR_NAME 647
 - DLFM_BACKUP_TARGET_LIBRARY 647
 - DLFM_GC_MODE 647
 - DLFM_INSTALL_PATH 647
 - DLFM_PORT 647
 - DLFM_START_ASNCOPYD 647
 - DLFM_TSM_MGMTCLASS 647
 - Schlüsselwort NO_SORT_MGJOIN 624
 - Schlüsselwort NO_SORT_NLJOIN 624
 - Übersicht 601
- release, Konfigurationsparameter 537
- Releasestand der Datenbankkonfiguration, Konfigurationsparameter 537
- REOPT, Bindeoption
 - Beschreibung 205
- REORG INDEXES, Befehl 366
- REORG TABLE, Befehl
 - am Platz, im Onlinemodus 372
 - Auswahl der Reorg-Methode 372
 - klassisch, im Offlinemodus 372
- Reorganisieren
 - Indizes
 - automatisch 374
 - Tabellen 369
 - automatisch 374
- REORGANIZE TABLE, Befehl
 - Indizes und Tabellen 366
- restore_pending, Konfigurationsparameter 540
- restrict_access, Konfigurationsparameter 589
- RESTRICTIVE, Option
 - CREATE DATABASE
 - Datenbank, Konfigurationsparameter 589
- resync_interval, Konfigurationsparameter 530
- RETURN, Operator
 - Definition 705
- REXX, Programmiersprache
 - Isolationsstufe angeben 66
- RIDSCN, Operator
 - Definition 706
- rollforward, Dienstprogramm
 - aktualisierende Recovery anstehend, Anzeiger 540
- rollfwd_pending, Konfigurationsparameter 540
- RPD, Operator
 - Definition 706
- rqrioblk, Konfigurationsparameter 464
- RUNSTATS, Befehl
 - automatische Statistikerfassung 358, 359
 - erfasste Statistiken 319
 - Stichprobenstatistiken 329
 - verwenden 322
- S**

S-Modus (Share)

 - Sperrmodusbeschreibung 71

Satzkennung (RID)

 - in Standardtabellen 384

sched_enable, Konfigurationsparameter 595

sched_userid, Konfigurationsparameter 596

Schlüsselkardinalitäten

 - berechnen 204

Seiten

 - Daten 384

Seitengröße

 - Datenbankstandardwert 537

Seitenlöschfunktionen

 - Anzahl optimieren 136
 - für Datenseitenverwaltung 138

Seitenspiegelung

 - lange Objekte 151

Selbstoptimierender Speicher

 - aktivieren 296, 444
 - nicht einheitliche Umgebungen 302
 - Einschränkungen 304
 - inaktivieren 297
 - Übersicht 295
 - Überwachung 299
 - Umgebungen mit partitionierten Datenbanken 300, 302

SELECT, Anweisung

 - DISTINCT-Klauseln eliminieren 240

self_tuning_mem

 - Konfigurationsparameter 444

seqdetect, Konfigurationsparameter 481

Sequenzieller Vorabesezugriff

 - Beschreibung 142

SET CURRENT QUERY OPTIMIZATION, Anweisung 184

sheapthres, Konfigurationsparameter 456

sheapthres_shr, Konfigurationsparameter 446

SHIP, Operator

 - Definition 706

Sicherheit

 - Plug-ins
 - Konfigurationsparameter für Aktivierung 575
 - Konfigurationsparameter für Aktivierung von Plug-ins 577, 578, 581, 583

Sichten

 - Mischen durch das Optimierungsprogramm 238
 - Vergleichselemente mit Optimierungsprogramm verschieben 240

SIX-Modus (Share with Intent Exclusive)

 - Sperrmodusbeschreibung 71

smtp_server, Konfigurationsparameter 596

softmax, Konfigurationsparameter 514

SORT, Operator

 - Definition 707

sorheap, Konfigurationsparameter

 - Auswirkung auf die Abfrageoptimierung 186
 - Beschreibung 458

Sortieren

 - Auswirkung auf Zugriffsplan 202
 - Schwellenwert für Sortierspeicher, Konfigurationsparameter 456
 - Schwellenwert für Sortierspeicher für gemeinsame Sortiervorgänge 446
 - Sortierzwischenspeichergröße, Konfigurationsparameter 458
 - verwalten 129

Spalten

 - manuelles Aktualisieren von Statistiken, Regeln 354
 - Unterelemente, Statistiken erfassen für 340
 - Verteilungsstatistiken zu bestimmten erfassen 324

Spaltenfunktionen

 - Ausgabe des Tools db2expln 732

Spaltengruppen, Statistik 204

- Speicher-Tracker, Befehl
 - Beispielausgabe 131
- Speicheranforderungen
 - FCM-Pufferpool 291
- Speicherkonfiguration
 - selbstoptimierender Speicher 295
- Speichermodell
 - gemeinsamer Speicher des Datenbankmanagers 289
- Speicheroptimierung
 - Umgebungen mit partitionierten Datenbanken 302
- Sperrdauer
 - Beschreibung 71
- Sperren
 - Auswirkung des Anwendungstyps 73
 - Auswirkung des Datenzugriffsplans 85
 - Blockindexsuche, Modi 101
 - Deadlocks 78
 - gleichzeitig erteilen (Grant) 84
 - Kompatibilität 84
 - maximale Anzahl von Sperren pro Anwendung 474
 - maximaler Speicher für Sperrenliste 438
 - optimieren 82
 - Sperren der nächsten Schlüssel 91
 - Sperreneskalation
 - Fehlerbehebung 76
 - Standardtabellen
 - Modi und Zugriffspfade 87
 - Steuerung des gemeinsamen Zugriffs 69
 - Verhalten bei partitionierten Tabellen 104
 - Verzögerung 92
 - Wartestatus 84
 - Zeitintervall für Prüfung von Deadlocks, Konfigurationsparameter 471
- Sperren der nächsten Schlüssel
 - Indextyp, Wirkung 91
 - Indizes zur Minimierung konvertieren 366
 - Typ-2-Indizes 390
- Sperreneskalation
 - Fehlerbehebung 76
- Sperrenumwandlung 75
- Sperrgranularität
 - Übersicht 74
- Sperrmodi
 - Beschreibung 71
 - IN-Modus (Intent None) 71
 - IS-Modus (Intent Share) 71
 - IX-Modus (Intent Exclusive) 71
 - Kompatibilität 84
 - Konvertierung 75
 - MDC-Tabellen (MDC = mehrdimensionales Clustering)
 - Tabellen- und RID-Indexsuchen 96
 - NS-Modus (Next Key Share) 71
 - NW-Modus (Next Key Weak Exclusive) 71
 - S-Modus (Share) 71
 - SIX-Modus (Share with Intent Exclusive) 71
 - Standardtabellen 87
 - U-Modus (Update) 71
 - W-Modus (Weak Exclusive) 71
 - X-Modus (Exclusive) 71
 - Z-Modus (Super Exclusive) 71
- Sperrobjekte
 - Beschreibung 71
- spm_log_file_sz, Konfigurationsparameter 530
- spm_log_path, Konfigurationsparameter 531
- spm_max_resync, Konfigurationsparameter 532
- spm_name, Konfigurationsparameter 532
- SQL- und XQuery-Compiler
 - Prozessbeschreibung 175
- SQL-Anweisungen
 - EXPLAIN-Tools für 711
 - Größe des Anweisungszwischenspeichers, Konfigurationsparameter 460
 - Hilfe anzeigen 768
 - neu schreiben 236
 - optimieren
 - SELECT-Anweisungen 126
 - Optimierung
 - Konfigurationsparameter 186
 - Vergleichstests durchführen 410
- SQL und XQuery, Explain, Befehl 712
- SQLDBCON, Konfigurationsdatei 275
- SQLJ (eingebettetes SQL für Java)
 - Isolationsstufe angeben 66
- srv_plugin_mode, Konfigurationsparameter 583
- srvcon_auth, Konfigurationsparameter 581
- srvcon_gssplugin_list, Konfigurationsparameter 581
- srvcon_pw_plugin, Konfigurationsparameter 582
- Standardanzahl von SMS-Containern, Konfigurationsparameter 481
- Standarddatenbankpfad, Konfigurationsparameter 578
- start_stop_time, Konfigurationsparameter 557
- stat_heap_sz, Konfigurationsparameter 459
- Statische Abfragen
 - Optimierungsklasse einstellen 184
- Statisches SQL
 - Isolationsstufe angeben 66
- Statistiken
 - automatische Erfassung 358, 359
 - manuell aktualisieren 353
 - Richtlinien aktualisieren 326
 - Richtlinien erfassen 326
 - Spaltengruppe 204
 - Stichproben
 - Erfassung 329
- Statistikprofil
 - automatisch generieren 360
 - generieren 361
- Statistiksichten
 - erstellen 212
 - Kardinalitätsschätzungen verbessern 214
 - relevante Statistikdaten 213
 - Übersicht 211
- Statusangaben
 - Sperrmodi 71
- STEPWISE_REDISTRIBUTE_DBPG, Prozedur
 - zur Datenumverteilung verwenden 401
- Steuerung des gemeinsamen Zugriffs
 - föderierte Datenbanken 61
 - maximale Anzahl aktiver Anwendungen 487
 - Probleme 61
 - Sperren verwenden 69
- Steuerzentrale
 - Designadvisor verwenden 55
 - Event Analyzer 405
 - Snapshot Monitor 405
- STMM (Self Tuning Memory Manager)
 - aktivieren 296, 444
 - Einschränkungen 304
 - Überwachung 299
- STMM2 296
- STMM3 296
- stmheap, Konfigurationsparameter 460
 - Auswirkung auf die Abfrageoptimierung 186

- svcname, Konfigurationsparameter 551
- sysadm_group, Konfigurationsparameter 583
- sysctrl_group, Konfigurationsparameter 584
- sysmaint_group, Konfigurationsparameter 585
- sysmon_group, Konfigurationsparameter 586
- Systemmonitor
 - Windows 757
- Systemprozesse 158
- Szenarios
 - Kardinalitätsschätzungen verbessern mit Statistikansichten 214

T

- Tabellen
 - MDC-Tabellen (MDC = mehrdimensionales Clustering) 13
 - mehrdimensionales Clustering 387
 - partitionierte Tabellen 13
 - Reorganisation
 - am Platz, im Onlinemodus 365
 - Ermitteln der Notwendigkeit 369
 - klassisch, im Offlinemodus 365
 - verringerte Notwendigkeit 365
 - reorganisieren 372
 - automatisch 374
 - Sperrmodi 87
 - Standard
 - verwalten 384
 - Warteschlangen, für Joinstrategien in partitionierten Datenbanken 229
 - Zugriff
 - Informationen, von db2expln angezeigt 720
 - Zugriffspfade
 - Sperrmodi 87
- Tabellenbereiche
 - Auswirkung auf die Abfrageoptimierung 189
 - OVERHEAD, Einstellung 189
 - TRANSFERRATE, Einstellung 189
- Tabellenstatistik
 - manuell aktualisieren 356
- TABLESAMPLE
 - Verwendungen 128
- TBSCAN, Operator
 - Definition 707
- TCP/IP-Servicename, Konfigurationsparameter 551
- TEMP, Operator
 - Definition 708
- Temporäre Tabellen
 - Informationen zur Verwendung, db2expln 725
- territory, Konfigurationsparameter 538
- Threads
 - Beschreibung 166
 - in DB2 158
- Tivoli Storage Manager (TSM)
 - Eigername, Konfigurationsparameter 528
 - Kennwort, Konfigurationsparameter 529
 - Knotenname, Konfigurationsparameter 528
 - Konfigurationsparameter für Verwaltungsklasse 527
- tm_database, Konfigurationsparameter 533
- toolscat_db, Konfigurationsparameter 597
- toolscat_inst, Konfigurationsparameter 597
- toolscat_schema, Konfigurationsparameter 598
- tp_mon_name, Konfigurationsparameter 571
- TP-Monitore
 - TP-Monitorname, Konfigurationsparameter 571
- TQUEUE, Operator
 - Definition 709

- trackmod, Konfigurationsparameter 527
- Trigger
 - EXPLAIN-Tabellen 663
- trust_allclnts, Konfigurationsparameter 587
- trust_clntauth, Konfigurationsparameter 588
- tsm_mgmtclass, Konfigurationsparameter 527
- tsm_nodename, Konfigurationsparameter 528
- tsm_owner, Konfigurationsparameter 528
- tsm_password, Konfigurationsparameter 529
- Typ-2-Indizes
 - Beschreibung 31
 - Sperrungen der nächsten Schlüssel 91
 - Vorteile 390

U

- U-Modus (Update)
 - Sperrmodusbeschreibung 71
- Überlaufsätze
 - Auswirkung auf Leistung 369
 - in Standardtabellen 384
- Übersichtstabellen
 - siehe MQT 11
- Überwachung
 - Anwendungsverhalten
 - Governor-Tool 107
 - Datenpartitionen 375
 - Vorgehensweise 405
- Überwachung mit dem Diagnosemonitor, Konfigurationsparameter 563
- Überwachung zu einem Zeitpunkt 405
- Umgebungsvariablen
 - Übersicht 601
- UNION, Operator
 - Definition 709
- UNIQUE, Operator
 - Definition 709
- Unterabfragen
 - korreliert
 - Art des Umschreibens 240
- UPDATE, Operator
 - Definition 710
- user_exit_status, Konfigurationsparameter 540
- userexit, Datenbankkonfigurationsparameter 516
- util_heap_sz, Konfigurationsparameter 448
- util_impact_lim, Konfigurationsparameter
 - Beschreibung 573

V

- vendoropt, Konfigurationsparameter 517
- Verbindungen
 - Antwortzeit 554
- Verbindungskonzentratoren
 - Clientverbindungen, Verbesserungen 171
 - Verwendung von Agenten in partitionierter Datenbank 163
 - Verwendungsbeispiele 171
- Verfallene Tasks ausführen, Konfigurationsparameter 593
- Vergleichselemente
 - anwenden 240
 - implizierte
 - vom Optimierungsprogramm hinzugefügt 242
 - Merkmale 200
 - von Optimierungsprogramm übersetzt 236
- Verschachtelungsschleife, Joins 221

- Verteilungsstatistik
 - Beschreibung 335
 - erweitertes Verwendungsbeispiel 345
 - manuelle Aktualisierung, Regeln 355
 - Optimierung 343
- Verzeichniscacheunterstützung, Konfigurationsparameter
 - Beschreibung 467
- Visual Explain
 - Abfragen auf föderierte Datenbanken auswerten 249
 - föderierte Datenbanken 250
 - Lernprogramm 772
- Vor bedingtem Prüfpunkt zu schreibende Protokollsätze, Konfigurationsparameter 514
- Vorablesezugriff
 - Beschreibung 141
 - blockorientierte Pufferpools 144
 - E/A-Serverkonfiguration 146
 - parallele E/A 147
 - partitionsinterne Leistung 141
 - sequenziell 142
 - über Listen, sequenziell 145
- Vorablesezugriff über Listen 145
- Vorkompilieren
 - Isolationsstufe angeben 66

W

- W-Modus (Weak Exclusive)
 - Sperrmodusbeschreibung 71
- Warten auf Sperren
 - Strategien zum Auflösen 81
 - Zeitlimit 80
- Weltzeit 557
- WHERE-Klausel
 - Terminologie für Vergleichselemente, Definitionen 200
- Wiederholungen für Knotenverbindung, Konfigurationsparameter 556
- Windows
 - Systemmonitor 757

X

- X-Modus (Exclusive)
 - Sperrmodusbeschreibung 71
- XML-Daten
 - indexieren
 - Übersicht 39
- XML-Datentyp
 - indexieren
 - Übersicht 39
- XML-Spalten
 - indexieren
 - Übersicht 39
- XQuery-Anweisungen
 - EXPLAIN-Tools für 711
 - Größe des Anweisungszwischenspeichers, Konfigurationsparameter 460
 - Isolationsstufe angeben 66
 - neu schreiben 236
 - Optimierung
 - Konfigurationsparameter 186

Z

- Z-Modus (Super Exclusive)
 - Sperrmodusbeschreibung 71

- Zeilenblockung
 - angeben 125
- Zeilenkennungen
 - vor Tabellenzugriff vorbereiten 731
- Zeit
 - Deadlocks, Konfigurationsparameter für Prüfintervall 471
 - Unterschied zwischen Knoten, maximal 557
- Zeitlimit
 - Sperre 80
- Zeitlimit für DB2START und DB2STOP, Konfigurationsparameter 557
- Zeitpunkt für Indexneuerstellung, Konfigurationsparameter 523
- Zugriff
 - EXPLAIN-Informationen zur Analyse des Typs 271
- Zugriffspfad
 - Standardtabellen
 - Sperrmodi 87
- Zugriffspläne
 - Ausgabebeispiel
 - keine Parallelität 739
 - Auswirkung auf Sperren 85
 - Auswirkung auf Sperrgranularität 69
- Daten
 - Beschreibung 192
 - Informationen erfassen
 - EXPLAIN-Einrichtung 257
 - mit Indizes 31, 193
 - Spaltenkorrelation mit mehreren Vergleichselementen 227
 - Standardtabellen
 - Sperrmodi 87
- Zwischenspeicher für Anwendungssteuerung, Konfigurationsparameter 449
- Zwischenspeicher für Datenbank, Konfigurationsparameter 437
- Zwischenspeichergröße für Anwendungsunterstützungsebene, Konfigurationsparameter 461

Kontaktaufnahme mit IBM

Informationen zum nächsten IBM Ansprechpartner in Ihrem Land oder Ihrer Region finden Sie im IBM Verzeichnis für weltweite Kontakte, das Sie im Web unter <http://www.ibm.com/planetwide> abrufen können.

Weitere Informationen zu DB2-Produkten finden Sie unter <http://www.ibm.com/software/data/db2/>.



SC12-3629-00

