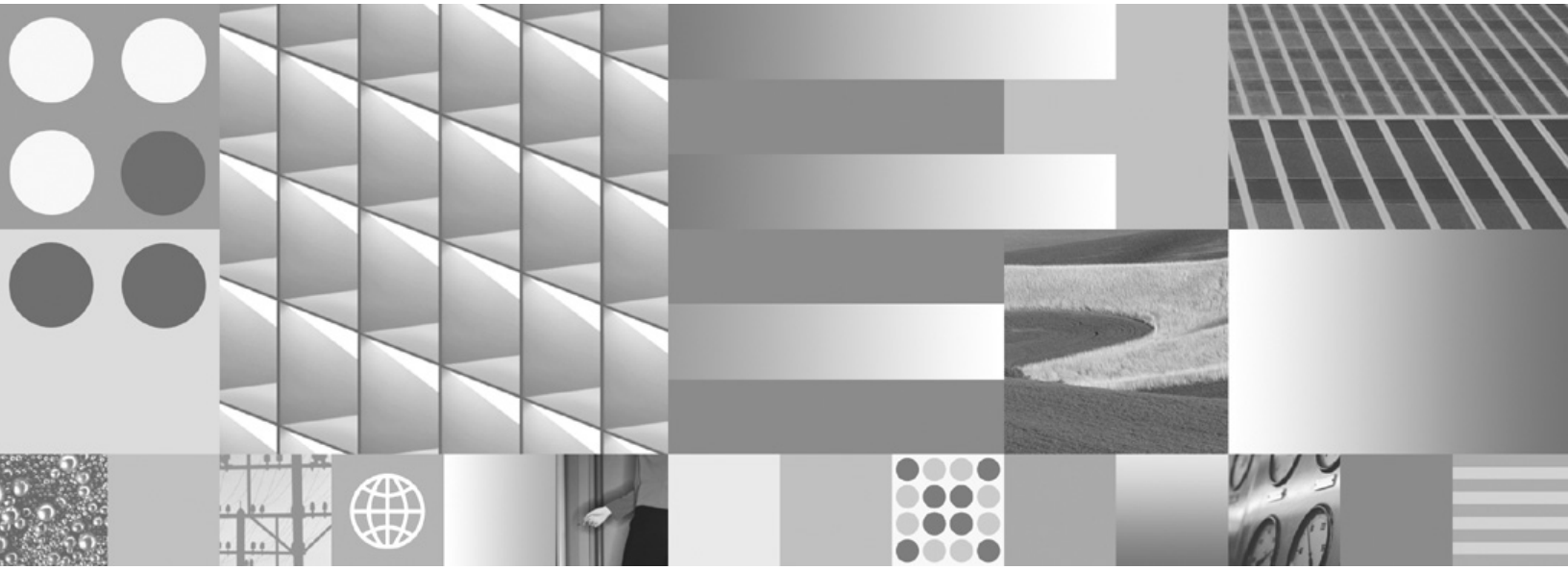
DB2
for Linux, UNIX and Windows

Version 9 Release 7

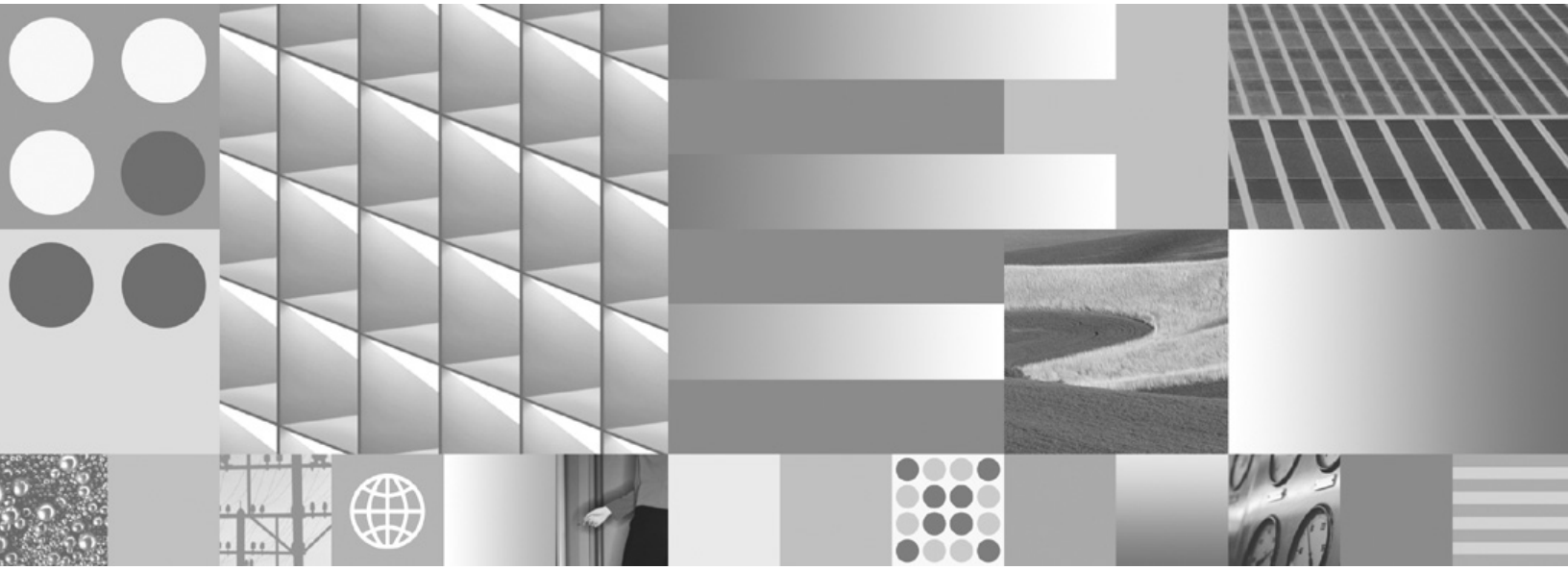


Fehlerbehebung und Optimieren der Datenbankleistung
Aktualisierung: Juli 2012

SC12-4289-03

DB2
for Linux, UNIX and Windows

Version 9 Release 7



Fehlerbehebung und Optimieren der Datenbankleistung
Aktualisierung: Juli 2012

SC12-4289-03

Hinweis

Vor Verwendung dieser Informationen und des darin beschriebenen Produkts sollten die allgemeinen Informationen in Anhang B, „Bemerkungen", auf Seite 715 gelesen werden.

Diese Veröffentlichung ist eine Übersetzung des Handbuchs
IBM DB2 for Linux, UNIX, and Windows, Version 9 Release 7, Troubleshooting and Tuning Database Performance,
IBM Form SC27-2461-03,
herausgegeben von International Business Machines Corporation, USA

Herausgegeben von:
TSC Germany
Kst. 2877
Mai 2012

Inhaltsverzeichnis

Zu diesem Handbuch

Dieses Handbuch enthält Informationen zur Optimierung der Datenbankleistung und zur Lösung von Problemen mit DB2-Datenbankclients und -Datenbankservern.

Sie erhalten Unterstützung für die folgenden Themen:

- Entwickeln einer Strategie zur Leistungsüberwachung und -optimierung
- Entwickeln einer Strategie zur Fehlerbehebung für tägliche Operationen
- Anpassen der Konfiguration des Datenbankservers
- Ausführen von Änderungen an Anwendungen, die den Datenbankserver verwenden
- Präzises Identifizieren von Problemen und Fehlern
- Lösen von Problemen auf der Basis der Fehlersymptome
- Abrufen von Informationen über verfügbare Diagnosetools

Zielgruppe

Dieses Handbuch richtet sich an Kunden, Benutzer, Systemadministratoren, Datenbankadministratoren, Kommunikationsspezialisten, Anwendungsentwickler und Ansprechpartner der technischen Unterstützung, die an einer Optimierung der Datenbankleistung sowie an der Behebung von Problemen mit DB2-Datenbankclients und -Datenbankserver interessiert sind. Für die Verwendung dieses Handbuchs sollten Sie mit Folgendem vertraut sein:

- Kommunikations- und LAN-Konzepte sowie Konzepte relationaler Datenbanken
- Hardware- und Softwarevoraussetzungen und entsprechende Optionen
- Gesamtkonfiguration Ihres Netzes
- Anwendungsprogramme und andere Funktionen, die in Ihrem Netz ausgeführt werden
- Grundlegende DB2-Datenbankverwaltungstasks
- Informationen zur Installation und zu früheren Tasks, die in den Handbüchern zum Einstieg für die installierten Produkte beschrieben werden

Aufbau des Handbuchs

Die in diesem Dokument bereitgestellten Informationen enthalten das erforderliche Hintergrundmaterial, das Sie als Unterstützung bei der Leistungsüberwachung und -optimierung des Datenbanksystems zum Verständnis der Faktoren benötigen, von denen die Datenbankleistung beeinflusst wird, sowie Anweisungen, die Ihnen bei der Optimierung der Leistung Ihres Systems helfen. Die Informationen zu Fehlerbehebung und Unterstützung beinhalten Anweisungen für die Fehlerbehebung mithilfe der Fehlerbehebungsressourcen, die zum Lieferumfang der DB2-Produkte gehören, um Ihnen das Feststellen, Eingrenzen und Beheben von Fehlern, die mit Ihrer DB2-Software auftreten, zu erleichtern.

Teil 1. Optimieren der Datenbankleistung

Als Datenbankadministrator begegnen Ihnen möglicherweise Situationen, in denen Benutzer berichten, dass ihre Datenbankanwendungen recht langsam arbeiten. Die hier bereitgestellten Informationen beschreiben, wie eine Strategie zur Leistungsüberwachung entwickelt wird, um objektive Beurteilungen der Datenbanksystem-

leistung im Vergleich zu früheren Ergebnissen zu ermöglichen. Darüber hinaus wird beschrieben, wie die Konfiguration des Datenbankmanagers angepasst wird und wie Änderungen an den Anwendungen durchgeführt werden, die den Datenbankserver verwenden. Allen Informationen liegt die Zielsetzung zugrunde, die Datenbanksystemleistung zu verbessern, ohne die Verarbeitungskosten zu erhöhen oder den Service für Benutzer zu beeinträchtigen.

- Kapitel 1, „Tools und Vorgehensweise bei der Leistungsoptimierung", beschreibt, wie ein Vergleichstestprogramm (Benchmarktest) entworfen und implementiert wird, das Sie bei der Leistungsverbesserung unterstützt.
- Kapitel 2, „Tools und Vorgehensweise bei der Leistungsüberwachung", enthält Informationen zur Bedeutung einer Betriebsüberwachungsstrategie, die regelmäßig Schlüsseldaten zur Systemleistung erfasst.
- Kapitel 3, „Faktoren mit Auswirkung auf die Leistung", enthält Informationen zu den verschiedenen Faktoren, die sich auf die Leistung des Datenbanksystems auswirken können. Einige dieser Faktoren können optimiert bzw. rekonfiguriert werden.
- Kapitel 4, „Entwickeln einer Strategie zur Leistungsoptimierung", beschreibt das DB2-Designadvisortool, das Ihnen helfen kann, die Auslastungsleistung Ihres Systems erheblich zu verbessern.

Teil 2. Fehlerbehebung

Die Informationen in diesem Abschnitt beschreiben, wie die Quelle eines Problems ermittelt und Diagnoseinformationen erfasst werden, wo Korrekturen erhältlich sind und welche Wissensbasen für die Suche nach zusätzlichen Informationen verfügbar sind, um Sie bei der Lösung eines Problems auch ohne fremde Hilfe zu unterstützen. Wenn Sie sich an IBM Software Support wenden müssen, finden Sie hier Informationen dazu, wie Sie mit der Unterstützungsfunktion Kontakt aufnehmen und welche Diagnoseinformationen von den Kundendiensttechnikern benötigt werden, um Ihnen zu helfen.

- Kapitel 5, „Tools für die Fehlerbehebung", beschreibt die Fehlerbehebungstools, die zur Unterstützung einer systematischen Strategie zur Lösung eines Problems verwendet werden können. Das Ziel dabei ist, herauszufinden, warum etwas nicht wie erwartet funktioniert und wie sich das Problem lösen lässt.
- Kapitel 6, „Fehlerbehebung bei DB2-Datenbank", enthält Informationen zu verschiedenen bekannten Problemen, die auftreten können, sowie zu geeigneten Behebungsmaßnahmen.
- Kapitel 7, „Fehlerbehebung - DB2 Connect", enthält Informationen zu verschiedenen bekannten Problemen, die auftreten können, sowie zu geeigneten Behebungsmaßnahmen.
- Kapitel 8, „Durchsuchen von Wissensbasen", enthält Informationen zum Auffinden von Lösungen für Probleme in IBM Wissensbasen. In diesem Kapitel wird beschrieben, wie Sie Ihre Ergebnisse mit den verfügbaren Ressourcen, Unterstützungstools und Suchmethoden optimieren können.
- Kapitel 9, „Abrufen von DB2-Produktkorrekturen", bietet Informationen dazu, wie eine Produktkorrektur abgerufen werden kann, die zur Lösung eines Problems bereits vorhanden ist. Sie können Programmkorrekturen abrufen, indem Sie die in diesem Kapitel beschriebenen Schritte ausführen.
- Kapitel 10, „Weitere Informationen zur Fehlerbehebung", beschreibt, wie die enthaltenen Themen Ihnen Informationen zu Konzepten und Begriffen zur Verfügung stellen, die Sie zu einer effizienten Behebung von Fehlern benötigen, die im Zusammenhang mit dem DB2-Datenbankserver auftreten können.

- Kapitel 11, „Kontaktaufnahme mit IBM Software Support", enthält Informationen dazu, wie Sie sich an IBM Software Support wenden und welche Informationen bereitzustellen sind, um Hilfe bei der Behebung von Produktfehlern und Datenbankproblemen zu erhalten.

Teil 3. Anhänge

- Anhang A, „Übersicht über die technischen Informationen zu DB2"
- Anhang B, „Bemerkungen"

Teil 1. Leistung - Übersicht

Der Begriff *Leistung* bezieht sich auf die Art und Weise, wie sich ein Computersystem in Bezug auf eine bestimmte Auslastung (Workload) verhält. Die Leistung wird an der Antwortzeit, am Durchsatz und an der Ressourcennutzung gemessen.

Die Leistung wird außerdem 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 mit dem Ziel, das Kosten-Nutzen-Verhältnis zu verbessern. Dabei können die folgenden speziellen Optimierungsziele verfolgt werden:

- Verarbeiten größerer oder anspruchsvollerer Auslastungen ohne steigende Verarbeitungskosten
- Erreichen schnellerer Systemantwortzeiten bzw. eines höheren Durchsatzes ohne steigende Verarbeitungskosten
- Reduzieren von Verarbeitungskosten ohne negative Auswirkungen für Benutzer

Einige Vorteile der Leistungsoptimierung, wie zum Beispiel eine effizientere Nutzung von Ressourcen und die Möglichkeit, dem System weitere Benutzer hinzuzufügen, zeigen sich sehr praktisch. Andere Vorteile, wie größere Zufriedenheit seitens der Benutzer aufgrund schnellerer Antwortzeiten, sind weniger fassbar.

Richtlinien zur Leistungsoptimierung

Beachten Sie bei der Entwicklung eines allgemeinen Ansatzes zur Leistungsoptimierung die folgenden Richtlinien.

- **Behalten Sie das Gesetz der abnehmenden Ertragsgewinne im Hinterkopf:** Die größten Leistungsvorteile werden in der Regel durch die ersten Maßnahmen erzielt.
- **Optimieren Sie nicht nur des Optimierens wegen:** Optimieren Sie, um erkannten Engpässen abzuhelfen. Eine Optimierung von Ressourcen, die nicht die Hauptursache für Leistungsprobleme darstellen, kann die nachfolgende Optimierungsarbeit tatsächlich erschweren.
- **Betrachten Sie das System als Ganzes:** Ein Parameter bzw. eine Ressource lässt sich nicht isoliert optimieren. Bevor Sie eine Anpassung vornehmen, überlegen Sie, wie sich diese Änderung auf das System als Ganzes auswirken wird. Die Leistungsoptimierung erfordert Kompromisslösungen zwischen verschiedenen Systemressourcen. Zum Beispiel könnten Sie die Werte für Pufferpoolgrößen erhöhen, um eine bessere Ein-/Ausgabeleistung zu erzielen, jedoch erfordern größere Pufferpools mehr Speicher und können daher andere Aspekte der Leistung wiederum beeinträchtigen.
- **Ändern Sie jeweils nur einen Parameter gleichzeitig:** Ändern Sie immer nur einen Faktor gleichzeitig. Selbst wenn Sie sich sicher sind, dass alle Änderungen vorteilhaft sind, haben Sie hinterher keine Möglichkeit, den Beitrag jeder einzelnen Änderung zu bewerten.
- **Führen Sie Messungen und Konfigurationen nach Ebenen durch:** Optimieren Sie jeweils nur eine Ebene Ihres Systems gleichzeitig. Systemebenen sind zum Beispiel:

  - Hardware
  - Betriebssystem
  - Anwendungsserver und -requester
  - Datenbankmanager
  - SQL- und XQuery-Anweisungen
  - Anwendungsprogramme
- **Prüfen Sie auf Hardware- und Softwareprobleme:** Einige Leistungsprobleme können 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, bevor Sie eine Hardwarewartung oder eine Softwarekorrektur durchgeführt haben.
- **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.
- **Implementieren Sie Rücksetzprozeduren, bevor Sie mit der Optimierung beginnen:** Wenn Optimierungsmaßnahmen zu einer unerwarteten Leistungsverschlechterung führen, sollten die vorgenommenen Änderungen rückgängig gemacht werden, bevor eine alternative Lösung versucht wird. Speichern Sie Ihre ursprünglichen Einstellungen, sodass Sie Änderungen, die Sie nicht beibehalten möchten, leicht rückgängig machen können.

Entwickeln eines Prozesses zur Leistungsverbesserung

Ein Leistungsverbesserungsprozess ist ein iteratives Verfahren zur Überwachung und Optimierung von Leistungsbereichen. Abhängig von den Ergebnissen dieser Leistungsüberwachung passen Sie 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 Ihnen bekannten Datenzugriffsmustern aus. Verschiedene Arten von Anwendungen haben unterschiedliche Leistungsanforderungen.

Jeder Leistungsverbesserungsprozess enthält die folgenden grundlegenden Schritte:

1. Definieren Sie die 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 Überwachungsergebnisse fortlaufend, um zu ermitteln, welche Ressourcen optimiert werden müssen.
5. Nehmen Sie jeweils nur eine Anpassung vor.

Wenn Sie an einem bestimmten Punkt keine weitere Verbesserung der Leistung durch Optimieren des Datenbankservers und der Anwendungen erzielen können, ist möglicherweise der Zeitpunkt gekommen, die Hardware aufzurüsten.

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 auseinander setzen, indem Sie Ihren Benutzern zuhören. Beginnen Sie, indem Sie einige einfache Fragen stellen, wie zum Beispiel die folgenden:

- Was meinen Sie mit „langsamer Reaktion"? Heißt dies, um zehn Prozent langsamer, als Sie erwarten, oder um das Zigfache 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 das gleiche Problem hat, sind diese Benutzer mit demselben lokalen Netz (LAN) verbunden?
- Scheint das Problem mit einem bestimmten Typ von Transaktions- oder Anwendungsprogramm zusammenzuhängen?
- Erkennen Sie ein Muster im Auftreten des Problems? Zum Beispiel: Tritt dieses Problem zu einer bestimmten Tageszeit auf oder ist es permanent bemerkbar?

Grenzen der Leistungsoptimierung

Die durch die Leistungsoptimierung realisierbaren Vorteile sind begrenzt. Wenn Sie überlegen, wie viel Zeit und Geld in die Verbesserung der Systemleistung investiert werden sollte, müssen Sie unbedingt eine Beurteilung des möglichen Grads vornehmen, bis zu dem eine zusätzliche Investition von Zeit und Geld den Benutzern des Systems hilft.

Eine Optimierung kann die Leistung häufig verbessern, wenn das System Probleme mit der Antwortzeit oder dem Durchsatz hat. Es gibt allerdings einen Punkt, ab dem eine weitere Optimierung keine Hilfe mehr ist. An diesem Punkt müssen Sie Ihre Ziele und Erwartungen überprüfen. Wenn Sie weitere wesentliche Leistungsverbesserungen erreichen wollen, müssen Sie vielleicht mehr Plattenspeicher, schnellere CPUs, zusätzliche CPUs, mehr Arbeitsspeicher, schnellere Kommunikationsverbindungen oder eine Kombination aus diesen Möglichkeiten hinzufügen.

Kapitel 1. Tools und Vorgehensweise bei der Leistungsoptimierung

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).

Vergleichstests werden für ein System ausgeführt, um das aktuelle Leistungsverhalten zu ermitteln. Mit ihrer Hilfe kann die Leistung von Anwendungen verbessert werden. Wenn der Code einer Anwendung bereits mit größtmöglicher Effizienz arbeitet, können weitere Leistungsvorteile eventuell durch eine Optimierung von Konfigurationsparametern der Datenbank und des Datenbankmanagers realisiert werden.

Durch verschiedene Typen von Vergleichstests lassen sich bestimmte Arten von Informationen gewinnen. Beispiel:

- Ein *Infrastrukturvergleichstest* dient zur Ermittlung der Durchsatzkapazitäten des Datenbankmanagers unter bestimmten, eingeschränkten Laborbedingungen.
- Ein *Anwendungsvergleichstest* dient zur Ermittlung der Durchsatzkapazitäten des Datenbankmanagers unter Bedingungen, die einer Produktsumgebung näher kommen.

Die Optimierung von Konfigurationsparametern mithilfe von Vergleichstests basiert auf kontrollierten Bedingungen. Derartige Tests beinhalten eine wiederholte Ausführung von SQL aus der Anwendung heraus, wobei die Systemkonfiguration (und eventuell das SQL) geändert wird, bis die Anwendung mit größtmöglicher Effizienz arbeitet.

Derselbe Ansatz kann zur Optimierung weiterer leistungsrelevanter Faktoren verwendet werden, wie zum Beispiel Indizes, die Konfiguration von Tabellenbereichen und die Konfiguration von Hardwarekomponenten, um nur einige zu nennen.

Vergleichstests helfen Ihnen bei der Untersuchung, wie der Datenbankmanager auf verschiedene Bedingungen reagiert. Es können Szenarios entwickelt werden, um die Behandlung von Deadlocks, die Leistung von Dienstprogrammen, die verschiedenen Methoden zum Laden von Daten, die Transaktionsgeschwindigkeiten bei wachsenden Benutzerzahlen und sogar die Auswirkungen der Verwendung eines neuen Release des Datenbankprodukts auf die Anwendung zu testen.

Bei der Durchführung von Vergleichstests wird eine reproduzierbare Umgebung zugrunde gelegt, sodass der gleiche Test unter den gleichen Bedingungen Ergebnisse liefert, deren Vergleich legitim ist. Sie können damit 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 besitzen folgende Merkmale:

- Die Tests sind reproduzierbar.
- Jede Iteration eines Tests beginnt im gleichen Systemstatus.
- Es sind keine anderen Funktionen oder Anwendungen unbeabsichtigt im System aktiv.
- Die Hardware und die Software, die für Vergleichstests verwendet werden, entsprechen der realen Produktionsumgebung.

Beachten Sie, dass gestartete Anwendungen Hauptspeicher belegen, auch wenn sie inaktiv sind. Dadurch erhöht sich die Wahrscheinlichkeit, dass Seitenauslagerungen zu einer Verzerrung der Vergleichstestergebnisse führen, was gegen das Reproduzierbarkeitskriterium verstößt.

Vorbereiten von Vergleichstests

Es müssen bestimmte Voraussetzungen erfüllt sein, bevor die Durchführung von Leistungsvergleichstests eingeleitet werden kann.

Gehen Sie wie folgt vor, bevor Sie mit Vergleichstests beginnen:

- Schließen Sie den logischen und den physischen Entwurf der Datenbank ab, für die Ihre Anwendung ausgeführt werden soll.
- Erstellen Sie Tabellen, Sichten und Indizes.
- Normalisieren Sie Tabellen, binden Sie Anwendungspakete und füllen Sie Tabellen mit realistischen Daten. Stellen Sie sicher, dass aussagekräftige Statistiken verfügbar sind.
- Planen Sie die Ausführung an einer Datenbank in einer Größe wie in der Produktionsumgebung, sodass die Anwendung repräsentative Speicheranforderungen testen kann. Falls dies nicht möglich ist, versuchen Sie sicherzustellen, dass die Proportionen der verfügbaren Systemressourcen zu den Daten im Test- und im Produktionssystem identisch sind. (Wenn das Testsystem z. B. 10 % der Daten hat, verwenden Sie 10 % der Prozessorzeit und 10 % des Hauptspeichers, der für das Produktionssystem verfügbar ist.)
- Platzieren Sie Datenbankobjekte an ihre endgültigen Datenträgerpositionen, definieren Sie die Größe von Protokolldateien, legen Sie die Position von Arbeitsdateien und Backup-Images fest und testen Sie die Backup-Prozeduren.
- Prüfen Sie Pakete, um sicherzustellen, dass Leistungsoptionen wie Zeilenblockung aktiviert werden, wenn dies möglich ist.

Während der Vergleichstests können die praktischen Grenzen einer Anwendungen zutage treten. Jedoch liegt das Ziel von Vergleichstests in der Messung der Leistung und nicht in der Feststellung von Fehlern.

Ihr Vergleichstestprogramm sollte in einer präzisen Nachbildung der endgültigen Produktionsumgebung ausgeführt werden. Im Idealfall sollte dasselbe Servermodell mit derselben Speicher- und Festplattenkonfiguration verwendet werden. Dies ist besonders dann von Bedeutung, wenn die Anwendung letztendlich eine große Anzahl von Benutzern bedienen und große Datenvolumen verarbeiten soll. Das Betriebssystem und alle Kommunikations- und Speichereinrichtungen, die direkt vom Vergleichstestprogramm verwendet werden, sollten ebenfalls zuvor optimiert worden sein.

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. Welche Anweisungen ausgewählt werden, hängt von der Spezifik der Anwendung ab. 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.

Extremfall-SQL

Zu dieser Kategorie gehören Anweisungen mit folgenden Merkmalen:

- Anweisungen, die häufig ausgeführt werden.
- Anweisungen, die umfangreiche Datenvolumen verarbeiten.
- Anweisungen, die zeitkritisch sind. Beispiel: Anweisungen in einer Anwendung, die Kundeninformationen abruft und aktualisiert, während der Kunde am Telefon wartet.
- Anweisungen mit einer hohen Anzahl von Joins oder die komplexesten Anweisungen in der Anwendung. Beispiel: Anweisungen in einer Finanzanwendung, die Zusammenfassungen der monatlichen Vorgänge für alle Konten eines Kunden generiert. Eine allgemeine Tabelle enthält vielleicht die Kundenadressen und die Kontonummern. Jedoch müssen mehrere andere Tabellen verknüpft werden, um alle benötigten Daten über Kontotransaktionen zu verarbeiten und zusammenzustellen.
- Anweisungen, die einen ungünstigen Zugriffspfad verwenden, zum Beispiel eine, die nicht durch einen verfügbaren Index unterstützt wird.
- Anweisungen, die eine lange Ausführungszeit haben.
- Anweisungen, die nur bei der Initialisierung einer Anwendung ausgeführt werden, jedoch überproportional großen Ressourcenbedarf haben. Beispiel: Anweisungen in einer Anwendung, die eine Liste von 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 jeden Tag ausgeführt, jedoch nimmt ihre Ausführung einige Minuten in Anspruch, wenn sie nicht ordnungsgemäß optimiert wurde.

Erstellen von Vergleichstests

Sie müssen eine Reihe von Faktoren berücksichtigen, wenn Sie ein Vergleichstestprogramm entwerfen und implementieren.

Da der Hauptzweck des Testprogramms darin besteht, eine Benutzeranwendung zu simulieren, ist die allgemeine Struktur des Programms unterschiedlich. Sie können die gesamte Anwendung zum Vergleichstest verwenden und nur die entsprechenden Mittel zur Erfassung der Zeiten für die SQL-Anweisungen einfügen, die analysiert werden sollen. 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 nur die Anweisungen in das Vergleichstestprogramm aufnehmen und die erforderlichen Anweisungen CONNECT, PREPARE, OPEN und andere sowie einen Mechanismus zur Zeiterfassung hinzufügen.

Ein weiterer wichtiger Gesichtspunkt ist der Typ von Vergleichstest, der zu verwenden ist. Eine Möglichkeit ist die, eine Gruppe von SQL-Anweisungen über ein

bestimmtes Zeitintervall hinweg wiederholt auszuführen. Die Anzahl von Anweisungen, die in diesem Zeitintervall ausgeführt werden, ist ein Maß 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 wird ein zuverlässiges und geeignetes Verfahren zum Messen der abgelaufenen Zeit benötigt. Zur Simulation einer Anwendung, in der einzelne SQL-Anweisungen isoliert ausgeführt werden, kann sich als bestes Verfahren eine Messung der Zeiten für PREPARE-, EXECUTE- oder OPEN-, FETCH- oder CLOSE-Anweisungen anbieten. Für andere Anwendungen kann eine Messung der Transaktionsdauer von der ersten SQL-Anweisung bis zur COMMIT-Anweisung besser geeignet sein.

Obwohl die abgelaufene 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-Belastung, über Sperren und Pufferpoolein-/ausgaben Hinweise darauf geben, dass eine Anweisung durch die Ein-/Ausgabeaktivitäten gebremst wird und nicht die volle CPU-Kapazität 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. Die Ergebnismenge könnte zum Beispiel eine Eingabe für eine andere Anwendung sein. 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 speziellen Zeilenbehandlungsaktivitäten der Anwendung berücksichtigen. Wenn Zeilen an eine Ausgabeeinheit gesendet werden, könnte ineffizientes Formatieren den Hauptanteil der CPU-Zeit in Anspruch nehmen und so zu einer Verzerrung der tatsächlichen Leistungsdaten für die SQL-Anweisung als solche führen.

Obwohl sich der DB2-Befehlszeilenprozessor (CLP) einfach verwenden lässt, ist er aufgrund des Verarbeitungsaufwands, den er zusätzlich verursacht, für Vergleichstests nicht geeignet. Es steht ein Vergleichstesttool (**db2batch**) im Unterverzeichnis `bin` des Verzeichnisses `sqllib` Ihrer Instanz zur Verfügung. Dieses Tool kann SQL-Anweisungen entweder aus einer unstrukturierten Datei oder von der Standardeingabeeinheit lesen, die Anweisungen dynamisch vorbereiten und ausführen und eine Ergebnismenge zurückliefern. Es gibt Ihnen außerdem die Möglichkeit, die Zahl der Zeilen, die an **db2batch** zurückgegeben werden, und die Zahl der Zeilen, die angezeigt werden, zu steuern. Sie können die Stufe der leistungsbezogenen Daten angeben, die zurückgegeben werden, einschließlich abgelaufener Zeit, Prozessorzeit, Pufferpoolnutzung, Sperren und anderer statistischer Daten aus dem Datenbankmonitor. Bei der Zeitmessung für eine Gruppe von SQL-Anweisungen erstellt **db2batch** auch eine Übersicht über die Leistungsergebnisse und berechnet arithmetische und geometrische Mittelwerte.

Durch eine Wiederverwendung von **db2batch**-Aufrufen in einem Perl- oder Korn-Shell-Script können Sie bequem eine Mehrbenutzerumgebung simulieren. Stellen Sie sicher, dass Verbindungsattribute wie die Isolationsstufe übereinstimmen, indem Sie die entsprechenden Optionen von **db2batch** auswählen.

Dabei ist zu beachten, dass **db2batch** in Umgebungen mit partitionierten Datenbanken nur zum Messen der verstrichenen Zeit geeignet ist. Andere Informationen, die zurückgegeben werden, beziehen sich lediglich auf die Aktivität auf der Koordinatordatenbankpartition.

Sie können ein Treiberprogramm schreiben, das Sie bei der Durchführung der Vergleichstests unterstützt. Auf Linux- oder UNIX-Systemen kann ein Treiberprogramm mithilfe von Shellprogrammen geschrieben werden. Ein Treiberprogramm kann das Vergleichstestprogramm ausführen, die richtigen Parameter übergeben, den Test durch mehrere Iterationen führen, die Umgebung in einen konsistenten Zustand zurückversetzen, den nächsten Test mit neuen Parameterwerten vorbereiten und die Testdaten sammeln und konsolidieren. 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 optimalen Parameterwerte für einen bestimmten Test verwendet werden können.

Ausführen von Vergleichstests

Beim allgemeinsten Typ 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 eine wiederholte Ausführung der Anwendung (z. B. fünf oder zehn Iterationen) mit demselben Parameterwert beinhalten. Dadurch erhalten Sie einen zuverlässigeren Durchschnittswert für die Leistung, mit dem Sie die Ergebnisse mit anderen Parameterwerten vergleichen können.

Die erste Ausführung, die als Aufwärmdurchlauf bezeichnet wird, sollte von den nachfolgenden, so genannten Normaldurchläufen separat betrachtet werden. Der Aufwärmdurchlauf umfasst einige Startaktivitäten, wie zum Beispiel die Initialisierung des Pufferpools, und dauert daher etwas länger als die Normaldurchläufe. Die Informationen aus einem Aufwärmdurchlauf sind statistisch nicht relevant. Bei der Berechnung von Durchschnittswerten für eine bestimmte Gruppe von Parameterwerten verwenden Sie nur die Ergebnisse aus Normaldurchläufen. Es ist in der Regel sinnvoll, die hohen und die niedrigen Werte vor dem Berechnen von Durchschnittswerten zu senken.

Zur Sicherstellung der größtmöglichen Konsistenz zwischen Durchläufen sorgen Sie vor jedem neuen Durchlauf dafür, dass der Pufferpool auf einen bekannten Status zurückgesetzt wird. Tests können zur Folge haben, dass sich der Pufferpool mit Daten füllt, sodass nachfolgende Durchläufe möglicherweise deshalb schneller sind, weil weniger E/A-Aktivitäten erforderlich sind. Der Pufferpoolinhalt kann zwangsweise entfernt werden, indem andere irrelevante Daten in den Pufferpool eingelesen werden oder indem der Pufferpool neu zugeordnet wird, wenn alle Datenbankverbindungen zeitweise getrennt wurden.

Nach der Ausführung der Tests für eine Gruppe von Parameterwerten, können Sie den eines einzelnen Parameters ändern. Zwischen den einzelnen Durchläufen müssen Sie 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 Testdaten müssen konsistent sein, wenn sie während der Tests aktualisiert werden. Dies kann folgendermaßen sichergestellt werden:
  - Durch Verwenden des Restoredienstprogramms, 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 ermöglicht ei-

nen 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.

Zusammengefasst: Führen Sie die folgenden Schritte zur Durchführung von Vergleichstests für eine Datenbankanwendung aus:

Schritt 1

Behalten Sie die empfohlenen Standardwerte der Konfigurationsparameter für die DB2 Registrierdatenbank, die Datenbank und den Datenbankmanagers sowie die Pufferpools bei. Dies kann für folgende Werte gelten:

- Werte, die bekanntermaßen für eine ordnungsgemäße und fehlerfreie Ausführung von Anwendungen erforderlich sind
- Werte, die bei vorherigen Optimierungsmaßnahmen Leistungsverbesserungen gebracht haben
- Werte, die vom Befehl **AUTOCONFIGURE** vorgeschlagen wurden
- Standardwerte, die jedoch möglicherweise nicht angemessen sind:
  - Für Parameter, die für die Auslastung und für die Zielsetzung des Tests von Bedeutung sind.
  - Für Protokolldateigrößen, die während der Einheiten- oder Systemtests für Ihre Anwendung bestimmt werden sollten.
  - Für alle Parameter, die geändert werden müssen, um die Ausführung Ihrer Anwendung zu ermöglichen.

Führen Sie Ihre Reihe von Durchläufen (Iterationen) für diesen Anfangsfall aus und berechnen Sie die durchschnittliche abgelaufene Zeit, den durchschnittlichen Durchsatz oder die durchschnittliche Prozessorzeit. Die Ergebnisse sollten möglichst konsistent sein. Im Idealfall sollten sie sich von Ausführung zu Ausführung nur um wenige Prozentpunkte unterscheiden. Leistungsmessungen, die sich erheblich von Ausführung zu Ausführung unterscheiden, können eine Optimierung sehr erschweren.

Schritt 2

Wählen Sie nur eine einzige Methode bzw. einen einzigen Optimierungsparameter für den Test aus und ändern Sie den zugehörigen Wert.

Schritt 3

Führen Sie eine weitere Reihe von Durchläufen (Iterationen) aus und berechnen Sie die durchschnittliche abgelaufene Zeit oder die durchschnittliche Prozessorzeit.

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 aus. Wiederholen Sie diese Prozedur, bis alle Parameter getestet wurden.

Vergleichstestanalyse - Beispiel

Die Ausgabe eines Vergleichstestprogamms sollte eine Kennung für jeden Test, Iterationsnummer, Anweisungsnummern und die abgelaufenen Zeiten für jede Ausführung enthalten.

Beachten Sie, dass die Daten in diesen Beispielberichten nur Illustrationszwecken dienen. Es handelt sich nicht um tatsächlich gemessene Ergebnisse.

Eine Zusammenfassung der Vergleichstestergebnisse könnte zum Beispiel wie folgt aussehen:

```
Test     Iter.   Stmt    Timing         SQL Statement
Numbr    Numbr   Numbr   (hh:mm:ss.ss)
 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 1. Beispiel für Vergleichstestergebnisse

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 Anweisung FETCH (Anweisung 15) sieben Zeilen mit der längsten Verzögerung von 0,28 Sekunden lieferte, 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 ASCII-Format ohne universelle Zeilenbegrenzer (Delimited ASCII) ausgeben könnte, könnten diese später in eine Datenbanktabelle oder ein Arbeitsblatt eines Tabellenkalkulationsprogramms zur weiteren statistischen Analyse importiert werden.

Ein zusammenfassender Vergleichstestbericht könnte wie folgt 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
stmtheap         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 2. Beispiel für einen Vergleichstestbericht zu Ausführungszeiten

Kapitel 2. Tools und Vorgehensweise bei der Leistungsüberwachung

Leistungsbezogene Betriebsüberwachung

Bei der Betriebsüberwachung werden in regelmäßigen Abständen über einen längeren Zeitraum hinweg wichtige Messdaten zur Systemleistung erfasst. Diese Informationen liefern Ihnen die kritischen Daten, die Sie benötigen, um die Ausgangskonfiguration an Ihre Anforderungen anzupassen und neue Probleme zu lösen, die durch unbekannte Ursachen oder im Anschluss an Softwareaktualisierungen, durch erhöhte Daten- oder Benutzermengen oder die Implementierung neuer Anwendungen auftreten.

Wichtige Aspekte bei der Betriebsüberwachung

Überwachungsstrategien für den Systembetrieb müssen verschiedenen Aspekten gerecht werden.

Die Betriebsüberwachung muss äußerst belastungsarm (d. h. sie darf das System, das sie misst, nicht sehr beanspruchen) und generisch (d. h. sie muss ganz allgemein auf potenzielle Probleme ausgerichtet, die an beliebiger Stelle im System auftreten könnten) sein.

Da Sie eine regelmäßige Erfassung von Betriebsmessdaten über den gesamten Lebenszyklus des Systems hinweg planen, ist es wichtig, über eine Methode zur Verwaltung all dieser Daten zu verfügen. Für viele mögliche Verwendungszwecke der Daten, wie zum Beispiel bei der Ermittlung langfristiger Leistungstrends, ist es wünschenswert, Vergleiche zwischen beliebigen Datenerfassungen durchführen zu können, die potenziell viele Monate auseinander liegen. Das DB2-Produkt selbst bietet eine gute Unterstützung für diese Art des Datenmanagements. Analysen und Vergleiche von Überwachungsdaten gestalten sich recht einfach und Sie haben bereits eine leistungsfähige Infrastruktur für eine langfristige Datenspeicherung und -verwaltung zur Hand.

Ein DB2-Datenbanksystem („DB2") stellt einige hervorragende Quellen für Überwachungsdaten zur Verfügung. Die primären Quellen sind Überwachungsprogramme für Momentaufnahmen (Snapshot Monitor) sowie ab DB2 Version 9.5 Tabellenfunktionen des Workload-Managements für Datenzusammenfassung. Beide Arten von Quellen konzentrieren sich auf Summendaten, wobei Tools wie Zähler, Zeitgeber und Histogramme laufende Summen in Bezug auf die im System ausgeführten Aktivitäten verwalten. Durch die Entnahme von Stichproben mithilfe dieser Monitorelemente über einen Zeitraum hinweg können Sie das durchschnittliche Aktivitätsvolumen ableiten, das zwischen den Start- und Endzeitpunkten stattgefunden hat. Dies kann einen hohen Informationsgehalt haben.

Es gibt keinen Grund, sich nur auf die Messdaten zu beschränken, die vom DB2-Produkt bereitgestellt werden. Tatsächlich sind Daten außerhalb von DB2 mehr wesentlich als nur eine nette Dreingabe. Kontextinformationen sind ein wichtiger Schlüssel bei der Bestimmung von Leistungsproblemen. Die Benutzer, die Anwendung, das Betriebssystem, das Speichersubsystem und das Netz können jeweils wertvolle Informationen zur Systemleistung beisteuern. Das Einschließen von Messdaten, die außerhalb der DB2-Datenbanksoftware erfasst werden, spielt eine wichtiger Rolle bei der Erstellung eines Gesamtbilds der Systemleistung.

Der Trend in den jüngsten Releases des DB2-Datenbankprodukts bestand darin, immer mehr Überwachungsdaten durch SQL-Schnittstellen verfügbar zu machen. Dies macht die Verwaltung von Überwachungsdaten mit DB2 sehr einfach, weil Sie zum Beispiel die Daten aus Verwaltungssichten problemlos an DB2-Tabellen zurückgeben können. Für eingehendere Untersuchungen können auch Ereignis- und Aktivitätsmonitordaten in DB2-Tabellen geschrieben werden, um ähnliche Vorteile zur Verfügung zu stellen. Angesichts dieser einfachen Möglichkeit, die weitaus meisten Ihrer Überwachungsdaten in DB2 zu speichern, erscheint eine kleine Investition zum Speichern von Systemmessdaten (wie z. B. CPU-Auslastungsdaten aus dem Befehl **vmstat**) in DB2 ebenfalls tragbar.

Für die Betriebsüberwachung zu erfassende Datentypen

Es empfiehlt sich bestimmte Datentypen für die kontinuierliche Betriebsüberwachung zu erfassen.

- Eine Gruppe von DB2-Hauptmessdaten für die Überwachung der Systemleistung
- DB2-Konfigurationsdaten

  Eine regelmäßige Erstellung von Kopien der Datenbank- und Datenbankmanagerkonfiguration, der DB2-Registrierdatenbankvariablen und der Schemadefinition helfen bei der Protokollierung aller Änderungen, die vorgenommen werden. Anhand dieser protokollierten Konfigurationsdaten lassen sich möglicherweise Änderungen besser erklären, die in den Überwachungsdaten auftreten.
- Allgemeine Systembelastung

  Wenn zugelassen wird, dass die CPU-Belegung oder das E/A-Aktivitätsvolumen einer Vollauslastung nahe kommen, kann dies einen Systemengpass verursachen, der nur mit DB2-Momentaufnahmen schwer zu erkennen ist. Es bietet sich daher als bewährtes Verfahren an, die Systembelastung regelmäßig mit **vmstat** und **iostat** (und möglicherweise **netstat** für Netzprobleme) auf Linux- und UNIX-basierten Systemen und mit dem Systemmonitor (**perfmon**) unter Windows zu überwachen. Sie können auch die Verwaltungssichten wie zum Beispiel ENV_SYS_RESOURCES verwenden, um Informationen zum Betriebssystem, zur CPU und zum Hauptspeicher sowie weitere systembezogene Informationen abzurufen. In der Regel suchen Sie in Daten, die für Ihr System normal sind, nach Änderungen und nicht nach bestimmten allgemeingültigen Werten.
- Auf der Ebene der Geschäftslogik gemessene Daten für Durchsatz und Antwortzeit

  Eine Anwendungssicht über die Leistung, gemessen oberhalb von DB2 auf der Ebene der Geschäftslogik, hat den Vorteil der höchstmöglichen Relevanz für den Endbenutzer. Darüber hinaus umfasst sie in der Regel alle Komponenten, die einen Engpass verursachen könnten, wie zum Beispiel Darstellungslogik, Anwendungsserver, Web-Server, mehrere Netzschichten usw. Solche Daten können für den Prozess zur Festlegung oder Prüfung eines Service-Level-Agreements (SLA) von entscheidender Bedeutung sein.

Die DB2-Monitorelemente für Systemleistung und die Daten zur Systembelastung sind ausreichend kompakt, sodass selbst bei einem Erfassungsintervall von fünf bis fünfzehn Minuten das Gesamtdatenvolumen im Verlauf der Zeit für die meisten Systeme irrelevant ist. Ebenso liegt der Systemaufwand zur Erfassung dieser Daten in der Regel im Bereich von 1 bis 3 Prozent zusätzlicher CPU-Belegung, die jedoch einen geringen Preis für ein fortlaufendes Protokoll wichtiger Systemmessdaten darstellen. Konfigurationsdaten ändern sich gewöhnlich relativ selten, sodass eine einmalige Erfassung solcher Daten pro Tag in der Regel ausreichen sollte, um von Nutzen zu sein, ohne ein übermäßiges Datenvolumen zu erzeugen.

Gruppe der Hauptmonitorelemente für die Systemleistung

Etwa 10 Messwerte zur Systemleistung liefern eine gute Basis für eine kontinuierliche Betriebsüberwachung.

Es stehen Hunderte von Messdaten zur Auswahl. Sämtliche Messdaten zu erfassen, kann sich indessen aufgrund des schieren Datenvolumens, das generiert wird, als kontraproduktiv erweisen. Wünschenswert sind Messdaten, die sich durch folgende Merkmale auszeichnen:

- Einfache Erfassung: Ein erforderlicher Einsatz komplexer oder kostenintensiver Tools zur täglichen Überwachung ist nicht wünschenswert und die Überwachung soll an sich keine bedeutende Belastung des Systems darstellen.
- Verständlichkeit: Es ist nicht von Vorteil, wenn die Bedeutung eines Messwerts jedes Mal, wenn er auftritt, nachgeschlagen werden muss.
- Systemrelevanz: Nicht alle Messdaten stellen aussagekräftige Informationen für jede Umgebung zur Verfügung.
- Angemessene Empfindlichkeit: Eine Änderung des Messwerts sollte eine reale Änderung im System wiedergeben. Der Messwert sollte keine inhärenten Schwankungen aufweisen.

Die Einstiegslösung beinhaltet 10 Messungen:

- Die Anzahl der ausgeführten Transaktionen:

  ```
  TOTAL_COMMITS
  ```

  Diese Angabe stellt eine hervorragende Basismessung für die Systemaktivität dar.
- Pufferpooltrefferquoten, für Daten, Indizes und temporäre Daten separat gemessen:

  ```
  100 * (POOL_DATA_L_READS – POOL_DATA_P_READS) / POOL_DATA_L_READS
  100 * (POOL_INDEX_L_READS – POOL_INDEX_P_READS) / POOL_INDEX_L_READS
  100 * (POOL_TEMP_DATA_L_READS - POOL_TEMP_DATA_P_READS) / POOL_TEMP_DATA_L_READS
  100 * (POOL_TEMP_INDEX_L_READS - POOL_TEMP_INDEX_P_READS) / POOL_TEMP_INDEX_L_READS
  ```

  Die Werte für die Effektivität der Zugriffe auf Pufferpools ('Pufferpooltrefferquoten') gehören zu den grundlegendsten Messdaten und geben einen wichtigen Gesamtmesswert darüber wieder, wie effektiv das System den Speicher nutzt, um Platten-E/A-Operationen zu vermeiden. Trefferquoten von 80 - 85 % oder höher für Daten sowie von 90 - 95 % oder höher für Indizes gelten für eine OLTP-Umgebung im Allgemeinen als gut. Diese Trefferquoten lassen sich natürlich für einzelne Pufferpools anhand der Daten aus der Pufferpoolmomentaufnahme berechnen.

  Obwohl diese Messdaten im Allgemeinen nützlich sind, sind die Datentrefferquoten für Systeme wie Data-Warehouses, in denen häufig umfangreiche Tabellensuchläufe ausgeführt werden, nicht selten vernachlässigbar gering, weil Daten in den Pufferpool eingelesen werden und anschließend nicht mehr verwendet werden, bevor sie bereinigt werden, um anderen Daten Platz zu machen.
- Physische Lese- und Schreibvorgänge für Pufferpools pro Transaktion:

  ```
  (POOL_DATA_P_READS + POOL_INDEX_P_READS +
    POOL_TEMP_DATA_P_READS + POOL_TEMP_INDEX_P_READS)
    / TOTAL_COMMITS

  (POOL_DATA_WRITES + POOL_INDEX_WRITES)
    / TOTAL_COMMITS
  ```

Diese Messdaten haben eine enge Beziehung zu den Pufferpooltrefferquoten, dienen jedoch einem etwas anderen Zweck. Sie können zwar Zielwerte für Trefferquoten ins Auge fassen, jedoch gibt es keine möglichen Zielwerte für Lese- und Schreibvorgänge pro Transaktion. Warum also sollten diese Berechnungen eine Rolle spielen? Weil die Platten-E/A einen derart wichtigen Faktor in der Datenbankleistung darstellt, ist es nützlich, mehrere Methoden zu haben, diese zu untersuchen. Darüber hinaus schließen diese Berechnungen Schreibvorgänge mit ein, während Trefferquoten nur Lesevorgänge erfassen. Und schließlich ist es bei isolierter Betrachtung zum Beispiel schwierig zu erkennen, ob eine Indextrefferquote von 94 % die Mühe einer Optimierung wert ist. Wenn nur 100 logische Indexlesevorgänge pro Stunde erfolgen und 94 von diesen Vorgängen Daten im Pufferpool vorfinden, ist die Arbeitszeit für den Versuch, die verbleibenden 6 nicht als physische Lesevorgänge ausführen zu lassen, nicht gut investiert. Wenn allerdings eine Indextrefferquote von 94 % von einer Statistik begleitet wird, die besagt, dass jede Transaktion 20 physische Lesevorgänge ausgeführt hat (was nach Daten und Indizes sowie regulären und temporären Daten weiter aufgeschlüsselt werden könnte), könnten die Pufferpooltrefferquoten in der Tat eine Untersuchung rechtfertigen.

Die Messdaten sind nicht einfach nur physische Lese- und Schreibvorgänge, sondern werden pro Transaktion normalisiert. Dieser Trend wird bei vielen Messdaten verfolgt. Der Zweck besteht darin, die Messdaten von der Dauer der Zeit, über die sie erfasst wurden, abzukoppeln und davon unabhängig zu machen, ob das System zu diesem Zeitpunkt sehr hoch oder weniger hoch ausgelastet war. Im Allgemeinen stellt dieses Verfahren sicher, dass ähnliche Werte für Messdaten ermittelt werden, unabhängig davon, wie und wann Überwachungsdaten erfasst wurden. Ein gewisser Grad an Konsistenz bei der Auswahl des Zeitpunkts und der Länge der Datenerfassung ist sinnvoll. Jedoch verringert die Normalisierung, dass er eine entscheidende Rolle spielt.

- Das Verhältnis von gelesenen Zeilen zu ausgewählten Zeilen in der Datenbank:
  `ROWS_READ / ROWS_RETURNED`

  Diese Berechnung gibt einen Anhaltspunkt über die durchschnittliche Zahl von Zeilen, die aus Datenbanktabellen gelesen werden, um die Zeilen zu finden, die den Auswahlbedingungen entsprechen. Niedrige Werte sind ein Hinweis auf die Effizienz der Datensuche und geben im Allgemeinen an, dass Indizes effektiv genutzt werden. Dieser Wert kann zum Beispiel sehr hoch sein, wenn das System zahlreiche Tabellensuchen ausführt und Millionen von Zeilen geprüft werden müssen, um festzustellen, ob sie den Bedingungen für die Ergebnismenge entsprechen. Andererseits kann dieser Statistikwert sehr niedrig sein, wenn auf eine Tabelle über einen vollständig qualifizierten eindeutigen Index zugegriffen wird. Zugriffspläne mit reinem Indexzugriff (bei denen keine Zeilen aus der Tabelle gelesen werden müssen) erhöhen den Wert für ROWS_READ nicht.

  In einer OLTP-Umgebung ist dieser Messwert im Allgemeinen nicht höher als 2 oder 3, was darauf hinweist, dass der Zugriff überwiegend über Indizes und nicht durch Tabellensuchen erfolgt. Dieser Messwert bietet eine einfache Methode, die Planstabilität über einen Zeitraum hinweg zu überwachen: Eine unerwartete Erhöhung ist häufig ein Hinweis darauf, dass ein Index nicht mehr verwendet wird. Dies sollte untersucht werden.
- Der Zeitaufwand für das Sortieren pro Transaktion:
  `TOTAL_SORT_TIME / TOTAL_COMMITS`

  Dies ist eine effiziente Methode zur Behandlung von Sortierstatistiken, weil alle zusätzlichen Aufwände aufgrund von Sortierüberläufen automatisch in diesem Wert berücksichtigt werden. Trotzdem kann es auch sinnvoll sein, die Werte TO-

TAL_SORTS und SORT_OVERFLOWS zur bequemen Analyse zu erfassen, insbesondere wenn Ihr System bereits in der Vergangenheit Sortierprobleme hatte.

- Aufgelaufene Wartezeit pro Tausend Transaktionen:

  ```
  1000 * LOCK_WAIT_TIME / TOTAL_COMMITS
  ```

  Übermäßige Wartezeiten auf Sperren schlagen sich häufig in einer schlechten Antwortzeit nieder, sodass es wichtig ist, die Wartezeit zu überwachen. Der Wert wird auf ein Tausend Transaktionen normalisiert, weil die Wartezeit auf Sperren für eine einzelne Transaktion in der Regel recht gering ist. Die Skalierung auf ein Tausend Transaktionen stellt einfach Messwerte zur Verfügung, die leichter zu handhaben sind.

- Die Anzahl von Deadlocks und Überschreitungen von Sperrzeitlimits pro Tausend Transaktionen:

  ```
  1000 * (DEADLOCKS + LOCK_TIMEOUTS) / TOTAL_COMMITS
  ```

  Während Deadlocks in den meisten Produktionssystemen vergleichsweise selten auftreten, können Überschreitungen von Sperrzeitlimits häufiger vorkommen. Die Anwendung muss sie gewöhnlich auf ähnliche Weise behandeln: Die Ausführung der Transaktion muss von Anfang an wiederholt werden. Die Überwachung der Rate, mit der solche Fälle auftreten, hilft bei der Vermeidung einer Situation, in der viele Deadlocks oder Überschreitungen von Sperrzeitlimits eine erhebliche Zusatzbelastung im System verursachen, ohne dass sich der Datenbankadministrator dessen bewusst ist.

- Die Anzahl von Triggern zum Stehlen (Neuzuordnung) genutzter Seiten pro Tausend Transaktionen:

  ```
  1000 * POOL_DRTY_PG_STEAL_CLNS / TOTAL_COMMITS
  ```

  Das „Stehlen (Neuzuordnen) genutzter Seiten" ist die am wenigsten wünschenswerte Methode zur Auslösung einer Pufferpoolbereinigung. Im Wesentlichen wird die Verarbeitung einer SQL-Anwendung, die eine neue Pufferpoolseite benötigt, unterbrochen, während Aktualisierungen an der zur Bereinigung ausgewählten Seiten auf die Platte geschrieben werden. Wenn ein häufiges Stehlen genutzter Zeilen zugelassen wird, kann sich dies erheblich auf den Durchsatz und die Antwortzeit auswirken.

- Die Anzahl der Paketcacheeinfügungen pro Tausend Transaktionen:

  ```
  1000 * PKG_CACHE_INSERTS / TOTAL_COMMITS
  ```

  Paketcacheeinfügungen gehören zur normalen Ausführung des Systems. In großer Häufigkeit können sie jedoch einen signifikanten Konsumenten von CPU-Zeit darstellen. In vielen geeignet konfigurierten Systemen treten nach dem Erreichen eines stabilen Systembetriebszustands sehr wenige Paketcacheeinfügungen auf, weil das System statische SQL-Anweisungen oder zuvor vorbereitete dynamische SQL-Anweisungen verwendet bzw. wiederverwendet. In Systemen mit einem hohen Aufkommen an dynamischen Ad-hoc-SQL-Anweisungen sind SQL-Kompilierungen und Paketcacheeinfügungen unvermeidbar. Allerdings ist dieser Messwert dazu gedacht, auf einen dritten Typ von Situation zu überwachen, bei dem Anwendungen unabsichtlich Paketcacheänderungen verursachen, indem sie vorbereitete Anweisungen nicht wiederverwenden oder keine Parametermarken in ihrem häufig ausgeführten SQL verwenden.

- Der Zeitraum, in dem ein Agent auf das Schreiben von Protokollsätzen auf die Platte wartet:

  ```
  LOG_WRITE_TIME
    / TOTAL_COMMITS
  ```

Das Transaktionsprotokoll birgt ein bedeutendes Potenzial, zu einem Systemengpass zu werden. Dies ist gegebenenfalls auf einen hohen Aktivitätsgrad, auf eine ungeeignete Konfiguration oder auch auf andere Ursachen zurückzuführen. Durch eine Überwachung der Protokollaktivitäten können Sie Probleme sowohl auf der DB2-Seite (d. h. an einer steigenden Anzahl von Protokollanforderungen durch die Anwendung) als auch auf der Systemseite (häufig an einer Verschlechterung der Leistung des Protokollsubsystems, die durch Hardware- oder Konfigurationsprobleme verursacht wird) erkennen.

- In Umgebungen mit partitionierten Datenbanken - die Anzahl der Fast Communications Manager-Puffer (FCM-Puffer), die zwischen Partitionen gesendet und empfangen werden:

  ```
  FCM_SENDS_TOTAL, FCM_RECVS_TOTAL
  ```

  Diese Messwerte liefern die Rate des Datenflusses zwischen verschiedenen Partitionen im Cluster und geben insbesondere Auskunft darüber, ob der Datenfluss ausgeglichen ist. Signifikante Unterschiede in den Anzahlen der Puffer, die von verschiedenen Partitionen empfangen werden, können auf ungleichmäßige Datenvolumen hinweisen, die durch das Hashverfahren auf die einzelnen Partitionen verteilt wurden.

Partitionsübergreifende Überwachung in Umgebungen mit partitionierten Datenbanken

Fast alle der einzelnen oben erwähnten Monitorelementwerte werden auf der Basis einer einzelnen Partition geliefert.

Im Allgemeinen ist zu erwarten, dass sich die meisten Überwachungsstatistiken relativ gleichmäßig über alle Partitionen in derselben DB2-Partitionsgruppe hinweg darstellen. Bedeutende Differenzen können auf eine ungleiche Datenverteilung hinweisen. Die folgenden partitionsübergreifenden Vergleiche sind zum Beispiel zu verfolgen:

- Logische und physische Pufferpoollesevorgänge für Daten, Indizes und temporäre Tabellen
- Gelesene Zeilen auf der Partitionsebene und für große Tabellen
- Sortierzeiten und Sortierüberläufe
- Gesendete und empfangene FCM-Puffer
- CPU- und E/A-Auslastung

Abnormale Werte in Überwachungsdaten

Bei der Behebung von Leistungsproblemen liefert die Ermittlung abnormaler Werte oft den Schlüssel zu einer richtigen Interpretation der Systemleistungswerte.

Wenn sich bestimmte Leistungswerte verschlechtert haben und somit abnormal sind, liefern Monitorelemente Hinweise auf die Art des Leistungsproblems. Im Allgemeinen handelt es sich bei einem schlechten Wert um einen Wert, der über dem erwarteten Wert liegt. Ein Beispiel dafür ist ein Wartestatus für Sperren, der länger andauert als normal. Ein abnormaler Wert kann jedoch auch unter dem erwarteten Wert liegen, z. B. bei der Pufferpooltrefferquote. Je nach Situation können Sie nur eine oder gleich mehrere Methoden anwenden, um festzustellen, ob ein Wert bedenklich ist.

Eine Methode besteht darin, sich nach den in der Branche etablierten Faustregeln oder bewährten Verfahren zu richten. Eine Faustregel besagt z. B., dass eine Pufferpooltrefferquote von 80 - 85 % für Daten in einer OLTP-Umgebung ein gutes Er-

gebnis darstellt. Dabei ist zu beachten, dass sich diese Faustregel auf OLTP-Umgebungen bezieht und keine sinnvolle Richtschnur für Data-Warehouses darstellt, bei denen die Datentrefferquote naturgemäß oft wesentlicher niedriger liegt.

Eine andere Methode besteht darin, die aktuellen Werte mit den Grundwerten zu vergleichen, die zuvor erfasst wurden. Diese Vorgehensweise liefert oft aussagefähigere Werte. Sie setzt allerdings voraus, dass bereits unter normalen Bedingungen eine adäquate Strategie zum Überwachen von Betriebsdaten beim Erfassen und Speichern der wichtigsten Leistungsmessdaten angewendet wurde. Angenommen z. B., Sie stellen fest, dass die aktuelle Pufferpooltrefferquote bei 85 % liegt. Den Branchenstandards nach handelt es sich dabei um einen normalen Wert. Wenn jedoch vor dem Auftreten des Leistungsproblems ein Wert von 99 % aufgezeichnet wurde, gilt dieser Wert als abnormal.

Schließlich gibt es noch die Möglichkeit, die aktuellen Werte mit den aktuellen Werten eines vergleichbaren Systems zu vergleichen. Eine Pufferpooltrefferquote von 85 % kann z. B. abnormal sein, wenn die Pufferpooltrefferquote bei vergleichbaren Systemen bei 99 % liegt.

Dienstprogramm Governor

Das Dienstprogramm Governor überwacht das Verhalten von Anwendungen, die für eine Datenbank ausgeführt werden, und kann abhängig von den Regeln, die Sie in der Konfigurationsdatei des Governors angeben, dieses Verhalten ändern.

Wichtig: Bedingt durch die neuen strategischen Funktionen des DB2-Workload-Managers, die mit DB2 Version 9.5 eingeführt wurden, gilt das DB2-Dienstprogramm Governor in Version 9.7 als veraltet und wird in einem zukünftigen Release möglicherweise entfernt. Weitere Informationen zur Unterstützung des Dienstprogramms Governor finden Sie in dem Abschnitt „DB2 Governor und Query Patroller gelten als veraltet". Nähere Informationen zum DB2-Workload-Manager und den Funktionen des Workload-Managers, die das Dienstprogramm Governor ersetzen, finden Sie in der Einführung zu den Konzepten, die dem DB2-Workload-Manager zugrunde liegen, und den häufig gestellten Fragen zum DB2-Workload-Manager.

Eine Governor-Instanz besteht aus einem Front-End-Dienstprogramm und mindestens einem Dämon. Jede Governor-Instanz 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.

Der Governor verwaltet Anwendungstransaktionen, wie es durch die Regeln in der Konfigurationsdatei angegeben ist. Zum Beispiel könnte durch das Anwenden einer Regel festgestellt werden, dass eine Anwendung eine bestimmte Ressource übermäßig beansprucht. Die Regel gibt darüber hinaus 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 der 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.

Der Governor protokolliert alle von ihm durchgeführten Aktionen.

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

Starten des Governors

Das Dienstprogramm 'Governor' überwacht Anwendungen, die mit einer Datenbank verbunden sind, und ändert das Verhalten dieser Anwendungen gemäß den Regeln, die Sie in einer Konfigurationsdatei des Governors für diese Datenbank angeben.

Informationen zu diesem Vorgang

Wichtig: Mit der Einführung der neuen Workload-Management-Features in DB2 Version 9.5 gilt das DB2-Dienstprogramm Governor in Version 9.7 als veraltet und wird möglicherweise in einen zukünftigen Release entfernt. Weitere Informationen hierzu finden Sie im Abschnitt „ DB2 Governor und Query Patroller gelten als veraltet" in der Veröffentlichung *Neuerungen in DB2 Version 9.7*.

Bevor Sie den Governor starten, müssen Sie die Konfigurationsdatei erstellen.

Zum Starten des Governors benötigen Sie die Berechtigung *sysadm* oder *sysctrl*.

Führen Sie zum Starten des Governors den Befehl **db2gov** aus, indem Sie die folgenden erforderlichen Parameter angeben:

START *datenbankname*
Der von Ihnen angegebene Datenbankname muss mit dem Namen der Datenbank in der Konfigurationsdatei des Governors übereinstimmen.

konfigurationsdatei
Der Name der Konfigurationsdatei des Governors für diese Datenbank. Wenn sich die Datei nicht an der Standardposition (d. h. im Verzeichnis `sqllib`) befindet, müssen Sie außer dem Dateinamen auch den Dateipfad angeben.

protokolldatei
Der Basisname der Protokolldatei für diesen Governor. Für eine partitionierte Datenbank wird die Nummer der Datenbankpartition für jede Datenbankpartition hinzugefügt, in der ein Dämon für diese Instanz des Governors ausgeführt wird.

Zum Starten des Governors in einer einzelnen Datenbankpartition einer partitionierten Datenbank geben Sie die Option **dbpartitionnum** an.

Um den Governor zum Beispiel in Datenbankpartition 3 einer Datenbank mit dem Namen SALES unter Verwendung einer Konfigurationsdatei mit dem Namen `salescfg` und einer Protokolldatei mit dem Namen `saleslog` zu starten, geben Sie den folgenden Befehl ein:

```
db2gov start sales dbpartitionnum 3 salescfg saleslog
```

Zum Starten des Governors in allen Datenbankpartitionen geben Sie den folgenden Befehl ein:

```
db2gov start sales salescfg saleslog
```

Governor-Dämon

Der Governor-Dämon erfasst Informationen über die Anwendungen, die für eine Datenbank ausgeführt werden.

Der Governor-Dämon führt bei jedem Start die folgende wiederkehrende Abfolge von Tasks aus.

1. Der Dämon 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. Der Dämon fordert Momentaufnahmeninformationen zur Ressourcennutzungsstatistik für jede Anwendung und jeden Agenten an, die bzw. der die Datenbank bearbeitet.
3. Der Dämon überprüft die Statistik für jede einzelne Anwendung anhand der Regeln in der Governor-Konfigurationsdatei. Wenn eine Regel zutrifft, führt der Governor die angegebene Aktion aus. Der Governor vergleicht aufgelaufene Informationen mit Werten, die in der Konfigurationsdatei definiert sind. Dies bedeutet: Falls die Konfigurationsdatei mit neuen Werten aktualisiert wird, die eine Anwendung möglicherweise bereits überschritten hat, werden die Regeln, die für die betreffende Überschreitung gelten, im nächsten Governor-Intervall auf die Anwendung angewandt.
4. Der Dämon schreibt für jede ausgeführte Aktion einen Datensatz in die Governor-Protokolldatei.

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

Stellt der Governor einen Fehler oder ein Stoppsignal fest, führt er vor der Beendigung eine Bereinigung durch. Bei der Bereinigung werden mithilfe 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. Hierdurch wird sichergestellt, dass keine Agenten nach Beendigung des Governors mit anderen als den Standardprioritäten aktiv bleiben. Falls ein Fehler auftritt, schreibt der Governor eine Nachricht in das Protokoll mit Benachrichtigungen für die Verwaltung, die die abnormale Beendigung angibt.

Der Governor kann nicht zur Anpassung der Agentenprioritäten verwendet werden, wenn der Wert des Konfigurationsparameters **agentpri** des Datenbankmanagers vom Systemstandardwert abweicht.

Obwohl der Governor-Dämon keine Datenbankanwendung ist und daher nicht über eine Verbindung zur Datenbank verfügt, besteht für ihn dennoch eine Instanzzuordnung. Da er Anforderungen für Momentaufnahmen absetzen kann, kann der Governor-Dämon erkennen, wenn der Datenbankmanager beendet wird.

Konfigurationsdatei des Governors

Die Konfigurationsdatei des Governors enthält die Regeln für die Steuerung von Anwendungen, die für eine Datenbank ausgeführt werden.

Der Governor wertet jede Regel aus und führt die angegebenen Aktionen aus, wenn die entsprechende Regel zutrifft.

Die Konfigurationsdatei des Governors enthält allgemeine Klauseln zur Identifizierung der zu überwachenden Datenbank (erforderlich), des Intervalls für das Schreiben von Abrechnungssätzen mit CPU-Auslastungsstatisktikdaten sowie des Ruheintervalls für Governor-Dämonen. Die Konfigurationsdatei kann darüber hinaus ein oder mehrere optionale Regelanweisungen zur Anwendungsüberwachung enthalten. Die folgenden Richtlinien gelten sowohl für allgemeine Klauseln als auch für Regelanweisungen:

- Allgemeine Kommentare müssen in geschweifte Klammer {...} eingeschlossen werden.
- In den meisten Fällen werden Werte in Großschreibung, in Kleinschreibung oder in gemischter Groß-/Kleinschreibung angegeben. Ein Ausnahme bildet der Anwendungsname (der nach der Klausel `applname` angegeben wird), bei dem die Groß-/Kleinschreibung beachtet werden muss.
- Jede allgemeine Klausel oder Regelanweisung muss mit einem Semikolon ; beendet werden.

Wenn eine Regel aktualisiert werden muss, bearbeiten Sie die Konfigurationsdatei, ohne den Governor zu stoppen. Jeder Governor-Dämon erkennt die geänderte Datei und liest sie erneut.

In einer Umgebung mit partitionierten Datenbanken muss die Konfigurationsdatei des Governors in einem Verzeichnis erstellt werden, dass für alle Datenbankpartitionen angehängt ist, sodass der Governor-Dämon in jeder Datenbankpartition dieselbe Konfigurationsdatei lesen kann.

Allgemeine Klauseln

Die folgenden Klauseln können in einer Konfigurationsdatei des Governors nur einmal angegeben werden.

dbname
: Der Name oder Aliasname der zu überwachenden Datenbank. Diese Klausel ist erforderlich.

account *n*
: Das Intervall (in Minuten), nach dem Abrechnungssätze mit CPU-Auslastungsstatistikdaten für die einzelnen Verbindungen geschrieben werden. Diese Option ist unter Windows-Betriebssystemen nicht verfügbar. Auf einigen Plattformen stehen keine CPU-Statistikdaten für Snapshot Monitor zur Verfügung. In diesem Fall wird die Klausel 'account' ignoriert.

  Findet eine kurze Sitzung 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 der 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 *n*
: Das Intervall in Sekunden, nach dem der Dämon jeweils aktiv wird. Wenn Sie diese Klausel nicht angeben, wird der Standardwert von 120 Sekunden verwendet.

Regelklauseln

Regelanweisungen geben an, wie Anwendungen zu steuern sind, und werden aus kleineren Komponenten zusammengesetzt, die als Regelklauseln bezeichnet werden. Wenn die Regelklauseln eingesetzt werden, müssen sie in einer bestimmten Reihenfolge in der Regelanweisung stehen, und zwar:

1. **desc**: Ein Kommentar zur jeweiligen Regel, der in einfache oder doppelte Anführungszeichen eingeschlossen ist.
2. **time**: Der Zeitpunkt, zu dem die Regel ausgewertet wird.
3. **authid**: Eine oder mehrere Berechtigungs-IDs, unter der die Anwendung Anweisungen ausführt.
4. **applname**: Der Name der ausführbaren Datei oder Objektdatei, die eine Verbindung zur Datenbank herstellt. Dieser Name ist von der Groß-/Kleinschreibung abhängig. Wenn der Anwendungsname Leerzeichen enthält, muss er in doppelte Anführungszeichen eingeschlossen werden.
5. **setlimit**: Grenzwerte, die der Governor überprüft, wie beispielsweise die CPU-Zeit, die Anzahl der zurückgegebenen Zeilen oder die Leerlaufzeit. Auf einigen Plattformen stehen keine CPU-Statistikdaten für Snapshot Monitor zur Verfügung. Ist dies der Fall, wird die Klausel 'setlimit' ignoriert.
6. **action**: Die beim Erreichen eines Grenzwerts auszuführende Aktion. Wenn keine Aktion angegeben wird, verringert der Governor die Priorität von Agenten, die für die Anwendung aktiv sind, um den Wert 10, wenn ein Grenzwert erreicht wird. Zu den Aktionen, die für eine Anwendung ausgeführt werden können, gehören das Reduzieren der Agentenpriorität, eine erzwungene Trennung der Verbindung zur Datenbank und das Festlegen von Zeitplanungsoptionen für die zugehörigen Operationen.

Zur Definition einer Regel kombinieren Sie die Regelklauseln, wobei jede Klausel in einer Regelanweisung nur einmal verwendet werden darf.

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

Wenn mehrere Regeln auf eine Anwendung zutreffen, werden alle Regeln angewendet. Normalerweise wird die Aktion, die dem zuerst festgestellten Grenzwert zugeordnet ist, zuerst angewendet. Zu einer Ausnahme kommt es, wenn Sie den Wert -1 für eine Regelklausel angeben: Ein anschließend angegebener Wert für dieselbe Klausel kann nur den zuvor angegebenen Wert überschreiben; andere Klauseln in der vorhergehenden Regelanweisung sind weiterhin wirksam.

Beispiel: Eine Regelanweisung verwendet die Klauseln `rowssel 100000` und `uowtime 3600`, um anzugeben, dass die Priorität einer Anwendung reduziert wird, falls die abgelaufene Zeit der Anwendung eine Stunde überschreitet oder falls 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 zutrifft.

Reihenfolge der Regelanwendung

Der Governor verarbeitet die Regeln vom Anfang der Konfigurationsdatei bis zum Ende. Wenn jedoch die Klausel `setlimit` in einer bestimmten Regelanweisung we-

niger restriktiv ist als dieselbe Klausel in einer vorhergehenden Regelanweisung, gilt die restriktivere Regel. Im folgenden Beispiel gilt für ADMIN weiterhin eine Grenze von 5000 Zeilen, da die erste Regel restriktiver ist.

```
desc "Verbindung für alle trennen, die 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 vorhergehende, restriktivere Regel überschreibt, müssen Sie -1 angeben, um die vorhergehende Regel zu löschen, bevor Sie die neue Regel anwenden. 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 "Verbindung für alle trennen, die 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;
```

Beispiel für eine Governor-Konfigurationsdatei

```
{ Der Datenbankname lautet SAMPLE; Abrechnungsdatensätze alle 30 Minuten;
  einmal pro Sekunde aktiv werden. }
dbname sample; account 30; interval 1;

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, quryA
setlimit cpu 3 locks 1000 rowssel 500 rowsread 5000;

desc "Der 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 "Erhöhen der Priorität einer wichtigen Anwendung, sodass sie immer
      schnell verarbeitet wird."
applname V1app setlimit cpu 1 locks 1 rowssel 1 action priority -20;
desc "Für einige Personen, z. B. den Datenbankadministrator (und andere),
     sollten keine Grenzwerte festgelegt werden. Da es sich um die letzte
     Spezifikation in der Datei handelt, überschreibt sie die vorhergehenden."
authid gene, hershel, janet setlimit cpu -1 locks -1 rowssel -1 uowtime -1;
```

Klauseln für Governor-Regeln

Jede Regel in der Konfigurationsdatei des Governors 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 für die müssen in der angezeigten Reihenfolge angegeben werden.

Optionale Startklauseln

desc Gibt eine Beschreibung für die Regel an. Die Beschreibung muss in einfachen oder doppelten Anführungszeichen stehen.

time Gibt die Zeitspanne an, während der die Regel auswertet werden soll. Die Zeitspanne muss im folgenden Format angegeben werden: `time hh:mm hh:mm`, zum Beispiel: `time 8:00 18:00`. Wird diese Klausel nicht angegeben, wird die Regel 24 Stunden am Tag ausgewertet.

authid Gibt eine oder mehrere Berechtigungs-IDs an, unter denen die Anwendung ausgeführt wird. Mehrere Berechtigungs-IDs müssen durch ein Komma (,) voneinander getrennt werden; zum Beispiel: `authid gene, michael, james`. Wird diese Klausel nicht angegeben, gilt die Regel für alle Berechtigungs-IDs.

applname
Gibt den Namen der ausführbaren Datei oder der Objektdatei an, für die eine Verbindung zur Datenbank hergestellt wird. Mehrere Anwendungsnamen müssen durch ein Komma (,) voneinander getrennt werden; zum Beispiel: `applname db2bp, batch, geneprog`. Wird diese Klausel nicht angegeben, gilt die Regel für alle Anwendungsnamen.

Anmerkung:

1. Bei Anwendungsnamen muss Groß-/Kleinschreibung beachtet werden.
2. Der Datenbankmanager schneidet alle Anwendungsnamen auf 20 Zeichen ab. Stellen Sie sicher, dass die Anwendung, für die Sie den Governor verwenden möchten, eindeutig durch die ersten 20 Zeichen des Anwendungsnamen identifiziert wird. Anwendungsnamen in der Governor-Konfigurationsdatei werden auf 20 Zeichen abgeschnitten, damit sie mit ihrer internen Darstellung übereinstimmen.

Grenzwertklauseln

setlimit
Gibt mindestens einen Grenzwerte an, den der Governor überprüfen soll. Die Grenzwerte dürfen nur den Wert -1 oder Werte größer als 0 annehmen (z. B. `cpu -1 locks 1000 rowssel 10000`). Es muss mindestens ein Grenzwert angegeben werden, und jeder Grenzwert, der nicht in einer Regelanweisung angegeben wird, wird nicht durch die entsprechende Regel eingeschränkt. Der Governor kann folgende Grenzwerte überprüfen:

cpu *n* Gibt die Anzahl der CPU-Sekunden an, die eine Anwendung nutzen kann. Wenn Sie den Wert -1 angeben, wird die CPU-Nutzung der Anwendung nicht begrenzt.

idle *n* Gibt die für eine Verbindung zulässige Leerlaufzeit in Sekunden an. Wenn Sie den Wert -1 angeben, wird die Leerlaufzeit der Verbindung nicht eingeschränkt.

Anmerkung: Einige Datenbankdienstprogramme, wie beispielsweise die Dienstprogramme für Backup und Restore, stellen eine Verbindung zur Datenbank her und führen die anfallenden Arbeiten dann über EDUs (Engine-Dispatchable-Units, von der Steuerkomponente zuteilbare Einheiten) aus, die für den Governor nicht sichtbar sind. Diese Datenbankverbindungen werden als im Leerlauf befindlich identifiziert, sodass es zu einer Überschreitung des für die Leerlaufzeit definierten Zeitlimits kommen kann. Um zu verhindern, dass der Governor entsprechende Maßnahmen in Bezug auf diese Dienstprogramme einleitet, geben Sie für diese über die Berechtigungs-ID, über die sie aufgerufen wurden, den Wert -1 an. Um beispielsweise zu verhindern, dass der Governor für Dienstprogramme aktiv wird, die unter der Berechtigungs-ID DB2SYS ausgeführt werden, geben Sie `authid DB2SYS setlimit idle -1` an.

locks *n*
Gibt die Anzahl der Sperren an, die eine Anwendung halten kann. Wenn Sie den Wert -1 angeben, wird die Anzahl der von der Anwendung gehaltenen Sperren nicht eingeschränkt.

rowsread *n*
Gibt die Anzahl der von einer Anwendung auswählbaren Zeilen an. Wenn Sie den Wert -1 angeben, wird die Anzahl der von einer Anwendung auswählbaren Zeilen nicht eingeschränkt. Der Maximalwert, der angegeben werden kann, ist 4.294.967.298.

Anmerkung: Dieser Grenzwert ist nicht mit `rowssel` identisch. Der Unterschied besteht darin, dass sich `rowsread` auf die Anzahl der Zeilen bezieht, die gelesen werden müssen, um die Ergebnismenge zurückzugeben. Diese Anzahl umfasst die von der Engine ausgeführten Lesevorgänge und kann bei der Verwendung von Indizes reduziert werden.

rowssel *n*
Gibt die Anzahl von Zeilen an, die an eine Anwendung zurückgegeben werden können. Dieser Wert ist nur für die Koordinatordatenbankpartition ungleich null. Wenn Sie den Wert -1 angeben, wird die Anzahl der Zeilen, die zurückgegeben werden können, nicht eingeschränkt. Der Maximalwert, der angegeben werden kann, ist 4.294.967.298.

uowtime *n*
Gibt die Zeitspanne in Sekunden an, die verstreichen kann, nachdem eine UOW (Unit of Work, Arbeitseinheit) zum ersten Mal aktiv wird. Wenn Sie den Wert -1 angeben, wird die verstrichene Zeit nicht eingeschränkt.

Anmerkung: Wenn Sie die API sqlmon zum Inaktivieren des Monitorschalters für die UOW oder des Monitorschalters für die Zeitmarke verwendet haben, wird die Fähigkeit des Governors beein-

trächtigt, Anwendungen auf der Grundlage der verstrichenen Zeit der UOW zu regeln. Der Governor verwendet das Überwachungsprogramm zum Sammeln von Systeminformationen. Wenn eine UOW (Unit of Work) der Anwendung vor dem Governor gestartet wird, regelt der Governor diese UOW nicht.

Aktionsklauseln

action Gibt die Aktion an, die ausgeführt werden soll, wenn einer oder mehrere der angegebenen Grenzwerte überschritten werden. Wenn ein Grenzwert überschritten wird und die Klausel `action` nicht angegeben ist, verringert der Governor die Priorität der für die Anwendung aktiven Agenten um 10.

force Gibt an, dass der Agent, der für die Anwendung ausgeführt wird, beendet werden soll. (Mit dem Befehl **FORCE APPLICATION** wird der Koordinatoragent beendet.)

Anmerkung: In Umgebungen mit partitionierten Datenbanken wird die Aktion `force` nur ausgeführt, wenn der Governordämon in der Koordinatordatenbankpartition der Anwendung ausgeführt wird. Wenn also ein Governordämon in Datenbankpartition A ausgeführt wird und ein Grenzwert für eine Anwendung überschritten wird, deren Koordinatordatenbankpartition die Datenbankpartition B ist, wird die Aktion `force` übersprungen.

nice *n* Gibt eine Änderung der relativen Priorität der für die Anwendung aktiven Agenten an. Gültige Wertebereiche liegen im Bereich von -20 bis +20 (UNIX-Systeme) bzw. im Bereich von -1 bis 6 (Windows-Plattformen).

- Auf Linux- und UNIX-Systemen muss der Konfigurationsparameter **agentpri** des Datenbankmanagers auf den Standardwert gesetzt sein, da er andernfalls den Wert für `nice` überschreibt.
- Auf Windows-Plattformen können der Konfigurationsparameter **agentpri** des Datenbankmanagers und der Wert für `nice` gemeinsam verwendet werden.

Sie können mit dem Governor die Priorität von Anwendungen steuern, die in der Superklasse des Standardbenutzerserivce SYSDEFAULTUSERCLASS ausgeführt werden. Wenn Sie den Governor verwenden, um die Priorität einer Anwendung zu senken, die in dieser Service-Superklasse ausgeführt wird, trennt sich der Agent selbst von seinem abgehenden Korrelator (falls ihm einer zugeordnet ist) und legt seine relative Priorität entsprechend der durch Governor angegebenen Priorität fest. Sie können mit dem Governor keine Prioritäten von Agenten in benutzerdefinierten Service-Superklassen oder -Unterklassen ändern. Stattdessen müssen Sie die Einstellung für die Agentenpriorität der Service-Superklasse oder -Unterklasse verwenden, um Anwendungen zu steuern, die in diesen Serviceklassen ausgeführt werden. Sie können den Governor jedoch verwenden, um Verbindungen in jeder Serviceklasse zwangsweise zu trennen.

Anmerkung: Unter AIX 5.3 muss der Eigner die Berechtigungsgruppe (Capability) CAP_NUMA_ATTACH haben, um die relative Priorität von Agenten zu erhöhen, die für die Anwendung tätig sind. Zum Erteilen dieser Berechtigungsgruppe melden Sie sich mit Rootberechtigung an und führen den folgenden Befehl aus:

```
chuser capabilities=CAP_NUMA_ATTACH,CAP_PROPAGATE <Benutzer-ID>
```

Dabei ist <Benutzer-ID> der Instanzname.

Unter Solaris 10 und höheren Versionen muss der Eigner über das Zugriffsrecht 'proc_priocntl' verfügen, um die relative Priorität von Agenten erhöhen zu können, die für die Anwendung ausgeführt werden. Zum Erteilen dieses Zugriffsrechts melden Sie sich als `Root` an und führen Sie den folgenden Befehl aus:

```
usermod -K defaultpriv=basic,proc_priocntl db2user
```

In diesem Beispiel wird 'proc_priocntl' zur Standardgruppe der Zugriffsrechte des Benutzers 'db2user' hinzugefügt.

Darüber hinaus muss bei der Ausführung von DB2 in einer nicht globalen Zone von Solaris das Zugriffsrecht 'proc_priocntl' zur Zugriffsrechtsgruppe 'Limit' (L) für die Zone hinzugefügt werden. Um der Zone dieses Zugriffsrecht zu erteilen, melden Sie sich als `Root` an und führen Sie den folgenden Befehl aus:

```
global# zonecfg -z db2zone
zonecfg:db2zone> set limitpriv="default,proc_priocntl"
```

In diesem Beispiel wird 'proc_priocntl' zur Zugriffsrechtsgruppe 'Limit' (L) für die Zone 'db2zone' hinzugefügt.

Unter Solaris 9 ist keine Funktion vorhanden, die es DB2 ermöglicht, die relative Priorität von Agenten zu erhöhen. Führen Sie ein Upgrade auf Solaris 10 oder eine höhere Version durch, um die Klausel ACTION NICE des DB2-Governors verwenden zu können.

schedule [class]

Die Zeitplanung verbessert die Prioritäten von Agenten, die für Anwendungen ausgeführt werden. Das Ziel ist, die durchschnittliche Antwortzeit zu minimieren, ohne Anwendungen zu benachteiligen.

Der Governor wählt die Anwendungen mit den jeweils höchsten Werten für die Zeitplanung auf der Basis der folgenden Kriterien aus:

- Die Anwendung, die die größten Anzahl von Sperren hält (hierdurch soll die Anzahl der Situationen, in denen auf Sperren gewartet wird, reduziert werden).
- Die älteste Anwendung.
- Die Anwendung mit der kürzesten geschätzten Restlaufzeit (hierdurch sollen möglichst viele kurzzeitig aktive Anweisungen während des Intervalls abgeschlossen werden).

Die drei beim jeweiligen Kriterium höchstplatzierten Anwendungen erhalten höhere Prioritäten als alle anderen Anwendungen. Das bedeutet, der höchstplatzierten Anwendung in jeder Kriteriumsgruppe wird die höchste Priorität, der nächsthöheren Anwendung die zweithöchste Priorität und der dritthöchsten Anwendung die dritthöchste Priorität zugewiesen. Wenn eine einzelne Anwendung auf einem der drei höchsten Plätze für mehrere Kriterien rangiert, erhält sie die entsprechende Priorität für das Kriterium, bei dem sie den höchsten Rang innehat. Die nächsthöhere Anwendung erhält die nächsthöhere Priorität für die anderen Kriterien. Wenn beispielsweise Anwendung A die meisten Sperren hält, jedoch die

drittkürzeste geschätzte Restlaufzeit aufweist, erhält sie die höchste Priorität für das erste Kriterium. Die Anwendung, die bei der kürzesten geschätzten Restlaufzeit auf Platz vier rangiert, erhält dadurch die dritthöchste Priorität für dieses Kriterium.

Die Anwendungen, die von dieser Governor-Regel ausgewählt werden, werden in bis zu drei Klassen unterteilt. Für jede Klasse wählt der Governor neun Anwendungen aus, welche nach den oben beschriebenen Kriterien jeweils die drei höchstplatzierten Anwendungen jeder Klasse sind. Wenn Sie die Option `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 regiert, in der sie ausgewählt wird.

Anmerkung: Wenn Sie die API sqlmon verwendet haben, um den Anweisungsschalter zu inaktivieren, wird die Fähigkeit des Governors beeinträchtigt, Anwendungen auf der Grundlage der verstrichenen Zeit der Anweisung zu steuern. Der Governor verwendet das Überwachungsprogramm zum Sammeln von Systeminformationen. Wenn Sie die Schalter in der Konfigurationsdatei des Datenbankmanagers inaktivieren, werden sie für die gesamte Instanz inaktiviert, und der Governor erhält diese Informationen nicht mehr.
Eine Zeitplanungsaktion kann folgenden Zwecken dienen:

- Sicherstellen, dass Anwendungen in verschiedenen Gruppen Zeit zugewiesen bekommen, ohne dass die Zeit unter allen Anwendungen gleichmäßig 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. Mithilfe der Prioritäten kann der 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 Zeitplanungsklasse zusammenfassen."
  applname medq1, medq2, medq3, medq4, medq5
  setlimit cpu -1
  action schedule class;

  desc "Lange Anwendungen in 1 Zeitplanungsklasse zusammenfassen."
  applname longq1, longq2, longq3, longq4, longq5, longq6
  setlimit cpu -1
  action schedule class;
  ```

- Sicherstellen, dass unterschiedliche Benutzergruppen (z. B. Abteilungen in Unternehmen) gleiche Kriterien für die Vergabe von Prioritäten erhalten. 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;
```

- Zulassen, dass der Governor alle Anwendungen zeitlich plant.

  Wenn die Option `class` nicht angegeben wird, erstellt der Governor eigene Klassen auf der Basis der Anzahl aktiver Anwendungen, für die die Zeitplanaktion gilt, und versetzt Anwendungen in die verschiedenen Klassen. Grundlage hierfür ist die Aufwandsschätzung des Abfragecompilers für die Abfrage, die die Anwendung ausführt. Der Administrator kann festlegen, dass alle Anwendung zeitlich geplant werden, indem er keine Qualifikationsmerkmale zur Auswahl der Anwendungen angibt, d. h., indem er die Klauseln `applname`, `authid` bzw. `setlimit` nicht angibt.

Governor-Protokolldateien

Wenn ein Governor-Dämon eine Aktion ausführt, schreibt er einen Datensatz in seine Protokolldatei.

Zu diesen Aktionen gehören:

- Starten oder Stoppen des Governors
- Lesen der Governor-Konfigurationsdatei
- Ändern der Priorität einer Anwendung
- Erzwungene Beendigung einer Anwendung
- Feststellen einer Fehler- oder Warnbedingung

Jeder Governor-Dämon verfügt über eine separate Protokolldatei, wodurch Engpässe aufgrund von Dateisperrungen vermieden werden, die auftreten können, wenn eine große Anzahl von Governor-Dämonen gleichzeitig versucht, in dieselbe Datei zu schreiben. Mithilfe des Befehls **db2govlg** können Sie die Protokolldateien des Governors abfragen.

Die Protokolldateien werden im Unterverzeichnis `log` des Verzeichnisses `sqllib` gespeichert. Eine Ausnahme bilden die Windows-Betriebssysteme, auf denen sich das Unterverzeichnis `log` im Verzeichnis für gemeinsame Anwendungsdaten (Com-

mon Application Data) befindet, das die Windows-Betriebssysteme zum Speichern von Anwendungsprotokolldateien verenden. 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 Nummer der Datenbankpartition, in der der Governor-Dämon ausgeführt wird, automatisch an den Protokolldateinamen angefügt.

Datensatzformat der Protokolldatei

Ein Datensatz in der Protokolldatei weist das folgende Format auf:

```
Datum Uhrzeit Datenbankpartitionsnummer Satztyp Nachricht
```

Das Format der Felder *Datum* und *Uhrzeit* ist *jjjj-mm-tt-hh.mm.ss*. Sie können die Protokolldateien in jeder Datenbankpartition durch Sortieren nach diesem Feld zusammenfügen. Das Feld *Datenbankpartitionsnummer* enthält die Nummer der Datenbankpartition, in der der Governor ausgeführt wird.

Das Feld *Satztyp* enthält unterschiedliche Werte, je nach Typ des in das Protokoll geschriebenen Satzes. Folgende Werte können eingetragen werden:

- ACCOUNT: Die Abrechnungsstatistik der Anwendung.
- ERROR: Ein Fehler ist aufgetreten.
- FORCE: Eine Anwendung wurde zwangsweise getrennt.
- NICE: Die Priorität einer Anwendung wurde geändert.
- READCFG: Der Governor hat die Konfigurationsdatei gelesen.
- SCHEDGRP: Eine Änderung der Agentenpriorität ist aufgetreten.
- START: Der Governor wurde gestartet.
- STOP: Der Governor wurde gestoppt.
- WARNING: Eine Warnung ist aufgetreten.

Einige dieser Werte werden im Folgenden näher beschrieben.

ACCOUNT

Der Datensatz ACCOUNT wird in den folgenden Situationen geschrieben:

- Der Wert des Monitorelements **agent_usr_cpu_time** oder **agent_sys_cpu_time** für eine Anwendung wurde geändert, seit der letzte Datensatz ACCOUNT für diese Anwendung geschrieben wurde.
- Eine Anwendung ist nicht mehr aktiv.

Der Datensatz ACCOUNT weist folgendes Format auf:

```
<authentifizierungs-id> <anwendungs-id> <anwendungsname> <verbindungsdauer>
<agent_usr_cpu-delta>
  <agent_sys_cpu-delta>
```

ERROR

Der Datensatz ERROR wird geschrieben, wenn der Governor-Dämon beendet werden muss.

FORCE

Der Datensatz FORCE wird geschrieben, wenn der Governor die Verbindung für eine Anwendung trennt, wie in den Regeln der Konfigurationsdatei des Governors festgelegt. Der Datensatz FORCE weist folgendes Format auf:

```
<anwendungsname> <authentifizierungs-id> <anwendungs-id>
<koordinierende_partition> <konfigurationszeile> <einschränkung_überschritten>
```

Dabei gilt:

koordinierende_partition
: Gibt die Nummer der Koordinatordatenbankpartition der Anwendung an.

konfigurationszeile
: Gibt die Zeilennummer in der Governor-Konfigurationsdatei an, in der sich die Regel befindet, die das Erzwingen der Anwendungsbeendigung verursacht.

einschränkung_überschritten
: Gibt Details zur Überschreitung der Regel an. Gültige Werte:

- CPU: Die gesamte USR CPU- plus SYS CPU-Zeit für die Anwendung in Sekunden.
- Locks: Die Gesamtzahl der Sperren, die von der Anwendung gehalten werden.
- Rowssel: Die Gesamtzahl der Zeilen, die von der Anwendung ausgewählt wurden.
- Rowsread: Die Gesamtzahl der Zeilen, die von der Anwendung gelesen wurden.
- Idle: Der Zeitraum, während dessen die Anwendung inaktiv war.
- ET: Die Zeit, die seit dem Start der aktuellen UOW der Anwendung vergangen ist (`uowtime setlimit` wurde überschritten).

NICE Der Datensatz NICE wird geschrieben, wenn der Governor die Priorität einer Anwendung ändert, wie in den Regeln der Konfigurationsdatei des Governors festgelegt. Der Datensatz NICE weist folgendes Format auf:

```
<anwendungsname> <authentifizierungs-id> <anwendungs-id>
<nice-wert> <konfigurationszeile> <einschränkung_überschritten>
```

Dabei gilt:

nice-wert
: Gibt das Inkrement oder Dekrement an, das auf den Prioritätswert des Agentenprozesses der Anwendung angewendet wird.

konfigurationszeile
: Gibt die Zeilennummer in der Konfigurationsdatei des Governors an, in der sich die Regel befindet, die das Ändern der Anwendungspriorität bewirkt.

einschränkung_überschritten
: Gibt Details zur Überschreitung der Regel an. Gültige Werte:

- CPU: Die gesamte USR CPU- plus SYS CPU-Zeit für die Anwendung in Sekunden.
- Locks: Die Gesamtzahl der Sperren, die von der Anwendung gehalten werden.
- Rowssel: Die Gesamtzahl der Zeilen, die von der Anwendung ausgewählt wurden.
- Rowsread: Die Gesamtzahl der Zeilen, die von der Anwendung gelesen wurden.

- Idle: Der Zeitraum, während dessen die Anwendung inaktiv war.
- ET: Die Zeit, die seit dem Start der aktuellen UOW der Anwendung vergangen ist (`uowtime setlimit` wurde überschritten).

SCHEDGRP

Der Datensatz SCHEDGRP wird geschrieben, wenn eine Anwendung zur einer Zeitplanungsgruppe hinzugefügt oder von einer Zeitplanungsgruppe in eine andere versetzt wird. 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 die Terminierung der Anwendung bewirkt.

einschränkung_überschritten
Gibt Details zur Überschreitung der Regel an. Gültige Werte:

- CPU: Die gesamte USR CPU- plus SYS CPU-Zeit für die Anwendung in Sekunden.
- Locks: Die Gesamtzahl der Sperren, die von der Anwendung gehalten werden.
- Rowssel: Die Gesamtzahl der Zeilen, die von der Anwendung ausgewählt wurden.
- Rowsread: Die Gesamtzahl der Zeilen, die von der Anwendung gelesen wurden.
- Idle: Der Zeitraum, während dessen die Anwendung inaktiv war.
- ET: Die Zeit, die seit dem Start der aktuellen UOW der Anwendung vergangen ist (`uowtime setlimit` wurde überschritten).

START

Der Datensatz START wird geschrieben, wenn der Governor gestartet wird. Der Datensatz START weist folgendes Format auf:

```
Database = <datenbankname>
```

STOP Der Datensatz STOP wird geschrieben, wenn der Governor gestoppt wird. Er hat folgendes Format:

```
Database = <datenbankname>
```

WARNING

Der Datensatz WARNING wird in den folgenden Situationen geschrieben:

- Die API sqlefrce wurde aufgerufen, um die Beendigung einer Anwendung zu erzwingen, es wurde jedoch ein positiver SQLCODE-Wert zurückgegeben.
- Ein Momentaufnahmenaufruf gab einen positiven SQLCODE-Wert zurück, bei dem es sich nicht um 1611 (SQL1611W) handelte.
- Ein Momentaufnahmenaufruf gab einen negativen SQLCODE-Wert zurück, bei dem es sich nicht um -1224 (SQL1224N) oder -1032 (SQL1032N) handelte. Diese Rückkehrcodes treten auf, wenn eine zuvor aktive Instanz gestoppt wurde.

- In einer Linux- oder UNIX-Umgebung ist ein Versuch, eine Signalroutine zu installieren, fehlgeschlagen.

Da standardisierte Werte geschrieben werden, können Sie die Protokolldateien nach unterschiedlichen Aktionstypen abfragen. Das Feld *Nachricht* enthält weitere nicht standardisierte Informationen, die vom Satztyp abhängen. 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 Protokolldatei des Governors kann in etwa wie das folgende Beispiel aussehen:

```
2007-12-11-14.54.52    0 START      Database = TQTEST
2007-12-11-14.54.52    0 READCFG    Config = /u/db2instance/sqllib/tqtest.cfg
2007-12-11-14.54.53    0 ERROR      SQLMON Error: SQLCode = -1032
2007-12-11-14.54.54    0 ERROR      SQLMONSZ Error: SQLCode = -1032
```

Stoppen des Governors

Das Dienstprogramm 'Governor' überwacht Anwendungen, die mit einer Datenbank verbunden sind, und ändert das Verhalten dieser Anwendungen gemäß den Regeln, die Sie in einer Konfigurationsdatei des Governors für diese Datenbank angeben.

Informationen zu diesem Vorgang

Wichtig: Mit der Einführung der neuen Workload-Management-Features in DB2 Version 9.5 gilt das DB2-Dienstprogramm Governor in Version 9.7 als veraltet und wird möglicherweise in einen zukünftigen Release entfernt. Weitere Informationen hierzu finden Sie im Abschnitt „ DB2 Governor und Query Patroller gelten als veraltet" in der Veröffentlichung *Neuerungen in DB2 Version 9.7*.

Zum Stoppen des Governors benötigen Sie die Berechtigung *sysadm* oder *sysctrl*.

Zum Stoppen des Governors verwenden Sie den Befehl **db2gov**, indem Sie die Option STOP angeben.

Beispiel

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

```
db2gov STOP sales
```

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

```
db2gov START sales nodenum 3
```

Kapitel 3. Faktoren mit Auswirkung auf die Leistung

Systemarchitektur

Übersicht über die Architektur und Prozesse von DB2

Auf der Clientseite werden lokale oder ferne Anwendungen mit der Clientbibliothek von DB2 verbunden. Lokale Clients kommunizieren über gemeinsamen Speicher und Semaphore, ferne Clients verwenden ein Protokoll wie benannte Pipes (Named Pipes, NPIPE) oder TCP/IP. Auf der Serverseite werden die Aktivitäten durch zuteilbare Einheiten der Steuerkomponente, so genannte EDUs (Engine Dispatchable Units), gesteuert.

Abb. 3 zeigt eine allgemeine Übersicht über die Architektur und die Prozesse von DB2.

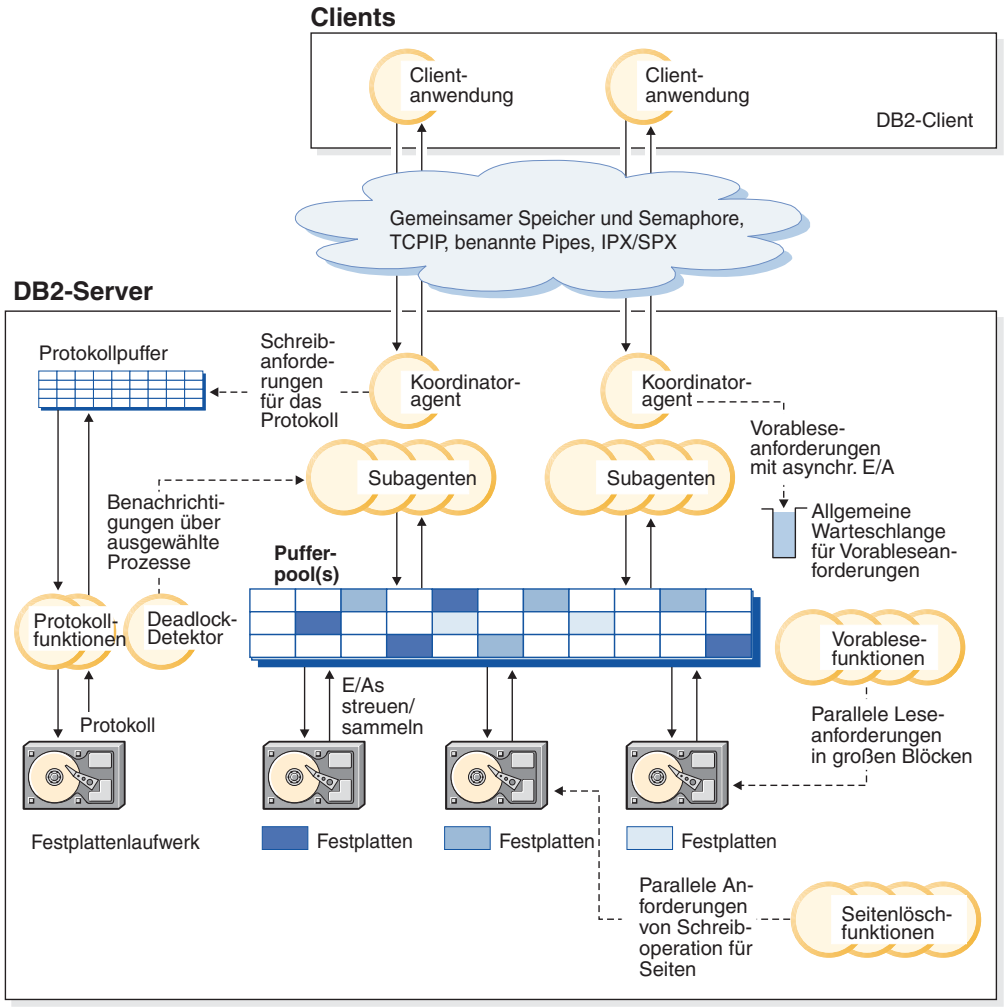


Abbildung 3. Clientverbindungen und Datenbankserverkomponenten

EDUs werden als Kreise oder Gruppen von Kreisen dargestellt.

EDUs werden auf allen Plattformen als Threads implementiert. DB2-Agenten sind der gängigste Typ von EDU. Diese Agenten führen den größten Teil der SQL- und XQuery-Verarbeitung für Anwendungen aus. Vorablesefunktionen und Seitenlöschfunktionen sind weitere häufig genutzte Typen 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 mithilfe eines Poolfunktionsalgorithmus verwaltet, der die Häufigkeit der Erstellung und Vernichtung von EDUs so gering wie möglich hält.

Pufferpools sind Bereiche des Datenbankserverspeichers, in die Seiten mit Benutzerdaten, 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.

Die Konfiguration von Pufferpools sowie der EDUs für Vorablesefunktionen und Seitenlöschfunktionen steuert, wie schnell auf Daten durch Anwendungen zugegriffen werden kann.

- *Vorablesefunktionen* rufen Daten von der Festplatte ab und lesen sie in einen Pufferpool ein, bevor Anwendungen die Daten benötigen. Beispielsweise müssten Anwendungen, die große Volumina von Daten durchsuchen müssen, darauf warten, dass Daten vom Plattenspeicher in einen 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 von Daten haben, können die Daten über die Plattendatenträger einheitenübergreifend gespeichert werden (Striping). Das einheitenübergreifendes Speichern von Daten gibt den Vorablesefunktionen die Möglichkeit, mehrere Platten zu verwenden, um Daten gleichzeitig abzurufen.
- *Seitenlöschfunktionen* versetzen Daten aus einem Pufferpool zurück auf den Plattenspeicher. Seitenlöschfunktionen sind im Hintergrund aktive EDUs, die unabhängig von den Anwendungsagenten arbeiten. Sie suchen nach Seiten, die geändert wurden, und schreiben diese geänderten Seiten auf die Platte. 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 einem Pufferpool und dem Plattenspeicher selbst durchführen.

DB2-Prozessmodell

Kenntnisse über das DB2-Prozessmodell erleichtern Ihnen das Verständnis, wie der Datenbankmanager und die zugeordneten Komponenten interagieren. Dieses Wissen kann Ihnen bei der Untersuchung von Problemen, die möglicherweise auftreten, von Nutzen sein.

Das Prozessmodell, das von allen DB2-Datenbankservern verwendet wird, ermöglicht die Kommunikation zwischen Datenbankservern und Clients. Es stellt zudem sicher, dass Datenbankanwendungen von Ressourcen wie Datenbanksteuerblöcken und kritischen Datenbankdateien isoliert werden.

Der DB2-Datenbankserver muss viele verschiedene Tasks ausführen. So verarbeitet er zum Beispiel Anforderungen von Datenbankanwendungen oder stellt sicher, dass Protokollsätze auf die Platte geschrieben werden. Jede Task wird in der Regel durch eine separate *Engine-Dispatchable-Unit* (EDU, von der Steuerkomponente zuteilbare Einheit) ausgeführt.

Die Verwendung einer Multithread-Architektur für den DB2-Datenbankserver bietet zahlreiche Vorteile. Ein neuer Thread erfordert weniger Speicher- und Betriebssystemressourcen als ein Prozess, da einige Betriebssystemressourcen von allen Threads innerhalb desselben Prozesses gemeinsam genutzt werden können. Auf einigen Plattformen ist darüber hinaus der Zeitaufwand von Kontextwechseln für Threads geringer als für Prozesse, was zur einer Leistungsverbesserung führen kann. Die Verwendung eines Multithread-Modells auf allen Plattformen vereinfacht die Konfiguration des DB2-Datenbankservers, weil es leichter ist, bei Bedarf mehr EDUs zuzuordnen, und es zudem möglich ist, Speicher dynamisch zuzuordnen, der von mehreren EDUs gemeinsam genutzt werden muss.

Für jede Datenbank, auf die zugegriffen wird, werden separate EDUs gestartet, um verschiedene Datenbanktasks, wie den Vorablesezugriff, die Kommunikation und die Protokollierung, auszuführen. Datenbankagenten sind eine spezielle Klasse von EDU, die zur Verarbeitung von Anwendungsanforderungen für eine Datenbank erstellt werden.

Im Allgemeinen garantiert der DB2-Datenbankserver eine zuverlässige Verwaltung der EDUs. Es sind jedoch DB2-Tools verfügbar, die für die EDUs konzipiert sind. Sie können z. B. den Befehl **db2pd** mit der Option **-edus** verwenden, um alle aktiven EDU-Threads aufzulisten.

Jede Clientanwendungsverbindung hat genau einen Koordinatoragenten, der in einer Datenbank arbeitet. Ein *Koordinatoragent* arbeitet im Auftrag einer Anwendung und kommuniziert mit anderen Agenten, indem er je nach Bedarf privaten Speicher, Interprozesskommunikation (IPC) oder Protokolle zur Fernkommunikation nutzt.

Die DB2-Architektur stellt eine Firewall bereit, die dafür sorgt, dass Anwendungen in einem anderen Adressraum als der DB2-Datenbankserver ausgeführt werden (siehe Abb. 4 auf Seite 38). Die Firewall schützt die Datenbank und den Datenbankmanager vor Anwendungen, gespeicherten Prozeduren und benutzerdefinierten Funktionen (UDFs). Die Firewall wahrt die Integrität der Daten in den Datenbanken, weil sie verhindert, dass interne Puffer oder Datenbankmanagerdateien durch Programmierfehler in Anwendungen überschrieben werden. Die Firewall erhöht außerdem die Zuverlässigkeit, weil Anwendungsfehler den Datenbankmanager nicht zum Absturz bringen können.

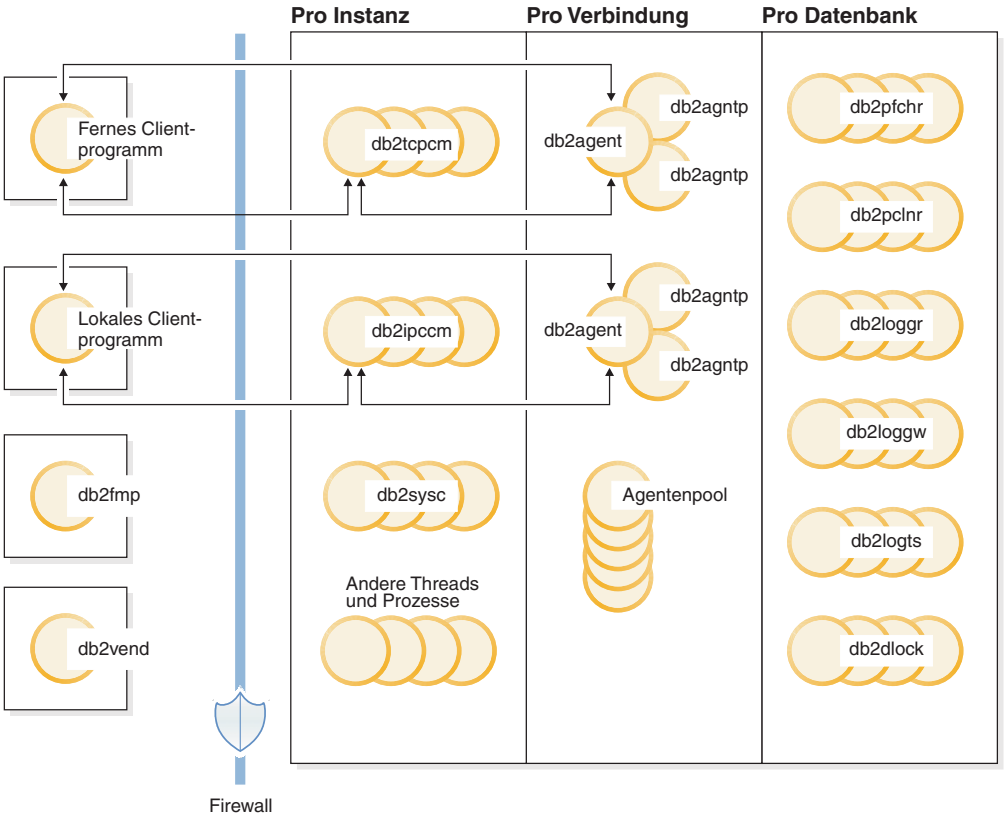


Abbildung 4. Prozessmodell für DB2-Datenbanksysteme

Clientprogramme

Clientprogramme können fern oder lokal, d. h. auf dem gleichen System wie der Datenbankserver, ausgeführt werden. Clientprogramme stellen den ersten Kontakt zu einer Datenbank über einen kommunikationsbereiten Agenten (Kommunikationslistenerfunktion) her.

Listenerfunktionen

Die kommunikationsbereiten Agenten (bzw. Listenerfunktionen) werden gestartet, wenn der DB2-Datenbankserver gestartet wird. Für jedes Kommunikationsprotokoll ist eine Listenerfunktion konfiguriert. Eine Listenerfunktion für Interprozesskommunikation (**db2ipccm**) ist für lokale Clientprogramme konfiguriert. Folgende Listenerfunktionen sind verfügbar:

- **db2ipccm** für lokale Clientverbindungen
- **db2tcpcm** für TCP/IP-Verbindungen
- **db2tcpdm** für TCP/IP-Anforderungen des Tools Discovery

Agenten

Allen Verbindungsanforderungen von lokalen oder fernen Clientprogrammen (Anwendungen) wird ein entsprechender Koordinatoragent (**db2agent**) zugeordnet. Wenn der Koordinatoragent erstellt ist, führt er alle Datenbankanforderungen im Auftrag der Anwendung aus.

In Umgebungen mit partitionierten Datenbanken oder auf Systemen mit aktivierter *abfrageinterner Parallelität* verteilt der Koordinatoragent die Datenbankanforderungen an Subagenten (**db2agntp** bzw. **db2agnts**). Subagenten, die zwar einer Anwendung zugeordnet, jedoch momentan inaktiv sind, haben den Namen **db2agnta**.

Ein Koordinatoragent kann folgende Verbindungen haben:

- Eine Verbindung (durch CONNECT) zur Datenbank mit einem Aliasnamen. Beispiel: **db2agent (DATA1)** hat eine Verbindung zum Datenbankaliasnamen DATA1.
- Eine Verbindung (durch ATTACH) zu einer Instanz. Beispiel: **db2agent (user1)** hat eine Verbindung zur Instanz 'user1'.

Der DB2-Datenbankserver instanziiert weitere Typen von Agenten, wie zum Beispiel unabhängige Koordinatoragenten oder Subkoordinatoragenten, um bestimmte Operationen auszuführen. Zum Beispiel dienen der unabhängige Koordinatoragent **db2agnti** zur Ausführung von Ereignismonitoren und der Subkoordinatoragent **db2agnsc** zur Parallelisierung von Datenbankneustartoperationen nach einer abnormalen Beendigung.

Ein Gateway-Agent (**db2agentg**) ist ein Agent, der einer fernen Datenbank zugeordnet ist. Er stellt indirekte Konnektivität bereit, die Clients den Zugriff auf die Hostdatenbank ermöglicht.

Inaktive 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. Durch einen Pool inaktiver Agenten in geeigneter Größe kann sich die Leistung bei hoher Anwendungsauslastung verbessern. In diesem Fall können inaktive Agenten verwendet werden, sobald sie erforderlich sind, ohne dass ein völlig neuer Agent für jede Anwendungsverbindung zugeordnet werden muss. Eine solche Zuordnung erfordert die Erstellung eines Threads sowie die Zuordnung und Initialisierung von Speicher- und anderen Ressourcen. Der DB2-Datenbankserver garantiert eine automatische Verwaltung der Größe des Pools inaktiver Agenten.

Ein zu einem Pool gehörender Agent kann einer fernen oder einer lokalen Datenbank zugeordnet werden. Ein Agent, der einem Pool angehört und einer fernen Datenbank zugeordnet ist, wird als Pool-Gateway-Agent bezeichnet (**db2agntgp**). Ein Agent, der einem Pool angehört und einer lokalen Datenbank zugeordnet ist, wird als Pooldatenbankagent bezeichnet (**db2agentdp**).

db2fmp

Der Prozess für abgeschirmten Modus (fmp - fenced-mode process) ist zuständig für die Ausführung von abgeschirmten gespeicherten Prozeduren und benutzerdefinierten Funktionen außerhalb der Firewall. Der Prozess **db2fmp** ist immer ein separater Prozess, kann jedoch je nach Art der ausgeführten Routinen mehrere Threads (Multithreading) enthalten.

db2vend

Dies ist ein Prozess zur Ausführung von Code anderer Anbieter im Auftrag einer EDU, zum Beispiel zur Ausführung des Benutzerexitprogramms für die Protokollarchivierung (nur UNIX).

Datenbank-EDUs

In der folgenden Liste sind einige der wichtigen EDUs aufgeführt, die von jeder Datenbank verwendet werden:

- **db2dlock** zur Erkennung von Deadlocks. In einer Umgebung mit partitionierten Datenbanken wird ein zusätzlicher Thread (**db2glock**) zur Koordinierung der Informationen verwendet, die von den **db2dlock**-EDUs in den einzelnen Partitionen erfasst werden. Der Thread **db2glock** wird nur in der Katalogpartition ausgeführt.
- **db2fw**, das Fast-Write-Programm des Ereignismonitors, das für das parallele Schreiben von großen Mengen an Ereignismonitordaten in Tabellen, Dateien oder Pipes verwendet wird.
- **db2hadrp** als Thread für den primären High Availability Disaster Recovery-Server (HADR-Server).
- **db2hadrs** ist ein Thread für den HADR-Bereitschaftsserver.
- **db2lfr** für Protokolldatei-Leseeinheiten, die einzelne Protokolldateien verarbeiten.
- **db2loggr** zur Behandlung von Protokolldateien für die Durchführung der Transaktionsverarbeitung und -recovery.
- **db2loggw** zum Schreiben von Protokollsätzen in Protokolldateien.
- **db2logmgr** für den Protokollmanager. Verwaltet Protokolldateien für eine wiederherstellbare Datenbank.
- **db2logts** zur Verfolgung. Es wird erfasst, welche Tabellenbereiche Protokollsätze in welchen Protokolldateien haben. Diese Informationen werden in der Datei `DB2TSCHG.HIS` im Datenbankverzeichnis aufgezeichnet.
- **db2lused** zur Aktualisierung der Objektnutzung.
- **db2pfchr** für Vorablesefunktionen für Pufferpools.
- **db2pclnr** für Seitenlöschfunktionen für Pufferpools.
- **db2redom** für den Master für aktualisierende Wiederherstellung. Bei der Recovery verarbeitet diese EDU Protokollsätze für die aktualisierende Wiederherstellung und ordnet Protokollsätzen Worker-Threads für die aktualisierende Wiederherstellung zu.
- **db2redow** für die Worker für aktualisierende Wiederherstellung. Bei der Recovery verarbeitet diese EDU Protokollsätze für die aktualisierende Wiederherstellung auf Anforderung des Masters für aktualisierende Wiederherstellung.
- **db2shred** zur Verarbeitung einzelner Protokollsätze auf Protokollseiten.
- **db2stmm** für die STMM-Funktion (automatische Speicherverwaltungsoptimierung).
- **db2taskd** zur Verteilung von im Hintergrund ausgeführten Datenbanktasks. Diese Tasks werden durch Threads mit dem Namen **db2taskp** ausgeführt.
- **db2wlmd** für die automatische Erfassung von Auslastungsmanagementstatistiken.
- Ereignismonitorthreads werden wie folgt angegeben:
  - db2evm*%1%2* (*%3*)

    Dabei kann *%1* folgende Werte haben:
    - `g` - globaler Dateiereignismonitor
    - `gp` - globaler, über eine Pipe geleiteter Ereignismonitor
    - `l` - lokaler Dateiereignismonitor
    - `lp` - lokaler, über eine Pipe geleiteter Ereignismonitor
    - `t` - Tabellenereignismonitor

Und *%2* kann folgende Werte haben:

- `i` - Koordinator
- `p` - kein Koordinator

und *%3* ist der Name des Ereignismonitors.

- Backup- und Restore-Threads 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* ist die EDU-ID des Agenten, der die Backup- bzw. Restoresitzung steuert.
    - *%2* ist ein sequenzieller Wert, der zur eindeutigen Unterscheidung der (möglicherweise vielen) Threads verwendet wird, die zu einer bestimmten Backup- bzw. Restoresitzung gehören.

    Beispiel: **db2bm.13579.2** gibt den zweiten Thread **db2bm** an, der durch den Thread **db2agent** mit der EDU-ID 13579 gesteuert wird.

Datenbankserverthreads und -prozesse

Der Systemcontroller (**db2sysc** unter UNIX und **db2syscs.exe** unter Windows-Betriebssystemen) muss vorhanden sein, wenn der Datenbankserver funktionieren soll. Die folgenden Threads und Prozesse führen eine ganze Reihe von Tasks aus:

- **db2acd** ist der Autonomic Computing-Dämon, der als Host für den Diagnosemonitor, für die Dienstprogramme zur automatischen Pflege und den Scheduler für Verwaltungstasks fungiert. Dieser Prozess war zuvor unter der Bezeichnung **db2hmon** bekannt.
- **db2aiothr** verwaltet asynchrone E/A-Anforderungen für eine Datenbankpartition (nur UNIX).
- **db2alarm** benachrichtigt EDUs, wenn ihr angeforderter Zeitgeber abgelaufen ist (nur UNIX).
- **db2cart** dient zur Archivierung von Protokolldateien (wenn der Datenbankkonfigurationsparameter **userexit** aktiviert ist).
- **db2disp** ist der Dispatcher für den Clientverbindungskonzentrator.
- **db2fcms** ist der FCM-Senderdämon.
- **db2fcmr** ist der FCM-Empfängerdämon.
- **db2fmd** ist der Fehlermonitordämon.
- **db2fmtlg** dient zur Formatierung von Protokolldateien (wenn der Datenbankkonfigurationsparameter **logretain** aktiviert und der Datenbankkonfigurationsparameter **userexit** inaktiviert ist).
- **db2licc** verwaltet die installierten DB2-Lizenzen.
- **db2panic** ist der Notfallagent, der dringende Anforderungen verarbeitet, nachdem Agentengrenzwerte auf einem bestimmten Knoten erreicht wurden.
- **db2pdbc** ist der parallele Systemcontroller, der Parallelanforderungen von fernen Datenbankpartitionen verarbeitet (nur in einer Umgebung mit partitionierten Datenbanken).
- **db2resync** ist der Resynchronisationsagent, der die globale Resynchronisationsliste (resync) durchsucht.
- **db2sysc** ist die Hauptsystemcontroller-EDU, die kritische Ereignisse des DB2-Servers verarbeitet.
- **db2thcln** macht Ressourcen wieder verfügbar, wenn eine EDU beendet wird (nur UNIX).

- **db2wdog** ist der Wachprozess unter UNIX- und Linux-Betriebssystemen, der abnormale Beendigungen behandelt.

Datenbankagenten

Wenn eine Anwendung auf eine Datenbank zugreift, beginnen mehrere Prozesse oder Threads mit der Ausführung der verschiedenen Anwendungstasks. Zu diesen Tasks gehören das Protokollieren, die Kommunikation und der Vorablesezugriff. Datenbankagenten sind Threads innerhalb des Datenbankmanagers, die zur Ausführung von Anwendungsanforderungen verwendet werden. In Version 9.5 werden Agenten auf allen Plattformen 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. *Koordinatoragenten* sind am längsten einer Anwendung zugeordnet, da sie ihr zugeordnet bleiben, bis die Anwendung die Verbindung trennt. Die einzige Ausnahme von dieser Regel liegt vor, wenn der Konzentrator der Steuerkomponente aktiviert ist. In diesem Fall kann ein Koordinatoragent diese Zuordnung an Transaktionsgrenzen (COMMIT- oder ROLLBACK-Operationen) trennen.

Es gibt drei Typen von Verarbeitungsagenten:

- Inaktive (nicht zugeordnete) Agenten

  Dies ist die einfachste Form eines Verarbeitungsagenten. Er hat keine abgehende Verbindung und keine lokale Datenbankverbindung oder Instanzverbindung.

- Aktive Koordinatoragenten

  Jede Datenbankverbindung einer Clientanwendung hat genau einen aktiven Agenten, der die Arbeit der Anwendung in der Datenbank koordiniert. Wenn der Koordinatoragent erstellt ist, führt er alle Datenbankanforderungen für die zugehörige Anwendung aus und kommuniziert mit anderen Agenten über die Interprozesskommunikation (IPC) oder über Fernübertragungsprotokolle. 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 Koordinatoragenten Folgendes:
  - Er wird ein aktiver Koordinatoragent, wenn weitere Verbindungen warten.
  - Er wird freigegeben und als inaktiv (nicht zugeordnet) markiert, wenn keine Verbindungen warten und die maximale Zahl von Poolagenten automatisch verwaltet wird oder 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

  Der Koordinatoragent verteilt 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. In DB2 Version 9.5 können Subagenten auch in nicht partitionierten Umgebungen sowie in Umgebungen, in denen die abfrageinterne Parallelität nicht aktiviert ist, vorhanden sein.

Agenten, die keine Arbeit für eine Anwendung ausführen und darauf warten, zugeordnet zu werden, gelten als inaktive 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 des Konfigurationsparameters **num_poolagents** des Datenbankmanagers abhängig.

Falls kein inaktiver 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 inaktiver Agent im Bereitschaftsmodus für einen Client aktiviert werden kann.

Wenn ein Subagent für eine Anwendung aktiv ist, ist 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 auf zugeordnete inaktive Agenten, bevor er einen neuen Agenten erstellt.

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 vom Datenbankserver verarbeitet werden können. Es ist jedoch möglich, dass in Ihrer Umgebung eine Viele-zu-eins-Beziehung zwischen der Zahl der verbundenen Anwendungen und der Zahl der Anwendungsanforderungen, die verarbeitet werden können, erforderlich ist.

Diese Faktoren werden von zwei Konfigurationsparametern des Datenbankmanagers separat gesteuert:

- Der Parameter **max_connections** gibt die maximale Anzahl verbundener Anwendungen an.
- Der Parameter **max_coordagents** gibt die maximale Anzahl von Anwendungsanforderungen an, die gleichzeitig 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 Datenbankressourcen erfordert, steigt mit wachsender Zahl dieser Agenten auch die Wahrscheinlichkeit, dass die Obergrenze an verfügbaren globalen Ressourcen erreicht wird. Um dies zu vermeiden, setzen Sie den Parameter **max_connections** auf einen höheren Wert als den Parameter **max_coordagents** oder Sie setzen beide Parameter auf den Wert AUTOMATIC.

In zwei speziellen Szenarios ist eine Einstellung dieser Parameter auf den Wert AUTOMATIC besonders zu empfehlen:

- Wenn Sie sicher sind, dass Ihr System alle Verbindungen verarbeiten kann, die möglicherweise benötigt werden, jedoch der Umfang der genutzten globalen Ressourcen begrenzt werden soll (durch Begrenzen der Anzahl von Koordinatoragenten), setzen Sie den Parameter **max_connections** auf AUTOMATIC. Wenn der Wert des Parameters **max_connections** größer ist als der Wert des Parameters **max_coordagents**, ist der Verbindungskonzentrator aktiviert.
- Wenn Sie die maximale Anzahl von Verbindungen oder koordinierenden Agenten nicht begrenzen wollen, jedoch wissen, dass Ihr System eine Viele-zu-eins-Beziehung zwischen der Anzahl der verbundenen Anwendungen und der An-

zahl der Anwendungsanforderungen, die verarbeitet werden, erfordert oder von einer solchen Beziehung profitieren würde, setzen Sie beide Parameter auf AUTOMATIC. Wenn beide Parameter auf AUTOMATIC gesetzt sind, verwendet der Datenbankmanager die Werte, die Sie angeben, als ideales Verhältnis von Verbindungen zu koordinierenden Agenten. Beachten Sie, dass beide Parameter mit einem Anfangswert und der Einstellung AUTOMATIC konfiguriert werden können. Mit dem folgenden Befehl wird dem Parameter **max_coordagents** zum Beispiel sowohl der Wert 200 als auch die Einstellung AUTOMATIC zugeordnet: `update dbm config using max_coordagents 200 automatic`.

Beispiel

Betrachten Sie folgendes Szenario:

- Der Parameter **max_connections** ist auf AUTOMATIC gesetzt und hat momentan den Wert 300.
- Der Parameter **max_coordagents** ist auf AUTOMATIC gesetzt und hat momentan den Wert 100.

Das Verhältnis von **max_connections** zu **max_coordagents** ist 300:100. Der Datenbankmanager erstellt neue koordinierende Agenten, wenn neue Verbindungen eintreffen und die Verbindungskonzentratorfunktion wird nur bei Bedarf angewendet. Diese Einstellungen führen zu folgenden Aktionen:

- Für die Verbindungen 1 bis 100 werden neue koordinierende Agenten erstellt.
- Für die Verbindungen 101 bis 300 werden keine neuen koordinierenden Agenten erstellt, sondern sie nutzten die 100 bereits erstellten Agenten gemeinsam.
- Für die Verbindungen 301 bis 400 werden neue koordinierende Agenten erstellt.
- Für die Verbindungen 401 bis 600 werden keine neuen koordinierenden Agenten erstellt, sondern sie nutzten die 200 bereits erstellten Agenten gemeinsam.
- Und so weiter...

In diesem Beispiel wird angenommen, dass die verbundenen Anwendungen genügend Arbeit verursachen, um die Erstellung neuer koordinierender Agenten in jedem Schritt zu rechtfertigen. Wenn die verbundenen Anwendungen nach einer gewissen Zeit nicht mehr genügend Arbeit anfordern, werden die koordinierenden Agenten inaktiv und möglicherweise beendet.

Wenn sich die Anzahl der Verbindungen reduziert, der Umfang der Arbeit, der von den verbleibenden Verbindungen verursacht wird, jedoch hoch ist, wird die Anzahl der koordinierenden Agenten möglicherweise nicht sofort verringert. Die Parameter **max_connections** und **max_coordagents** wirken sich nicht direkt auf den Zusammenschluss (Pooling) von Agenten oder die Beendigung von Agenten aus. Die normalen Regeln für die Beendigung von Agenten sind weiterhin gültig. Das heißt, dass das Verhältnis von Verbindungen zu koordinierenden Agenten möglicherweise nicht exakt den von Ihnen angegebenen Werten entspricht. Agenten werden möglicherweise zur Wiederverwendung an den Agentenpool zurückgegeben, bevor sie beendet werden.

Wenn ein feinerer Grad an Steuerung erforderlich ist, geben Sie ein einfacheres Verhältnis an. Zum Beispiel kann das Verhältnis von 300:100 aus dem vorherigen Beispiel als 3:1 ausgedrückt werden. Wenn der Parameter **max_connections** auf 3 (AUTOMATIC) und der Parameter **max_coordagents** auf 1 (AUTOMATIC) gesetzt wird, kann für je drei Verbindungen ein koordinierender Agent erstellt werden.

Client-/Serververarbeitungsmodell

Sowohl lokale als auch ferne Anwendungsprozesse können mit derselben Datenbank arbeiten. Eine ferne Anwendung ist eine Anwendung, die eine Datenbankaktion von einer Maschine aus initiiert, die über Fernzugriff mit der Maschine verbunden ist, auf der sich der Datenbankserver befindet. Lokale Anwendungen sind auf der Servermaschine direkt mit der Datenbank verbunden.

Wie Clientverbindungen verwaltet werden, hängt davon ab, ob der Verbindungskonzentrator aktiviert ist oder nicht. Der Verbindungskonzentrator ist aktiviert, wenn der Wert des Konfigurationsparameters **max_connections** des Datenbankmanagers größer als der Wert des Konfigurationsparameters **max_coordagents** 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 Verarbeitungsagenten zur Erledigung dieser Arbeit koordinieren. Für Internetanwendungen mit vielen relativ kurzfristigen Verbindungen oder Anwendungen mit zahlreichen relativ kleinen Transaktionen verbessert der Verbindungskonzentrator die Leistung, indem er wesentlich mehr Clientanwendungen eine gleichzeitige Herstellung von Verbindungen ermöglicht. Darüber hinaus verringert er die Belastung der Systemressourcen durch jede einzelne Verbindung.

In Abb. 5 auf Seite 46 stellt jeder Kreis im DB2-Server eine Engine-Dispatchable-Unit (EDU) dar, die mit Betriebssystemthreads implementiert ist.

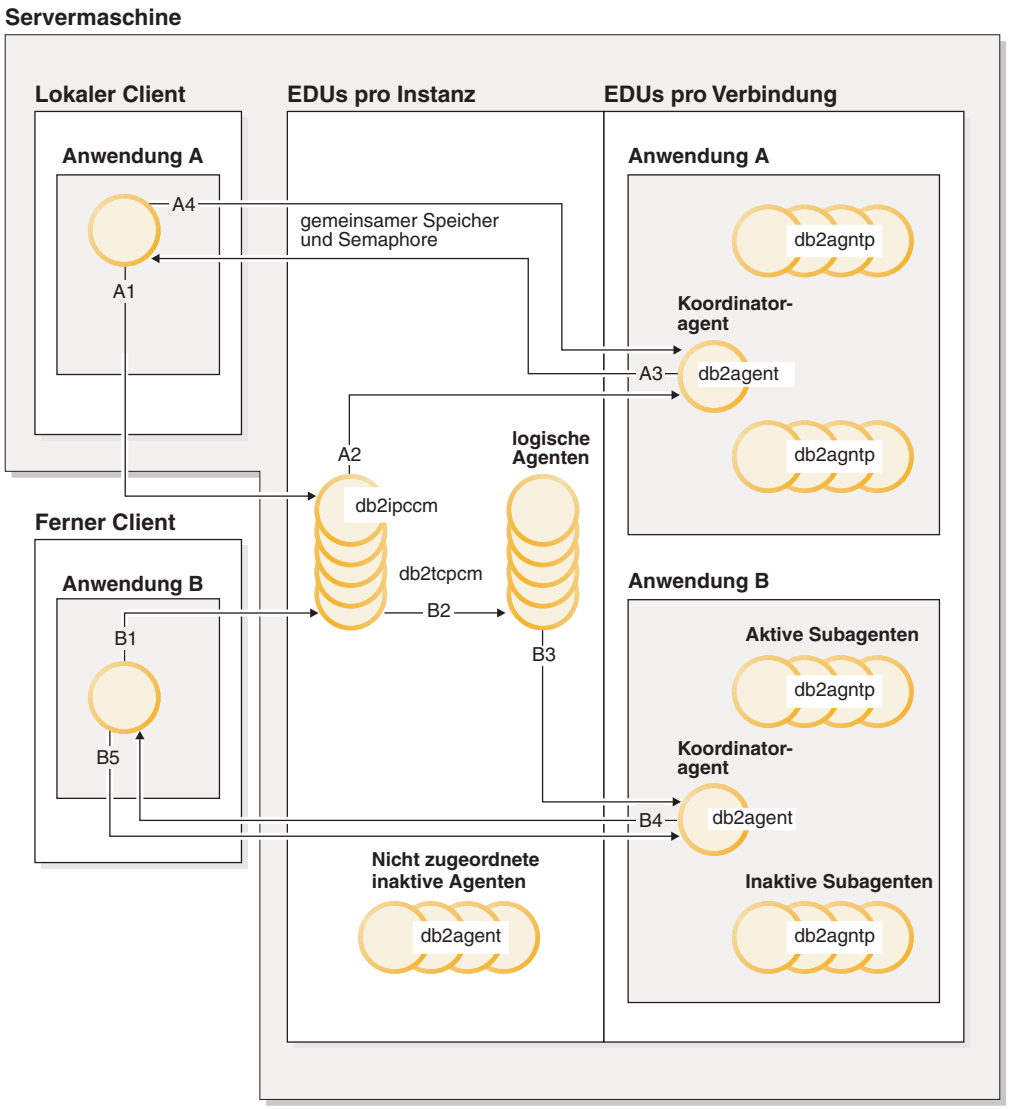


Abbildung 5. Übersicht über das Client-/Serververarbeitungsmodell

- Bei A1 stellt ein lokaler die Kommunikation durch eine db2ipccm-EDU her.
- Bei A2 arbeitet die db2ipccm-EDU mit einer db2agent-EDU, die zum Koordinatoragenten für Anwendungsanforderungen vom lokalen Client wird.
- Bei A3 nimmt der Koordinatoragent mit der Clientanwendung Kontakt auf, um zwischen der Clientanwendung und dem Koordinator eine Kommunikation über gemeinsam genutzten Speicher einzurichten.
- Bei A4 stellt die Anwendung auf dem lokalen Client eine Verbindung zur Datenbank her.
- Bei B1 stellt ein ferner Client die Kommunikation durch eine db2tcpcm-EDU her. Wenn ein anderes Kommunikationsprotokoll ausgewählt würde, würde der entsprechende Kommunikationsmanager verwendet.
- Bei B2 arbeitet die db2tcpcm-EDU mit einer db2agent-EDU, die zum Koordinatoragenten für die Anwendung wird, und übergibt die Verbindung an diesen Agenten.
- Bei B4 nimmt der Koordinatoragent Kontakt mit der fernen Clientanwendung auf.
- Bei B5 stellt die ferne Clientanwendung eine Verbindung zur Datenbank her.

Beachten Sie außerdem Folgendes:

- Verarbeitungsagenten führen Anwendungsanforderungen aus. Es gibt vier Typen von Verarbeitungsagenten: aktive Koordinatoragenten, aktive Subagenten, zugeordnete Subagenten und nicht zugeordnete, inaktive Agenten.
- Jede Clientverbindung wird mit einem aktiven Koordinatoragenten verknüpft.
- In einer Umgebung mit partitionierten Datenbanken oder in einer Umgebung mit aktivierter partitionsinterner Parallelität verteilen die Koordinatoragenten Datenbankanforderungen an Subagenten (db2agntp).
- Es gibt einen Agentenpool (db2agent), in dem inaktive Agenten im Bereitschaftsmodus auf neue Arbeit warten.
- Andere EDUs verwalten Clientverbindungen, Protokolle, zweiphasige Commitoperationen, Backup- und Restore-Operationen sowie weitere Tasks.

Abb. 6 zeigt weitere EDUs, 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).

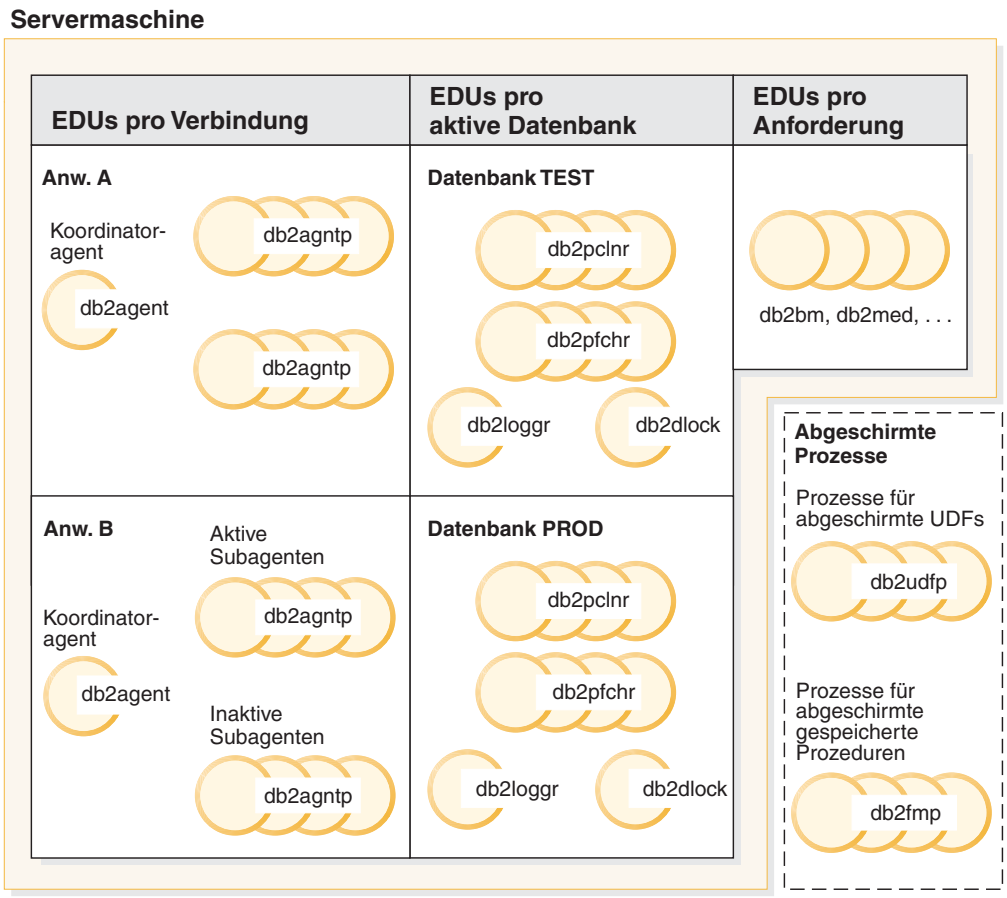


Abbildung 6. EDUs im Datenbankserver

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 Konfigurationsparameter **keepfenced** des Datenbankmanagers hat den Standardwert YES, durch den 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 gespeicherten Prozeduren jedoch über unbeschränkten Zugriff auf den Adressraum des Agenten verfügen, müssen sie vor ihrer Verwendung gründlich getestet werden.

Abb. 7 zeigt die Ähnlichkeiten und Unterschiede zwischen dem Prozessmodell mit einer einzigen Datenbankpartition und dem Prozessmodell mit mehreren Datenbankpartitionen auf.

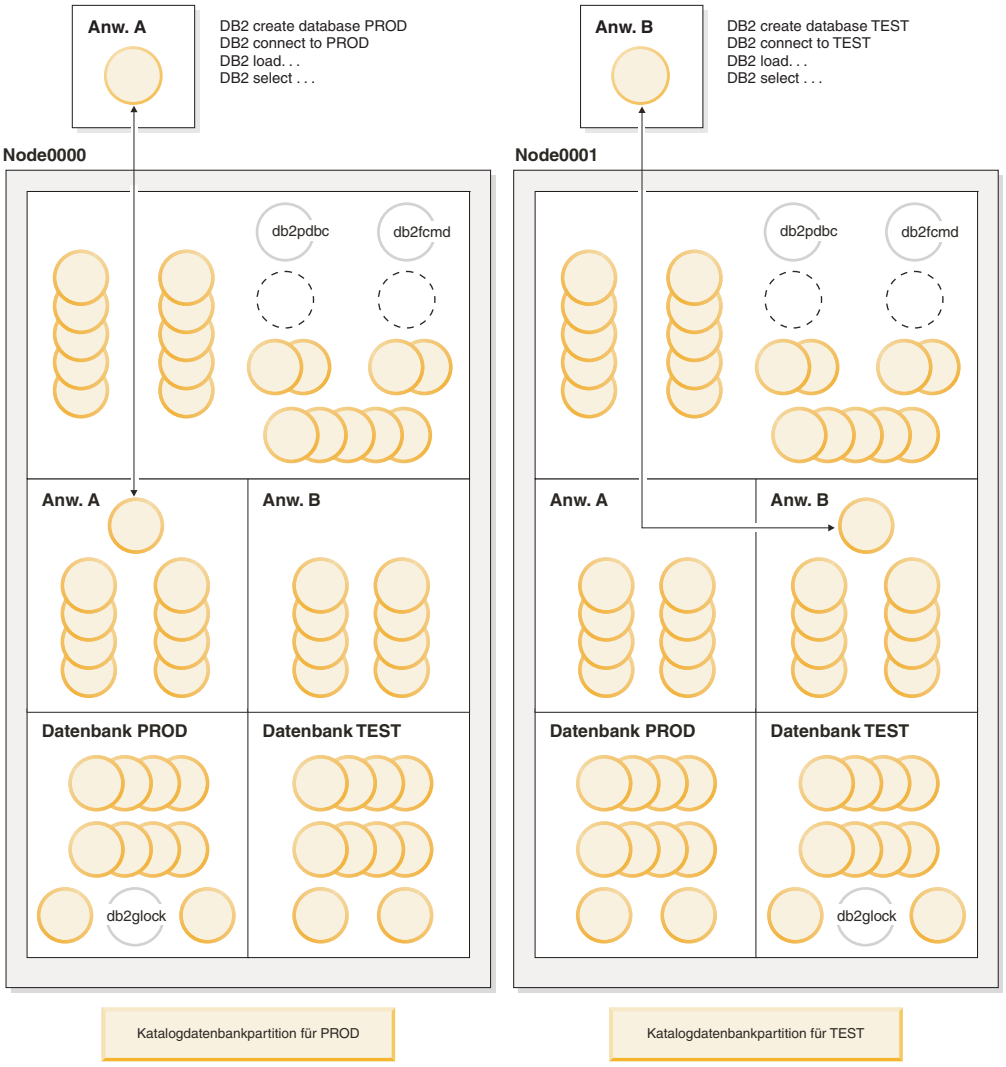


Abbildung 7. Prozessmodell für mehrere Datenbankpartitionen

In einer Umgebung mit mehreren Datenbankpartitionen wird die Datenbankpartition, in der der Befehl **CREATE DATABASE** ausgeführt wurde, als *Katalogdatenbankpartition* bezeichnet. In dieser Datenbankpartition werden die Systemkatalogtabellen gespeichert. Der Systemkatalog ist ein Repository für alle Informationen zu Objekten in der Datenbank.

Wie in Abb. 7 gezeigt, erstellt die Anwendung A die Datenbank PROD in der Datenbankpartition mit dem Namen Node0000, sodass auch der Katalog für die Datenbank PROD dieser Datenbankpartition erstellt wird. Analog wird der Katalog für die Datenbank TEST in der Datenbankpartition mit dem Namen Node0001 er-

stellt, weil Anwendung B die Datenbank TEST in dieser Datenbankpartition erstellt. Es ist empfehlenswert, die Datenbanken in verschiedenen Datenbankpartitionen zu erstellen, um die zusätzlichen mit den Katalogen für die Datenbanken verbundenen Aktivitäten gleichmäßig auf die Datenbankpartitionen in der Umgebung zu verteilen.

Der Instanz sind zusätzliche EDUs (db2pdbc und db2fcmd) zugeordnet, die sich in jeder Datenbankpartition einer Umgebung mit mehreren Datenbankpartitionen befinden. Diese EDUs werden für die Koordination von Anforderungen zwischen Datenbankpartitionen und zum Aktivieren des FCM (Fast Communications Manager) benötigt.

Der Katalogdatenbankpartition ist außerdem eine zusätzliche EDU (db2glock) zugeordnet. Diese EDU prüft auf globale Deadlocks unter den Datenbankpartitionen, in denen sich die aktive Datenbank befindet.

Jede Verbindungsanforderung von einer Anwendung wird durch eine Verbindung dargestellt, die einem Koordinatoragenten 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 *Koordinatordatenbankpartition* der Anwendung bezeichnet.

Teile der Datenbankanforderungen von einer Anwendung werden von der Koordinatordatenbankpartition an Subagenten in anderen Datenbankpartitionen gesendet. Alle Ergebnisse werden in der Koordinatordatenbankpartition zusammengeführt, bevor sie zurück zur Anwendung gesendet werden.

Eine beliebige Anzahl von Datenbankpartitionen kann zur Ausführung auf einer einzigen Maschine konfiguriert werden. Dies wird als Konfiguration mit *mehreren logischen Partitionen* 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 Semaphoren optimiert werden.

Verbesserungen des Verbindungskonzentrators für Clientverbindungen

Der Verbindungskonzentrator verbessert die Leistung von Anwendungen, die häufige, jedoch relativ kurzfristige Verbindungen herstellen, indem er eine effiziente Verarbeitung vieler gleichzeitiger Clientverbindungen ermöglicht. Darüber hinaus verringert er die Speicherbelegung bei jeder Verbindung und senkt die Anzahl von Kontextwechseln.

Der Verbindungskonzentrator ist aktiviert, wenn der Wert des Konfigurationsparameters **max_connections** des Datenbankmanagers größer als der Wert des Konfigurationsparameters **max_coordagents** 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 zuvor nur beim Verbindungspooling von DB2 Connect bekannt waren. Nach Herstellung der ersten Verbindung verringert der Verbindungskonzentrator die Zeit, die zur Herstellung einer Verbindung zu einem Host erforderlich ist. Wenn die Trennung einer Verbindung zu einem Host angefordert wird, wird die eingehende Verbindung gelöscht, während

die abgehende Verbindung zu diesem Host an einen Pool übergeben wird und erhalten bleibt. Wenn eine neue Verbindungsanforderung empfangen wird, versucht der Datenbankmanager eine vorhandene abgehende Verbindung aus dem Pool wiederzuverwenden.

Zur Erzielung der besten Leistung von Anwendungen, die mit einem Verbindungspooling oder mit dem Verbindungskonzentrator arbeiten, optimieren Sie die Parameter, die die Größe des Datenblocks steuern, der in einem Cache zwischengespeichert wird. Weitere Informationen finden Sie in der Produktdokumentation zu DB2 Connect.

Beispiele

- Betrachten Sie eine Einzelpartitionsdatenbank, mit der im Durchschnitt 1000 Benutzer gleichzeitig verbunden sind. Mitunter kann die Anzahl der verbundenen Benutzer auch höher liegen. Die Anzahl der gleichzeitig ausgeführten Transaktionen kann bis zu 200, jedoch nie über 250 betragen. Transaktionen sind kurz.

  Für diese Auslastung könnten Sie die folgenden Konfigurationsparameter des Datenbankmanagers definieren:

  – Setzen Sie den Parameter **max_coordagents** auf den Wert 250, um die maximale Anzahl gleichzeitiger Transaktionen zu unterstützen.
  – Setzen Sie den Parameter **max_connections** auf AUTOMATIC mit dem Wert 1000, um die Unterstützung für eine beliebige Anzahl von Verbindungen sicherzustellen. In diesem Beispiel stellt jeder Wert über 250 sicher, dass der Verbindungskonzentrator aktiviert ist.
  – Belassen Sie den Parameter **num_poolagents** auf dem Standardwert, wodurch sichergestellt werden sollte, dass Datenbankagenten zur Verarbeitung ankommender Clientanforderungen verfügbar sind und die Erstellung neuer Agenten nur geringen Aufwand verursacht.
- Betrachten Sie eine Einzelpartitionsdatenbank, mit der im Durchschnitt 1000 Benutzer gleichzeitig verbunden sind. Mitunter kann die Anzahl der verbundenen Benutzer 2000 erreichen. Im Durchschnitt wird erwartet, dass zu einem beliebigen Zeitpunkt 500 Benutzer gleichzeitig Arbeiten ausführen. Die Anzahl gleichzeitig ausgeführter Transaktionen liegt ungefähr bei 250. Fünfhundert koordinierende Agenten wären im Allgemeinen zu viele. Für 1000 verbundene Benutzer sollten 250 koordinierende Agenten ausreichend sein.

  Für diese Auslastung könnten Sie die Konfiguration des Datenbankmanagers wie folgt aktualisieren:

  ```
  update dbm cfg using max_connections 1000 automatic
  update dbm cfg using max_coordagents 250 automatic
  ```

  Dies bedeutet, dass, wenn die Anzahl von Verbindungen über 1000 ansteigt, zusätzliche koordinierende Agenten nach Bedarf erstellt werden, wobei die maximale Anzahl durch die Gesamtanzahl der Verbindungen festzulegen ist. Bei steigender Auslastung versucht der Datenbankmanager ein relativ stabiles Verhältnis von Verbindungen zu koordinierenden Agenten beizubehalten.
- Nehmen Sie an, dass Sie den Verbindungskonzentrator nicht aktivieren wollen, jedoch die Anzahl der verbundenen Benutzer begrenzt werden soll. Zur Begrenzung der Anzahl gleichzeitig verbundener Benutzer auf 250 könnten Sie zum Beispiel die folgenden Konfigurationsparameter des Datenbankmanagers definieren:

  – Setzen Sie den Parameter **max_coordagents** auf den Wert 250.
  – Setzen Sie den Parameter **max_connections** auf den Wert 250.

- Nehmen Sie an, dass Sie den Verbindungskonzentrator nicht aktivieren und die Anzahl der verbundenen Benutzer nicht begrenzen wollen. In diesem Fall könnten Sie die Konfiguration des Datenbankmanagers wie folgt aktualisieren:

```
update dbm cfg using max_connections automatic
update dbm cfg using max_coordagents automatic
```

Agenten in einer partitionierten Datenbank

In einer Umgebung mit partitionierten Datenbanken oder in einer Umgebung mit aktivierter partitionsinterner Parallelität verfügt jede Datenbankpartition 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, oder von neuen Anwendungen verwendet werden.

Der Einfluss auf die Leistung und die Speicherbelegung innerhalb des Systems ist eng mit der Optimierung des Agentenpools verbunden. Der Konfigurationsparameter des Datenbankmanagers für die Größe des Agentenpools (**num_poolagents**) betrifft die Gesamtzahl von Agenten und Subagenten, die Anwendungen in einer Datenbankpartition zugeordnet 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 erneut Anwendungen zugeordnet werden müssen, sinkt die Leistung.

Standardmäßig hat der Parameter **num_poolagents** die Einstellung AUTOMATIC mit einem Wert von 100 und der Datenbankmanager verwaltet die Anzahl der inaktiven Agenten im Pool automatisch.

Wenn der Parameter **num_poolagents** manuell auf einen zu niedrigen Wert gesetzt wird, könnte eine Anwendung allein den Pool mit zugeordneten Subagenten füllen. Wenn nun eine andere Anwendung einen neuen Subagenten benötigt und keine Agenten im zugehörigen Agentenpool hat, übernimmt und verwendet sie inaktive Subagenten aus den Agentenpools anderer Anwendungen. Dieses Verhalten stellt sicher, dass Ressourcen vollständig genutzt werden.

Wenn der Parameter **num_poolagents** manuell auf einen zu hohen Wert gesetzt wird, verbleiben zugeordnete Subagenten über einen langen Zeitraum ungenutzt im Pool und belegen Datenbankmanagerressourcen, die dadurch für andere Tasks nicht verfügbar sind.

Wenn der Verbindungskonzentrator aktiviert ist, gibt der Wert des Parameters **num_poolagents** nicht unbedingt die exakte Anzahl der Agenten wieder, die sich zu einem bestimmten Zeitpunkt inaktiv im Pool befinden können. Agenten werden möglicherweise vorübergehend zum Auffangen höherer Auslastungsaktivitäten benötigt.

Neben Datenbankagenten werden auch andere asynchrone Aktivitäten des Datenbankmanagers als eigene Prozesse bzw. Threads ausgeführt. Dazu gehören:
- E/A-Server oder E/A-Vorablesefunktionen der Datenbank
- Asynchrone Seitenlöschfunktionen der Datenbank
- Protokollfunktionen der Datenbank
- Deadlock-Detektoren für Datenbanken
- Übertragungs- und IPC-Listenerfunktionen
- Funktionen zur Neuverteilung von Daten in Tabellenbereichscontainern

Konfiguration im Hinblick auf gute Leistung

Einige Typen von DB2-Implementierungen, wie zum Beispiel InfoSphere Balanced Warehouse (BW) oder solche innerhalb von SAP-Systemen, haben hoch spezifizierte Konfigurationen.

Im Fall von Balanced Warehouse (BW) sind Hardwarefaktoren, wie die Anzahl der CPUs, das Verhältnis von Speicher zu CPU, die Anzahl und Konfiguration von Platten und die Versionen, auf der Basis gründlicher, im Hinblick auf die optimale Konfiguration durchgeführter Tests im Voraus spezifiziert. Im Fall von SAP-Systemen ist die Hardwarekonfiguration nicht ebenso präzise spezifiziert. Es ist jedoch eine große Anzahl von Beispielkonfigurationen verfügbar. Darüber hinaus geben die empfohlenen Methoden für SAP auch Empfehlungen für DB2-Konfigurationseinstellungen an. Wenn Sie eine DB2-Implementierung für ein System verwenden, das getestete Konfigurationsrichtlinien zur Verfügung stellt, sollten Sie in der Regel diese Richtlinien anstelle allgemeinerer Faustregeln befolgen.

Stellen Sie sich ein vorgeschlagenes System vor, für das Sie noch keine detaillierte Hardwarekonfiguration haben. Ihre Zielsetzung besteht darin, eine kleine Anzahl kritischer Konfigurationsentscheidungen zu ermitteln, die das System geeignet auf gute Leistung hin ausrichten. Dieser Schritt findet in der Regel statt, bevor das System betriebsbereit ist, sodass Ihre Kenntnisse darüber, wie sich das System tatsächlich verhalten wird, wahrscheinlich eher begrenzt sind. In gewisser Weise müssen Sie eine “wahrscheinlichste Annahme” auf der Basis Ihrer Kenntnisse darüber treffen, was das System leisten soll.

Hardwarekonfiguration

Die CPU-Kapazität ist eine der unabhängigen Hauptvariablen bei der Konfiguration eines Systems im Hinblick auf gute Leistung. Da sich in der Regel alle anderen Aspekte der Hardwarekonfiguration daraus ableiten, ist es nun leicht, vorherzusagen, wie viel CPU-Kapazität für eine bestimmte Auslastung erforderlich ist. In Business-Intelligence-Umgebungen (BI) sind 200 - 300 GB aktiver Rohdaten pro Prozessorkern eine angemessene Schätzung. Für andere Umgebungen ist es eine geeignete Strategie, die erforderliche CPU-Kapazität auf der Basis eines oder mehrerer DB2-Systeme zu messen. Wenn das neue System zum Beispiel 50 % mehr Benutzer verarbeiten muss, wobei jeder Benutzer SQL-Code ausführt, der mindestens so komplex wie der auf dem vorhandenen Vergleichssystem ist, wäre die Annahme angemessen, dass 50 % mehr CPU-Kapazität erforderlich sind. Ähnlich sollten andere Faktoren, die eine Änderung der CPU-Nutzung vorhersagen, wie zum Beispiel geänderte Durchsatzanforderungen oder Änderungen in der Verwendung von Triggern oder der referenziellen Integrität, ebenfalls berücksichtigt werden.

Nachdem Sie eine möglichst präzise Vorstellung vom CPU-Bedarf (aus den verfügbaren Informationen) gewonnen haben, rücken weitere Aspekte der Hardwarekonfiguration ins Blickfeld. Obwohl Sie natürlich die erforderliche Systemplattenkapazität in Gigabyte oder Terabyte nicht außer Acht lassen dürfen, sind die wichtigsten Faktoren im Hinblick auf die Leistung die Kapazität in E/A-Operationen pro Sekunde (IOPS) oder die Megabyte pro Sekunde bei der Datenübertragung. Praktisch wird dies durch Anzahl der einzelnen beteiligten Platten bestimmt.

Warum ist dies der Fall? Die Weiterentwicklung von CPUs war im letzten Jahrzehnt durch ungeahnte Geschwindigkeitszuwächse gekennzeichnet, während sich die Weiterentwicklung von Plattendatenträgern mehr im Bereich der Kapazität und der Kosten vollzog. Es wurden Verbesserungen bei Plattensuchzeiten und Übertragungsraten erzielt, die jedoch hinter den CPU-Geschwindigkeiten zurückgeblieben

sind. Um also eine notwendige Aggregatleistung mit modernen Systemen zu realisieren, ist der Einsatz mehrerer Platten wichtiger denn je. Dies gilt insbesondere für Systeme, die ein beträchtliches Volumen an wahlfreien Platten-E/A-Zugriffen ausführen. In vielen Fällen scheint es vermeintlich ausreichend, nicht wesentlich mehr als die Mindestanzahl von Platten zu verwenden, die zur Aufnahme des gesamten Datenvolumens im System benötigt werden. Dies führt im Allgemeinen jedoch zu einer sehr schlechten Leistung.

Bei einem RAID-Speicher oder bei einzeln adressierbaren Laufwerken besagt die Faustregel, dass mindestens 10 bis 20 Platten pro Prozessorkern zu konfigurieren sind. Für Speicherserver wird eine ähnliche Anzahl empfohlen. Jedoch ist in diesem Fall zusätzlich besondere Vorsicht geboten. Eine Zuordnung von Speicherbereich auf Speicherservern erfolgt häufig mehr mit Blick auf die Kapazität als auf Durchsatz. Es ist sehr zu empfehlen, sich mit dem physischen Layout des Datenbankspeichers vertraut zu machen, um sicherzustellen, dass es zu keiner unbeabsichtigten Überlappung von logisch separaten Speicherbereichen kommt. Zum Beispiel könnte eine angemessene Zuordnung für ein 4-Wege-System aus acht Arrays mit jeweils acht Laufwerken bestehen. Wenn jedoch alle acht Arrays dieselben acht zugrunde liegenden physischen Laufwerke nutzen würden, wäre der Durchsatz der Konfiguration gegenüber acht Arrays, die auf 64 physische Laufwerke verteilt sind, drastisch geringer.

Es ist ein bewährtes Verfahren, eine dedizierte (nicht gemeinsam genutzte) Platte für die DB2-Transaktionsprotokolle vorzusehen. Dies hat seinen Grund darin, dass sich die E/A-Merkmale der Protokolle zum Beispiel sehr von denen der DB2-Container unterscheiden, sodass eine Konkurrenzsituation zwischen E/A-Operationen für Protokolle und anderen Typen von E/A-Operationen insbesondere bei Systemen mit einem hohen Grad an Schreibaktivitäten zu einem Protokollengpass führen kann.

Im Allgemeinen kann ein RAID 1-Plattenpaar genügend Protokolldurchsatz für bis zu 400 angemessen schreibintensive DB2-Transaktionen pro Sekunde bereitstellen. Höhere Durchsatzraten oder volumenintensives Protokollieren (z. B. bei Massendateneinfügungen) erfordern einen höheren Protokolldurchsatz, der durch zusätzliche Platten in einer RAID 10-Konfiguration bereitgestellt werden kann, die über einen Plattencontroller mit Schreibcachingfunktion verbunden werden.

Da CPUs und Platten effektiv in verschiedenen Zeitskalen operieren (in Nanosekunden bzw. in Mikrosekunden), müssen Sie sie voneinander abkoppeln, um eine akzeptable Verarbeitungsleistung zu erzielen. Hier kommt der Hauptspeicher ins Spiel. In einem Datenbanksystem hat der Hauptspeicher in erster Linie die Aufgabe, E/A-Operationen zu vermeiden. Es gilt daher bis zu einem gewissen Punkt, dass ein System eine umso bessere Leistung erreicht, je mehr Hauptspeicher es besitzt. Infolge der stark gesunkenen Marktpreise für Speicher im Lauf der vergangenen Jahre sind heute Systeme mit Hauptspeichergrößen (RAM) von bis zu Hunderten Gigabyte (GB) durchaus nicht mehr ungewöhnlich. Im Allgemeinen sollten vier bis acht GB pro Prozessorkern für die meisten Anwendungen geeignet sein.

AIX-Konfiguration

Die Zahl der AIX-Parameter, die zur Realisierung einer guten Leistung geändert werden müssen, ist relativ gering. Gehen Sie für den Zweck dieser Empfehlungen mindestens von einer AIX-Version 5.3 aus. Es sei noch einmal darauf hingewiesen, dass bestimmte Einstellungen, die für Ihr System (z. B. bei einer BW- oder SAP-Konfiguration) bereits festgelegt wurden, Vorrang vor den nachfolgenden allgemeinen Richtlinien haben sollten.

- Der VMO-Parameter **LRU_FILE_REPAGE** sollte auf den Wert 0 gesetzt werden. Dieser Parameter steuert, ob AIX Berechnungsseiten oder Dateisystemcacheseiten als zu bereinigend auswählt. Darüber hinaus sollte der Parameter **minperm** auf den Wert 3 gesetzt werden. Beide Werte sind Standardwerte in AIX 6.1.
- Der AIO-Parameter **maxservers** (AIO, asynchrone Ein-/Ausgabe) kann zu Anfang mit seinem Standardwert von 10 pro CPU belassen werden. Wenn das System aktiv ist, wird **maxservers** wie folgt optimiert:
  1. Erfassen Sie die Ausgabe des Befehls `ps –elfk | grep aio` und stellen Sie fest, ob alle asynchronen E/A-Kernelprozesse ('aioserver', asynchrone E/A-Server) denselben Betrag an CPU-Zeit belegen.
  2. Ist dies der Fall, ist **maxservers** möglicherweise zu niedrig eingestellt. Erhöhen Sie den Wert für **maxservers** um 10 % und wiederholen Sie Schritt 1.
  3. Wenn einige asynchrone E/A-Server weniger CPU-Zeit als andere belegen, hat das System mindestens so viele von ihnen, wie es benötigt. Wenn mehr als 10 % der asynchronen E/A-Server weniger CPU-Zeit belegen, verringern Sie den Wert von **maxservers** um 10 % und wiederholen Schritt 1.
- Der AIO-Parameter **maxreqs** sollte auf `MAX(NUM_IOCLEANERS x 256, 4096)` gesetzt werden. Dieser Parameter steuert die maximale Anzahl von ausstehenden asynchronen E/A-Anforderungen (AIO-Anforderungen).
- Der Hdisk-Parameter **queue_depth** sollte auf der Anzahl der physischen Platten im Array basieren. Zum Beispiel hat **queue_depth** für IBM®-Platten den Standardwert 3 und der empfohlene Werte wäre 3 x *anzahl_der_einheiten*. Dieser Parameter steuert die Anzahl der in eine Warteschlange einreihbaren Plattenanforderungen.
- Der Plattenadapterparameter **num_cmd_elems** sollte auf die Summe der Parameter **queue_depth** für alle mit dem Adapter verbundenen Einheiten gesetzt werden. Dieser Parameter steuert die Anzahl der Anforderungen, die für den Adapter in eine Warteschlange eingereiht werden können.

Solaris- und HP-UX-Konfiguration

Für DB2 unter Solaris oder HP-UX steht das Dienstprogramm **db2osconf** zur Verfügung, mit dem Kernelparameter auf der Basis der Systemgröße geprüft und Werte empfohlen werden können. Das Dienstprogramm **db2osconf** gibt Ihnen die Möglichkeit, die Kernelparameter auf der Basis von Hauptspeicher und CPU oder mit einem allgemeinen Skalierungsfaktor anzugeben, der die aktuelle Systemkonfiguration mit einer für die Zukunft erwarteten Konfiguration vergleicht. Eine gute Methode besteht darin, einen Skalierungsfaktor von 2 oder höher anzusetzen, wenn große Systeme, wie zum Beispiel SAP-Anwendungen ausgeführt werden. Im Allgemeinen erhalten Sie durch **db2osconf** einen guten Ausgangspunkt zur Konfiguration von Solaris und HP-UX. Das Dienstprogramm liefert jedoch nicht den optimalen Wert, da es aktuelle und zukünftige Auslastungen nicht mit einkalkulieren kann.

Linux-Konfiguration

Der DB2-Datenbankmanager aktualisiert automatisch die wichtigsten Linux-Kernelparameter, um die Voraussetzungen einer Vielzahl unterschiedlicher Konfigurationen zu erfüllen.

Weitere Informationen hierzu finden Sie im Abschnitt „Voraussetzungen für Kernelparameter (Linux)" in der Veröffentlichung *DB2-Server - Installation*.

DB2 Database Partitioning Feature

Die Entscheidung, DB2 Database Partitioning Feature (DPF) zu verwenden, wird im Allgemeinen nicht allein auf der Basis des Datenvolumens, sondern eher auf der Basis der Auslastung getroffen. Als allgemeine Richtlinie gilt, dass die meisten DPF-Implementierungen im Bereich von Data-Warehousing und Business-Intelligence erfolgen. DPF ist für große und komplexe Abfrageumgebungen sehr zu empfehlen, weil diese Architektur mit exklusiv genutzten Systemen (Shared-Nothing-Architektur) eine hervorragende Skalierbarkeit zur Verfügung stellt. Für kleinere Datamarts (ca. bis zu 300 GB), die kein rasches Wachstum erwarten lassen, ist eine Konfiguration mit DB2 Enterprise Server Edition (ESE) häufig eine gute Wahl. Großen oder schnell wachsenden Business-Intelligence-Umgebungen (BI) bietet DPF jedoch beträchtliche Vorteile.

Ein typisches partitioniertes Datenbanksystem hat in der Regel einen Prozessorkern pro Datenpartition. Zum Beispiel hätte ein System mit *n* Prozessorkernen den Katalog wahrscheinlich in Partition 0 und zusätzlich *n* weitere Datenpartitionen. Wenn die Katalogpartition stark ausgelastet wird (z. B. durch die Speicherung von Dimensionstabellen einer Partition), kann ihr ebenfalls ein Prozessorkern zugeordnet werden. Wenn das System eine sehr große Zahl gleichzeitig aktiver Benutzer unterstützen soll, könnten zwei Kerne pro Partition erforderlich sein.

Als allgemeine Richtlinie sollten Sie ca. mit 250 GB an aktiven Rohdaten pro Partition planen.

Die InfoSphere Balanced Warehouse-Dokumentation enthält eingehende Informationen zu bewährten Verfahren bei der Konfiguration partitionierter Datenbanken. Diese Dokumentation bietet außerdem auch nützliche Informationen für andere Implementierungen als Balanced Warehouse-Implementierungen.

Auswahl von Codepage und Sortierfolge

Neben dem Einfluss auf das Datenbankverhalten kann sich die Auswahl der Codepage bzw. des codierten Zeichensatzes und der Sortierfolge auch auf die Leistung stark auswirken. Die Verwendung von Unicode ist inzwischen weit verbreitet, weil Unicode die Darstellung einer größeren Vielfalt von Zeichenfolgen in einer Datenbank ermöglicht, als dies mit den traditionellen Einzelbyte-Codepages der Fall war. Unicode ist die Standardeinstellung für neue Datenbanken in DB2 Version 9.5. Da codierte Unicode-Zeichensätze jedoch mehrere Byte zur Darstellung einiger Einzelzeichen verwenden, kann ein höherer Platten- und Hauptspeicherbedarf bestehen. Zum Beispiel verwendet der codierte Zeichensatz UTF-8, der einer der gängigsten codierten Unicode-Zeichensätze ist, zwischen ein und vier Byte pro Zeichen. Ein durchschnittlicher Faktor, um den sich Zeichenfolgen aufgrund einer Migration von einem codierten Einzelbytezeichensatz auf UTF-8 verlängern, ist sehr schwer zu schätzen, weil er davon abhängt, wie oft Mehrbytezeichen verwendet werden. Für typischen Inhalt in Nordamerika tritt zum Beispiel in der Regel keine Verlängerung auf. Für die meisten westeuropäischen Sprachen führt die Verwendung von Zeichen mit Akzent in der Regel zu einer Verlängerung von ca. 10 %.

Darüber hinaus kann die Verwendung von Unicode im Vergleich zu Einzelbyte-Codepages eine zusätzliche CPU-Belegung zur Folge haben. Erstens, wenn es zu einer Verlängerung kommt, erfordern die längeren Zeichenfolgen mehr Bearbeitungsaufwand. Zweitens, und wichtiger, können die Algorithmen, die von den komplexeren Unicode-Sortierfolgen verwendet werden, wie zum Beispiel UCA500R1_NO, wesentlich verarbeitungsaufwendiger sein als die typische Sortierfolge SYSTEM, die mit Einzelbyte-Codepages verwendet wird. Dieser höhere Auf-

wand ist auf die Komplexität des Sortierens von Unicode-Zeichenfolgen in einer landesspezifisch korrekten Weise zurückzuführen. Operationen, die davon betroffen werden, sind Sortierungen, Zeichenfolgevergleiche, die LIKE-Verarbeitung und die Indexerstellung.

Wenn Unicode für eine ordnungsgemäße Darstellung Ihrer Daten erforderlich ist, wählen Sie die Sortierfolge mit Sorgfalt aus.

- Wenn die Datenbank viele Sprachen enthalten soll und die korrekte Sortierreihenfolge dieser Daten von höchster Wichtigkeit sind, verwenden Sie eine der landesspezifisch korrekten Sortierfolgen (z. B. UCA500R1_*). Abhängig von den Daten und der Anwendung könnte dies zur einer 1,5- bis 3fachen Leistungseinbuße gegenüber der Sortierfolge IDENTITY führen.
- Es sind normalisierte und nicht normalisierte Arten von landesspezifisch korrekten Sortierfolgen vorhanden. Normalisierte Sortierfolgen (z. B. UCA500R1_NO) haben zusätzliche Prüfungen zur Behandlung von fehlerhaften Zeichen, während nicht normalisierte Sortierfolgen (z. B. UCA500r1_NX) dies nicht tun. Sofern die Behandlung von fehlerhaften Zeichen kein Problem ist, verwenden Sie die nicht normalisierte Version, weil sich durch die Vermeidung des Normalisierungscodes ein Leistungsvorteil ergibt. Nichtsdestoweniger sind sogar nicht normalisierte, landesspezifisch korrekte Sortierfolgen sehr aufwendig.
- Wenn eine Datenbank aus einer Einzelbyteumgebung in eine Unicode-Umgebung versetzt wird, jedoch keine strengen Anforderungen im Hinblick auf die Bereitstellung einer Reihe von Sprachen hat (was bei den meisten Implementierungen der Fall sein wird), kann eine sprachsensitive Sortierfolge angemessen sein. *Sprachsensitive Sortierfolgen* (z. B. SYSTEM_819_BE) nutzen die Tatsache aus, dass viele Unicode-Datenbanken Daten nur einer Sprache enthalten. Sie verwenden denselben auf einer Suchtabelle basierenden Sortierfolgealgorithmus wie Einzelbytesortierfolgen (z. B. SYSTEM_819) und sind daher sehr effizient. Als allgemeine Regel gilt, dass, wenn das Sortierfolgeverhalten in der ursprünglichen Einzelbytedatenbank annehmbar war, eine sprachsensitive Sortierfolge in Betracht gezogen werden sollte, solange sich der Sprachinhalt nach der Umstellung auf Unicode nicht wesentlich ändert. Dies kann große Leistungsvorteile im Vergleich zur landesspezifisch korrekten Sortierfolge erbringen.

Physischer Datenbankentwurf

- Im Allgemeinen zeigen dateibasierte reguläre DMS-Tabellenbereiche (vom Datenbankmanager verwaltete Tabellenbereiche) eine bessere Leistung als reguläre SMS-Tabellenbereiche (vom System verwaltete Tabellenbereiche). SMS wird häufig für Tabellenbereiche für temporäre Tabellen verwendet, insbesondere wenn die temporären Tabellen sehr klein sind. Der Leistungsvorteil von SMS schrumpft jedoch mit der Zeit.
- In der Vergangenheit hatten DMS-Tabellenbereiche auf Roheinheiten einen substanziellen Leistungsvorteil gegenüber DMS-Dateitabellenbereichen. Mit der Einführung der direkten E/A (die nun als Standardeinstellung durch die Klausel NO FILE SYSTEM CACHING in den Anweisungen CREATE TABLESPACE und ALTER TABLESPACE gültig ist) bieten DMS-Dateitabellenbereiche praktisch dieselbe Leistung wie DMS-Roheinheitentabellenbereiche.

Anfangseinstellungen für die DB2-Konfiguration

Der DB2-Konfigurationsadvisor (Befehl **AUTOCONFIGURE**) empfängt grundlegende Systemrichtlinien, die Sie angeben, und ermittelt einen guten Startsatz von DB2-Konfigurationswerten. Der Befehl **AUTOCONFIGURE** kann reale Verbesserungen gegenüber den Standardkonfigurationseinstellungen bereitstellen und wird als Methode zur Ermittlung von Erstkonfigurationswerten empfohlen. Eine zusätzliche Feinein-

stellung der Empfehlungen, die vom Befehl **AUTOCONFIGURE** generiert werden, ist in vielen Fällen abhängig von den Merkmalen des Systems erforderlich.

Beachten Sie die folgenden Empfehlungen zur Verwendung des Befehls **AUTOCONFIGURE**:

- Obwohl seit DB2 Version 9.1 der Befehl **AUTOCONFIGURE** bei einer Datenbankerstellung automatisch ausgeführt wird, ist es durchaus zu empfehlen, den Befehl **AUTOCONFIGURE** auch explizit auszuführen. Dies bietet Ihnen die Möglichkeit, Schlüsselwort/Wert-Paare anzugeben, die bei der Anpassung der Ergebnisse an Ihr System helfen können.
- Führen Sie den Befehl **AUTOCONFIGURE** aus (bzw. erneut aus), nachdem die Datenbank mit einem geeigneten Volumen an aktiven Daten gefüllt wurde. Dadurch werden dem Tool zusätzliche Informationen über die Spezifik der Datenbank zur Verfügung gestellt. Das Datenvolumen, mit dem die Datenbank gefüllt wird, ist von Bedeutung, da es sich z. B. auf die Berechnungen für die Pufferpoolgröße auswirkt. Zu viele oder zu wenige Daten beeinträchtigen die Genauigkeit solcher Berechnungen.
- Testen Sie verschiedene Werte für wichtige Schlüsselwörter des Befehls **AUTOCONFIGURE**, zum Beispiel **mem_percent**, **tpm** und **num_stmts**, um einen Eindruck davon zu bekommen, welche Konfigurationswerte in welchem Grad von diesen Änderungen betroffen sind.
- Verwenden Sie beim Experimentieren mit verschiedenen Schlüsselwörtern und Werten die Option APPLY NONE. Dies bietet Ihnen die Möglichkeit, die Empfehlungen mit den aktuellen Einstellungen zu vergleichen.
- Geben Sie Werte für alle Schlüsselwörter an, da die Standardwerte für Ihr System möglicherweise nicht geeignet sind. Der Standardwert für **mem_percent** ist zum Beispiel 25 %. Dieser Wert ist für einen dedizierten DB2-Server zu niedrig. In diesem Fall wäre der empfohlene Wert 85 %.

Autonome und automatische DB2-Parameter

Die neueren Releases des DB2-Datenbankprodukts besitzen eine erheblich höhere Anzahl von Parametern, die entweder automatisch beim Start der Instanz bzw. der Datenbank eingestellt oder während des Betriebs dynamisch optimiert werden. Für die meisten Systeme bieten die automatischen Einstellungen eine bessere Leistung als alle anderen, sofern die Systeme nicht sehr sorgfältig von Hand optimiert wurden. Dies liegt insbesondere am DB2-Speichermanager mit automatischer Leistungsoptimierung (Self-Tuning Memory Manager, STMM), der die gesamte Datenbankspeicherzuordnung sowie vier Hauptkonsumenten in einem DB2-System dynamisch optimiert: die Pufferpools, die Sperrenliste, den Paketcache und den Sortierspeicher.

Da diese Parameter jeweils nur für eine einzelne Partition gelten, muss die Verwendung von STMM in einer Umgebung mit partitionierten Datenbanken mit einiger Vorsicht erfolgen. In partitionierten Datenbanksystemen misst STMM kontinuierlich den Speicherbedarf einer einzelnen Partition (deren Auswahl vom DB2-System automatisch getroffen wird, jedoch überschrieben werden kann), und sendet Aktualisierungen der Werte für Zwischenspeichergrößen an alle Partitionen, in denen STMM aktiviert ist. Da in allen Partitionen dieselben Werte verwendet werden, funktioniert STMM am besten in Umgebungen mit partitionierten Datenbanken, in denen die Datenvolumen, der Speicherbedarf und die allgemeinen Aktivitätsgrade sehr gleichmäßig auf alle Partitionen verteilt sind. Wenn eine kleine Anzahl von Partitionen abweichende Datenvolumen oder andere Speicheranforderungen hat, sollte STMM in diesen Partitionen inaktiviert werden. In den Partitio-

nen mit gleichmäßigeren Anforderungen kann STMM zur Optimierung beibehalten werden. Zum Beispiel sollte STMM im Allgemeinen in der Katalogpartition inaktiviert werden.

Für Umgebungen mit partitionierten Datenbanken, die eine ungleichmäßige Datenverteilung haben und für die eine clusterübergreifende Speicheroptimierung nicht empfohlen wird, kann STMM selektiv und vorübergehend während einer 'Optimierungsphase' eingesetzt werden, um gute manuelle Speichereinstellungen zu ermitteln:

- Aktivieren Sie STMM in einer 'typischen' Partition. In anderen Partitionen bleibt STMM weiterhin inaktiviert.
- Nachdem sich die Speichereinstellungen stabilisiert haben, inaktivieren Sie STMM und legen die betroffenen Parameter mit ihren optimierten Werten manuell fest.
- Implementieren Sie die optimierten Werte in anderen Datenbankpartitionen mit ähnlichen Datenvolumen und ähnlichem Speicherbedarf (z. B. Partitionen in derselben Datenbankpartitionsgruppe).
- Wiederholen Sie den Prozess, wenn mehrere disjunkte Gruppen von Datenbankpartitionen vorhanden sind, die ähnliche Volumen und Typen von Daten enthalten und ähnliche Aufgaben im System erfüllen.

Der Konfigurationsadvisor wählt im Allgemeinen die Aktivierung der automatischen Einstellungen aus, wo dies möglich ist. Dies umfasst automatische Aktualisierungen von Statistiken durch den Befehl **RUNSTATS** (sehr nützlich), schließt jedoch automatische Reorganisationen und automatische Backup-Operationen aus. Diese können ebenfalls sehr nützlich sein, müssen jedoch entsprechend Ihrer Umgebung konfiguriert und zwecks optimaler Ergebnisse durch einen Zeitplan terminiert werden. Die automatische Statistikprofilerstellung sollte standardmäßig inaktiviert bleiben. Sie verursacht einen recht hohen Systemaufwand und ist dazu gedacht, zeitlich begrenzt unter kontrollierten Bedingungen und mit komplexen Anweisungen verwendet zu werden.

Explizite Konfigurationseinstellungen

Einige Parameter haben keine automatischen Einstellungen und werden vom Konfigurationsadvisor nicht festgelegt. Diese müssen explizit eingestellt werden. Im vorliegenden Dokument werden nur Parameter behandelt, die Auswirkung auf die Leistung haben.

- Der Parameter **logpath** oder **newlogpath** legt die Position des Transaktionsprotokolls fest. Auch der Konfigurationsadvisor kann nicht für Sie entscheiden, wo die Protokolle gespeichert werden sollen. Wie zuvor erwähnt, ist der wichtigste Punkt, dass sie keine Platteneinheiten mit anderen DB2-Objekten, wie zum Beispiel Tabellenbereichen, gemeinsam nutzen sollten, oder an der Standardposition, die sich unter dem Datenbankpfad befindet, verbleiben sollten. Im Idealfall sollten die Transaktionsprotokolle auf einem dedizierten Speicher mit genügend Durchsatzkapazität platziert werden, um sicherzustellen, dass kein Engpass entsteht.
- Der Parameter **logbufsz** legt die Größe des internen Puffers für die Transaktionsprotokollfunktion in 4-KB-Seiten fest. Der Standardwert von nur acht Seiten ist wesentlich zu klein für eine gute Leistung in einer Produktionsumgebung. Der Konfigurationsadvisor setzt diesen Wert abhängig von den Eingabeparametern immer herauf, jedoch möglicherweise nicht hoch genug. Ein Wert von 256 - 1000 Seiten ist im Allgemeinen ein guter Bereich und stellt nur eine sehr kleine Gesamtgröße von Speicher im Gesamtschema eines Datenbankservers dar.

- Der Parameter **mincommit** steuert die *Anzahl von Gruppencommits*, die ein DB2-System veranlasst, nach Möglichkeit *n* Anwendungen, die eine Commitoperation ausführen, zu einem Stapel zusammenzufassen. Angesichts des aktuellen Aufbaus der Transaktionsprotokollfunktion ist dies nur selten ein wünschenswertes Verhalten. Belassen Sie den Parameter **mincommit** auf dem Standardwert 1.
- Der Parameter **buffpage** legt die Anzahl von Seiten fest, die jedem Pufferpool zugeordnet werden, der mit einer Größe von -1 definiert ist. Die empfohlene Methode ist, den Parameter **buffpage** zu ignorieren und entweder die Größe von Pufferpools, die einen Eintrag in der Katalogsicht SYSCAT.BUFFERPOOLS haben, explizit festzulegen oder STMM die Pufferpoolgröße automatisch optimieren zu lassen.
- Der Parameter **diagpath** legt die Position verschiedener nützlicher DB2-Diagnosedateien fest. Er hat im Allgemeinen wenig Einfluss auf die Leistung, außer möglicherweise in einer Umgebung mit partitionierten Datenbanken. Die durch **diagpath** angegebene Standardposition in allen Partitionen ist in der Regel ein gemeinsam genutzter, über NFS-Mount zugeordneter Pfad. Die empfohlene Methode besteht darin, den Wert des Parameters **diagpath** mit einem lokalen, nicht über NFS-Mount zugeordneten Verzeichnis für jede Partition zu überschreiben. Dies verhindert, dass alle Partitionen versuchen, dieselbe Datei mit Diagnosenachrichten zu aktualisieren. Stattdessen werden diese lokal in jeder Partition behalten, sodass die Wahrscheinlichkeit von Konkurrenzsituationen erheblich geringer ist.
- **DB2_PARALLEL_IO** ist kein Konfigurationsparameter, sondern eine DB2-Registrierdatenbankvariable. DB2-Systeme arbeiten sehr häufig mit Speicher, der aus Plattenarrays besteht, die sich dem Betriebssystem gegenüber wie eine Einheit verhalten, oder mit Dateisystemen, die sich über mehrere Einheiten erstrecken. Die Folge ist, dass ein DB2-Datenbanksystem nur eine Vorablesezugriffsanforderung gleichzeitig an einen Tabellenbereichscontainer absetzt. Dies geschieht unter der Auffassung, dass mehrere Anforderungen an eine einzige Einheit ohnehin serialisiert würden. Wenn sich ein Container jedoch auf einem Array von Platten befindet, besteht die Möglichkeit, mehrere Vorablesezugriffsanforderungen gleichzeitig an das Array abzusetzen, ohne dass diese serialisiert werden. An diesem Punkt kommt die Variable **DB2_PARALLEL_IO** ins Spiel. Sie teilt dem DB2-System mit, dass Vorablesezugriffsanforderungen parallel an einen einzigen Container abgesetzt werden können. Die einfachste Einstellung ist `DB2_PARALLEL_IO=*` (d. h. alle Container befinden sich auf mehreren - hier werden sieben angenommen - Platten), jedoch steuern auch andere Einstellungen den Grad der Parallelität und die Tabellenbereiche, die betroffen sind. Wenn Sie zum Beispiel wissen, dass sich Ihre Container auf einem RAID-5-Array mit vier Platten befinden, könnten Sie **DB2_PARALLEL_IO** auf den Wert *:3 setzen. Ob bestimmte Werte die Leistung fördern oder nicht, hängt auch vom Wert der Speicherbereichsgröße (EXTENTSIZE), von der RAID-Segmentgröße sowie von der Anzahl der Container ab, die dieselbe Gruppe von Platten verwenden.

Hinweise für SAP-Umgebungen und andere Umgebungen unabhängiger Softwareanbieter

Wenn Sie einen DB2-Datenbankserver für eine Anwendung eines unabhängigen Softwareanbieters (ISV-Anwendung) wie zum Beispiel SAP ausführen, sind möglicherweise Richtlinien zu bewährten Verfahren verfügbar, die speziell für diese Anwendung erarbeitet wurden. Der einfachste Mechanismus ist die DB2-Registrierdatenbankvariable **DB2_WORKLOAD**, die auf einen Wert gesetzt werden kann, der eine Optimierung kumulativer Registrierdatenbankvariablen für bestimmte Umgebun-

gen und Auslastungen ermöglicht. Für die Variable **DB2_WORKLOAD** sind die folgenden Einstellungen gültig: 1C, CM, COGNOS_CS, FILENET_CM, MAXIMO, MDM, SAP, TPM, WAS, WC und WP.

Möglicherweise gelten weitere Empfehlungen und bewährte Verfahren, wie zum Beispiel die Auswahl einer Codepage oder eines codierten Zeichensatzes und einer Sortierfolge, da diese auf einen vordefinierten Wert gesetzt werden müssen. Detaillierte Informationen dazu finden Sie in der Dokumentation des Anwendungsanbieters.

Für viele Anwendungen unabhängiger Softwareanbieter, wie SAP Business One, kann der Befehl **AUTOCONFIGURE** erfolgreich zur Definition der Erstkonfiguration verwendet werden. Er sollte jedoch nicht in SAP NetWeaver-Installationen verwendet werden, weil ein Anfangssatz von DB2-Konfigurationsparametern bei der SAP-Installation angewendet wird. Darüber hinaus bietet SAP eine leistungsfähige Strategie für bewährte Verfahren (SAP Notes), die die bevorzugten DB2-Parametereinstellungen beschreibt, wie zum Beispiel 'SAP Note 1086130 - DB6: DB2 9.5 Standard Parameter Settings'.

Richten Sie besondere Aufmerksamkeit auf SAP-Anwendungen, wenn Sie DB2 Database Partitioning Feature (DPF) verwenden. SAP verwendet DPF in erster Linie im Produkt SAP NetWeaver Business Intelligence (Business Warehouse). Beim empfohlenen Layout befinden sich der DB2-Systemkatalog, die Dimensions- und Mastertabellen und die SAP-Basistabellen in Partition 0. Dies führt in dieser Partition zu einer anderen Auslastung im Vergleich zu anderen Installationen mit DB2 DPF. Da der SAP-Anwendungsserver in dieser Partition ausgeführt wird, können bis zu acht Prozessoren nur dieser Partition zugeordnet werden. Wenn die SAP BW-Auslastung in höherem Maße parallelisiert wird, wobei viele kurze Abfragen gleichzeitig ausgeführt werden, ist die Anzahl von Partitionen für SAP BI in der Regel kleiner als für andere Anwendungen. Das heißt mit anderen Worten, dass mehr als eine CPU pro Datenpartition erforderlich ist.

Konfiguration der Instanz

Wenn Sie eine neue DB2-Instanz starten, können Sie eine Reihe von Schritten ausführen, um eine Basiskonfiguration einzurichten.

- Mithilfe des Konfigurationsadvisors können Sie Empfehlungen für die Anfangswerte der Pufferpoolgröße, der Datenbankkonfigurationsparameter und der Konfigurationsparameter des Datenbankmanagers abrufen. Zur Verwendung des Konfigurationsadvisors geben Sie den Befehl **AUTOCONFIGURE** für eine vorhandene Datenbank an oder Sie geben beim Erstellen einer Datenbank die Option AUTOCONFIGURE im Befehl **CREATE DATABASE** an. Sie können die empfohlenen Werte anzeigen oder anwenden, indem Sie die Option APPLY im Befehl **CREATE DATABASE** verwenden. Die Empfehlungen basieren auf Ihrer Eingabe sowie auf Systeminformationen, die von der Advisorfunktion erfasst werden.
- Mithilfe des Konfigurationsassistenten können Sie Ihre Datenbankobjekte konfigurieren und verwalten, neue Objekte hinzufügen, Anwendungen binden, Konfigurationsparameter des Datenbankmanagers einstellen sowie Konfigurationsinformationen importieren und exportieren. Zum Öffnen des Konfigurationsassistenten rufen Sie den Befehl **db2ca** auf. Bei der Instanzkonfiguration hilft Ihnen der Konfigurationsassistent, Konfigurationsparameter des Datenbankmanagers einzustellen, DB2-Registrierdatenbankvariablen zu definieren, eine andere Instanz zu konfigurieren oder die Konfiguration zurückzusetzen.

- Lesen Sie die Übersichtstabellen (siehe „Konfigurationsparameter - Zusammenfassung"), in denen die einzelnen Konfigurationsparameter aufgelistet und kurz beschrieben sind, die für den Datenbankmanager bzw. eine Datenbank zur Verfügung stehen. Diese Übersichtstabellen enthalten eine Spalte, die angibt, ob die Optimierung eines bestimmten Parameters wahrscheinlich eine hohe, mittlere, geringe oder keine Änderung der Leistung bewirkt. Ermitteln Sie anhand dieser Tabellen die Parameter, durch die sich in Ihrer Umgebung möglicherweise die größten Leistungsverbesserungen erzielen lassen.
- Verwenden Sie den Befehl **ACTIVATE DATABASE**, um eine Datenbank zu aktivieren und alle erforderlichen Datenbankservices zu initialisieren, sodass die Datenbank für Verbindungen und zur Verwendung durch eine Anwendung verfügbar ist. In einer Umgebung mit partitionierten Datenbanken 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.

Aufbau von Tabellenbereichen

Leistungsfaktoren für Plattenspeicher

Hardwaremerkmale, wie zum Beispiel die Plattenspeicherkonfiguration, können die Leistung Ihres Systems erheblich beeinflussen.

Die Leistung kann von folgenden Aspekten der Plattenspeicherkonfiguration beeinflusst werden:

- Speichereinteilung

  Wie geeignet Sie eine begrenzte Speichergröße zwischen Indizes und Daten sowie unter Tabellenbereichen aufteilen, bestimmt in hohem Maße, welche Leistung das System in verschiedenen Situationen erreicht.
- Verteilung der Platten-E/A

  Wie ausgewogen Sie den Bedarf an Platten-E/A auf mehrere Einheiten und Controller verteilen, kann sich auf die Geschwindigkeit auswirken, mit der der Datenbankmanager Daten vom Plattenspeicher abrufen kann.
- Wichtige Leistungsmessdaten zum Plattensubsystem

  Die Anzahl der Plattenoperationen pro Sekunde und die Kapazität, gemessen in pro Sekunde übertragenen Megabyte, haben einen sehr starken Einfluss auf die Leistungsdaten des Systems insgesamt.

Auswirkung von Tabellenbereichen auf die Abfrageoptimierung

Bestimmte Merkmale der verwendeten Tabellenbereiche können die Auswahl der Zugriffspläne durch den Abfragecompiler beeinflussen.

Zu diesen Merkmalen gehören die folgenden:

- 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 bei Zugriffen auf Daten verschiedener Tabellenbereiche. Zwei Spalten in der Katalogsicht SYSCAT.TABLESPACES werden vom Optimierungsprogramm zur Abschätzung der E/A-Aufwände für den Zugriff auf Daten in einem Tabellenbereich herangezogen:

– Die Spalte OVERHEAD enthält einen Schätzwert für die Zeit (in Millisekunden), 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.

  Mithilfe der folgenden Formel lässt sich der Systemaufwand (OVERHEAD) abschätzen:

  ```
  OVERHEAD = durchschnittliche Suchzeit in Millisekunden
             + (0,5 * rotationsbedingte Latenzzeit)
  ```

  Dabei gilt:

  - 0,5 stellt den durchschnittlichen Aufwand für eine halbe Umdrehung (Rotation) dar.
  - Die rotationsbedingte Latenzzeit (in Millisekunden) wird für jede vollständige Umdrehung wie folgt berechnet:

    ```
    (1 / Umdrehungen pro Minute) * 60 * 1000
    ```

    Dabei gilt:

    • Sie dividieren durch die Umdrehungen pro Minute, um die Minuten pro Umdrehung zu erhalten.
    • Sie multiplizieren mit 60 Sekunden pro Minute.
    • Sie multiplizieren mit 1000 Millisekunden pro Sekunde.

  Nehmen Sie zum Beispiel an, dass eine Platte 7200 Umdrehungen pro Minute ausführt. In diesem Fall sieht die Formel für die rotationsbedingte Latenzzeit wie folgt aus:

  ```
  (1 / 7200) * 60 * 1000 = 8,328 Millisekunden
  ```

  Dieser Wert kann in die Formel für die Aufwandsschätzung unter Annahme einer durchschnittlichen Plattensuchzeit von 11 Millisekunden wie folgt eingesetzt werden:

  ```
  OVERHEAD = 11 + (0,5 * 8,328)
           = 15,164
  ```

– Die Spalte TRANSFERRATE enthält einen Schätzwert in Millisekunden für die Zeit, die zum Einlesen einer Datenseite in den Hauptspeicher benötigt wird.

  Wenn jeder Tabellenbereichscontainer eine einzelne physische Platte ist, können Sie mithilfe der folgenden Formel den Übertragungsaufwand pro Seite in Millisekunden abschätzen:

  ```
  TRANSFERRATE = (1 / spezifikationsrate) * 1000 / 1024000 * Seitengröße
  ```

  Dabei gilt:

  - Sie dividieren durch *spezifikationsrate*, d. h. durch den Spezifikationswert für die Übertragungsrate der Platte (in MB pro Sekunde), um die Sekunden pro MB zu berechnen.
  - Sie multiplizieren mit 1000 Millisekunden pro Sekunde.
  - Sie dividieren durch 1.024.000 Byte pro MB.
  - Sie multiplizieren mit der Seitengröße (in Byte), zum Beispiel mit 4096 Byte für eine 4-KB-Seite.

  Beispiel: Nehmen Sie an, dass als Spezifikationsrate für eine Platte 3 MB pro Sekunde angegeben sind. In diesem Fall gilt:

  ```
  TRANSFERRATE = (1 / 3) * 1000 / 1024000 * 4096
               = 1,333248
  ```

Dies wären also 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 Werts für TRANSFERRATE zu beachten.

Wenn die Platteneinheit relativ klein ist, können Sie den Wert für *spezifikationsrate* mit der Anzahl Platten multiplizieren, da anzunehmen ist, dass der Engpass auf Plattenebene liegt. Wenn die Platteneinheit jedoch groß ist, liegt der Engpass möglicherweise nicht auf Plattenebene, sondern an einer der E/A-Subsystemkomponenten, wie zum Beispiel Plattencontrollern, E/A-Bussen oder dem Systembus. In diesem Fall kann nicht angenommen werden, dass die E/A-Durchsatzkapazität das Produkt aus *spezifikationsrate* und der Anzahl der Platten ist. Stattdessen muss die tatsächliche E/A-Geschwindigkeit (in MB) während einer sequenziellen Tabellensuche gemessen werden. Zum Beispiel könnte eine sequenzielle Suche mit einer Anweisung wie `select count(*) from grosse_tabelle` einen Umfang von mehreren MB haben. In diesem Fall dividieren Sie das Ergebnis durch die Anzahl der Container, die den Tabellenbereich bilden, in dem die Tabelle GROSSE_TABELLE gespeichert ist. Setzen Sie dieses Ergebnis für *spezifikationsrate* 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 / 1.024.000 * 4096 = 0,16 Millisekunden pro Seite bedeuten.

Container, die einem Tabellenbereich zugeordnet sind, können sich auf verschiedenen physischen Platten befinden. Um die besten Ergebnisse erzielen zu können, sollten alle physischen Platten, die für einen bestimmten Tabellenbereich verwendet werden, die gleichen Werte für OVERHEAD und TRANSFERRATE besitzen. Wenn diese Merkmale nicht übereinstimmen, sollten Sie bei der Einstellung der Werte für OVERHEAD und TRANSFERRATE Mittelwerte 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.

- 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 Abfrageleistung haben kann. Der Vorablesezugriff kann den Aufwand verringern, der mit dem Einlesen von Daten in den Pufferpool verbunden ist.

  Das Optimierungsprogramm verwendet die Informationen aus den Spalten PREFETCHSIZE und EXTENTSIZE der Katalogsicht SYSCAT.TABLESPACES, um das Volumen der durch den Vorablesezugriff gelesenen Daten abzuschätzen.

  – Der Wert für EXTENTSIZE kann nur bei der Erstellung eines Tabellenbereichs festgelegt werden. Ein Bereichsgröße für EXTENTSIZE von 4 oder 8 Seiten ist in der Regel ausreichend.
  – Der Wert für PREFETCHSIZE kann bei der Erstellung oder bei einer Änderung des Tabellenbereichs festgelegt werden. Der Standardwert für PREFETCHSIZE wird durch den Wert des Datenbankkonfigurationsparameters **dft_prefetch_sz** bestimmt. Lesen Sie die Empfehlungen zur Einstellung dieses Parameters und nehmen Sie Änderungen nach Bedarf vor oder definieren Sie den Parameter mit AUTOMATIC.

Ziehen Sie nach der Ausführung von Änderungen an den Tabellenbereichen in Betracht, das Dienstprogramm RUNSTATS zum Erfassen der neuesten Statistikdaten zu Indizes auszuführen, um sicherzustellen, dass das Abfrageoptimierungsprogramm die bestmöglichen Datenzugriffspläne auswählt, bevor Sie einen Rebind für die Anwendungen durchführen.

Datenbankentwurf

Tabellen

Tabellen- und Indexverwaltung für Standardtabellen

In Standardtabellen werden Tabellendaten logisch in Form einer Liste von Datenseiten organisiert. Diese Datenseiten 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 Datenseiten enthalten sind, kann je nach Größe der Datenseite und Größe der Datensätze variieren. Die meisten Seiten enthalten nur Benutzerdatensätze. Eine kleine Zahl von Seiten enthält jedoch besondere interne Datensätze, die vom Datenserver zur Verwaltung der Tabelle verwendet werden. Auf jeder 500sten Seite befindet sich beispielsweise ein FSCR-Datensatz (Free Space Control Record, Steuersatz für freien Speicherbereich, siehe Abb. 8 auf Seite 65). Diese Datensätze ordnen den freien Speicherbereich, der für neue Datensätze auf jeder der folgenden 500 Datenseiten verfügbar ist, (bis zum nächsten FSCR-Datensatz) zu.

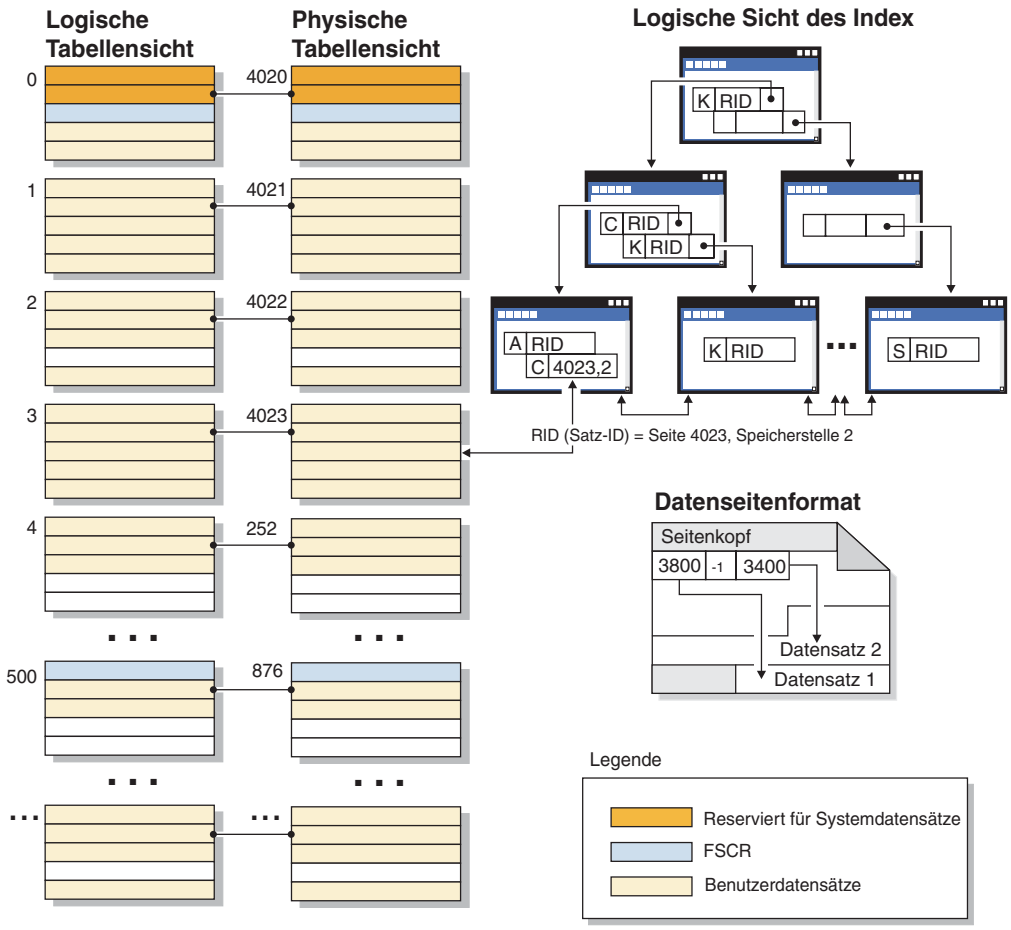


Abbildung 8. Logische Tabellen-, Datensatz- und Indexstruktur für Standardtabellen

Logisch werden Indexseiten als B-Baumstruktur organisiert, durch die Tabellendatensätze, die einen bestimmten 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 vom Datenbankmanager verwalteten Tabellenbereichen (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. Eine Seite beginnt mit einem Seitenkopf, dem ein Speicherstellenverzeichnis folgt. Jeder Eintrag im Speicherstellenverzeichnis entspricht einem anderen Datensatz auf der Seite. Ein Eintrag im Speicherstellenverzeichnis stellt die relative Byteadresse auf der Datenseite dar, an er ein Datensatz beginnt. Einträge mit dem Wert -1 entsprechen gelöschten Datensätzen.

Satz-IDs und Seiten

Bei Satz-IDs (Record Identifier, RID) handelt es sich um eine Seitennummer gefolgt von einer Speicherstellennummer (siehe Abb. 9 auf Seite 66). Indexdatensätze enthalten ein zusätzliches Feld mit der Bezeichnung 'ridFlag'. Das Feld 'ridFlag' speichert Informationen über den Status von Schlüsseln im Index, zum Beispiel, ob sie als gelöscht markiert wurden. Nachdem der Index zur Identifizierung einer Satz-ID verwendet wurde, wird diese Satz-ID dazu verwendet, die richtige Datenseite und

Speicherstellennummer auf der betreffenden Seite zu ermitteln. Wenn dem Datensatz eine Satz-ID zugeordnet wurde, wird diese erst wieder bei einer Reorganisation der Tabelle geändert.

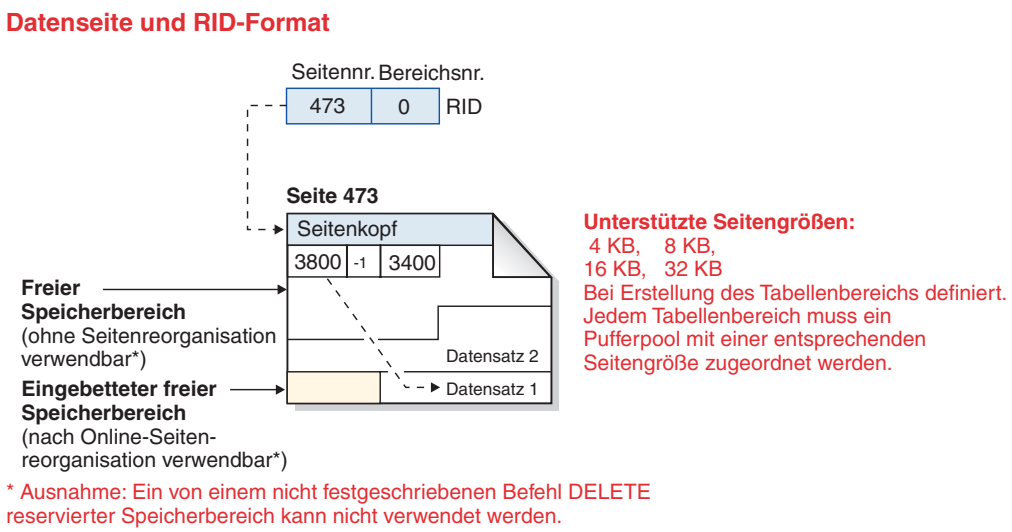


Abbildung 9. Format der Datenseite und der Satz-ID (RID)

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.

Der DB2-Datenserver unterstützt verschiedene Seitengrößen. Verwenden Sie größere Seiten für Auslastungen, bei denen auf Zeilen eher sequenziell zugegriffen wird. Ein sequenzieller Zugriff wird beispielsweise häufig für Anwendungen zur Entscheidungshilfe oder bei extensiver Nutzung von temporären Tabellen verwendet. Verwenden Sie geringere Seitengrößen für Auslastungen, bei denen eher ein wahlfreier Zugriff auf Zeilen erfolgt. Ein wahlfreier Zugriff wird beispielsweise häufig in OLTP-Umgebungen (Onlinetransaktionsverarbeitung) verwendet.

Indexverwaltung in Standardtabellen

DB2-Indizes verwenden eine optimierte B-Baumstrukurimplementierung, die auf einer effizienten Indexverwaltungsmethode mit einem hohen Grad an gemeinsamem Zugriff und einer im Voraus schreibenden Protokollierung (Write Ahead Logging) basiert. Ein B-Baumstrukturindex (B-Tree-Index) ist als gleichmäßige Hierarchie von Seiten angeordnet, die die Zugriffszeit minimiert, indem sie Datenschlüssel beim Einfügen oder Löschen von Elementen neu ausrichtet.

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 Seite mit den höchsten Indexschlüsseln (HIGHKEY) eine Ausnahme bildet, da bei ihr eine 90/10-Teilung erfolgt. Das heißt, die höchsten zehn Prozent von Indexschlüsseln werden auf einer neuen Seite gespeichert. Diese Art der Indexseitenteilung ist bei Auslastungen nützlich, bei denen häufig Einfügeoperationen mit neuen höchsten Werten für die Indexschlüssel ausgeführt werden.

Gelöschte Indexschlüssel werden von einer Indexseite nur entfernt, wenn eine X-Sperre für die Tabelle aktiviert wurde. Wenn Schlüssel nicht sofort entfernt werden können, werden sie als gelöscht markiert und später physisch entfernt.

Wenn Sie die Online-Indexdefragmentierung aktiviert haben, indem Sie einen positiven Wert für MINPCTUSED angegeben haben, als der Index erstellt wurde, können Indexblattseiten online zusammengefügt werden. Der MINPCTUSED-Wert gibt den minimalen Prozentsatz des verwendeten Speicherplatzes auf einer Indexblattseite an. Wenn nach dem Entfernen eines Schlüssels die Größe des verwendeten Speicherplatzes auf einer Indexseite unter diesen Wert fällt, 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 Indexblattseite gelöscht. Da die Onlinedefragmentierung nur stattfindet, wenn Schlüssel von einer Indexseite entfernt werden, findet sie nicht statt, wenn Schlüssel nur als gelöscht markiert, jedoch nicht physisch von der Seite entfernt wurden. Die Online-Indexdefragmentierung kann die Wiederverwendung von Speicherbereich verbessern. Wenn jedoch der Wert für MINPCTUSED zu hoch ist, wird mehr Zeit für das Zusammenfügen von Seiten benötigt, und ein erfolgreiches Zusammenfügen wird weniger wahrscheinlich. Der empfohlene Wert für MINPCTUSED ist 50 % oder weniger.

Die Klausel INCLUDE der Anweisung CREATE INDEX ermöglicht die Angabe einer oder mehrerer zusätzlicher Spalten (über die Schlüsselspalten hinaus) für die Indexblattseiten. Diese INCLUDE-Spalten, die nicht an Sortieroperationen über die B-Baumstruktur des Index beteiligt sind, können die Zahl der Abfragen erhöhen, bei denen ein reiner Indexzugriff möglich ist. Allerdings können sie auch den Indexspeicherbedarf und möglicherweise den Indexverwaltungsaufwand erhöhen, wenn die eingeschlossenen Spalten häufig aktualisiert werden. Der Verwaltungsaufwand zur Aktualisierung von INCLUDE-Spalten ist geringer als der Aufwand zur Aktualisierung von Schlüsselspalten, jedoch höher als der Aufwand zur Aktualisierung von Spalten, die nicht in einem Index enthalten sind.

Tabellen- und Indexverwaltung für MDC-Tabellen

Die Tabellen- und Indexorganisation für Tabellen mit mehrdimensionalem Clustering (MDC-Tabellen) 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 Satz-IDs (RIDs) gekennzeichnet. Die Seiten von MDC-Tabellen werden jedoch in EXTENTSIZE große Blöcke gruppiert. Zum Beispiel zeigt Abb. 10 auf Seite 68 eine Tabelle mit dem EXTENTSIZE-Wert 4. Die ersten vier Seiten mit den Nummern 0 bis 3 bilden den ersten Block in der Tabelle. Die nächsten vier Seiten mit den Nummern 4 bis 7 bildet den zweiten Block in der Tabelle.

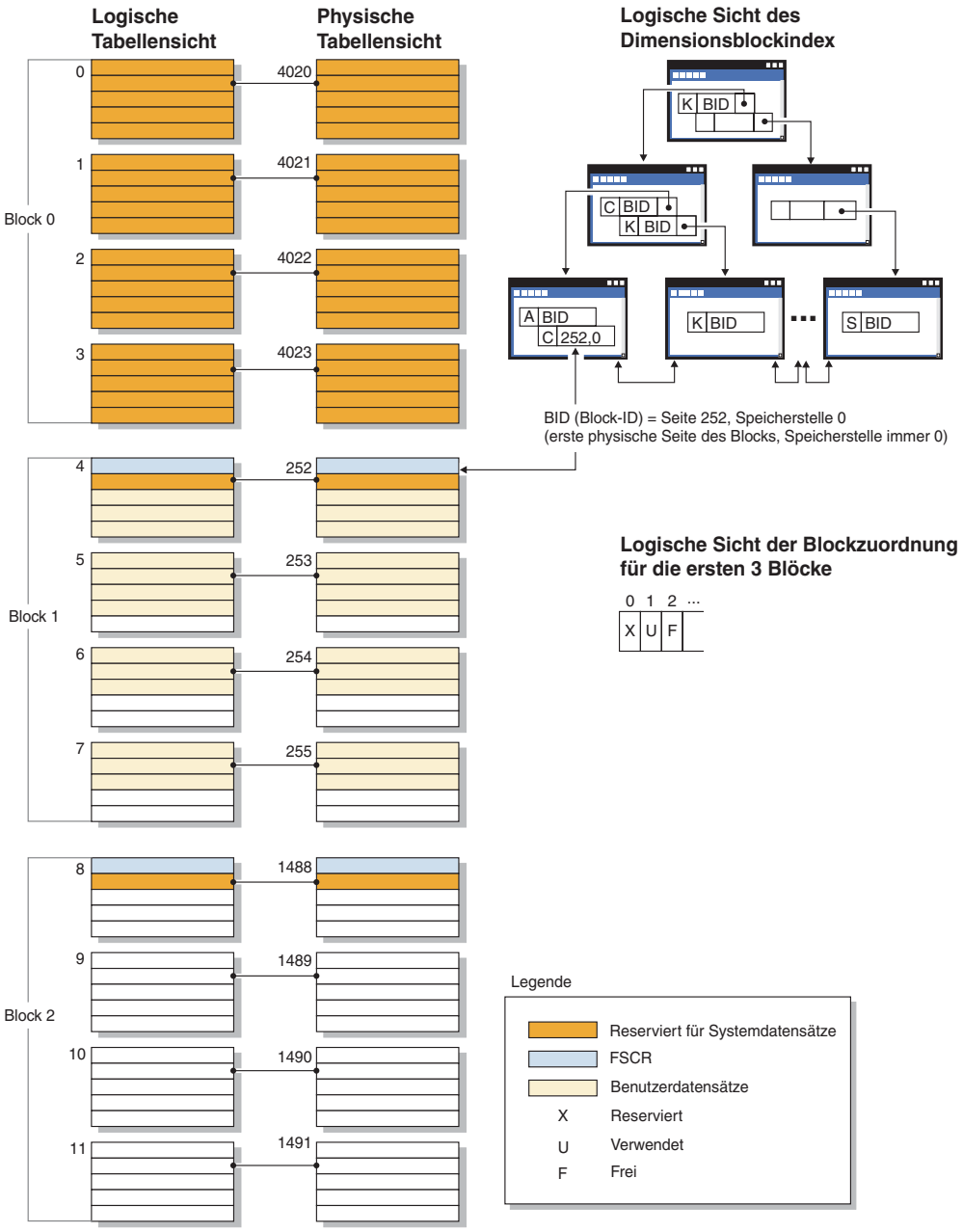


Abbildung 10. Logische Tabellen-, Datensatz- und Indexstruktur für MDC-Tabellen

Der erste Block enthält besondere interne Datensätze, einschließlich des Steuersatzes für freien Speicherbereich (Free Space Control Record, FSCR), die vom DB2-Server zur Verwaltung der Tabelle verwendet werden. In den nachfolgenden Blöcken enthält jeweils die erste Seite den FSCR-Satz. Ein FSCR-Satz ordnet den freien Speicher für neue Datensätze zu, der auf jeder Seite des Blocks 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 Indizes automatisch erstellt:

- Ein Dimensionsblockindex, der Zeiger auf jeden belegten Block für eine einzelne Dimension enthält
- Ein zusammengesetzter Blockindex, der alle Dimensionsschlüsselspalten enthält und zur Aufrechterhaltung des Clusterings bei Einfüge- und Aktualisierungsaktivitäten verwendet wird

Das Optimierungsprogramm zieht Zugriffspläne, die Dimensionsblockindizes verwenden, 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 Daten aus diesen abzurufen. Da sich diese Speicherbereiche in physisch aufeinander folgenden Seiten auf der Platte befinden, wird der E/A-Aufwand minimiert und eine bessere Leistung erzielt.

Sie können auch spezielle Satz-ID-Indizes (RID-Indizes) erstellen, wenn eine Analyse von Datenzugriffsplänen nahe legt, dass solche Indizes die Abfrageleistung verbessern würden.

Indizes

Indexstruktur

Der Datenbankmanager verwendet eine B+-Baumstruktur zur Indexspeicherung.

Eine B+-Baumstruktur besitzt mehrere Stufen, wie in Abb. 11 auf Seite 70 dargestellt. Das Kürzel „rid" steht für Satz-ID (RID).

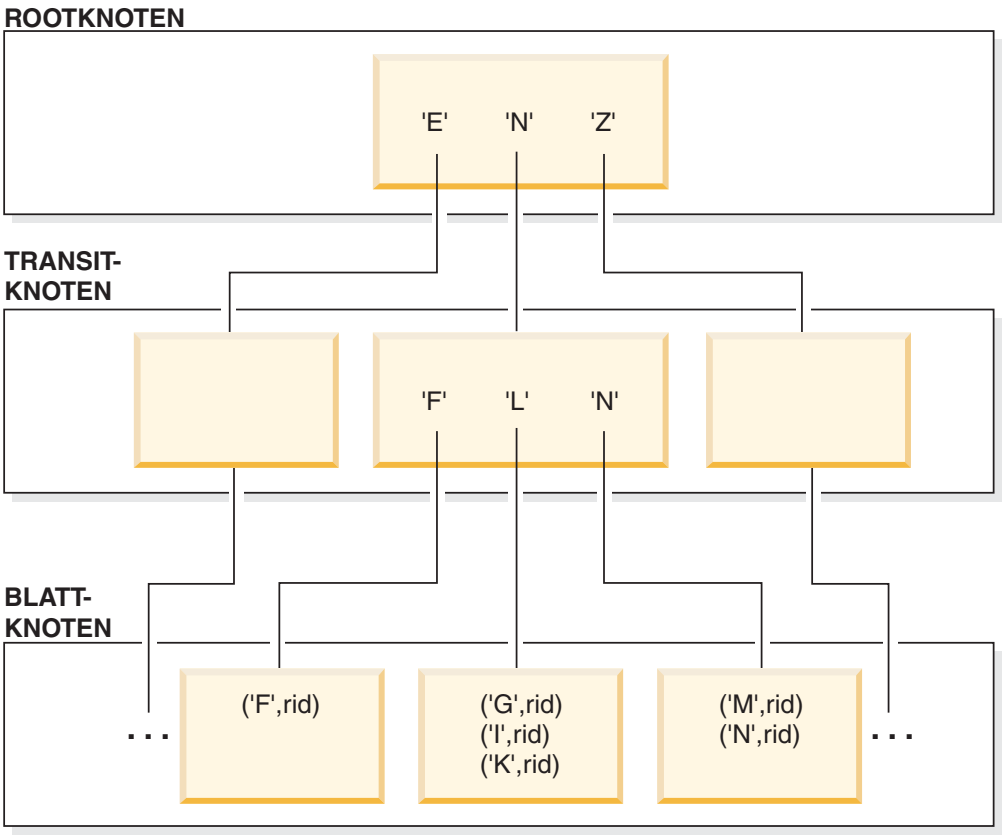


Abbildung 11. Struktur eines Index mit B+-Baumstruktur

Die oberste Stufe wird als *Rootknoten* bezeichnet. Die unterste Stufe besteht aus *Blattknoten* (engl. leaf nodes), in denen Indexschlüsselwerte mit Zeigern auf die Tabellenzeilen gespeichert werden, die die entsprechenden Daten enthalten. Die Stufen zwischen dem Rootknoten und den 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 (oder Transitknoten) auf der folgenden Stufe. Der Wert jedes dieser Schlüssel ist jeweils der größte vorhandene Schlüsselwert für den entsprechenden Knoten auf der nächsten Stufe. Nehmen Sie zum Beispiel an, dass ein Index drei Stufen hat, wie in der Abbildung dargestellt. Zum Auffinden eines bestimmten Indexschlüsselwerts durchsucht der Indexmanager den Rootknoten nach dem ersten Schlüsselwert, der größer oder gleich dem gesuchten Schlüsselwert ist. Der Rootknotenschlüssel enthält einen Zeiger auf einen bestimmten Transitknoten. Der Indexmanager setzt dieses Verfahren durch die einzelnen Transitknoten fort, bis er den Blattknoten findet, der den benötigten Indexschlüssel enthält.

Nehmen Sie an, dass in Abb. 11 nach dem Schlüssel „I" gesucht wird. Der erste Schlüssel im Rootknoten, der größer oder gleich „I" ist, ist „N". Dieser Wert verweist 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 wiederum auf einen bestimmten Blattknoten, in dem der Indexschlüssel für „I" zusammen mit der entsprechenden Satz-ID (RID) zu finden ist. Die 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 Option ALLOW REVERSE SCANS erstellt wurde.

Im Fall einer Tabelle mit mehrdimensionalem Clustering (MDC-Tabelle) wird automatisch ein Blockindex für jede Clusteringdimension erstellt, die Sie für die Tabelle angeben. Außerdem wird ein zusammengesetzter Blockindex erstellt. Dieser Index enthält einen Schlüsselbestandteil für jede Spalte enthält, die in einer Dimension der Tabelle enthalten ist. Solche Indizes enthalten Zeiger auf Block-IDs (BIDs) anstelle von Satz-IDs (RIDs) und bieten Verbesserungen beim Datenzugriff.

Mit einer ein Byte großen *ridFlag*-Markierung, die für jede Satz-ID auf der Blattseite eines Index gespeichert wird, wird die Satz-ID ggf. als logisch gelöscht markiert, sodass sie später physisch entfernt werden kann. Wenn eine Aktualisierungs- oder Löschoperation durch Commit festgeschrieben wird, können die als gelöscht markierten Schlüssel entfernt werden. Für jede Spalte variabler Länge im Index wird in zwei zusätzlichen Byte die tatsächliche Länge des Spaltenwerts gespeichert.

Indexbereinigung und Indexpflege

Nach der Erstellung eines Index kann sich die Leistung mit der Zeit möglicherweise verschlechtern, wenn Sie den Index nicht kompakt und effizient organisiert halten.

Die folgenden Empfehlungen sollten Ihnen helfen, 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 auf Tabellenebene, entweder explizit oder durch Sperreneskalation, falls häufige Commits nicht möglich sind.

  Indexschlüssel, die als gelöscht markiert sind, können nach einem Commit physisch aus der Tabelle entfernt werden. X-Sperren für Tabellen ermöglichen es, die gelöschten Schlüssel physisch zu entfernen, wenn sie als gelöscht markiert sind, wie weiter unten erläutert wird.
- Verwenden Sie den Befehl **REORGCHK**, um festzustellen, wann Indizes oder Tabellen zu reorganisieren sind und wann der Befehl **REORG INDEXES** mit der Klausel CLEANUP ONLY zu verwenden ist.

  Um einen Schreib- und Lesezugriff auf einen Index bei der Reorganisation zuzulassen, verwenden Sie den Befehl **REORG INDEXES** mit der Option ALLOW WRITE ACCESS.

  Um einen Schreib- und Lesezugriff auf einen Index bei der Bereinigung zuzulassen, verwenden Sie den Befehl **REORG INDEXES** mit der Option ALLOW WRITE ACCESS. Bei partitionierten Tabellen kann die Klausel ALLOW WRITE ACCESS für den Befehl **REORG INDEXES...ALL** nur in Verbindung mit der Option CLEANUP ONLY oder der Option ON DATA PARTITION angegeben werden.

  Bei DB2 Version 9.7 Fixpack 1 und späteren Releases führen Sie den Befehl **REORG INDEXES** mit der Klausel `ON DATA PARTITION` für eine datenpartitionierte Tabelle aus, um die partitionierten Indizes für die angegebene Partition zu reorganisie-

ren. Während der Indexreorganisation bleibt der Zugriff auf die nicht betroffenen Partitionen bestehen. Nur der Lese- und Schreibzugriff auf die betroffene Partition wird beschränkt.

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 wurden, bereinigt, wenn dies die Durchführung einer Seitenteilung 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 der physischen Löschung werden alle gelöschten Schlüssel auf derselben Seite ebenfalls entfernt, wenn sie als gelöscht markiert sind und bekannt ist, dass diese Löschung festgeschrieben wurde.

- 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, von Seiten, die auch einige nicht gelöschte Satz-IDs enthalten. Diese Option versucht außerdem, benachbarte Blattseiten zusammenzufügen, wenn dies zu einer zusammengefügten Blattseite führt, die mindestens den durch PCTFREE angegebenen freien Speicherbereich enthält. Der PCTFREE-Wert wird bei der Erstellung eines Index definiert. Der Standardwert für PCTFREE ist 10 %. Wenn zwei Seiten zusammengefügt werden können, wird eine der Seiten freigegeben.

  Bei datenpartitionierten Tabellen wird empfohlen, den Befehl **RUNSTATS** nach Abschluss einer asynchronen Indexbereinigung aufzurufen. Um festzustellen, ob in der Tabelle Datenpartitionen mit aufgehobener Zuordnung vorhanden sind, führen Sie eine Abfrage auf das Feld STATUS in der Katalogsicht SYSCAT.DATAPARTITIONS aus und suchen nach dem Wert 'L' ('Logically detached', Zuordnung logisch aufgehoben), dem Wert 'D' ('Detached', Zuordnung aufgehoben) oder dem Wert 'I' (Indexbereinigung).

- Beim erneuten Erstellen eines Index (bzw. bei datenpartitionierten Indizes, beim erneuten Erstellen einer Indexpartition)

  Zu den Dienstprogrammen, die Indizes neu erstellen, gehören folgende:

  – **REORG INDEXES** ohne eine der CLEANUP-Optionen
  – **REORG INDEXES** mit der Klausel ON DATA PARTITION
  – **REORG TABLE** mit der Klausel ON DATA PARTITION
  – **REORG TABLE** ohne die Option INPLACE
  – **IMPORT** mit der Option REPLACE
  – **LOAD** mit der Option INDEXING MODE REBUILD

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 Index können die Einträge Satz-IDs (RIDs) oder Block-IDs (BIDs) sein. Ungültige Indexeinträge werden von Indexbereinigungsfunktionen entfernt, die asynchron im Hintergrund ausgeführt werden.

Die asynchrone Indexbereinigung beschleunigt den Prozess des Aufhebens der Zuordnung einer Datenpartition zu einer partitionierten Tabelle und wird eingeleitet, wenn die partitionierte Tabelle mindestens einen nicht partitionierten Index hat. In diesem Fall entfernt die asynchrone Indexbereinigung 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. In DB2 Version 9.7 Fixpack 1 und späteren Releases wird die asynchrone Indexbereinigung durch eine Task zur asynchronen Aufhebung von Partitionszuordnungen eingeleitet.

Wenn in einer Version vor DB2 Version 9.7 Fixpack 1 die partitionierte Tabelle abhängige MQTs (Materialized Query Tables) hat, wird die asynchrone Indexbereinigung erst nach der Ausführung einer Anweisung SET INTEGRITY ausgeführt.

Der normale Tabellenzugriff bleibt während der Ausführung der asynchronen Indexbereinigung erhalten. Abfragen, die auf die Indizes zugreifen, ignorieren 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 Tabellenpartitionen sowie für die Zuweisung von Datenbankagenten zuständig. Der Verteilungsdämonprozess und die Bereinigungsagenten sind interne Systemanwendungen, die in der Ausgabe des Befehls **LIST APPLICATIONS** mit den Anwendungsnamen **db2taskd** bzw. **db2aic** angegeben werden. 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 Sie die Datenbank reaktivieren.

Auswirkung der asynchronen Indexbereinigung auf die Leistung

Die asynchrone Indexbereinigung wirkt sich nur minimal auf die Leistung aus.

Ein sofortiger Zeilensperrentest ist erforderlich, um festzustellen, ob ein pseudogelöschter 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). Diese Sperren werden freigegeben, wenn eine Bereinigungsfunktion feststellt, dass andere Anwendungen auf Sperren warten. Wenn dieser Fall eintritt, setzt die Bereinigungsfunktion die Verarbeitung für fünf Minuten aus.

Bereinigungsfunktionen sind in die Drosselungseinrichtung für Dienstprogramme integriert. Standardmäßig ist für jede Bereinigungsfunktion der Wert 50 für die Priorität der Auslastungswirkung von Dienstprogrammen eingestellt. Sie können die

Priorität mit dem Befehl **SET UTIL_IMPACT_PRIORITY** oder mit der API db2UtilityControl ändern.

Überwachen der asynchronen Indexbereinigung

Sie können die asynchrone Indexbereinigung mit dem Befehl **LIST UTILITIES** überwachen. Jede Indexbereinigungsfunktion wird in der Ausgabe als separates Dienstprogramm aufgeführt. Das folgende Beispiel zeigt eine Ausgabe des Befehls **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. Die 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 einer Bereinigungsfunktion an. Der normale Status ist 'Wird ausgeführt'. Jedoch kann sich die Bereinigungsfunktion 'Im Wartestatus' befinden, wenn sie darauf wartet, einem verfügbaren Datenbankagenten zugeordnet zu werden oder wenn sie wegen eines Sperrenkonflikts vorübergehend ausgesetzt ist.

Beachten Sie, dass verschiedene Tasks in verschiedenen Datenbankpartitionen die gleiche Dienstprogramm-ID haben können, da jede Datenbankpartition IDs den Tasks zuordnet, die in dieser Datenbankpartition ausgeführt werden.

Asynchrone Indexbereinigung für MDC-Tabellen

Sie können die Leistung einer Rolloutlöschung durch die Nutzung einer asynchronen Indexbereinigung (AIC, Asynchronous Index Cleanup) verbessern. Eine Rolloutlöschung ist eine effiziente Methode zum Löschen von Datenblöcken, die bestimmten Kriterien entsprechen, aus Tabellen mit mehrdimensionalem Clustering

(MDC). Die asynchrone Indexbereinigung ist die verzögerte Bereinigung von Indizes, die im Anschluss an Operationen erfolgt, durch die Indexeinträge ungültig werden.

Indizes werden synchron bei einer Standardrolloutlöschung bereinigt. Wenn eine Tabelle viele Satz-ID-Indizes (RID-Indizes) hat, fällt ein beträchtlicher Zeitaufwand für das Entfernen von Indexschlüsseln an, die auf die Tabellenzeilen verweisen, die gelöscht werden. Sie können den Rollout beschleunigen, indem Sie angeben, dass diese Indizes nach dem Festschreiben der Löschoperation zu bereinigen sind.

Zur Nutzung der asynchronen Indexbereinigung für MDC-Tabellen müssen Sie den Mechanismus für den *Rollout mit verzögerter Indexbereinigung* explizit aktivieren. Ein Rollout mit verzögerter Indexbereinigung kann durch zwei Methoden angegeben werden: Durch Setzen der Registrierdatenbankvariablen **DB2_MDC_ROLLOUT** auf den Wert DEFER und durch Ausführen der Anweisung SET CURRENT MDC ROLLOUT MODE. Während einer Rolloutoperation mit verzögerter Indexbereinigung werden Blöcke als durch Rollout gelöscht markiert, wobei die Aktualisierung an den Satz-ID-Indizes erst erfolgt, wenn die Transaktion festgeschrieben wurde. Block-ID-Indizes (BID-Indizes) werden während der Löschoperation bereinigt, weil sie keine Verarbeitung auf Zeilenebene erfordern.

Die Rolloutoperation der verzögerten Indexbereinigung wird aufgerufen, wenn eine Rolloutlöschung festgeschrieben wird oder, falls die Datenbank beendet wurde, wenn auf die Tabelle nach dem Datenbankneustart zum ersten Mal zugegriffen wird. Während der Ausführung der asynchronen Indexbereinigung werden Abfragen, die auf die Indizes, einschließlich des Index, der gerade bereinigt wird, zugreifen, erfolgreich ausgeführt.

Pro MDC-Tabelle ist eine koordinierende Bereinigungsfunktion vorhanden. Die Indexbereinigung für mehrere Rollouts wird innerhalb der Bereinigungsfunktion konsolidiert, die wiederum einen Bereinigungsagenten für jeden Satz-ID-Index startet. Bereinigungsagenten aktualisieren die Satz-ID-Indizes parallel. Bereinigungsfunktionen sind darüber hinaus in die Drosselungseinrichtung für Dienstprogramme integriert. Standardmäßig ist jede Bereinigungsfunktion auf den Wert 50 für die Priorität der Auslastungswirkung von Dienstprogrammen eingestellt. (Akzeptable Werte liegen zwischen 1 und 100, wobei 0 keine Drosselung angibt.) Sie können diese Priorität mit dem Befehl **SET UTIL_IMPACT_PRIORITY** oder mit der API db2UtilityControl ändern.

Anmerkung: In DB2 Version 9.7 und späteren Releases wird ein Rollout mit verzögerter Bereinigung für eine datenpartitionierte MDC-Tabelle mit partitionierten Satz-ID-Indizes nicht unterstützt. Es werden nur die Modi NONE und IMMEDIATE unterstützt. Der Modus des Rollouts mit Bereinigung ist IMMEDIATE, wenn die Registrierdatenbankvariable **DB2_MDC_ROLLOUT** auf den Wert DEFER gesetzt ist oder wenn das Sonderregister CURRENT MDC ROLLOUT MODE auf den Wert DEFERRED gesetzt ist, um die Einstellung der Variablen **DB2_MDC_ROLLOUT** zu überschreiben.

Wenn nur nicht partitionierte Satz-ID-Indizes für die MDC-Tabelle vorhanden sind, wird ein Rollout mit verzögerter Indexbereinigung unterstützt. MDC-Blockindizes können partitioniert oder nicht partitioniert sein.

Überwachen des Fortschritts einer Rolloutoperation mit verzögerter Indexbereinigung

Da die durch Rollout gelöschten Blöcke in einer MDC-Tabelle erst wieder verwendet werden können, wenn die Bereinigung abgeschlossen ist, ist es nützlich, den Fortschritt einer Rolloutoperation mit verzögerter Indexbereinigung zu überwachen. Mit dem Befehl **LIST UTILITIES** können Sie einen Dienstprogrammüberwachungseintrag für jeden Index anzeigen, der momentan bereinigt wird. Sie können auch mit der Tabellenfunktion SYSPROC.ADMIN_GET_TAB_INFO_V95 oder mit dem Befehl **GET SNAPSHOT** die Gesamtzahl der MDC-Tabellenblöcke in der Datenbank abrufen, für die eine asynchrone Bereinigung nach einer Rolloutlöschung ansteht (BLOCKS_PENDING_CLEANUP).

In der folgenden Beispielausgabe für den Befehl **LIST UTILITIES SHOW DETAIL** wird der Fortschritt durch die Anzahl der Seiten in jedem Index angezeigt, die bereinigt wurden. Jede Phase stellt einen Satz-ID-Index dar.

```
ID                               = 2
Typ                              = MDC ROLLOUT INDEX CLEANUP
Datenbankname                    = WSDB
Partitionsnummer                 = 0
Beschreibung                     = TABLE.<schemaname>.<tabellenname>
Startzeit                        = 06-12-2006 08:56:33.390158
Status                           = Wird ausgeführt
Aufruftyp                        = Automatisch
Drosselung:
   Priorität                     = 50
Fortschrittsüberwachung:
   Geschätzte Fertigstellung (%) = 83
   Phasennummer                  = 1
      Beschreibung               = <schemaname>.<indexname>
      Gesamte Arbeit             = 13 Seiten
      Abgeschlossene Arbeit      = 13 Seiten
      Startzeit                  = 06-12-2006 08:56:33.391566
   Phasennummer                  = 2
      Beschreibung               = <schemaname>.<indexname>
      Gesamte Arbeit             = 13 Seiten
      Abgeschlossene Arbeit      = 13 Seiten
      Startzeit                  = 06-12-2006 08:56:33.391577
   Phasennummer                  = 3
      Beschreibung               = <schemaname>.<indexname>
      Gesamte Arbeit             = 9 Seiten
      Abgeschlossene Arbeit      = 3 Seiten
      Startzeit                  = 06-12-2006 08:56:33.391587
```

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 dieser Schwellenwert überschritten wird, werden die benachbarten Indexblattseiten daraufhin überprüft, ob zwei Blattseiten zusammengefügt werden können. Wenn auf einer Seite ausreichend Platz vorhanden und das Zusammenfügen zweier benachbarter Seiten möglich ist, wird das Zusammenfügen unverzüglich im Hintergrund ausgeführt und die resultierende leere Indexblattseite gelöscht.

Wenn für vorhandene Indizes die Möglichkeit der Onlinezusammenfügung erforderlich ist, müssen sie gelöscht und mithilfe der Anweisung CREATE INDEX unter Angabe der Klausel MINPCTUSED erneut erstellt werden. Der empfohlene Wert für MINPCTUSED ist kleiner als 50, da der Zweck darin besteht, zwei benachbarte

Indexblattseiten zusammenzufügen. Durch den Wert 0, der gleichzeitig der Standardwert ist, wird die Onlinedefragmentierung inaktiviert.

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 (DMS-Speicher = vom Datenbankmanager verwalteter Tabellenbereich) oder um Plattenspeicherplatz in einem SMS-Speichermodell (SMS-Speicher = vom System verwalteter Tabellenbereich) freizugeben, führen Sie eine vollständige Reorganisation der Tabelle und der Indizes aus. Dadurch werden die Indizes auf ihre kleinstmögliche Größe reduziert. Die Anzahl der Stufen in einem Index wird bei einer Online-Indexdefragmentierung nicht verringert.

Wenn eine X-Sperre (exklusive Sperre) für eine Tabelle aktiviert ist, werden Schlüssel beim Löschen von Schlüsseln physisch von einer Seite entfernt. In diesem Fall 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. In diesem Fall wird keine Indexdefragmentierung versucht.

Zur Defragmentierung von Indizes unabhängig vom Wert für MINPCTUSED rufen Sie den Befehl **REORG INDEXES** mit der Option CLEANUP ONLY ALL auf. Zwei benachbarte Blattseiten werden zusammengefügt, wenn mindestens ein freier Speicherbereich von der Größe des PCTFREE-Werts auf der zusammengefügten Seite frei bleibt. PCTFREE kann bei der Erstellung eines Index angegeben werden. Der Standardwert ist 10 (Prozent).

Verwenden von relationalen Indizes zur Leistungsverbesserung

Indizes können zur Leistungsverbesserung beim Zugreifen auf Tabellendaten verwendet werden. Relationale Indizes werden beim Zugriff auf relationale Daten, Indizes zu XML-Daten beim Zugriff auf XML-Daten verwendet.

Obwohl das Abfrageoptimierungsprogramm entscheidet, ob ein relationaler Index für den Zugriff auf Daten relationaler Tabellen verwendet wird, ist es Ihre Aufgabe, zu ermitteln, welche Indizes die Leistung verbessern könnten, und diese Indizes zu erstellen. Die einzigen Ausnahmen von dieser Regel sind die Dimensionsblockindizes und die zusammengesetzten Blockindizes, die für jede Dimension automatisch erstellt werden, wenn Sie eine MDC-Tabelle (MDC, mehrdimensionales Clustering) erstellen.

Führen Sie das Dienstprogramm RUNSTATS aus, um neue Indexstatistikdaten zu erfassen, nachdem Sie einen relationalen Index erstellt oder die Vorablesezugriffsgröße (PREFETCHSIZE) geändert haben. Sie sollten 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.

Um festzustellen, ob ein relationaler Index in einem bestimmten Paket verwendet wird, verwenden Sie die EXPLAIN-Funktion. Wenn Sie Empfehlungen zu relationalen Indizes abrufen wollen, die von einer oder mehreren SQL-Anweisung genutzt werden könnten, starten Sie den Designadvisor mit dem Befehl **db2advis**.

IBM InfoSphere Optim Query Workload Tuner stellt Tools für die Leistungsverbesserung einzelner SQL-Anweisungen und die Leistung von Gruppen von SQL-Anweisungen bereit, die als Abfrageworkloads bezeichnet werden. Weitere Informati-

onen zu diesem Produkt finden Sie auf der Seite mit der Produktübersicht unter http://www.ibm.com/software/data/optim/query-workload-tuner-db2-luw/index.html. In Version 3.1.1 oder späteren Versionen des Produkts können Sie auch den Designadvisor für Workloads verwenden, um zahlreiche Operationen auszuführen, die im Assistenten für den DB2-Designadvisor verfügbar waren. Weitere Informationen finden Sie in der Dokumentation des Designadvisors für Workloads unter http://publib.boulder.ibm.com/infocenter/dstudio/v3r1/topic/com.ibm.datatools.qrytune.workloadtunedb2luw.doc/topics/genrecsdsgn.html.

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 bei einer Tabellensuche auf jede Tabellenzeile sequenziell zugegriffen werden muss. Obwohl eine Tabellensuche für eine komplexe Abfrage, die die meisten der Zeilen in einer Tabelle abruft, 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, insbesondere wenn die Tabelle sehr groß ist. Darüber hinaus ist es vielleicht gar nicht nötig, einen gesamten Index zu durchsuchen. Alle Vergleichselemente, die auf einen Index angewendet werden, verringern die Anzahl der Zeilen, die aus 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 abgerufen werden, ohne dass eine Sortieroperation erforderlich wird. Beachten Sie, dass das Vorhandensein eines relationalen Index für die Tabelle, die abgefragt wird, keine sortierte Ergebnismenge garantiert. Nur durch die Angabe der Klausel ORDER BY lässt sich die Reihenfolge der Ergebnismenge sicherstellen.

Ein relationaler Index kann auch INCLUDE-Spalten enthalten, bei denen es sich um nicht indexierte Spalten in einer indexierten Zeile handelt. Solche Spalten können dem Optimierungsprogramm die Möglichkeit geben, angeforderte Informationen nur aus dem Index abzurufen, ohne auf die Tabelle selbst zugreifen zu müssen.

Nachteile eines relationalen Index gegenüber keinem Index

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. Wählen Sie Indizes sorgfältig, um den Anforderungen der Anwendungsprogramme gerecht zu werden.

- Jeder Index erfordert Speicherplatz. Der exakte Speicherbedarf hängt von der Größe der Tabelle sowie von der Größe und der Anzahl der Spalten im relationalen Index ab.

- Jede für eine Tabelle ausgeführte Einfüge- oder Löschoperation erfordert eine zusätzliche Aktualisierung aller Indizes für diese Tabelle. Dies gilt auch für jede Aktualisierungsoperation, mit der der Wert eines Indexschlüssels geändert wird.
- Jeder relationale Index stellt einen anderen potenziellen Zugriffsplan dar, den das Optimierungsprogramm berücksichtigen muss. Dadurch erhöht sich der Zeitaufwand für die Abfragekompilierung.

Tipps zur Planung von relationalen Indizes

Ein geeignet definierter Index kann Abfragen den Zugriff auf relationale Daten erleichtern.

Mithilfe des Designadvisors (Befehl **db2advis**) können Sie die besten Indizes für eine bestimmte Abfrage oder eine Gruppe von Abfragen, die eine Auslastung definiert, ermitteln. Dieses Tool kann Empfehlungen zur Leistungsverbesserung geben. Es kann zum Beispiel INCLUDE-Spalten oder Indizes empfehlen, die für Rückwärtssuchläufe eingerichtet sind.

Die folgenden Richtlinien können Ihnen ebenfalls helfen, nützliche relationale Indizes zu erstellen:

- Effizientes Abrufen von Daten
  - Zur Verbesserung des Datenabrufs fügen Sie eindeutigen Indizes *INCLUDE-Spalten* hinzu. Zu diesem Zweck bieten sich Spalten mit folgenden Eigenschaften an:
    - Auf sie wird häufig zugegriffen und die Leistung ließe sich durch einen reinen Indexzugriff verbessern.
    - Sie sind zur Begrenzung des Bereichs einer Indexsuche nicht erforderlich.
    - Sie haben keinen Einfluss auf die Reihenfolge oder die Eindeutigkeit des Indexschlüssels.

    Beispiel:

    ```
    create unique index idx on employee (workdept) include (lastname)
    ```

    Die Angabe von LASTNAME als INCLUDE-Spalte und nicht als Teil des Indexschlüssels bedeutet, dass LASTNAME nur auf den Blattseiten des Index gespeichert wird.
  - Erstellen Sie relationale Indizes für Spalten, die in WHERE-Klauseln häufig ausgeführter Abfragen verwendet werden.

    Im folgenden Beispiel lässt sich für die WHERE-Klausel wahrscheinlich ein Vorteil durch einen Index für die Spalte WORKDEPT erzielen, sofern die Spalte WORKDEPT nicht zahlreiche doppelten Werte enthält.

    ```
    where workdept='A01' or workdept='E21'
    ```
  - Erstellen Sie relationale Indizes 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 Abfrage:

    ```
    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 Abfrage möglicherweise effizienter verarbeitet werden, indem nur der Index und nicht die gesamte Tabelle

durchsucht wird. Da das Vergleichselement auf die Spalte WORKDEPT verweist, sollte diese Spalte die erste Schlüsselspalte des relationalen Index sein.

- Effizientes Durchsuchen von Tabellen

  Treffen Sie eine Entscheidung zwischen aufsteigender und absteigender Reihenfolge der Schlüssel, je nachdem, welche Reihenfolge am häufigsten verwendet wird. Obwohl die Werte in umgekehrter Richtung gesucht werden können, wenn Sie die Option ALLOW REVERSE SCANS in der Anweisung CREATE INDEX angeben, zeigen Suchoperationen in der angegebenen Indexreihenfolge eine etwas bessere Leistung als umgekehrte Suchen.

- Effizientes Zugreifen auf größere Tabellen

  Verwenden Sie relationale Indizes, um häufige Abfragen auf Tabellen mit einer größeren Anzahl von Datenseiten, wie in der Spalte NPAGES 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.

- Verbessern der Leistung von Aktualisierungs- und Löschoperationen
  – Erstellen Sie relationale Indizes für Fremdschlüssel, um die Leistung solcher Operationen an einer übergeordneten Tabelle zu verbessern.
  – Erstellen Sie zur Verbesserung der Leistung solcher Operationen an MQTs (Materialized Query Tables), die mit REFRESH IMMEDIATE und INCREMENTAL definiert sind, eindeutige relationale Indizes für den implizierten eindeutigen Schlüssel der MQT, der sich aus den Spalten in der GROUP BY-Klausel der MQT-Definition zusammensetzt.

- Verbessern der Joinleistung

  Wenn Sie mehr als eine Auswahlmöglichkeit für die erste Schlüsselspalte in einem mehrspaltigen relationalen Index haben, verwenden Sie die Spalte, die am häufigsten in einem Equijoin-Vergleichselement (*ausdruck1 = ausdruck2*) angegeben wird, oder die Spalte mit der größten Anzahl unterschiedlicher Werte als erste Schlüsselspalte.

- Sortieren
  – Erstellen Sie relationale Indizes für Spalten, die häufig zum Sortieren der relationalen Daten verwendet werden, um schnelle Sortieroperationen zu ermöglichen.
  – Definieren Sie zur Vermeidung einiger Sortieroperationen mithilfe der Anweisung CREATE INDEX an allen möglichen Stellen Primärschlüssel und eindeutige Schlüssel.
  – Erstellen Sie einen relationalen Index, um die Zeilen in der Reihenfolge zu sortieren, die für eine häufig ausgeführte Abfrage erforderlich ist. Die Sortierung ist für die Klauseln DISTINCT, GROUP BY und ORDER BY erforderlich.

    Im folgenden Beispiel wird die Klausel DISTINCT verwendet:

    ```
    select distinct workdept
      from employee
    ```

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

    ```
    select workdept, average(salary)
      from employee
      group by workdept
    ```

- Aufrechterhalten des Clusterings neu eingefügter Zeilen und Vermeiden von Seitenteilungen

  Definieren Sie einen Clusterindex. Dadurch wird der Bedarf an Reorganisationen der Tabelle meist erheblich verringert. Verwenden Sie die Option PCTFREE in der Anweisung CREATE TABLE, um anzugeben, wie viel freier Speicherplatz auf jeder Seite behalten werden soll, damit Zeilen geeignet eingefügt werden können. Sie können auch den Dateitypmodifikator `pagefreespace` im Befehl **LOAD** angeben.
- Einsparen von Indexpflegeaufwand und Speicherplatz
  - Vermeiden Sie die Erstellung von Indizes, deren Schlüssel Teilschlüssel von anderen vorhandenen Indizes sind. Wenn zum Beispiel ein Index für die Spalten A, B und C vorhanden ist, ist ein weiterer Index für die Spalten A und B im Allgemeinen nicht von Nutzen.
  - Erstellen Sie nicht willkürlich Indizes für viele Spalten. Unnötige Indizes verschwenden nicht nur Speicherplatz, sondern haben zudem längere Vorbereitungszeiten (PREPARE) zur Folge.
    - Erstellen Sie für OLTP-Umgebungen (OLTP, Onlinetransaktionsverarbeitung) nur einen oder zwei Indizes pro Tabelle.
    - Für Umgebungen mit reinen Leseabfragen können Sie mehr als fünf Indizes pro Tabelle erstellen.
    - Für gemischte Abfrage- und OLTP-Umgebungen eignen sich wahrscheinlich zwischen zwei und fünf Indizes pro Tabelle.
- Aktivieren der Online-Indexdefragmentierung

  Verwenden Sie die Option MINPCTUSED, wenn Sie relationale Indizes erstellen. Die Option MINPCTUSED aktiviert die Online-Indexdefragmentierung. Sie gibt die Mindestgröße von Speicherplatz an, der auf einer Indexblattseite belegt sein muss.

Tipps zur Leistung von relationalen Indizes

Durch eine Reihe von Maßnahmen können Sie sicherstellen, dass Ihre relationalen Indizes eine gute Leistung erzielen.

- Geben Sie einen großen Dienstprogrammzwischenspeicher an.

  Wenn Sie ein hohes Aktualisierungsaufkommen an der Tabelle erwarten, für die ein relationaler Index erstellt oder reorganisiert wird, ziehen Sie die Konfiguration eines großen Dienstprogrammzwischenspeichers (Datenbankkonfigurationsparameter **util_heap_sz**) in Betracht, um die Geschwindigkeit dieser Operationen zu erhöhen.
- Zur Vermeidung von Sortierüberläufen in einer symmetrischen Multiprozessorumgebung (SMP-Umgebung) erhöhen Sie den Wert des Konfigurationsparameters **sheapthres** des Datenbankmanagers.
- Erstellen Sie separate Tabellenbereiche für relationale Indizes.

  Sie können Indextabellenbereiche auf schnelleren physischen Einheiten erstellen oder Indextabellenbereiche einem anderen Pufferpool zuweisen, wodurch die Indexseiten vielleicht länger im Puffer verbleiben, weil sie nicht mit Datenseiten konkurrieren.

  Wenn Sie einen anderen Tabellenbereich für Indizes verwenden, können Sie die Konfiguration dieses Tabellenbereichs für Indizes optimieren. Da Indizes in der Regel kleiner als Tabellen sind und sich über weniger Container erstrecken, haben Indizes in der Regel auch kleinere Speicherbereichsgrößen (EXTENTSIZE). Das Abfrageoptimierungsprogramm beachtet bei der Auswahl eines Zugriffsplans die Geschwindigkeit der Einheit, die einen Tabellenbereich enthält.
- Stellen Sie einen hohen Grad der Clusterbildung sicher.

Wenn für Ihre SQL-Anweisung eine Sortierung des Ergebnisses erforderlich ist (z. B. wenn sie eine Klausel ORDER BY, GROUP BY oder DISTINCT enthält), wählt das Optimierungsprogramm einen verfügbaren Index in folgenden Fällen möglicherweise nicht aus:

- Wenn der Grad der Index-Clusterbildung gering ist. Zum Abrufen von Informationen über den Grad der Clusterbildung in einem bestimmten Index führen Sie eine Abfrage auf die Spalten CLUSTERRATIO und CLUSTERFACTOR der Katalogsicht SYSCAT.INDEXES aus.
- 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.

Ein Clusterindex versucht, eine bestimmte Reihenfolge der Daten beizubehalten, wodurch die vom Dienstprogramm RUNSTATS gesammelten statistischen Werte für CLUSTERRATIO bzw. CLUSTERFACTOR verbessert werden. Führen Sie nach der Erstellung eines Clusterindex eine Offlinetabellenreorganisation aus. Im Allgemeinen ist die Clusterbildung in einer Tabelle nur einem Index entsprechend möglich. Erstellen Sie weitere Indizes, nachdem Sie den Clusterindex erstellt haben.

Der PCTFREE-Wert einer Tabelle legt die Größe des Speicherplatzes auf einer Seite fest, der für zukünftige Dateneinfügungen freigehalten wird, sodass diese Daten entsprechend dem Clustering geeignet eingefügt werden können. Wenn Sie keinen PCTFREE-Wert für eine Tabelle angeben, wird der gesamte freie Speicherplatz durch eine Reorganisation beseitigt.

Außer bei Bereichsclustertabellen wird das Datenclustering bei UPDATE-Operationen nicht beibehalten. Das heißt, wenn Sie einen Datensatz aktualisieren, sodass sich der Schlüsselwert im Clusterindex ändert, wird dieser Datensatz nicht unbedingt auf eine entsprechend andere Seite versetzt, um die Clusterreihenfolge beizubehalten. Wenn das Clustering beibehalten werden soll, löschen Sie den Datensatz und fügen anschließend eine aktualisierte Version des Datensatzes ein, anstatt eine UPDATE-Operation zu verwenden.

- Halten Sie Tabellen- und Indexstatistiken auf aktuellem Stand.

  Führen Sie nach der Erstellung eines neuen relationalen Index das Dienstprogramm RUNSTATS aus, um Indexstatistikdaten zu sammeln. Diese Statistikdaten unterstützen das Optimierungsprogramm bei der Bestimmung, ob durch die Verwendung des Index die Datenzugriffsleistung verbessert werden kann.

- Aktivieren Sie die Online-Indexdefragmentierung.

  Die Online-Indexdefragmentierung ist aktiviert, wenn für den relationalen Index die Option MINPCTUSED auf einen Wert größer null gesetzt ist. Die Online-Indexdefragmentierung ermöglicht eine Komprimierung von Indizes durch Zusammenfügen von Blattseiten, wenn der freie Speicherplatz auf einer Seite den angegebenen MINPCTUSED-Wert unterschreitet.

- Reorganisieren Sie relationale 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 können, 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.

  Zur Vermeidung häufiger Reorganisationen geben Sie einen geeigneten Wert für PCTFREE in der Anweisung CREATE INDEX an, um ausreichend Speicherplatz auf jeder Indexblattseite, wenn sie erstellt wird, freizuhalten. So ist bei zukünftigen Aktivitäten, durch die Datensätze in den Index eingefügt werden, die Wahr-

scheinlichkeit von Indexseitenteilungen geringer. Seitenteilungen beeinträchtigen die richtige Abfolge von Seiten und setzen dementsprechend die Effizienz des Vorablesezugriffs auf Indexseiten herab. Der PCTFREE-Wert, den Sie bei der Erstellung eines relationalen Index angegeben haben, wird beibehalten, wenn der Index reorganisiert wird.

- Analysieren Sie EXPLAIN-Informationen zur Nutzung von relationalen Indizes.

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

  Anhand von EXPLAIN-Informationen können Sie auch feststellen, ob eine große Tabelle, die durchsucht wird, als innere Tabellen bei Joins mit Verschachtelungsschleife verarbeitet wird. Dies würde darauf hinweisen, dass ein Index für die Spalte des Joinvergleichselements entweder fehlt oder als ineffektiv für die Anwendung des Joinvergleichselements betrachtet wird.

- Deklarieren Sie Tabellen, deren Größe stark variiert, als flüchtig („VOLATILE").

  Eine *flüchtige Tabelle* ist eine Tabelle, deren Kardinalität während der Ausführung stark variieren kann. Für diese Art von Tabelle kann das Optimierungsprogramm einen Zugriffsplan generieren, der eine Tabellensuche einer Indexsuche vorzieht.

  Verwenden Sie die Anweisung ALTER TABLE mit der Klausel VOLATILE, um eine solche Tabelle als flüchtig zu deklarieren. In den folgenden Fällen verwendet das Optimierungsprogramm unabhängig von den statistischen Daten eine Indexsuche anstelle einer Tabellensuche für solche Tabellen:

  – Alle Spalten, auf die verwiesen wird, sind im Index enthalten.
  – Der Index kann bei der Indexsuche ein Vergleichselement anwenden.

  Bei typisierten Tabellen wird die Anweisung ALTER TABLE...VOLATILE nur für die Stammtabelle einer Hierarchie von typisierten Tabellen unterstützt.

Partitionierung und Clustering

Indexverhalten bei partitionierten Tabellen

Indizes für partitionierte Tabellen verhalten sich ähnlich wie Indizes für nicht partitionierte Tabellen, sie werden jedoch mit einem anderen Speichermodell gespeichert, je nachdem, ob es sich bei ihnen um partitionierte oder nicht partitionierte Indizes handelt.

Während sich die Indizes für reguläre nicht partitionierte Tabellen in einem gemeinsam genutzten Indexobjekt befinden, werden *nicht partitionierte Indizes* für partitionierte Tabellen in einem eigenen Indexobjekt und in einem einzelnen Tabellenbereich erstellt, auch wenn die Datenbankpartitionen mehrere Tabellenbereiche umfassen. Sowohl vom Datenbankmanager verwaltete Tabellenbereiche (DMS-Tabellenbereiche) als auch vom System verwaltete Tabellenbereiche (SMS-Tabellenbereiche) unterstützen die Verwendung von Indizes, die sich an einer anderen Position als die Tabellendaten befinden. Jeder nicht partitionierte Index kann in einem eigenen Tabellenbereich, auch in großen Tabellenbereichen (LARGE), gespeichert 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. Alle Tabellenbereiche müssen sich in derselben Datenbankpartitionsgruppe befinden.

Ein *partitionierter Index* verwendet ein Indexorganisationsschema, bei dem Indexdaten entsprechend dem Partitionierungsschema der Tabelle auf mehrere

Indexpartitionen verteilt werden. Jede Indexpartition bezieht sich ausschließlich auf Tabellenzeilen in der entsprechenden Datenpartition. Alle Indexpartitionen für eine bestimmte Datenpartition befinden sich in demselben Indexobjekt.

Ab DB2 Version 9.7 Fixpack 1 können Indizes zu XML-Daten für XML-Spalten in partitionierten Tabellen partitioniert oder nicht partitioniert sein. Standardmäßig werden partitionierte Indizes erstellt. Vom System generierte Indizes zu XML-Regionen sind immer partitioniert, während vom System generierte Indizes zu Spaltenpfaden immer nicht partitioniert sind. In DB2 V9.7 sind Indizes zu XML-Daten nicht partitioniert.

Nicht partitionierte Indizes zeichnen sich durch folgende Vorteile aus:

- Es besteht die Möglichkeit, verschiedene Tabellenbereichsmerkmale für jeden Index zu definieren (z. B. können verschiedene Seitengrößen zu einer besseren Speicherplatznutzung beitragen).
- Indizes können unabhängig voneinander reorganisiert werden.
- Beim Löschen von Indizes wird eine bessere Leistung erzielt.
- Weniger E/A-Konkurrenzsituationen unterstützen einen effizienteren gemeinsamen Zugriff auf die Indexdaten.
- Wenn einzelne Indizes gelöscht werden, wird der Speicherplatz sofort für das System verfügbar, ohne dass eine Indexreorganisation erforderlich ist.

Partitionierte Indizes zeichnen sich durch folgende Vorteile aus:

- Bei der Ein- und Auslagerung von Daten (Rollin/Rollout) wird eine bessere Leistung erzielt.
- Durch die Partitionierung der Indizes entstehen weniger Konkurrenzsituationen bei Indexseiten.
- Es gibt eine B-Indexbaumstruktur für jede einzelne Indexpartition, die folgende Vorteile bieten kann:
  - Beim Einfügen, Aktualisieren, Löschen und Suchen wird eine bessere Leistung erzielt, da die B-Baumstruktur für eine Indexpartition im Allgemeinen weniger Ebenen enthält als ein Index, der auf alle Daten in der Tabelle verweist.
  - Beim Suchen wird eine bessere Leistung erzielt und es treten weniger Konkurrenzsituationen auf, wenn der Partitionsausschluss aktiviert ist. Partitionen können zwar sowohl beim Durchsuchen von partitionierten als auch von nicht partitionierten Indizes ausgeschlossen werden, ein Partitionsausschluss ist jedoch beim Suchen in partitionierten Indizes effektiver, da jede einzelne Indexpartition Schlüssel enthält, die sich ausschließlich auf die zugehörige Datenpartition beziehen. Dies kann dazu führen, dass weniger Schlüssel und weniger Indexseiten durchsucht werden müssen, als es bei einer entsprechenden Abfrage für einen nicht partitionierten Index erforderlich wäre.

Während ein nicht partitionierter Index die Reihenfolge der Indexspalten immer beibehält, kann eine über mehrere Partitionen zu wahrende Reihenfolge bei einem partitionierten Index in bestimmten Szenarien ein Stück weit verloren gehen. Dies ist zum Beispiel der Fall, wenn die Partitionierungsspalten nicht den Indexspalten entsprechen oder wenn ein Zugriff auf mehrere Partitionen erforderlich ist.

Bei der Onlineindexerstellung werden gleichzeitige Lese- und Schreibzugriffe auf die Tabelle zugelassen. Nach dem Erstellen eines derartigen Indexes werden Änderungen, die während der Indexerstellung an der jeweiligen Tabelle vorgenommen wurden, auf den neuen Index angewendet. Der Schreibzugriff auf die Tabelle wird blockiert, bis die Indexerstellung abgeschlossen ist und die Transaktion festge-

schrieben wurde. Bei partitionierten Indizes wird jede Datenpartition *nur* in den Quiescemodus versetzt (sodass nur ein Lesezugriff möglich ist), während an der Datenpartition vorgenommene Änderungen (beim Erstellen der Indexpartition) angewendet werden.

Die Unterstützung für partitionierte Indizes ist insbesondere dann hilfreich, wenn Daten mit der Anweisung ALTER TABLE...ATTACH PARTITION eingelagert werden (Rollin). Führen Sie nach dem Zuordnen einer Partition die Anweisung SET INTEGRITY aus, wenn nicht partitionierte Indizes vorhanden sind (ausgenommen XML-Spaltenpfadindizes, wenn die Tabelle XML-Daten enthält). Dies ist bei nicht partitionierten Indizes für die Verwaltung, die Bereichsprüfung, die Prüfung von Integritätsbedingungen und die MQT-Verwaltung erforderlich. Die Verwaltung nicht partitionierter Indizes kann sehr aufwendig sein und sehr viel Protokollspeicherplatz belegen. Umgehen Sie diesen Verwaltungsaufwand, indem Sie partitionierte Indizes verwenden.

In Abb. 12 sind zwei nicht partitionierte Indizes für eine partitionierte Tabelle zu sehen. Jeder Index befindet sich in einem separaten Tabellenbereich.

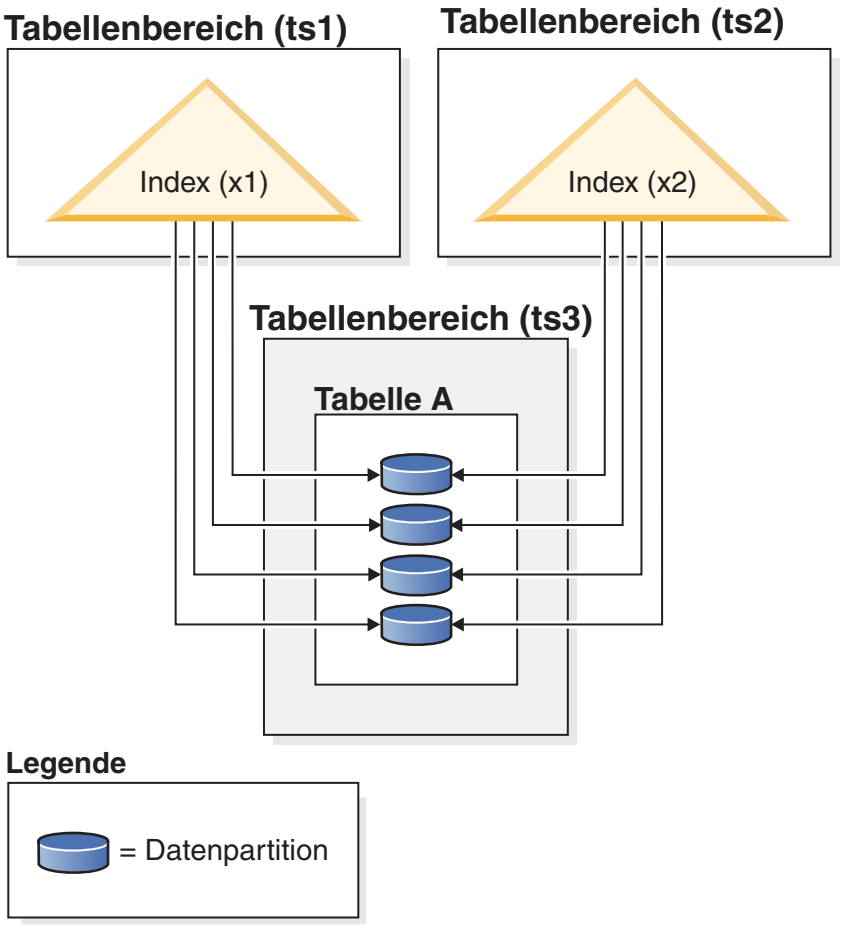


Abbildung 12. Nicht partitionierte Indizes für eine partitionierte Tabelle

In Abb. 13 auf Seite 86 ist ein nicht partitionierter Index für eine partitionierte Tabelle zu sehen, die zwei Datenbankpartitionen umfasst und sich in einem einzigen Tabellenbereich befindet.

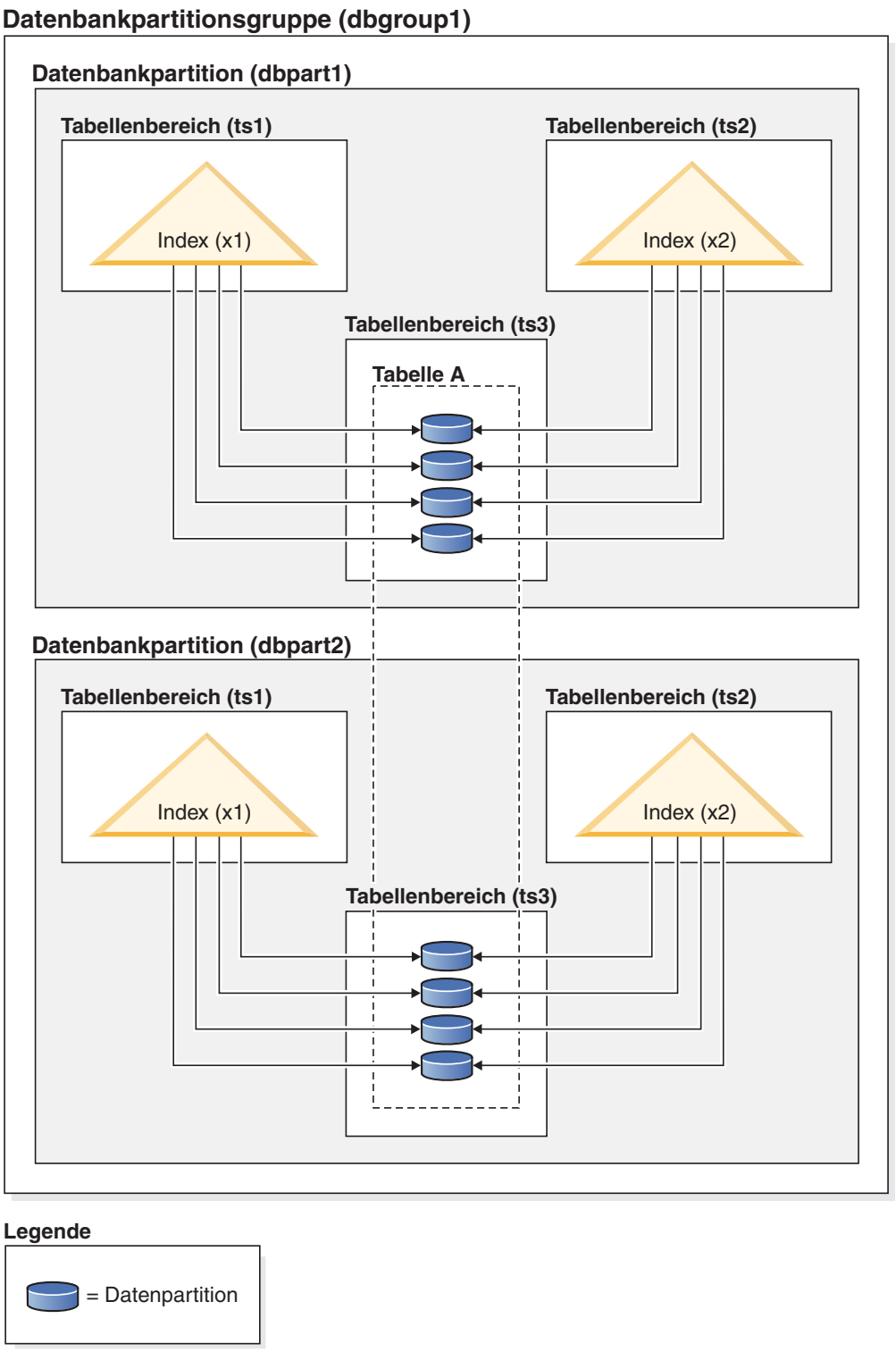


Abbildung 13. Nicht partitionierter Index für eine verteilte und partitionierte Tabelle

Abb. 14 auf Seite 87 zeigt eine Kombination aus partitionierten und nicht partitionierten Indizes für eine partitionierte Tabelle

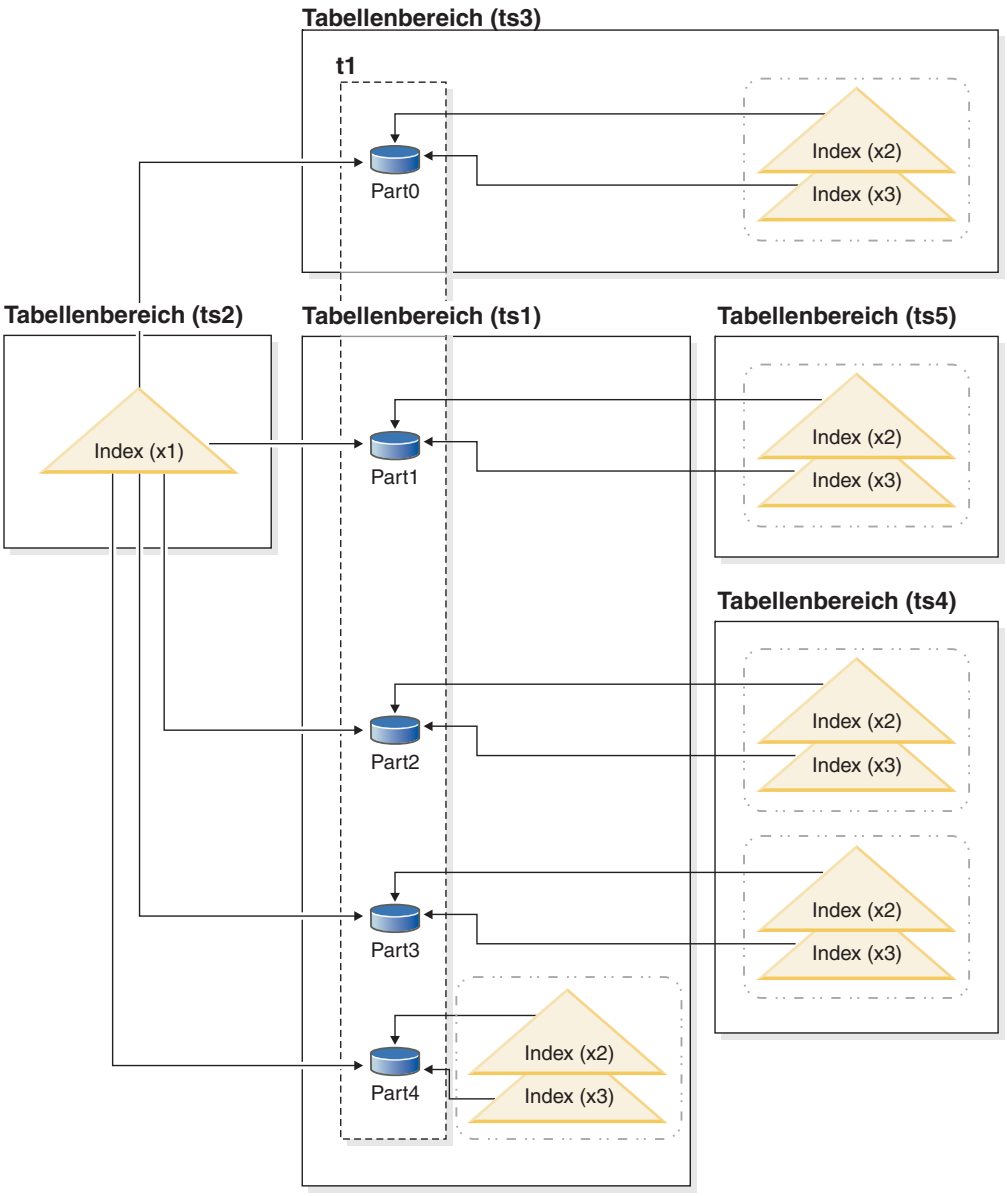


Abbildung 14. Partitionierter Index und nicht partitionierte Indizes für eine partitionierte Tabelle

Der nicht partitionierte Index X1 bezieht sich auf Zeilen in allen Datenpartitionen. Die partitionierten Indizes X2 und X3 beziehen sich dagegen nur auf Zeilen in der Datenpartition, der sie zugeordnet sind. Bei Tabellenbereich TS3 ist ebenfalls zu sehen, dass die Indexpartitionen den Tabellenbereich der Datenpartitionen, denen sie zugeordnet sind, gemeinsam nutzen. Dies ist das Standardverhalten bei partitionierten Indizes.

Sie können die Standardposition bei nicht partitionierten und bei partitionierten Indizes überschreiben, müssen dabei jedoch unterschiedlich vorgehen. Bei nicht partitionierten Indizes können Sie beim Erstellen des Index einen Tabellenbereich angeben. Bei partitionierten Indizes müssen Sie beim Erstellen der Tabelle festlegen, in welchen Tabellenbereichen Indexpartitionen gespeichert werden.

Nicht partitionierte Indizes

Die Indexposition für nicht partitionierte Indizes können Sie überschreiben, indem Sie die Anweisung CREATE INDEX mit der Klausel IN verwenden. Dies ermöglicht es Ihnen, eine andere Tabellenbereichsposition für den Index anzugeben. Sie können verschiedene Indizes nach Bedarf in unterschiedlichen Tabellenbereichen speichern. Wenn Sie eine partitionierte Tabelle erstellen, ohne die Speicherposition für die zugehörigen nicht partitionierten Indizes anzugeben, und anschließend einen Index mit einer Anweisung CREATE INDEX erstellen, ohne einen 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. Diese Auswertung zur Bestimmung der Tabellenbereichsposition für den Index wird beendet, wenn ein übereinstimmender Fall gefunden ist.

Fall 1:
Wenn ein Tabellenbereich für den Index in der Anweisung
CREATE INDEX...IN *tabellenbereich*
angegeben ist, wird der angegebene Tabellenbereich für diesen Index verwendet.

Fall 2:
Wenn ein Indextabellenbereich in der Anweisung
CREATE TABLE...INDEX IN *tabellenbereich*
angegeben ist, wird der angegebene Tabellenbereich für diesen Index verwendet.

Fall 3:
Wenn kein Tabellenbereich angegeben ist, wird der Tabellenbereich ausgewählt, der von der ersten zugeordneten oder sichtbaren Datenpartition verwendet wird.

Partitionierte Indizes

Standardmäßig werden Indexpartitionen in demselben Tabellenbereich platziert wie die Datenpartitionen, auf die sie verweisen. Dieses Standardverhalten können Sie überschreiben, indem Sie die Klausel INDEX IN für jede Datenpartition verwenden, die Sie mit der Anweisung CREATE TABLE definieren. Dies bedeutet, dass Sie beim Erstellen einer Tabelle bereits wissen müssen, an welcher Stelle die Indexpartitionen gespeichert werden sollen, wenn Sie partitionierte Indizes für eine partitionierte Tabelle verwenden möchten. Wenn Sie die Klausel INDEX IN beim Erstellen eines partitionierten Indexes verwenden, erhalten Sie eine Fehlernachricht.

Beispiel 1: Gegeben ist die partitionierte Tabelle SALES (a int, b int, c int). Es wird ein eindeutiger Index A_IDX erstellt.

```
create unique index a_idx on sales (a)
```

Da die Tabelle SALES partitioniert ist, wird der Index A_IDX ebenfalls als partitionierter Index erstellt.

Beispiel 2: Erstellen von Index B_IDX.

```
create index b_idx on sales (b)
```

Beispiel 3: Zum Überschreiben der Standardposition für die Indexpartitionen in einem partitionierten Index verwenden Sie die Klausel INDEX IN für jede Partition, die Sie beim Erstellen der partitionierten Tabelle definieren. Im folgenden Beispiel werden Indizes für die Tabelle Z im Tabellenbereich TS3 erstellt.

```
create table z (a int, b int)
      partition by range (a) (starting from (1)
      ending at (100)  index in ts3)

   create index c_idx on z (a) partitioned
```

Clustering bei nicht partitionierten Indizes für partitionierte 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 einen Clusterindex 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. Während einer Clustereinfügeoperation wird eine Indexsuche durchgeführt, um eine passende Satz-ID (RID) ausfindig zu machen. Diese Satz-ID wird als Ausgangspunkt in der Tabelle für die Suche nach einem Platz zum Einfügen des Datensatzes verwendet. Zur Erzielung einer optimalen 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. Nehmen Sie zum Beispiel an, dass eine Tabelle über die Spalte QUARTER (Quartal) partitioniert ist und ein Clusterindex für die Spalte DATE (Datum) definiert ist. Zwischen den Spalten QUARTER und DATE besteht eine Beziehung und es lässt sich ein optimales Clustering der Daten mit guter Leistung erzielen, weil alle Schlüssel einer Datenpartition im Index zusammen gruppiert werden. Abb. 15 auf Seite 90 veranschaulicht, dass sich eine optimale Suchleistung nur erzielen lässt, wenn das Clustering mit dem Tabellenpartitionierungsschlüssel korreliert.

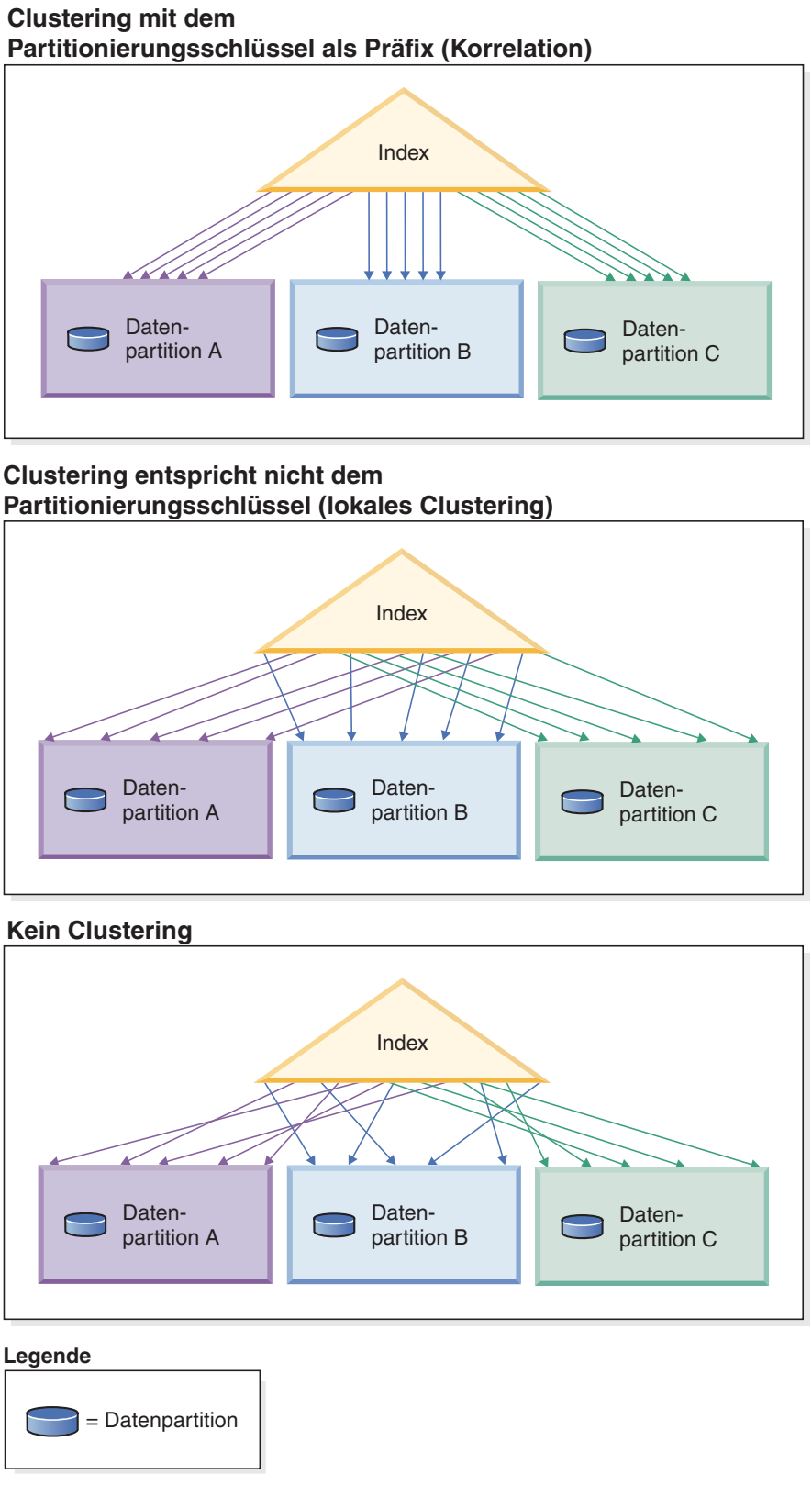


Abbildung 15. Die möglichen Effekte eines Clusterindex für eine partitionierte Tabelle.

Das Clustering bietet folgende Vorteile:

- Die Zeilen sind in jeder Datenpartition in der Reihenfolge des Schlüssels angeordnet.

- Clusterindizes verbessern die Leistung von Suchläufen, welche die Tabelle in der Reihenfolge der Schlüssel durchqueren, weil der Suchvorgang die erste Zeile der ersten Seite abruft, dann jede Zeile auf eben dieser Seite, bevor er mit der nächsten Seite fortfährt. Dies bedeutet, dass sich nur eine Seite der Tabelle zu einem gegebenen Zeitpunkt im Pufferpool befinden muss. Wenn die Tabelle keine Clusterbildung aufweist, werden Zeilen wahrscheinlich von verschiedenen Seiten abgerufen. Wenn der Pufferpool nicht die gesamte Tabelle aufnehmen kann, ist es wahrscheinlich, dass die meisten Seiten mehrmals abgerufen werden und sich die Suche erheblich verlangsamt.

Wenn der Clusterschlüssel jedoch nicht mit dem Tabellenpartitionierungsschlüssel korreliert ist, die Daten jedoch 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. Dies liegt daran, dass jede abgerufene Zeile aus einer bestimmten Datenpartition der Zeile nahe ist, die zuvor aus dieser selben Partition abgerufen wurde (siehe zweites Beispiel in Abb. 15 auf Seite 90).

Föderierte Datenbanken

Serveroptionen mit Einfluss auf föderierte Datenbanken

Ein System föderierter Datenbanken besteht aus einem DB2-Datenserver (der föderierten Datenbank) und mindestens einer Datenquelle. Sie geben die Datenquellen für die föderierte Datenbank an, wenn Sie Anweisungen CREATE SERVER absetzen. Sie können auch Serveroptionen angeben, die verschiedene Aspekte des Betriebs des föderierten Systems optimieren und steuern.

Sie müssen die Installationsoption für den verteilten Join installieren und den Konfigurationsparameter **federated** des Datenbankmanagers auf den Wert YES setzen, bevor Sie Server erstellen und Serveroptionen angeben können. Wenn Sie später Serveroptionen ändern wollen, verwenden Sie die Anweisung ALTER SERVER.

Die Werte der Serveroptionen, die Sie in der Anweisung CREATE SERVER angeben, haben Einfluss auf die Pushdown-Analyse von Abfragen, die globale Optimierung und andere Aspekte von Operationen mit föderierten Datenbanken. Sie können zum Beispiel Leistungsstatistikdaten als Werte für Serveroptionen angeben. Die Option *cpu_ratio* gibt die relativen Geschwindigkeiten der Prozessoren an der Datenquelle und auf dem Server mit föderierten Datenbanken an. Die Option *io_ratio* gibt die relativen Raten der Daten-E/A-Unterteilungen an der Datenquelle und auf dem Server mit föderierten Datenbanken an.

Werte für Serveroptionen werden in den Systemkatalog (SYSCAT.SERVEROPTIONS) geschrieben. Das Optimierungsprogramm verwendet diese Informationen bei der Ermittlung von Zugriffsplänen für die Datenquelle. Wenn sich ein Statistikwert ändert (z. B. wenn ein Datenquellenprozessor aufgerüstet wird), können Sie den Katalog mithilfe der Anweisung ALTER SERVER mit dem neuen Wert aktualisieren.

Ressourcennutzung

Hauptspeicherzuordnung

Die Hauptspeicherzuordnung und die Hauptspeicherfreigabe finden zu verschiedenen Zeiten statt. Hauptspeicher kann einem bestimmten Speicherbereich zugeordnet werden, wenn ein bestimmtes Ereignis auftritt (z. B. die Herstellung einer Ver-

bindung durch eine Anwendung) oder er kann infolge einer geänderten Konfigurationsparametereinstellung neu zugeordnet werden.

In Abb. 16 werden die unterschiedlichen Speicherbereiche gezeigt, die der Datenbankmanager für unterschiedliche Zwecke zuordnet, sowie die Konfigurationsparameter angegeben, mit denen Sie die Größe dieser Speicherbereiche steuern können. Beachten Sie, dass in einer Umgebung mit partitionierten Datenbanken jede Datenbankpartition über einen eigenen Bereich für den gemeinsam genutzten Speicher des Datenbankmanagers verfügt.

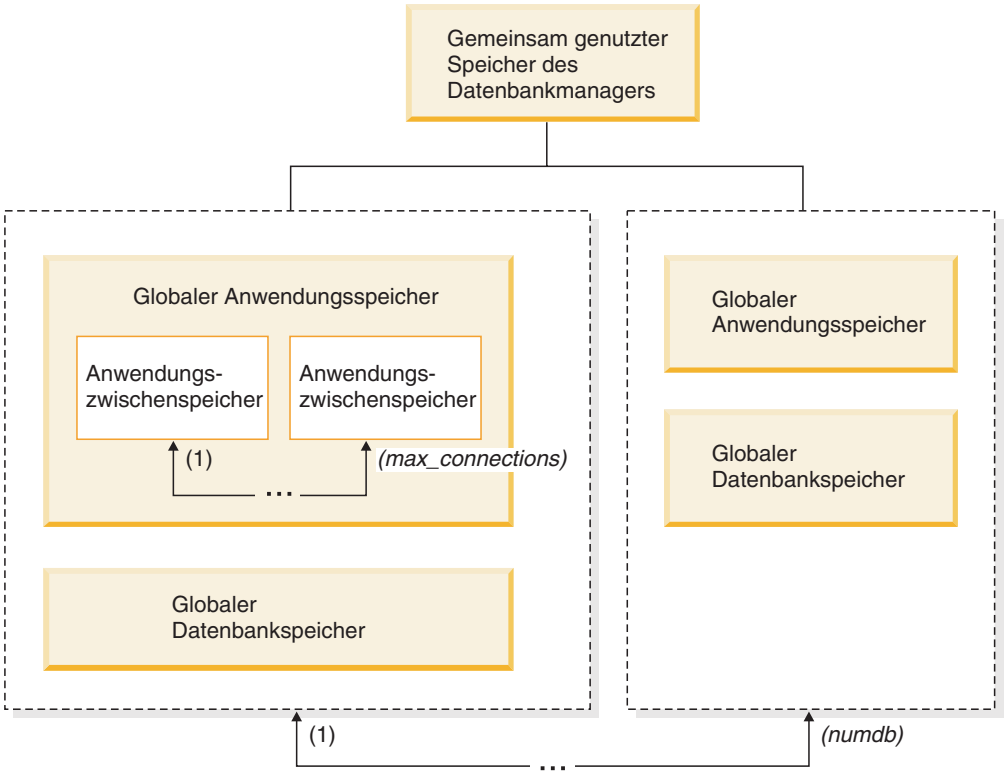


Abbildung 16. Vom Datenbankmanager zugeordnete Typen von Speicher

Speicher wird vom Datenbankmanager immer zugeordnet, wenn eines der folgenden Ereignisse auftritt:

Wenn der Datenbankmanager gestartet wird (`db2start`)
Der *gemeinsam genutzte Speicher des Datenbankmanagers* (auch als *gemeinsam genutzter Instanzspeicher* bezeichnet) 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. Die Größe des gemeinsam genutzten Speichers des Datenbankmanagers wird von DB2 automatisch gesteuert.

Wenn eine Datenbank aktiviert oder zum ersten Mal eine Verbindung zu ihr hergestellt wird
Der *globale Datenbankspeicher* wird für alle Anwendungen verwendet, die eine Verbindung zur Datenbank herstellen. Die Größe des globalen Datenbankspeichers wird durch den Datenbankkonfigurationsparameter **`database_memory`** festgelegt. Standardmäßig hat dieser Parameter den AUTOMATIC, sodass DB2 die Anfangsgröße des Speichers, der der Datenbank

zugeordnet wird, berechnen und die Größe des Datenbankspeichers während der Laufzeit automatisch je nach Bedarf der Datenbank konfigurieren kann.

Die folgenden Hauptspeicherbereiche können dynamisch angepasst werden:

- Pufferpools (unter Verwendung der Anweisung ALTER BUFFERPOOL)
- Datenbankzwischenspeicher (einschließlich Protokollpuffer)
- Zwischenspeicher für Dienstprogramme
- Paketcache
- Katalogcache
- Sperrenliste

Die Konfigurationsparameter **sortheap**, **sheapthres_shr** und **sheapthres** können auch dynamisch aktualisiert werden. Die einzige Einschränkung besteht darin, dass der Parameter **sheapthres** nicht dynamisch von 0 auf einen Wert größer 0 oder umgekehrt geändert werden kann.

Standardmäßig werden gemeinsame Sortieroperationen ausgeführt, und die Größe des gemeinsam genutzten Datenbankspeichers, der von Konsumenten des Sortierspeichers zu einem beliebigen Zeitpunkt verwendet werden kann, wird durch den Wert des Datenbankkonfigurationsparameters **sheapthres_shr** bestimmt. Private Sortieroperationen werden nur ausgeführt, wenn die partitionsinterne Parallelität, die Datenbankpartitionierung und der Verbindungskonzentrator inaktiviert sind und der Konfigurationsparameter **sheapthres** des Datenbankmanagers auf einen Wert ungleich null gesetzt ist.

Wenn eine Anwendung die Verbindung zu einer Datenbank herstellt
Jede Anwendung verfügt über einen eigenen *Anwendungszwischenspeicher*, der Teil des *globalen Anwendungsspeichers* ist. Sie können die Speicherkapazität, die jede einzelne Anwendung zuordnen kann, mithilfe des Datenbankkonfigurationsparameters **applheapsz** begrenzen. Sie können auch die Gesamtkapazität des Anwendungsspeichers mithilfe des Datenbankkonfigurationsparameters **appl_memory** begrenzen.

Wenn ein Agent erstellt wird
Der *private Agentenspeicher* wird für einen Agenten zugeordnet, wenn der Agent infolge einer Verbindungsanforderung oder einer neuen SQL-Anforderung in einer Umgebung mit partitionierten Datenbanken zugeordnet wird. Der private Agentenspeicher enthält Speicher, der nur von diesem speziellen Agenten verwendet wird. Wenn private Sortieroperationen aktiviert wurden, wird der private Sortierspeicher aus dem privaten Agentenspeicher zugeordnet.

Die folgenden Konfigurationsparameter begrenzen die Speicherkapazität, die für die einzelnen Typen von Speicherbereichen zugeordnet wird. Beachten Sie, dass dieser Speicher in einer Umgebung mit partitionierten Datenbanken in jeder Datenbankpartition zugeordnet wird.

numdb Dieser Konfigurationsparameter des Datenbankmanagers 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 zugeordnet werden kann, wenn Sie den Wert dieses Parameters erhöhen.

maxappls
Dieser Datenbankkonfigurationsparameter definiert die maximale Anzahl

von Anwendungen, die gleichzeitig eine Verbindung zu einer bestimmten Datenbank herstellen können. Der Wert dieses Parameters wirkt sich auf die Menge an Speicher aus, die sowohl für privaten Agentenspeicher als auch für den globalen Anwendungsspeicher der betreffenden Datenbank zugeordnet werden kann.

max_connections
Dieser Konfigurationsparameter des Datenbankmanagers begrenzt die Anzahl von Datenbankverbindungen (CONNECT) oder Instanzverbindungen (ATTACH), die auf den Datenserver zu einem gegebenen Zeitpunkt gleichzeitig zugreifen können.

max_coordagents
Dieser Konfigurationsparameter des Datenbankmanagers begrenzt die Anzahl von koordinierenden Agenten des Datenbankmanagers, die gleichzeitig für alle aktiven Datenbanken in einer Instanz (und pro Datenbankpartition in einer Umgebung mit partitionierten Datenbanken) vorhanden sein können. Zusammen mit den Parametern **maxappls** und **max_connections** begrenzt dieser 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 aktuelle Speicherzuordnung in der Instanz prüfen. Darüber hinaus können Sie mithilfe der Tabellenfunktion ADMIN_GET_DBP_MEM_USAGE die Gesamtspeicherbelegung für die gesamte Instanz oder nur für eine einzelne Datenbankpartition ermitteln. Mithilfe des Befehls **GET SNAPSHOT** können Sie die aktuelle Speicherbelegung auf Instanz-, Datenbank- oder Anwendungsebene untersuchen.

Unter UNIX und Linux können mit dem Befehl **ipcs** alle gemeinsam genutzten Speichersegmente aufgelistet werden; die Menge der genutzten Ressourcen wird jedoch nicht genau wiedergegeben. Alternativ zum Befehl **ipcs** kann der Befehl **db2mtrk** verwendet werden.

Speichergruppen - Übersicht

Der DB2-Speichermanager organisiert Speicherzuordnungen des Betriebssystems in *Speichergruppen*.

Der Speicher innerhalb einer bestimmten Speichergruppe weist gemeinsame Attribute auf, wie zum Beispiel den allgemeinen Verwendungszweck, die erwartete Flüchtigkeit und die Einschränkungen - falls vorhanden - hinsichtlich des Speicherwachstums. Pufferpools werden beispielsweise aus der Datenbankspeichergruppe zugeordnet und bleiben zugeordnet, solange die Datenbank aktiv ist. Anweisungszwischenspeicher werden aus der Anwendungsspeichergruppe für bestimmte SQL-Vorbereitungsanweisungen einer Anwendung zugeordnet und bleiben nur zugeordnet, solange die Anweisungskompilierungsoperation dauert.

Innerhalb jeder Speichergruppe werden bestimmte Speicherbereiche für Verwendungszwecke zugeordnet, die im Allgemeinen dem jeweiligen Speichergruppentyp entsprechen. So verwenden zum Beispiel bestimmte Verarbeitungsarten auf Datenbankebene Speicher aus der Datenbankspeichergruppe; für den Fast Communication Manager benötigter Speicher wird aus der FCM-Speichergruppe zugeordnet.

In Tabelle 1 auf Seite 95 sind die verschiedenen Speichergruppentypen aufgeführt.

Tabelle 1. Speichergruppen

Speichergruppentyp*	Beschreibung	Aus dieser Gruppe zugeordneter Speicherbereich
DBMS	Gruppe des Datenbankspeichermanagers. Der Großteil des Speichers in dieser Gruppe wird für grundlegende Infrastrukturoperationen verwendet, wie unter anderem Kommunikationsservices, die nicht für eine bestimmte Datenbank spezifisch sind. Für diese Speichergruppe ist keine Konfiguration erforderlich; der benutzerkonfigurierbare Speicher aus dieser Gruppe umfasst jedoch die Größe des Überwachungszwischenspeichers (**mon_heap_sz**) sowie die Größe des Prüfpuffers (**audit_buf_sz**).	Instanz
FMP	Speichergruppe für Prozesse im abgeschirmten Modus. Der aus dieser Gruppe zugeordnete Speicher wird für die Kommunikation zwischen Agenten und Prozessen im abgeschirmten Modus verwendet. Die Speicherzuordnungen aus dieser Gruppe können mit der Registrierdatenbankvariablen DB2_FMP_COMM_HEAPSZ und dem Konfigurationsparameter **aslheapsz** konfiguriert werden.	Instanz
PRIVATE	Der aus dieser Gruppe zugeordnete Speicher wird für allgemeine Zwecke verwendet, wie zum Beispiel die grundlegende Infrastruktur- und Diagnoseunterstützung. Mit Ausnahme der Systeme, die das Modell der privaten Sortierung verwenden, bei dem der Konfigurationsparameter **sheapthres** auf einen Wert ungleich null gesetzt wird, muss für den privaten Speicher keine Konfiguration vorgenommen werden.	Instanz
DATABASE	Datenbankspeichergruppe. Der aus dieser Gruppe zugeordnete Speicher wird in der Regel für die Verarbeitung verwendet, die für eine einzelne Datenbank, nicht jedoch für eine bestimmte Anwendung spezifisch ist. Beispiele für Speicher, der aus dieser Gruppe zugeordnet wird, sind Pufferpools, Datenbankzwischenspeicher, Sperrenlisten, Zwischenspeicher für Dienstprogramme, Paketcache, Katalogcache und gemeinsamer Sortierspeicher. Diese Gruppe kann über den Datenbankkonfigurationsparameter **database_memory** konfiguriert werden. Darüber hinaus kann dieser Speicherbereich auch mithilfe des Managers für Speicher mit automatischer Leistungsoptimierung (Self-Tuning Memory Manager, STMM) optimiert werden.	Datenbank

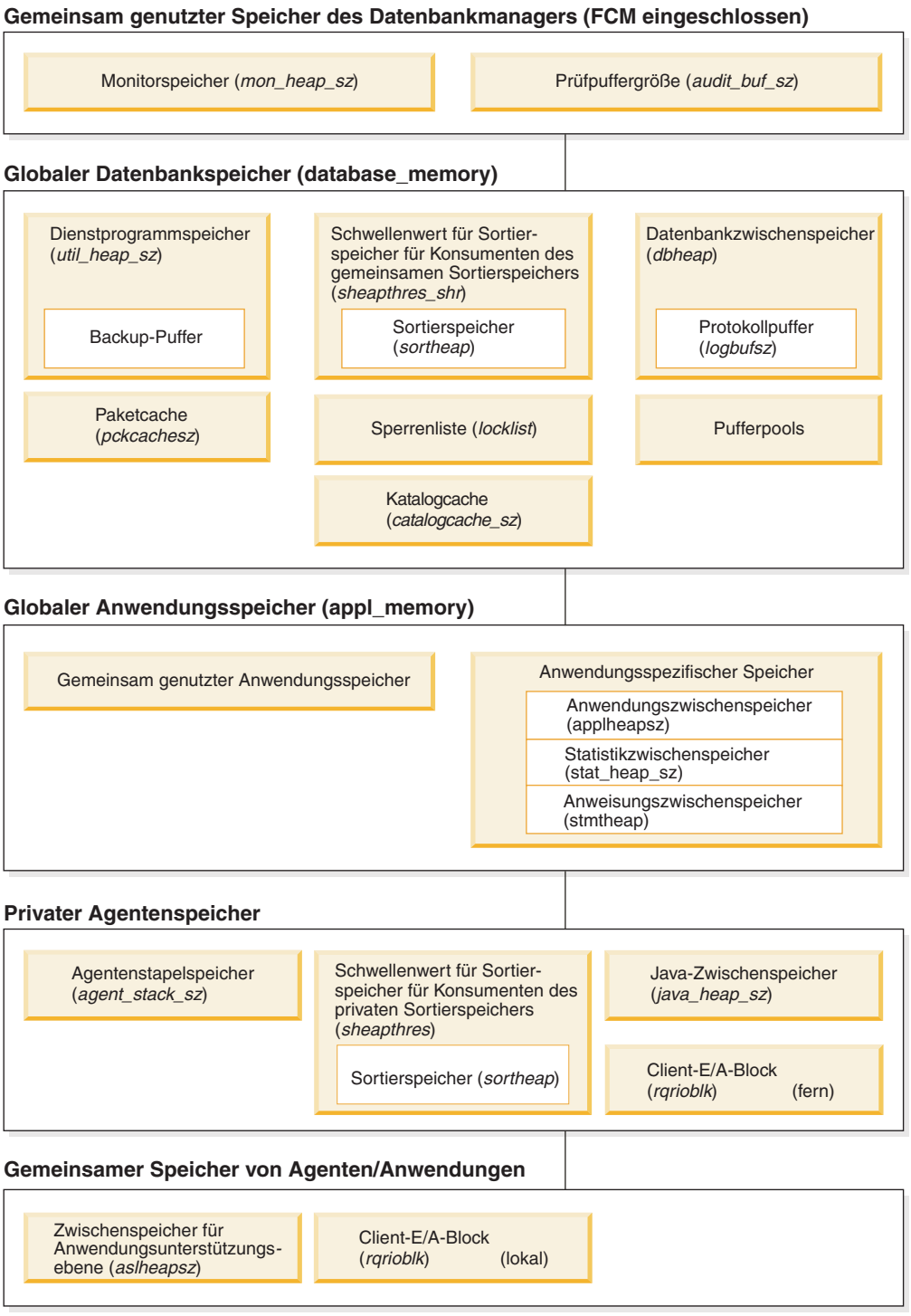
Tabelle 1. Speichergruppen (Forts.)

Speichergruppentyp*	Beschreibung	Aus dieser Gruppe zugeordneter Speicherbereich
APPLICATION	Anwendungsspeichergruppe. Der aus dieser Gruppe zugeordnete Speicher wird in der Regel für die anwendungsspezifische Verarbeitung verwendet. Er umfasst beispielsweise Anwendungs-, Statistik- und Anweisungszwischenspeicher sowie einen nicht konfigurierbaren gemeinsamen Arbeitsbereich. Diese Gruppe kann über den Datenbankkonfigurationsparameter **appl_memory** konfiguriert werden.	Datenbank
FCM	Speichergruppe des Fast Communication Manager. Der aus dieser Gruppe zugeordnete Speicher wird ausschließlich durch den Fast Communication Manager genutzt. Er kann mithilfe der Konfigurationsparameter **fcm_num_buffers** und **fcm_num_channels** des Datenbankmanagers konfiguriert werden.	Host
* Die in der ersten Spalte angezeigten Namen sind die Namen, die für das Monitorelement **memory_set_type** zurückgegeben werden.		

Gemeinsam genutzter Speicher des Datenbankmanagers

Der gemeinsam genutzte Speicher des Datenbankmanagers wird in vielen verschiedenen Speicherbereichen verwaltet. Mit Konfigurationsparametern können Sie die Größen der verschiedenen Speicherbereiche steuern.

Abb. 17 auf Seite 97 zeigt, wie der gemeinsam genutzte Speicher des Datenbankmanagers zugeordnet wird.



Hinweis: Die Rahmengröße gibt keine relative Speichergröße an.

Abbildung 17. Verwendung von Speicher durch den Datenbankmanager

Gemeinsam genutzter Speicher des Datenbankmanagers

Der gemeinsam genutzte Speicher des Datenbankmanagers wird durch folgende Konfigurationsparameter beeinflusst:

- Der Konfigurationsparameter **audit_buf_sz** legt die Größe des für Datenbankprüfungsaktivitäten verwendeten Puffers fest.

- Der Konfigurationsparameter **mon_heap_sz** legt die Größe des für Datenbanksystemmonitordaten verwendeten Speicherbereichs fest.
- Bei partitionierten Datenbanksystemen benötigt FCM einen beträchtlichen Speicherbereich, insbesondere wenn der Wert des Parameters **fcm_num_buffers** hoch eingestellt ist. Der FCM-Speicherbereich wird aus dem FCM-Pufferpool zugeordnet.

Globaler Datenbankspeicher

Der globale Datenbankspeicher wird durch die Größe der Pufferpools sowie durch folgende Datenbankkonfigurationsparameter beeinflusst:

- **catalogcache_sz**
- **database_memory**
- **dbheap**
- **locklist**
- **pckcachesz**
- **sheapthres_shr**
- **util_heap_sz**

Globaler Anwendungsspeicher

Der globale Anwendungsspeicher kann mit dem Konfigurationsparameter **appl_memory** gesteuert werden. Die folgenden Datenbankkonfigurationsparameter können zur Begrenzung des Speichers verwendet werden, der von einer beliebigen einzelnen Anwendung belegt werden kann:

- **applheapsz**
- **stat_heap_sz**
- **stmtheap**

Privater Agentenspeicher

Jeder Agent benötigt einen eigenen privaten Speicherbereich. Der Datenserver erstellt die erforderliche Anzahl von Agenten entsprechend den konfigurierten Speicherressourcen. Sie können die maximale Anzahl von Koordinatoragenten über den Konfigurationsparameter **max_coordagents** des Datenbankmanagers steuern. Die maximale Größe des privaten Speicherbereichs jedes Agenten wird durch die Werte der folgenden Konfigurationsparameter bestimmt:

- **agent_stack_sz**
- **sheapthres** und **sortheap**

Gemeinsamer Speicher von Agenten/Anwendungen

Die Gesamtzahl der Segmente für gemeinsamen Agenten-/Anwendungsspeicher für lokale Clients wird durch den niedrigeren der folgenden Werte begrenzt:

- Der Gesamtwert des Datenbankkonfigurationsparameters **maxappls** für alle aktiven Datenbanken
- Der Wert des Datenbankkonfigurationsparameters **max_coordagents**

Anmerkung: In Konfigurationen, in denen die Konzentratorfunktion der Steuerkomponente aktiviert ist (**max_connections** > **max_coordagents**), wird die Anwendungsspeicherbelegung durch den Parameter **max_connections** begrenzt.

Der gemeinsame Agenten-/Anwendungsspeicher wird außerdem durch folgende Datenbankkonfigurationsparameter beeinflusst:

- **aslheapsz**
- **rqrioblk**

Der FCM-Pufferpool und Speicheranforderungen

In einem partitionierten Datenbanksystem wird der gemeinsam genutzte FCM-Pufferspeicher (FCM - Fast Communications Manager) aus dem gemeinsam genutzten Speicher des Datenbankmanagers zugeordnet.

Dies ist in Abb. 18 dargestellt.

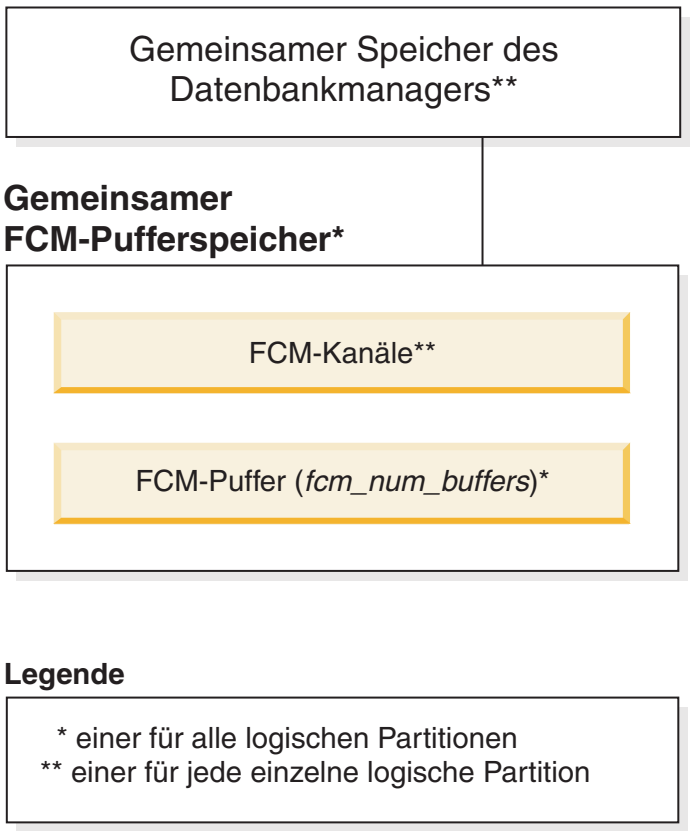


Abbildung 18. FCM-Pufferpool bei Verwendung mehrerer logischer Partitionen

Die Anzahl der FCM-Puffer für jede Datenbankpartition wird durch den Konfigurationsparameter **fcm_num_buffers** des Datenbankmanagers gesteuert. Standardmäßig ist dieser Parameter auf den Wert AUTOMATIC gesetzt. Zur manuellen Optimierung dieses Parameters verwenden Sie Daten aus den Systemmonitorelementen **buff_free** und **buff_free_bottom**.

Die Anzahl der FCM-Kanäle für jede Datenbankpartition wird durch den Konfigurationsparameter **fcm_num_channels** des Datenbankmanagers gesteuert. Standardmäßig ist dieser Parameter auf den Wert AUTOMATIC gesetzt. Zur manuellen Optimierung dieses Parameters verwenden Sie Daten aus den Systemmonitorelementen **ch_free** und **ch_free_bottom**.

Der DB2-Datenbankmanager kann eine automatische Verwaltung der FCM-Speicherressourcen bereitstellen, indem mehr FCM-Puffer und -Kanäle zugeordnet werden als erforderlich. Dies führt zu Leistungsverbesserungen und verhindert Laufzeitfehler durch nicht ausreichende FCM-Ressourcen. Beim Betriebssystem Linux kann der Datenbankmanager in größerem Umfang Systemspeicher (bis zu einem Standardmaximalwert von 4 GB) für FCM-Puffer und -Kanäle bereitstellen. Auswirkungen auf die Hauptspeicherkapazität ergeben sich nur, wenn zusätzliche FCM-Puffer oder -Kanäle erforderlich sind. Definieren Sie die Option FCM_MAXI-

MIZE_SET_SIZE der Registrierdatenbankvariablen **DB2_FCM_SETTINGS** mit YES (oder TRUE), um dieses Verhalten zu aktivieren. YES ist der Standardwert.

Richtlinien zum Optimieren von Parametern mit Wirkung auf die Speicherbelegung

Bei der Optimierung des Hauptspeichers (d. h. wenn die automatische Speicheroptimierungsfunktion (STMM) nicht verwendet wird) liefern Vergleichstests die besten Informationen zur Einstellung geeigneter Werte für Speicherparameter.

Bei der Durchführung von Vergleichstests werden repräsentative SQL-Anweisungen und Extremfall-SQL-Anweisungen auf dem Server ausgeführt und die Werte der Speicherparameter geändert, bis der Punkt gefunden wird, an dem die Leistung wieder sinkt. Dies ist der Punkt, an dem eine zusätzliche Speicherzuordnung keine weitere Leistungssteigerung für die Anwendung bedeutet.

Die oberen Speicherzuordnungsgrenzen für einige Parameter liegen möglicherweise über den Kapazitäten der vorhandenen Hardware und des Betriebssystems. Diese Grenzen wurden mit Blick auf steigende Anforderungen in der Zukunft definiert. Es hat sich bewährt, Speicherparameter nicht auf die jeweils höchsten Werte zu setzen, sofern diese Werte nicht gerechtfertigt werden können. Dies gilt sogar für Systeme, die beträchtliche Kapazitäten an verfügbarem Hauptspeicher besitzen. Die Grundgedanke besteht darin, zu verhindern, dass der Datenbankmanager rasch sämtlichen verfügbaren Hauptspeicher auf einem System einnimmt. Darüber hinaus entsteht durch die Verwaltung großer Speichergrößen zusätzlicher Systemaufwand.

Bei der Mehrzahl der Konfigurationsparameter wird der entsprechende Speicher erst reserviert, wenn er benötigt wird. Die Parametereinstellungen legen die maximale Größe eines bestimmten Zwischenspeicherbereichs fest. Für Pufferpools und die folgenden Konfigurationsparameter wird jedoch die gesamte angegebene Speichergröße zugeordnet:

- **aslheapsz**
- **fcm_num_buffers**
- **fcm_num_channels**
- **locklist**

Gültige Wertebereiche der Parameter finden Sie in den detaillierten Informationen zu den einzelnen Parametern.

Speicher mit automatischer Leistungsoptimierung - Übersicht

Der 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 dynamisch unter folgenden Speicherkonsumenten: Pufferpools, Sperrenspeicher, Paketcache und Sortierspeicher.

Der Speicher mit automatischer Leistungsoptimierung wird durch den Datenbankkonfigurationsparameter **self_tuning_mem** aktiviert.

Die folgenden für den Hauptspeicher relevanten Datenbankkonfigurationsparameter können automatisch optimiert werden:

- database_memory - Größe des gemeinsam genutzten Datenbankspeichers
- locklist - Maximaler Speicher für Sperrenliste

- maxlocks - Maximale Anzahl von Sperren vor Eskalation
- pckcachesz - Größe des Paketcache
- sheapthres_shr - Schwellenwert für Sortierspeicher für gemeinsame Sortiervorgänge
- sortheap - Sortierspeichergröße

Speicher mit automatischer Leistungsoptimierung

Seit DB2 Version 9 vereinfacht eine Speicheroptimierungsfunktion die Aufgabe der Speicherkonfiguration, indem sie automatisch Werte für verschiedene Speicherkonfigurationsparameter einstellt. Wenn sie aktiviert ist, verteilt die Speicheroptimierungsfunktion verfügbare Speicherressourcen dynamisch unter folgenden Speicherkonsumenten: Pufferpools, Sperrenspeicher, Paketcache und Sortierspeicher.

Die Optimierungsfunktion arbeitet innerhalb der Speicherbegrenzungen, die durch den Konfigurationsparameter **database_memory** definiert sind. Der Wert dieses Parameters kann ebenfalls automatisch optimiert werden. Wenn die automatische Speicheroptimierung aktiviert ist (d. h., wenn der Parameter **database_memory** den Wert AUTOMATIC hat), 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 zugeordnet. 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 Konfigurationsparameter **database_memory** nicht auf AUTOMATIC gesetzt ist, verwendet die Datenbank die Größe an Speicher, die für diesen Parameter angegeben wurde, und verteilt sie nach Bedarf an die Speicherkonsumenten. Sie können die Speichergröße auf eine von zwei Arten angeben: durch Setzen des Parameters **database_memory** auf einen bestimmten numerischen Wert oder durch Setzen dieses Parameters auf den Wert COMPUTED. Im letzteren Fall basiert der Gesamtspeicher auf der Summe der Anfangswerte der Datenbankzwischenspeicher beim Starten der Datenbank.

Sie können die automatische Speicheroptimierung für die Speicherkonsumenten auch wie folgt aktivieren:

- Für Pufferpools verwenden Sie die Anweisung ALTER BUFFERPOOL oder CREATE BUFFERPOOL (mit dem Schlüsselwort AUTOMATIC).
- Für den Sperrenspeicher verwenden Sie den Datenbankkonfigurationsparameter **locklist** oder **maxlocks** (mit dem Wert AUTOMATIC).
- Für den Paketcache verwenden Sie den Datenbankkonfigurationsparameter **pckcachesz** (mit dem Wert AUTOMATIC).
- Für den Sortierspeicher verwenden Sie den Datenbankkonfigurationsparameter **sheapthres_shr** oder **sortheap** (mit dem Wert AUTOMATIC).

Änderungen, die aus Operationen der automatischen Speicheroptimierung resultieren, werden in Protokolldateien für die Speicheroptimierung aufgezeichnet, die sich im Unterverzeichnis `stmmlog` befinden. Diese Protokolldateien enthalten Zusammenfassungen über den Ressourcenbedarf der einzelnen Speicherkonsumenten während bestimmter Optimierungsintervalle, die durch Zeitmarken in den Protokolleinträgen bestimmt werden.

Wenn nur wenig Speicher verfügbar ist, fallen die Leistungsvorteile durch die automatische Speicheroptimierung eher begrenzt aus. Da Optimierungsentscheidungen auf der Datenbankauslastung basieren, schränken Auslastungen mit rasch wechselnden Speicheranforderungen die Effektivität des Speichermanagers zur automatischen Leistungsoptimierung (STMM, Self-Tuning Memory Manager) ein. Wenn sich die Speichermerkmale Ihrer Auslastung ständig ändern, optimiert STMM weniger häufig und unter wechselnden Zielbedingungen. In diesem Szenario erreicht STMM keine absolute Konvergenz, sondern versucht stattdessen eine Hauptspeicherkonfiguration beizubehalten, die für die aktuelle Auslastung optimiert ist.

Aktivieren des Speichers mit automatischer Leistungsoptimierung

Der Speicher mit automatischer Leistungsoptimierung vereinfacht die Speicherkonfiguration, indem automatisch Werte für Speicherkonfigurationsparameter eingestellt und die Größe von Pufferpools gesteuert werden.

Informationen zu diesem Vorgang

Wenn sie aktiviert ist, verteilt die Speicheroptimierungsfunktion verfügbare Speicherressourcen dynamisch auf mehrere Speicherkonsumenten, zu denen Pufferpools, der Sperrenspeicher, der Paketcache und der Sortierspeicher gehören.

Vorgehensweise

1. Aktivieren Sie die automatische Speicheroptimierung für die Datenbank, indem Sie den Datenbankkonfigurationsparameter **self_tuning_mem** mithilfe des Befehls **UPDATE DATABASE CONFIGURATION** oder der API db2CfgSet auf den Wert ON setzen.
2. Zur Aktivierung der automatischen Optimierungsfunktion für Speicherbereiche, die durch Speicherkonfigurationsparameter gesteuert werden, setzen Sie die relevanten Konfigurationsparameter mithilfe des Befehls **UPDATE DATABASE CONFIGURATION** oder der API db2CfgSet auf den Wert AUTOMATIC.
3. Zur Aktivierung der automatischen Optimierungsfunktion für einen Pufferpool setzen Sie die Pufferpoolgröße mithilfe der Anweisung CREATE BUFFERPOOL oder ALTER BUFFERPOOL auf den Wert AUTOMATIC. In einer Umgebung mit partitionierten Datenbanken sollte dieser Pufferpool keine Einträge in der Katalogsicht SYSCAT.BUFFERPOOLDBPARTITIONS haben.

Ergebnisse

Anmerkung:

1. Da der Speicher mit automatischer Leistungsoptimierung zwischen verschiedenen Speicherkonsumenten verteilt wird, muss für mindestens zwei Hauptspeicherbereiche gleichzeitig die automatische Optimierung zu einem Zeitpunkt aktiviert sein, zum Beispiel für den Sperrenspeicher und den gemeinsam genutzten Datenbankspeicher. Die Speicheroptimierungsfunktion optimiert den Hauptspeicher im System aktiv (der Datenbankkonfigurationsparameter **self_tuning_mem** hat den Wert ON), wenn eine der folgenden Bedingungen zutrifft:
   - Ein Konfigurationsparameter oder eine Pufferpoolgröße ist auf AUTOMATIC gesetzt und der Datenbankkonfigurationsparameter **database_memory** ist entweder auf einen numerischen Wert oder auf AUTOMATIC gesetzt.
   - Beliebige zwei der Parameter **locklist**, **sheapthres_shr**, **pckcachesz** oder der Pufferpoolgröße sind auf AUTOMATIC gesetzt.

- Der Datenbankkonfigurationsparameter **sortheap** ist auf AUTOMATIC gesetzt.

2. Der Wert des Datenbankkonfigurationsparameters **locklist** wird zusammen mit dem Datenbankkonfigurationsparameter **maxlocks** optimiert. Die Inaktivierung der automatischen Optimierung des Parameters **locklist** inaktiviert automatisch auch die automatische Optimierung des Parameters **maxlocks** und die Aktivierung der automatischen Optimierung des Parameters **locklist** aktiviert automatisch auch die automatische Optimierung des Parameters **maxlocks**.
3. Die automatische Optimierung des Datenbankkonfigurationsparameters **sortheap** oder **sheapthres_shr** ist nur zulässig, wenn der Konfigurationsparameter **sheapthres** des Datenbankmanagers auf den Wert 0 gesetzt ist.
4. Der Wert des Parameters **sortheap** wird zusammen mit dem Parameter **sheapthres_shr** optimiert. Die Inaktivierung der automatischen Optimierung des Parameters **sortheap** inaktiviert automatisch auch die automatische Optimierung des Parameters **sheapthres_shr** und die Aktivierung der automatischen Optimierung des Parameters **sheapthres_shr** aktiviert automatisch auch die automatische Optimierung des Parameters **sortheap**.
5. Die automatische Speicheroptimierungsfunktion wird nur auf dem primären HADR-Server (High Availability Disaster Recovery) ausgeführt. Wenn die automatische Speicheroptimierung auf einem HADR-System aktiviert wird, wird sie nie auf dem sekundären Server ausgeführt, und sie wird auch nur dann auf dem primären Server ausgeführt, wenn die Konfiguration ordnungsgemäß eingestellt ist. Wenn die HADR-Datenbankrollen vertauscht werden, wird die Funktion der automatischen Speicheroptimierung ebenfalls übertragen, sodass sie auf dem neuen primären Server ausgeführt wird. Nach dem Starten der Primärdatenbank oder dem Wechsel eines Systems von einer Bereitschaftsdatenbank zu einer Primärdatenbank durch Funktionsübernahme wird die EDU (Engine Dispatchable Unit, von der Steuerkomponente zuteilbare Einheit) der automatischen Speicheroptimierungsfunktion (STMM, Self-Tuning Memory Manager) möglicherweise erst gestartet, wenn der erste Client eine Verbindung herstellt.

Inaktivieren des Speichers mit automatischer Leistungsoptimierung

Die automatische Speicheroptimierung kann für die gesamte Datenbank oder für einen oder mehrere Konfigurationsparameter bzw. Pufferpools inaktiviert werden.

Informationen zu diesem Vorgang

Wenn die automatische Speicheroptimierung für die gesamte Datenbank inaktiviert wird, bleiben die Speicherkonfigurationsparameter und Pufferpools, die auf AUTOMATIC gesetzt sind, für die automatische Optimierung aktiviert, jedoch behalten die Speicherbereiche ihre aktuelle Größe.

Vorgehensweise

1. Inaktivieren Sie die automatische Speicheroptimierung für die Datenbank, indem Sie den Datenbankkonfigurationsparameter **self_tuning_mem** mithilfe des Befehls **UPDATE DATABASE CONFIGURATION** oder der API db2CfgSet auf den Wert OFF setzen.
2. Zur Inaktivierung der automatischen Optimierungsfunktion für Speicherbereiche, die durch Speicherkonfigurationsparameter gesteuert werden, setzen Sie die relevanten Konfigurationsparameter mithilfe des Befehls **UPDATE DATABASE CONFIGURATION** oder der API db2CfgSet auf den Wert MANUAL.

3. Zur Inaktivierung der automatischen Optimierungsfunktion für einen Pufferpool setzen Sie die Pufferpoolgröße mithilfe der Anweisung ALTER BUFFERPOOL auf einen bestimmten Wert.

Ergebnisse

Anmerkung:

- In einigen Fällen kann ein Speicherkonfigurationsparameter für die automatische Optimierungsfunktion nur dann aktiviert werden, wenn ein anderer Speicherkonfigurationsparameter ebenfalls aktiviert ist. Dies bedeutet zum Beispiel, dass eine Inaktivierung der automatischen Speicheroptimierung für die Datenbankkonfigurationsparameter **locklist** oder **sortheap** die automatische Speicheroptimierung für die Datenbankparameter **maxlocks** bzw. **sheapthres_shr** inaktiviert.

Ermitteln der Speicherkonsumenten mit aktivierter automatischer Leistungsoptimierung

Sie können die Einstellungen für den Speicher mit automatischer Leistungsoptimierung anzeigen, die von Konfigurationsparametern gesteuert werden oder für Pufferpools gelten.

Informationen zu diesem Vorgang

- Zum Anzeigen der Einstellungen für Konfigurationsparameter über die Befehlszeile verwenden Sie den Befehl **GET DATABASE CONFIGURATION** unter Angabe der Option SHOW DETAIL. Die Speicherkonsumenten, für die die automatische Optimierung aktiviert werden kann, werden in der Ausgabe wie folgt zusammengruppiert:

```
Beschreibung                                                    Parameter    Aktueller Wert     Verzögerter Wert
-------------------------------------------------------------------------------------------------------------------
Speicher mit automatischer Leistungsoptimierung          (SELF_TUNING_MEM) = ON (Aktiv)         ON
Größe des gemeinsamen Datenbankspeichers (4 KB)          (DATABASE_MEMORY) = AUTOMATIC(37200)   AUTOMATIC(37200)
Max. Speicher für Sperrenliste (4 KB)                           (LOCKLIST) = AUTOMATIC(7456)    AUTOMATIC(7456)
Anzahl der Sperrenlisten pro Anwend. (in %)                     (MAXLOCKS) = AUTOMATIC(98)      AUTOMATIC(98)
Größe des Paketcache (4 KB)                                  (PCKCACHESZ) = AUTOMATIC(5600)    AUTOMATIC(5600)
Sortierspeicherschwelle für gemeinsame Sortierungen (4 KB)(SHEAPTHRES_SHR) = AUTOMATIC(5000)    AUTOMATIC(5000)
Zwischenspeicher für Sortierlisten (4 KB)                       (SORTHEAP) = 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
```

  SQLF_ON_ACTIVE gibt an, dass die automatische Optimierung aktiviert und aktiv ist, während SQLF_ON_INACTIVE anzeigt, dass die automatische Optimierung zwar aktiviert, jedoch zurzeit nicht aktiv ist.

Zum Anzeigen der Einstellungen für die automatische Optimierungsfunktion für Pufferpools können Sie eine der folgenden Methoden verwenden.

- Zum Abrufen einer Liste der Pufferpools, für die die automatische Optimierung aktiviert ist, verwenden Sie die folgende Abfrage:

```
SELECT BPNAME, NPAGES FROM SYSCAT.BUFFERPOOLS
```

  Wenn die automatische Optimierung für einen Pufferpool aktiviert ist, hat das Feld NPAGES in der Sicht SYSCAT.BUFFERPOOLS für den betreffenden Pufferpool den Wert -2. Wenn die automatische Optimierung inaktiviert ist, enthält das Feld NPAGES die aktuelle Größe des Pufferpools.

- Zur Ermittlung der aktuellen Größe von Pufferpools, für die die automatische Optimierung aktiviert wurde, verwenden Sie den Befehl **GET SNAPSHOT** und untersuchen die aktuelle Größe der Pufferpools (den Wert des Monitorelements **bp_cur_buffsz**):

  ```
  GET SNAPSHOT FOR BUFFERPOOLS ON datenbankaliasname
  ```

  Eine Anweisung ALTER BUFFERPOOL, die die Größe eines Pufferpools in einer bestimmten Datenbankpartition angibt, erstellt einen Ausnahmeeintrag (bzw. aktualisiert einen vorhandenen Eintrag) für diesen Pufferpool in der Katalogsicht SYSCAT.BUFFERPOOLDBPARTITIONS. Wenn ein Ausnahmeeintrag für einen Pufferpool vorhanden ist, wird dieser Pufferpool an Operationen zur automatischen Optimierung nicht beteiligt, wenn die Standardpufferpoolgröße auf den Wert AUTOMATIC gesetzt ist.

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 und die Leistungsvorteile durch eine Verkleinerung des Pufferpoolspeichers zugunsten einer Erweiterung des Sortierspeichers können vielleicht nicht sofort realisiert werden.

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 für partitionierte 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 Optimierungsentscheidungen in dieser Partition getroffen werden, werden die Speicheranpassungen an die anderen Datenbankpartitionen verteilt, um sicherzustellen, dass alle Datenbankpartitionen ähnliche Konfigurationen behalten.

Das auf einer Optimierungspartition basierende Modell geht davon aus, dass die Funktion nur verwendet wird, wenn alle Datenbankpartitionen ähnliche Speicheranforderungen haben. Beachten Sie die folgenden Richtlinien bei der Entscheidung, ob die automatische Speicheroptimierung für eine partitionierte Datenbank aktiviert werden sollte.

Fälle, in denen die automatische Speicheroptimierung für partitionierte 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 Datenbankpartitionen sind gleichmäßig auf mehrere physische Datenbankpartitionen verteilt.
- Es ist eine perfekte oder nahezu perfekte Verteilung von Daten vorhanden.
- Auslastungen werden gleichmäßig über Datenbankpartitionen verteilt. Das heißt, keine Datenbankpartition hat höheren Speicherbedarf für einen oder mehrere Zwischenspeicherbereiche als irgendeine der anderen Datenbankpartitionen.

Wenn in einer solchen Umgebung alle Datenbankpartitionen gleich konfiguriert sind, sorgt die automatische Speicheroptimierung für eine ordnungsgemäße Konfiguration des Systems.

Fälle, in denen die automatische Speicheroptimierung für partitionierte Datenbanken unter Vorkehrungen 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 Sorgfalt erfolgt. Solche Systeme haben möglicherweise nur 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.

Die automatische Speicheroptimierung sollte in allen Datenbankpartitionen aktiviert werden, die Daten enthalten, wobei eine dieser Datenbankpartitionen als Optimierungspartition vorgesehen werden sollte. Da die Koordinatorpartitionen und die Katalogpartitionen verschieden konfiguriert sein können, sollte außerdem die automatische Speicheroptimierung in diesen Partitionen inaktiviert werden. Zur Inaktivierung der automatischen Speicheroptimierung in den Koordinator- und Katalogpartitionen setzen Sie den Datenbankkonfigurationsparameter **self_tuning_mem** in diesen Partitionen auf den Wert OFF.

Fälle, in denen die automatische Speicheroptimierung für partitionierte Datenbanken nicht empfohlen wird

Wenn der Speicherbedarf für die einzelnen Datenbankpartitionen unterschiedlich ist oder verschiedene Datenbankpartitionen auf erheblich unterschiedlicher Hardware betrieben werden, ist es eine empfohlene Methode, die Funktion der automatischen Speicheroptimierung zu inaktivieren. Sie können die Funktion inaktivieren, indem Sie den Datenbankkonfigurationsparameter **self_tuning_mem** in allen Partitionen auf den Wert OFF setzen.

Vergleich des Speicherbedarfs 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 Datenbankpartitionen ähnliche Werte liefern (mit Abweichungen unter 20 %), können die Datenbankpartitionen als ausreichend ähnlich betrachtet werden.

Erfassen Sie die folgenden Daten, indem Sie den Befehl `get snapshot for database on <datenbankname>` ausführen:

```
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

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

Anzahl OLAP-Funktionen                                                      = 0
Anzahl Überläufe von OLAP-Funktionen                                        = 0
Aktive OLAP-Funktionen                                                      = 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
```

Verwenden von Speicher mit automatischer Leistungsoptimierung in Umgebungen mit partitionierten Datenbanken

Wenn die Funktion für Speicher mit automatischer Leistungsoptimierung in Umgebungen mit partitionierten Datenbanken aktiviert wird, überwacht eine einzige Datenbankpartition (die *Optimierungspartition*) die Speicherkonfiguration und gibt alle Konfigurationsänderungen an alle anderen Datenbankpartitionen weiter, um eine konsistente Konfiguration über alle beteiligten 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 Pufferpools.

- Zur Ermittlung, welche Datenbankpartition zurzeit als Optimierungspartition angegeben ist, rufen Sie die Prozedur **ADMIN_CMD** wie folgt auf:

  ```
  CALL SYSPROC.ADMIN_CMD( 'get stmm tuning dbpartitionnum' )
  ```

- Zum Ändern der Optimierungspartition rufen Sie die Prozedur **ADMIN_CMD** wie folgt auf:

  ```
  CALL SYSPROC.ADMIN_CMD('update stmm tuning dbpartitionnum <partitionsnummer>')
  ```

  Die Optimierungspartition wird asynchron oder beim nächsten Start der Datenbank aktualisiert. Wenn die Speicheroptimierungsfunktion die Optimierungspartition automatisch auswählen soll, geben Sie '-1' für *partitionsnummer* ein.

Starten der Speicheroptimierungsfunktion in Umgebungen mit partitionierten Datenbanken

In einer Umgebung mit partitionierten Datenbanken wird die Speicheroptimierungsfunktion nur gestartet, wenn die Datenbank explizit mit dem Befehl **ACTIVATE DATABASE** aktiviert wird, weil die automatische Speicheroptimierung voraussetzt, dass alle Partitionen aktiv sind.

Inaktivieren der automatischen Speicheroptimierungsfunktion für eine bestimmte Datenbankpartition

- Zur Inaktivierung der automatischen Speicheroptimierung für eine Teilgruppe von Datenbankpartitionen setzen Sie den Datenbankkonfigurationsparameter **self_tuning_mem** für diese Datenbankpartitionen auf den Wert OFF.
- Zur Inaktivierung der automatischen Speicheroptimierung 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. Es wird empfohlen, die Werte der Konfigurationsparameter für die automatische Speicheroptimierungsfunktion über alle aktiven Partitionen hinweg einheitlich zu definieren.
- Zur Inaktivierung der automatischen Speicheroptimierung für einen bestimmten Pufferpool in einer bestimmten Datenbankpartition führen Sie die Anweisung ALTER BUFFERPOOL aus, indem Sie einen Größenwert sowie die Partition angeben, in der die automatische Speicheroptimierung inaktiviert werden soll.

  Eine Anweisung ALTER BUFFERPOOL, die die Größe eines Pufferpools in einer bestimmten Datenbankpartition angibt, erstellt einen Ausnahmeeintrag (bzw. aktualisiert einen vorhandenen Eintrag) für diesen Pufferpool in der Katalogsicht SYSCAT.BUFFERPOOLDBPARTITIONS. Wenn ein Ausnahmeeintrag für einen Pufferpool vorhanden ist, wird dieser Pufferpool an Operationen zur automatischen Optimierung nicht beteiligt, wenn die Standardpufferpoolgröße auf den Wert AUTOMATIC gesetzt ist. Gehen Sie daher wie folgt vor, wenn Sie einen Ausnahmeeintrag entfernen möchten, sodass ein Pufferpool für die automatische Optimierung wieder aktiviert werden kann:

  1. Inaktivieren Sie die automatische Optimierung für diesen Pufferpool, indem Sie eine Anweisung ALTER BUFFERPOOL ausführen, die die Pufferpoolgröße auf einen bestimmten Wert setzt.
  2. Führen Sie eine weitere Anweisung ALTER BUFFERPOOL aus, um die Größe des Pufferpools in dieser Datenbankpartition auf den Standardwert zu setzen.
  3. Aktivieren Sie die automatische Optimierung für diesen Pufferpool, indem Sie eine weitere Anweisung ALTER BUFFERPOOL ausführen, die die Pufferpoolgröße auf den Wert AUTOMATIC setzt.

Aktivieren des Speichers mit automatischer Leistungsoptimierung in nicht einheitlichen Umgebungen

Im Idealfall sollten Daten gleichmäßig auf alle Datenbankpartitionen verteilt und die Auslastung, die in jeder Partition ausgeführt wird, durch ähnliche Speicheranforderungen gekennzeichnet sein. Wenn die Datenverteilung ungleichmäßig ist, sodass mindestens eine Datenbankpartition erheblich mehr oder weniger Daten als andere Datenbankpartitionen enthält, sollte für solche anomalen Datenbankpartitionen die automatische Speicheroptimierung nicht 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 Speicheroptimierung 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, ermitteln Sie eine Gruppe von Datenbankpartitionen, die ähnliche Daten- und Speicheranforderungen haben, und aktivieren für

diese die automatische Speicheroptimierung. Der Speicher in den übrigen Partitionen muss manuell konfiguriert werden.

Pufferpoolverwaltung

Ein Pufferpool stellt einen Arbeitsspeicher und einen Cache für Datenbankseiten bereit.

Pufferpools verbessern die Leistung des Datenbanksystems, indem sie die Möglichkeit schaffen, auf Daten im Hauptspeicher anstatt auf der Platte zuzugreifen. Aufgrund der Tatsache, dass die meisten Bearbeitungsschritte für Seitendaten in Pufferpools stattfinden, ist die Konfiguration von Pufferpools der wichtigste Einzelbereich bei der Optimierung.

Wenn eine Anwendung auf eine Tabellenzeile zugreift, sucht der Datenbankmanager die Seite, die diese Zeile enthält, im Pufferpool. Wenn die Seite dort nicht gefunden wird, liest der Datenbankmanager die Seite von der Platte und speichert sie im Pufferpool. Die Daten können anschließend zur Verarbeitung der Abfrage verwendet werden.

Der Speicher für Pufferpools wird zugeordnet, wenn eine Datenbank aktiviert wird. Die erste Anwendung, die eine Verbindung herstellt, kann eine implizite Aktivierung der Datenbank bewirken. Pufferpools können erstellt, in der Größe geändert oder gelöscht werden, während der Datenbankmanager aktiv ist. Mit der Anweisung ALTER BUFFERPOOL kann ein Pufferpool vergrößert werden. Standardmäßig wird die Größe des Pufferpools, sofern ausreichend Speicher verfügbar ist, unverzüglich bei der Ausführung der Anweisung geändert. Wenn keine ausreichende Speicherkapazität verfügbar ist, wenn die Anweisung ausgeführt wird, wird der Speicher zugeordnet, wenn die Datenbank reaktiviert wird. Wenn Sie den Pufferpool verkleinern, wird die Zuordnung von Speicher aufgehoben, wenn die Transaktion festgeschrieben (Commit) wird. Der Pufferpoolspeicher wird freigegeben, wenn die Datenbank inaktiviert wird.

Um sicherzustellen, dass in allen Situationen ein geeigneter Pufferpool verfügbar ist, erstellt DB2 kleine Systempufferpools, d. h. je einen mit den folgenden Seitengrößen: 4 KB, 8 KB, 16 KB und 32 KB. Die Größe der Pufferpools beträgt jeweils 16 Seiten. Diese Pufferpools sind verdeckt. Sie sind weder im Systemkatalog noch in den Pufferpoolsystemdateien zu finden. Diese Pufferpools können von Ihnen auch nicht direkt verwendet oder geändert werden, sondern werden von DB2 in den folgenden Situationen genutzt:

- Wenn ein angegebener Pufferpool nicht gestartet wird, weil er mit dem Schlüsselwort DEFERRED erstellt wurde, oder wenn ein Pufferpool der erforderlichen Seitengröße nicht aktiv ist, weil nicht genügend Speicher zu seiner Erstellung verfügbar ist.

  Eine Nachricht wird in das Protokoll mit Benachrichtigungen für die Systemverwaltung geschrieben. Falls erforderlich, werden Tabellenbereiche alternativ einem Systempufferpool zugeordnet. Die Leistung könnte sich beträchtlich verringern.
- Wenn Pufferpools bei der Herstellung einer Datenbankverbindung nicht aktiviert werden können.

  Dieses Problem hat wahrscheinlich eine ernst zu nehmende Ursache, wie zum Beispiel, dass kein Speicher mehr verfügbar ist. Obwohl DB2 aufgrund der Systempufferpools weiterhin voll funktionsfähig bleibt, 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 Protokoll mit Benachrichtigungen für die Systemverwaltung geschrieben.

Wenn Sie einen Pufferpool erstellen, ist seine Seitengröße diejenige, die bei der Erstellung der Datenbank angegeben wurde, sofern Sie nicht explizit 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.

Mit dem Speichertracker, den Sie mithilfe des Befehls **db2mtrk** aufrufen können, haben Sie die Möglichkeit, die Menge an Datenbankspeicher zu prüfen, die Pufferpools zugeordnet wurde. Sie können auch den Befehl **GET SNAPSHOT** verwenden und die aktuelle Größe der Pufferpools untersuchen (d. h. den Wert des Monitorelements **bp_cur_buffsz**).

Die Pufferpoolpriorität für Aktivitäten kann im Rahmen einer umfassenderen Funktionalität für das Workload-Management gesteuert werden, die von DB2-Workload-Manager bereitgestellt wird. Nähere Informationen hierzu finden Sie in den Abschnitten „Einführung zu den dem DB2-Workload-Manager zugrunde liegenden Konzepten" und „Pufferpoolpriorität von Serviceklassen".

Pufferpoolverwaltung von Datenseiten

Seiten im Pufferpool können entweder im Gebrauch sein oder nicht, und sie können benutzt (geändert) oder sauber (ungeändert) sein.

- *Im Gebrauch* befindliche Seiten werden momentan gelesen oder aktualisiert. Wenn eine Seite aktualisiert wird, kann nur die aktualisierende Komponente auf sie zugreifen. Wird die Seite jedoch nicht aktualisiert, kann sie von mehreren lesenden Komponenten gleichzeitig gelesen werden.
- *Benutzte Seiten* enthalten Daten, die geändert, jedoch noch nicht auf die Platte geschrieben wurden.

Seiten verbleiben im Pufferpool, bis die Datenbank gestoppt wird, bis der von einer Seite belegte Speicherbereich im Pufferpool für eine andere Seite benötigt wird oder bis die Seite explizit aus dem Pufferpool entfernt wird, zum Beispiel wenn ein Objekt gelöscht wird. Die folgenden Kriterien bestimmen, welche Seite entfernt wird, wenn eine andere Seiten diesen Speicherplatz benötigt:

- Der Zeitpunkt, zu dem zum letzten Mal auf die Seite verwiesen wurde
- Die Wahrscheinlichkeit, mit der erneut auf die Seite zugegriffen wird
- Der Typ von Daten, den die Seite enthält
- Der Änderungsstatus der Seite, d. h. ob sie im Speicher geändert, jedoch noch nicht auf Platte geschrieben wurde

Geänderte Seiten werden immer auf die Festplatte geschrieben, bevor sie überschrieben werden. Geänderte Seiten, die auf die Festplatte geschrieben werden, werden nicht automatisch aus dem Pufferpool entfernt, sofern der Speicherplatz nicht benötigt wird.

Agenten für Seitenlöschfunktionen

In einem entsprechend optimierten System schreiben meist Agenten für Seitenlöschfunktionen geänderte oder 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, die *proaktive Seitenbereinigung* zu aktivieren, sodass sich Seitenlöschfunktionen bei der Auswahl der benutzten Seiten, die zu einem Zeitpunkt auf die Platte zu schreiben sind, proaktiver verhalten. 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.

Abb. 19 veranschaulicht, wie die Arbeit zur Verwaltung des Pufferpools zwischen den Agenten für Seitenlöschfunktionen und den Datenbankagenten aufgeteilt werden kann.

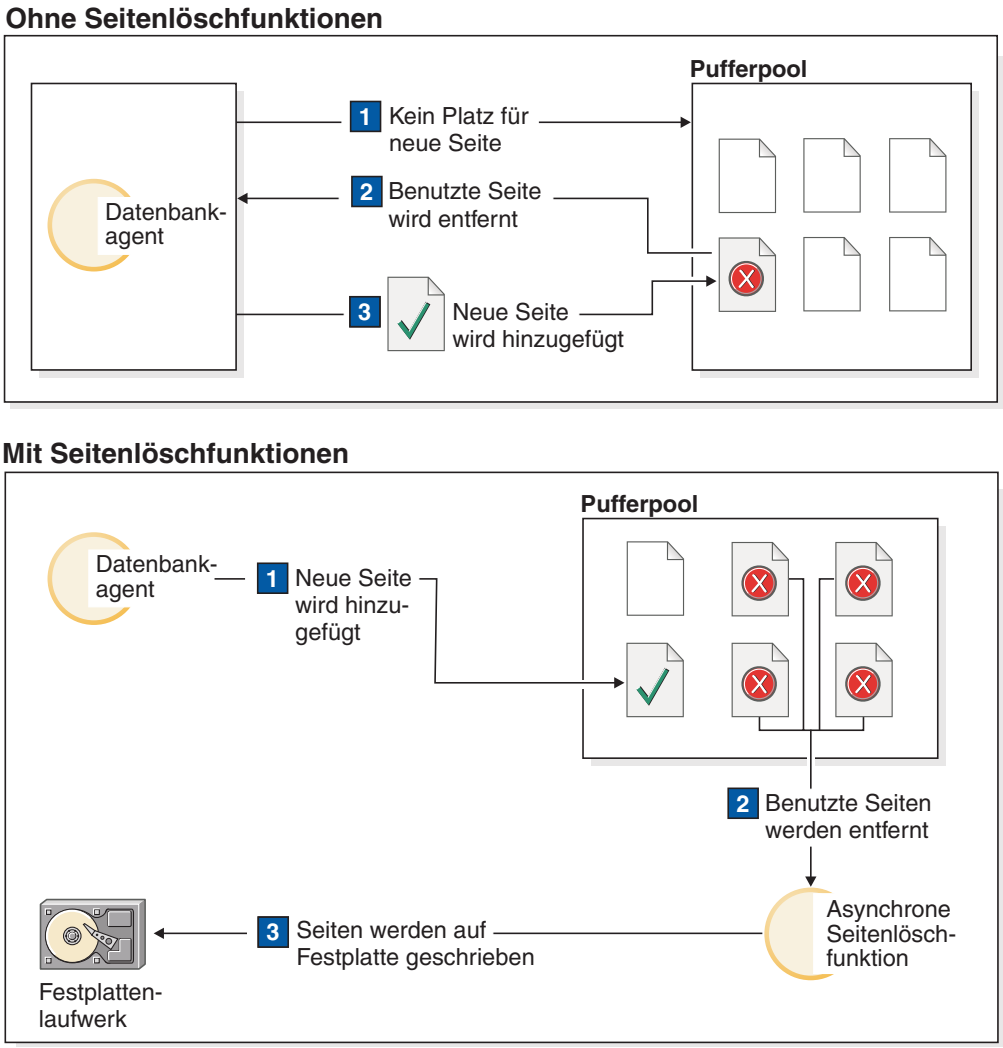


Abbildung 19. Asynchrone Seitenbereinigung. Geänderte bzw. benutzte Seiten werden auf Platte geschrieben.

Seitenbereinigung und schnelle Recovery

Eine Datenbankrecovery nach einem Systemabsturz ist schneller, wenn mehr Seiten auf die Festplatte geschrieben wurden, 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

Seitenlöschfunktionen schreiben benutze Seiten in der Weise auf die Platte, dass die Größe des Protokolls, das bei einer Recovery wiederholt werden müsste, den nach folgender Formel berechneten Wert nie übersteigt:

```
logfilsiz * softmax / 100 (in 4-KB-Seiten)
```

Dabei gilt:

- **logfilsiz** ist die Größe der Protokolldateien.
- **softmax** stellt den Prozentsatz der Protokolldateien dar, für die nach einem Datenbankabsturz eine Recovery durchgeführt werden muss. 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, mithilfe des Datenbanksystemmonitors die Häufigkeit, mit der Seitenlöschfunktionen aktiv werden. Das Monitorelement **pool_lsn_gap_clns** (für Pufferpoolprotokollbereich ausgelöste Seitenlöschfunktionen) stellt diese Information bereit, wenn Sie nicht die proaktive Seitenbereinigung für Ihre Datenbank aktiviert haben. Wenn Sie die proaktive Seitenbereinigung aktiviert haben, sollte diese Situation nicht eintreten und das Monitorelement **pool_lsn_gap_clns** zeigt den Wert 0.

Mithilfe des Monitorelements **log_held_by_dirty_pages** können Sie bestimmen, ob die Seitenlöschfunktionen nicht ausreichend Seiten bereinigen, um die vom Benutzer festgelegten Bedingungen für die Recovery zu erfüllen. Wenn das Monitorelement **log_held_by_dirty_pages** beständig einen erheblich größeren Wert als **logfilsiz** * **softmax** zeigt, sind entweder mehr Seitenlöschfunktionen erforderlich oder der Wert des Parameters **softmax** muss angepasst werden.

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.

Mit der Anweisung ALTER BUFFERPOOL können Sie die Größe eines Pufferpools ändern.

Eine neue Datenbank verfügt über einen Standardpufferpool mit dem Namen IBMDEFAULTBP mit einer Standardseitengröße, die auf der Seitengröße basiert, die bei der Erstellung der Datenbank angegeben wurde. Die Standardseitengröße wird im Informationsdatenbankkonfigurationsparameter **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 einem Upgrade einer Datenbank können Sie weitere Pufferpools erstellen. Wenn Sie eine Datenbank mit einer Seitengröße von 8 KB als Standardwert erstellen, wird der Standardpufferpool mit der Standardseitengröße (in diesem Fall 8 KB) erstellt. Alternativ 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, falls einer vorhanden ist, oder einem der Standardsystempufferpools zuzuordnen, die DB2 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 diesem Pufferpool zu.

Wenn Sie bei der Erstellung eines Pufferpools mit der Anweisung CREATE BUFFERPOOL keine Größe angeben, wird die Pufferpoolgröße auf AUTOMATIC gesetzt und von DB2 verwaltet. Wenn Sie die Pufferpoolgröße später ändern wollen, verwenden Sie dazu 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, sodass der Zugriff schneller erfolgen kann. Durch weniger E/A-Operationen können E/A-Konkurrenzsituationen besser vermieden werden, sodass 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 solche, auf denen Katalogtabellen, häufig verwendete Benutzertabellen und Indizes gespeichert werden. Sortierungen, die für Abfragen erforderlich sind, profitieren ebenfalls von verminderten E/A-Konkurrenzsituationen auf Plattenspeichereinheiten, die temporäre Tabellenbereiche enthalten.

Vorteile vieler Pufferpools

Verwenden Sie nur einen Pufferpool, wenn eine der folgenden Bedingungen auf Ihr System zutrifft:

- Der gesamte Pufferspeicherbereich beträgt weniger als 10.000 4-KB-Seiten.
- Es stehen keine Fachleute mit Anwendungskenntnissen zur Durchführung einer spezialisierten Optimierung zur Verfügung.
- Sie arbeiten auf einem Testsystem.

In allen anderen Fällen und aus folgenden Gründen sollten Sie die Verwendung mehrerer Pufferpools in Betracht ziehen:

- Tabellenbereiche für temporäre Tabellen können einem getrennten Pufferpool zugeordnet werden, um eine bessere Leistung für Abfragen (insbesondere sortierintensiven Abfragen) zu erzielen, die temporären Speicherbereich benötigen.
- 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.
- Sie können kleinere Pufferpools für Daten verwenden, auf die selten ausgeführte Anwendungen zugreifen, insbesondere Anwendungen, die einen sehr wahlfreien Zugriff auf eine sehr umfangreiche Tabelle benötigen. In solchen Fällen brauchen Daten nicht länger als für eine einzige Abfrage im Pufferpool behalten zu werden. Es ist günstiger, einen kleinen Pufferpool für diesen Typ von Daten zu verwenden und den übrigen Speicher für andere Pufferpools freizugeben.

Nach der Trennung der Daten in verschiedene Pufferpools können gute und relativ unaufwendige Daten zur Leistungsdiagnose aus Statistiken und aus Tracefunktionen für Benutzeraktivitäten generiert werden.

Die automatische Speicheroptimierungsfunktion (STMM, Self-Tuning Memory Manager) eignet sich ideal zur Optimierung von Systemen, die mehrere Pufferpools haben.

Zuordnung von Pufferpoolspeicher beim Start

Wenn Sie einen Pufferpool erstellen oder 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 ändern, während der Datenbankmanager online ist, sollte zusätzlicher Speicherplatz im globalen Datenbankspeicher verfügbar sein. Wenn Sie das Schlüsselwort DEFERRED 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 Änderungen bei der nächsten Aktivierung der Datenbank durch.

Wenn dieser Speicher beim Start der Datenbank nicht verfügbar ist, verwendet der Datenbankmanager nur die Systempufferpools (einen für jede Seitengröße) mit ei-

ner Minimalgröße von 16 Seiten und gibt eine Warnung zurück. Die Datenbank setzt den Betrieb in diesem Status fort, bis ihre Konfiguration geändert wird und die Datenbank vollständig neu gestartet werden kann. Obwohl die Leistung möglicherweise nicht optimal ist, können Sie eine Verbindung zu der Datenbank herstellen, die Konfiguration der Pufferpoolgrößen ändern und andere wichtige Tasks ausführen. Wenn Sie diese Tasks abgeschlossen haben, starten Sie die Datenbank erneut. Arbeiten Sie nicht über einen längeren Zeitraum mit der Datenbank in einem solchen Status.

Zur Vermeidung eines Starts der Datenbank nur mit Systempufferpools verwenden Sie die Registrierdatenbankvariable **DB2_OVERRIDE_BPF**, um die Verwendung des verfügbaren Speichers zu optimieren.

Proaktive Seitenbereinigung

Seit DB2 Version 8.1.4 steht eine alternative Methode zur Konfiguration der Seitenbereinigung in Ihrem System zur Verfügung. Mit dieser Methode verhalten sich Seitenlöschfunktionen proaktiver bei der Auswahl der benutzten (geänderten) Seiten, die zu einem gegebenen Zeitpunkt ausgelesen und auf die Platte geschrieben werden.

Diese proaktive Methode zur Seitenbereinigung unterscheidet sich von der Standardseitenbereinigung in zwei wesentlichen Punkten:

- Seitenlöschfunktionen reagieren nicht mehr auf den Wert des Datenbankkonfigurationsparameters **chngpgs_thresh**.

  Wenn die Anzahl gut geeigneter Seiten unter einen akzeptablen Wert fällt, durchsuchen Seitenlöschfunktionen den gesamten Pufferpool, wobei sie potenziell geeignete Seiten auf die Platte schreiben und die Agenten über die Positionen dieser Seiten informieren.
- Seitenfunktionen reagieren nicht mehr auf Trigger bei LSN-Abstimmungsverlusten (LSN - Protokollfolgenummer), die von der Protokollfunktion abgesetzt werden.

  Wenn die Größe des Protokollspeichers zwischen dem Protokollsatz, der die älteste Seite im Pufferpool aktualisiert hat, und der aktuellen Protokollposition den vom Datenbankkonfigurationsparameter **softmax** zugelassenen Wert überschreitet, wird diese Situation als *LSN-Abstimmungsverlust* (LSN-Gap) der Datenbank bezeichnet.

  Wenn eine Protokollfunktion unter der Standardseitenbereinigungsmethode einen LSN-Abstimmungsverlust erkennt, löst sie Seitenlöschfunktionen aus, die alle Seiten, die zu diesem Abstimmungsverlust beitragen, auf die Platte zu schreiben. Das heißt, die Seitenlöschfunktionen schreiben die Seiten auf die Platte, die älter sind, als durch den Wert des Parameters **softmax** zugelassen wird. Seitenlöschfunktionen alternieren zwischen Leerlauf und hoher Aktivität beim Schreiben großer Anzahlen von Seiten. Dies kann zu einer Vollauslastung 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 erkannt wird, die Seitenlöschfunktionen die Seiten nicht mehr schnell genug bereinigen können und DB2 am Ende keinen Protokollspeicherbereich mehr zur Verfügung hat.

  Die proaktive Seitenbereinigungsmethode moduliert dieses Verhalten, indem sie die gleiche Anzahl von Schreiboperationen über einen längeren Zeitraum verteilt. Die Seitenlöschfunktionen führen dies aus, indem sie nicht nur Seiten bereinigen, die zurzeit zu einem LSN-Abstimmungsverlust beitragen, sondern auch Seiten, die mit einiger Wahrscheinlichkeit angesichts des aktuellen Aktivitätsvolumens zu einem entstehenden LSN-Abstimmungsverlust beitragen werden.

Zur Verwendung der neuen Seitenbereinigungsmethode setzen Sie die Registrierdatenbankvariable **DB2_USE_ALTERNATE_PAGE_CLEANING** auf den Wert ON.

Verbessern der Aktualisierungsleistung

Wenn ein Agent eine Seite aktualisiert, arbeitet der Datenbankmanager mit einem Protokoll, um die für die Transaktion erforderlichen E/A-Aktivitäten zu minimieren und die Wiederherstellbarkeit zu gewährleisten.

Dieses Protokoll umfasst die folgenden Schritte:

1. Die Seite, die zu aktualisieren ist, wird fixiert und mit einer exklusiven Sperre belegt. In den Protokollpuffer wird ein Protokollsatz geschrieben, der beschreibt, wie die Änderung rückgängig gemacht und wiederholt werden kann. Während dieser Aktion wird auch eine Protokollfolgenummer (Log Sequence Number, LSN) empfangen und im Kopf der Seite gespeichert, die aktualisiert wird.
2. Die Aktualisierung wird auf die Seite angewendet.
3. Die Sperre der Seite wird aufgehoben. Die Seite gilt als „benutzt", da Änderungen an der Seite noch nicht auf Platte geschrieben wurden.
4. Der Protokollpuffer wird aktualisiert. Sowohl die Daten im Protokollpuffer als auch die benutzte Datenseite werden auf Platte geschrieben.

Zur Erzielung einer besseren Leistung werden diese E/A-Operationen verzögert, bis eine Phase mit niedriger Systemauslastung eintritt oder bis sie erforderlich sind, um die Wiederherstellbarkeit sicherzustellen oder um die Dauer einer Recovery zu begrenzen. Insbesondere wird eine benutzte Seite zu folgenden Zeitpunkten auf Platte geschrieben:

- Ein anderer Agent wählt sie als zu bereinigend aus.
- Eine Seitenlöschfunktion arbeitet an dieser Seite. Die kann unter folgenden Umständen geschehen:
  - Ein anderer Agent wählt die Seite als zu bereinigend aus.
  - Der Wert des Datenbankkonfigurationsparameters **chngpgs_thresh** wird überschritten, sodass asynchrone Seitenlöschfunktionen aktiviert werden und die geänderten Seiten auf Platte schreiben. Wenn die proaktive Seitenbereinigung für die Datenbank aktiviert ist, ist dieser Wert irrelevant und löst keine Seitenbereinigung aus.
  - Der Wert des Datenbankkonfigurationsparameters **softmax** wird überschritten, sodass asynchrone Seitenlöschfunktionen aktiviert werden und die geänderten Seiten auf Platte schreiben. 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 der sauberen Seiten fällt auf einen zu niedrigen Wert. Seitenlöschfunktionen reagieren auf diese Bedingung unter der proaktiven Seitenbereinigung.
  - Eine benutzte Seite trägt zu einer LSN-Abstimmungsverlustbedingung bei oder es ist zu erwarten, dass sie dies tut. Seitenlöschfunktionen reagieren auf diese Bedingung unter der proaktiven Seitenbereinigung.
- Die Seite ist Teil einer Tabelle, die mit der Klausel NOT LOGGED INITIALLY definiert wurde, und auf die Aktualisierung folgt eine COMMIT-Anweisung. Wenn die COMMIT-Anweisung ausgeführt wird, werden alle geänderten Seiten auf Platte geschrieben, um die Wiederherstellbarkeit sicherzustellen.

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 E/A-Wartezeiten verringert werden. Darüber hinaus wird die Effizienz des Vorablesezugriffs durch parallele E/A-Operationen erhöht.

Es gibt zwei Kategorien des Vorablesezugriffs:

- Beim *sequenziellen Vorablesezugriff* werden aufeinander folgende Seiten in den Pufferpool eingelesen, bevor die Seiten von der Anwendung angefordert werden.
- Beim *Vorablesezugriff über Listen* (auch als *sequenzieller Vorablesezugriff über Listen* bezeichnet) wird eine Gruppe nicht aufeinander folgender Datenseiten effizient vorab eingelesen.

Das Vorablesen von Datenseiten unterscheidet sich von einem Lesevorgang durch einen Datenbankmanageragenten, der verwendet wird, wenn nur eine oder wenige aufeinander folgende Seiten abgerufen werden, jedoch nur eine Seite von Daten an eine Anwendung übertragen wird.

Vorablesezugriff und partitionsinterne Parallelität

Das Vorablesen hat einen wichtigen Einfluss auf die Leistung der partitionsinternen Parallelität, bei der mehrere Subagenten beim Durchsuchen eines Index oder einer Tabelle verwendet werden. Solche Parallelsuchen führen zu höheren Datenverarbeitungsraten, die wiederum höhere Vorableseraten erfordern.

Der Nachteil durch ungeeignetes Vorablesen ist bei parallelen Suchoperationen höher als bei seriellen Suchoperationen. Wenn bei einer seriellen Suchoperation kein Vorablesezugriff stattfindet, arbeit die Abfrage langsamer, weil der Agent auf E/A-Operationen wartet. Wenn bei einer parallelen Suchoperation kein Vorablesezugriff stattfindet, müssen eventuell alle Subagenten warten, während ein Subagent auf eine E/A-Operation wartet.

Aufgrund seiner Bedeutung in diesem Kontext wird der Vorablesezugriff unter partitionsinterner Parallelität aggressiver ausgeführt. Der Mechanismus zur Sequenzerkennung toleriert größere Lücken zwischen benachbarten Seiten, sodass die Seiten als sequenziell betrachtet werden können. Die Breite dieser Lücken erhöht sich mit der Anzahl der an der Suchoperation beteiligten Subagenten.

Sequenzieller Vorablesezugriff:

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.

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 Tabellensortierungen wählt der Datenbankmanager automatisch den sequenziellen Vorablesezugriff aus. 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
```

Sequenzerkennung

Manchmal ist es nicht von vornherein offensichtlich, dass ein sequenzieller Vorablesezugriff die Leistung verbessert. In solchen Fällen kann der Datenbankmanager die E/A-Operationen überwachen und den Vorablesezugriff aktivieren, wenn sequenzielles Lesen von Seiten auftritt. Dieser Typ des sequenziellen Vorablesens wird als *Sequenzerkennung* bezeichnet und gilt für Index- und Datenseiten. Mit dem Datenbankkonfigurationsparameter **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 mithilfe eines Index für die Spalte EMPNO zu durchsuchen. Wenn die Tabelle eine hohe Clusterbildung in Bezug auf diesen Index aufweist, sind die Leseoperationen für die Datenseiten nahezu sequenziell und ein Vorablesezugriff könnte zu einer Leistungsverbesserung führen. Wenn analog eine große Anzahl von Indexseiten untersucht werden muss und der Datenbankmanager erkennt, dass ein sequenzielles Lesen der Indexseiten stattfindet, ist ein Vorablesezugriff auf die Indexseiten wahrscheinlich.

Auswirkungen der Option PREFETCHSIZE für Tabellenbereiche

Mit der Klausel PREFETCHSIZE in der Anweisung CREATE TABLESPACE bzw. ALTER TABLESPACE können Sie die Anzahl der vorabgelesenen Seiten angeben, die aus dem Tabellenbereich gelesen werden, wenn ein Datenvorablesezugriff ausgeführt wird. Der Wert, den Sie angeben (oder 'AUTOMATIC') wird in der Spalte PREFETCHSIZE der Katalogsicht 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 (die Anzahl von Seiten an, die der Datenbankmanager in einen Container schreibt, bevor er einen anderen Container verwendet) für Ihren Tabellenbereich zu definieren. Ist zum Beispiel für EXTENTSIZE ein Wert von 16 Seiten festgelegt und hat der Tabellenbereich zwei Container, könnten Sie den Wert für die Vorablesezugriffsgröße auf den Wert 32 setzen. Wenn fünf physische Platten pro Container vorhanden sind, könnten Sie die Vorablesezugriffsgröße 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 für den Tabellenbereich angegebenen Wert begrenzen.

Die Vorablesezugriffsgröße (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 der Vorablesezugriffsgröße für Ihre Tabellenbereiche. Zum Beispiel können Sie folgende Arten von Informationen erfassen:

- Mit Überwachungsprogrammen, die für Ihr Betriebssystem verfügbar sind, können Sie feststellen, ob E/A-Wartezeiten für die Abfrage auftreten.
- Mithilfe 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, während für eine Abfrage der Vorablesezugriff aktiv ist, können Sie den Wert für PREFETCHSIZE erhöhen. Wenn die Vorablesefunktion nicht die Ursache für diese 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 EXTENTSIZE großen Bereiche in separaten Containern befinden. Konfigurieren Sie die Container zwecks besserer Leistung so, dass sie separate physische Einheiten verwenden.

Blockbasierte 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 überlappt.

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 in großen Blockgrößen konkurrieren.

Standardmäßig sind Pufferpools seitenbasiert. Dies bedeutet, dass zusammenhängende Seiten auf dem Plattenspeicher in nicht zusammenhängende Seiten im Arbeitsspeicher vorab eingelesen werden. Der sequenzielle Vorablesezugriff kann verbessert werden, wenn zusammenhängende Seiten vom Plattenspeicher in zusammenhängende Seiten in einem Pufferpool eingelesen werden können.

Zu diesem Zweck können Sie blockbasierte Pufferpools erstellen. Ein blockbasierter 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 blockbasierten Pufferpools hängt von der angegebenen Blockgröße ab. Die Blockgröße ist die Granularität, mit der E/A-Server bei sequenziellen Vorablesezugriffen eine blockbasierte 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:

- Der Wert für EXTENTSIZE, durch den die Größe für Vorableseanforderungen festgelegt wird, ist kleiner als die für den Pufferpool angegebene Blockgröße.

- Einige Seiten in der Vorableseanforderung befinden sich bereits im Seitenbereich des Pufferpools.

Der E/A-Server lässt einige verschwendete Seiten in jedem Pufferpoolblock zu. Wenn jedoch zu viele Seiten verschwendet würden, führt der E/A-Server ein nicht blockbasiertes Vorablesen in den Seitenbereich des Pufferpools durch, was zu einer suboptimalen Leistung führt.

Zur Erzielung einer optimalen Leistung binden Sie Tabellenbereiche der gleichen EXTENTSIZE-Größe an einen Pufferpool, dessen Blockgröße 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 BUFFERPOOL oder ALTER BUFFERPOOL zur Erstellung eines blockbasierten Pufferpools.

Anmerkung: Blockbasierte Pufferpools sind für den sequenziellen Vorablesezugriff gedacht. Wenn Ihre Anwendungen keinen sequenziellen Vorablesezugriff nutzen, wird der Blockbereich im Pufferpool verschwendet.

Vorablesezugriff über Listen:

Der *Vorablesezugriff über Listen* (oder *sequenzielle Vorablesezugriff über Listen*) ist eine effiziente Methode, auf Daten zuzugreifen, auch wenn diese Seiten nicht zusammenhängend vorliegen.

Der Vorablesezugriff 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 Satz-IDs (RIDs) aus dem Index empfangen wurden. Zum Beispiel könnte das Optimierungsprogramm eine Indexsuche im folgenden Index durchführen, um die abzurufenden Zeilen und Datenseiten zu ermitteln:

```
INDEX IX1:  NAME    ASC,
            DEPT    ASC,
            MGR     DESC,
            SALARY  DESC,
            YEARS   ASC
```

Anschließend können die folgenden Suchbedingungen verwendet werden:

```
WHERE NAME BETWEEN 'A' and 'I'
```

Wenn die Daten in Bezug auf den Index keine Clusterbildung aufweisen, enthält der Vorablesezugriff über Listen einen Schritt zum Sortieren der durch die Indexsuche ermittelten Liste von Satz-IDs (RIDs).

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-Aktivität. Diese Vorableseanforderungen enthalten eine Beschreibung der E/A-Operationen, die zur Erfüllung der Abfrage benötigt werden.

Durch Konfigurieren einer ausreichenden Anzahl von E/A-Servern (mithilfe des Datenbankkonfigurationsparameters **num_ioservers**) kann die Leistung von Abfragen, die vom Vorablesezugriff profitieren können, 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 ohne Auswirkungen auf die Leistung ausgelagert. Jeder E/A-Serverprozess hat eine Nummer. Der Datenbankmanager verwendet immer den Prozess mit der niedrigsten Nummer, sodass 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

Ziehen Sie in Betracht, den Konfigurationsparameter **num_ioservers** auf den Wert AUTOMATIC zu setzen, sodass der Datenbankmanager intelligente Werte auf der Basis der Systemkonfiguration auswählen kann.

Illustration des Vorablesezugriffs mit paralleler E/A:

E/A-Server lesen Daten durch einen Vorablesezugriff in einen Pufferpool ein.

Dieser Prozess ist in Abb. 20 auf Seite 122 dargestellt.

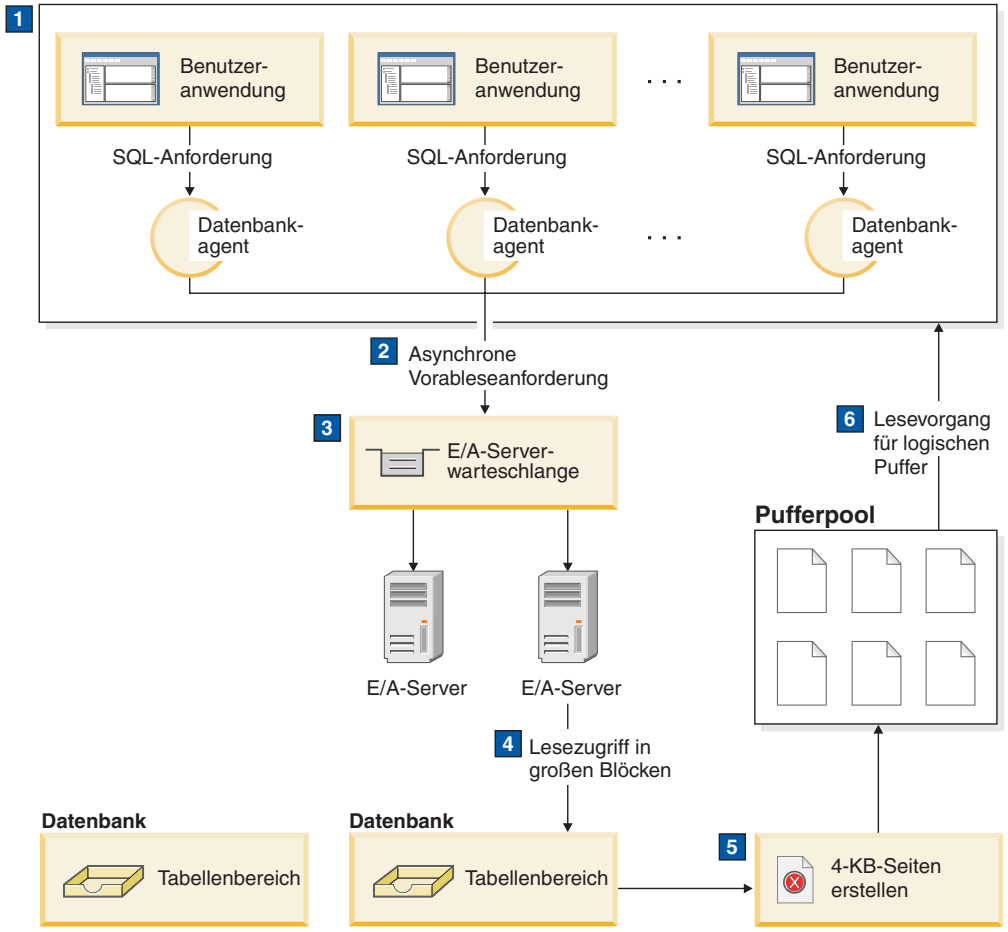


Abbildung 20. Vorablesezugriff auf Daten durch E/A-Server

1 Die Benutzeranwendung übergibt die Anforderung an den Datenbankagenten, der der Benutzeranwendung vom Datenbankmanager zugeordnet wurde.

2, **3**
Der Datenbankagent stellt 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 anschließend die Daten aus dem Tabellenbereich in den Pufferpool ein. 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 Datenbankkonfigurationsparameter **num_ioservers** angegeben ist.

6 Der Datenbankagent führt die erforderlichen Operationen an den Datenseiten im Pufferpool aus und liefert das Ergebnis an die Benutzeranwendung zurück.

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, sodass mehrere Container parallel gelesen werden können. Die parallele E/A 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:

- Beim *sequenziellen Vorablesezugriff* wird die parallele E/A initialisiert, wenn der Wert für PREFETCHSIZE (Vorablesezugriffsgröße) ein Vielfaches des Werts für EXTENTSIZE eines Tabellenbereichs ist. Jede Vorablesezugriffsanforderung wird in kleinere, an den EXTENTSIZE großen Speicherbereichsgrenzen ausgerichtete Anforderungen zerlegt. Diese kleinen Anforderungen werden dann verschiedenen E/A-Servern zugeordnet.
- Beim *Vorablesezugriff über Listen* wird jede Liste von Seiten entsprechend den Containern, in denen die Datenseiten gespeichert sind, in kleinere Listen unterteilt. Diese kleinen Listen werden dann verschiedenen E/A-Servern zugeordnet.
- Beim *Backup und Restore von Datenbank und Tabellenbereichen* ist die Anzahl paralleler E/A-Anforderungen gleich der Größe des Backup-Puffers dividiert durch den Wert von EXTENTSIZE, wobei der Maximalwert gleich der Anzahl von Containern ist.
- Beim *Restore von Datenbanken und Tabellenbereichen* werden die parallelen E/A-Anforderungen initialisiert und auf dieselbe Weise zerlegt, wie dies beim sequenziellen Vorablesezugriff der Fall ist. Die Daten werden nicht in einen Pufferpool zurückgeschrieben, sondern direkt aus dem Restore-Puffer auf den Plattendatenträger.
- Wenn Sie Daten *laden* (LOAD), können Sie den Grad der E/A-Parallelität mit der Befehlsoption DISK_PARALLELISM 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.

Stellen Sie für eine optimale parallele E/A-Leistung sicher, dass folgende Voraussetzungen erfüllen sind:

- 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 den optimierten parallelen Vorablesezugriff. 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 Vorablesezugriffs, 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 Vorablesezugriffs.
  – Wenn ein Container größer ist und alle anderen Container vollständig gefüllt werden, wird dieser größere Container zum einzigen Container, in dem weitere Daten gespeichert werden. Beim Zugriff auf diese weiteren Daten kann der Datenbankmanager keinen parallelen Vorablesezugriff 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 mehreren 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 einen Vorablesezugriff 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.

Konfigurieren von IOCP unter AIX:

Bei AIX 5.3 TL9 SP2 und AIX 6.1 TL2 gehört die IOCP-Dateigruppe (I/O Completion Ports, E/A-Konfigurationsprogramm) zur Basisinstallation. Wenn Sie jedoch kein neues Betriebssystem installiert, sondern lediglich ein Betriebssystemupgrade durchgeführt haben, um die Mindestvoraussetzungen in Bezug auf das Betriebssystem zu erfüllen, müssen Sie IOCP separat konfigurieren.

Vorbereitende Schritte

Bei einer DB2-Installation mit dem Befehl **db2setup** oder **db2_install** wird IOCP aktiviert und der IOCP-Port in den Status 'verfügbar' versetzt.

Vorgehensweise

1. Geben Sie den Befehl **lslpp** ein, um zu prüfen, ob das IOCP-Modul auf Ihrem System installiert ist.

   ```
   $ lslpp -l bos.iocp.rte
   ```

   Die daraus resultierende Ausgabe entspricht in etwa dem folgenden Beispiel:

```
Fileset                          Level  State      Description
----------------------------------------------------------------------------
 Path: /usr/lib/objrepos
   bos.iocp.rte              5.3.9.0   APPLIED    I/O Completion Ports API

 Path: /etc/objrepos
   bos.iocp.rte              5.3.0.50  COMMITTED  I/O Completion Ports API
```

2. Geben Sie den Befehl **lsdev** ein, um zu prüfen, ob der Status des IOCP-Ports den Wert Available ('Verfügbar') aufweist:

   ```
   $ lsdev -Cc iocp
   ```

   Die daraus resultierende Ausgabe entspricht in etwa dem folgenden Beispiel:

   ```
   iocp0 Available  I/O Completion Ports
   ```

   Wenn der IOCP-Portstatus den Wert Defined aufweist, ändern Sie den Status in Available:

   a. Melden Sie sich am Server mit Rootberechtigung an und geben Sie den folgenden Befehl ein:

      ```
      # smitty iocp
      ```

   b. Wählen Sie die Option zum Ändern / Anzeigen der Merkmale der E/A-Ausführungsports (Change / Show Characteristics of I/O Completion Ports) aus.

   c. Ändern Sie den konfigurierten Status bei Systemneustart von Defined ('Definiert') in Available ('Verfügbar').

   d. Geben Sie den Befehl **lsdev** erneut ein und vergewissern Sie sich, dass der Status des IOCP-Ports in **Available** ('Verfügbar') geändert wurde.

Datenbankinaktivierungsverhalten bei Szenarien mit erster Benutzerverbindung

Ein Datenbank wird aktiviert, sobald von einem Benutzer eine erste Verbindung zu der Datenbank hergestellt wird. In einer Umgebung mit einer Einzelpartition wird die Datenbank in den Speicher geladen und verbleibt dort, bis der letzte Benutzer die Verbindung beendet. Dasselbe Verhalten gilt bei Umgebungen mit mehreren Partitionen. Die Datenbank wird sowohl auf lokalen als auch auf Katalogpartitionen aktiviert, sobald ein Benutzer eine erste Verbindung zu der Datenbank herstellt.

Wenn der letzte Benutzer die Verbindung zur Datenbank beendet, führt die Datenbank einen Abschluss auf lokalen und allen fernen Partitionen durch, für die der Benutzer die letzte aktive Benutzerverbindung zur Datenbank hält. Dieses an der ersten Verbindung und dem letzten Verbindungsabbau ausgerichtete Aktivierungs- und Inaktivierungsverhalten der Datenbank wird als *implizite Aktivierung* bezeichnet. Die Aktivierung wird durch die erste Benutzerverbindung ausgelöst und die Datenbank bleibt aktiv, bis der letzte Benutzer die Anweisung CONNECT RESET ausgibt (oder die Verbindung beendet oder aufhebt) und die Datenbank dadurch implizit inaktiviert wird.

Beim Laden einer Datenbank in den Speicher handelt es sich um einen aufwendigen Prozess. Er beinhaltet die Initialisierung aller Datenbankkomponenten, einschließlich der Pufferpools, und gehört zu den Verarbeitungstypen, die insbesondere in leistungskritischen Umgebungen möglichst selten ausgeführt werden sollten. Dieses Verhalten ist vor allem in Umgebungen mit mehreren Partitionen wichtig, bei denen Abfragen von einer Datenbankpartition sich auf andere Partitionen auswirken, die Teile der Zielmengen beinhalten. Diese Datenbankpartitionen werden

aktiviert und inaktiviert, wie es das Verbindungs- und Verbindungsabbauverhalten der Benutzeranwendungen erfordert. Wird von einem Benutzer eine Abfrage ausgegeben, die für eine Datenbankpartition die erste Abfrage darstellt, wird der Aufwand für die erste Aktivierung dieser Partition dieser Abfrage zugeordnet. Wird die Verbindung von dem betreffenden Benutzer getrennt, wird die Datenbank inaktiviert, sofern bis zu diesem Zeitpunkt nicht zuvor andere Verbindungen für diese ferne Partition eingerichtet wurden. Bei der nächsten eingehenden Abfrage, die einen Zugriff auf die ferne Partition erfordert, muss die betreffende Partition zunächst erneut aktiviert werden. Dieser Aufwand fällt bei jeder einzelnen Aktivierung und Inaktivierung der Datenbank (bzw. der Datenbankpartition) an.

Ausgenommen von diesem Verhalten sind nur die Fälle, in denen ein Benutzer die Datenbank explizit mit dem Befehl **ACTIVATE DATABASE** aktiviert. Wird dieser Befehl erfolgreich ausgeführt, bleibt die Datenbank im Speicher, auch wenn die letzte Benutzerverbindung zur Datenbank beendet wird. Dies gilt sowohl für Umgebungen mit Einzelpartitionen als auch für Umgebungen mit mehreren Partitionen. Geben Sie den Befehl **DEACTIVATE DATABASE** aus, wenn die Datenbank inaktiviert werden soll. Beide Befehle haben eine globale Reichweite, sodass die Datenbank gegebenenfalls, soweit erforderlich, auf allen Datenbankpartitionen aktiviert wird. Ziehen Sie deshalb angesichts des hohen Verarbeitungsaufwands, den das Laden einer Datenbank in den Speicher darstellt, gegebenenfalls ein explizites Aktivieren von Datenbanken mithilfe des Befehls **ACTIVATE DATABASE** in Betracht, statt sich auf eine implizite Aktivierung über Datenbankverbindungen zu verlassen.

Optimieren der Sortierleistung

Da Abfragen häufig sortierte oder gruppierte Ergebnisse erfordern, spielt eine geeignete Konfiguration des Sortierspeichers eine wichtige Rolle bei der Realisierung einer guten Abfrageleistung.

Sortieren ist in folgenden Fällen erforderlich:

- Es ist kein Index vorhanden, der eine angeforderte Reihenfolge (z. B. durch eine SELECT-Anweisung mit der Klausel ORDER BY) liefert.
- 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:

- Einstellungen für die folgenden Konfigurationsparameter:
  - Die Sortierspeichergröße (**sortheap**), welche die Kapazität des Speichers angibt, der für jede Sortierung verwendet wird
  - Der Schwellenwert für Sortierspeicher (**sheapthres**) und der Schwellenwert für Sortierspeicher für gemeinsame Sortiervorgänge (**sheapthres_shr**), welche die Gesamtgröße des Speichers steuern, der für das Sortieren in der Instanz 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 die Notwendigkeit von Sortierungen nicht minimiert.

- Paralleles Sortieren, das die Sortierleistung erhöht, aber nur erfolgen kann, wenn die Anweisung partitionsinterne Parallelität verwendet.
- Ob die Sortierung einen *Überlauf* verursacht hat oder nicht. Wenn die sortierten Daten nicht vollständig in den Sortierspeicher passen, bei dem es sich um einen Speicherblock handelt, der jedes Mal zugeordnet wird, wenn eine Sortierung ausgeführt wird, laufen die Daten in eine temporäre Tabellen über, deren Eigner die Datenbank ist.
- Ob die Ergebnisse der Sortierung *über eine Pipe* geleitet werden oder nicht. Wenn sortierte Daten direkt zurückgegeben werden können, ohne dass eine temporäre Tabelle zum Speichern der sortierten Liste erforderlich ist, handelt es sich um einen Sortiervorgang mit Piping (d. h. die Daten werden über eine Pipe zurückgegeben).

  Bei einer Sortierung mit Piping wird der Sortierspeicher nicht freigegeben, bevor die Anwendung den Cursor schließt, der dieser Sortierung zugeordnet ist. Eine Sortierung mit Piping kann weiter Speicher belegen, bis der Cursor geschlossen wird.

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 Sortierspeichers 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 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 Abfragen mit Sortieroperationen zu ermitteln.
3. Verwenden Sie diese Anweisungen als Eingabe für den Designadvisor, der Indizes ermittelt und auch erstellen kann, um den Sortierbedarf zu reduzieren.

Sie können den automatischen Speicheroptimierungsmanager (STMM, Self-Tuning Memory Manager) verwenden, der automatisch und dynamisch für die Sortierung erforderliche Speicherressourcen zuordnet und wieder freigibt. Gehen Sie wie folgt vor, um diese Funktion zu verwenden:

- Aktivieren Sie die automatische Speicheroptimierung für die Datenbank, indem Sie den Konfigurationsparameter **self_tuning_mem** auf den Wert ON setzen.
- Setzen Sie die Konfigurationsparameter **sortheap** und **sheapthres_shr** auf den Wert AUTOMATIC.
- Setzen Sie den Konfigurationsparameter **sheapthres** auf den Wert 0.

Sie können auch den Datenbanksystemmonitor und Vergleichstestverfahren verwenden, um geeignete Werte für die Konfigurationsparameter **sortheap**, **sheapthres_shr** und **sheapthres** zu ermitteln. Gehen Sie für jeden Datenbankmanager und jede Datenbank folgendermaßen vor:

1. Erstellen Sie eine repräsentative Auslastung und führen Sie sie aus.

2. Erfassen Sie für jede betroffene Datenbank Durchschnittswerte für die folgenden Leistungsvariablen über den Auslastungszeitraum der Vergleichstests:
   - Gesamter verwendeter Sortierspeicher (der Wert des Monitorelements **sort_heap_allocated**)
   - Aktive Sortiervorgänge und aktive Hash-Joins (die Werte der Monitorelemente **active_sorts** und **active_hash_joins**)
3. Setzen Sie den Parameter **sortheap** auf den Durchschnittswert für den *gesamten verwendeten Sortierspeicher* für jede Datenbank.

   Anmerkung: 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 mithilfe von Vergleichstests diesen Wert optimieren.

IBM InfoSphere Optim Query Workload Tuner stellt Tools für die Leistungsverbesserung einzelner SQL-Anweisungen und die Leistung von Gruppen von SQL-Anweisungen bereit, die als Abfrageworkloads bezeichnet werden. Weitere Informationen zu diesem Produkt finden Sie auf der Seite mit der Produktübersicht unter http://www.ibm.com/software/data/optim/query-workload-tuner-db2-luw/index.html. In Version 3.1.1 oder späteren Versionen des Produkts können Sie auch den Designadvisor für Workloads verwenden, um zahlreiche Operationen auszuführen, die im Assistenten für den DB2-Designadvisor verfügbar waren. Weitere Informationen finden Sie in der Dokumentation des Designadvisors für Workloads unter http://publib.boulder.ibm.com/infocenter/dstudio/v3r1/topic/com.ibm.datatools.qrytune.workloadtunedb2luw.doc/topics/genrecsdsgn.html.

Datenorganisation

Mit der Zeit können Daten in Ihren Tabellen einer gewissen Fragmentierung unterliegen, die eine Vergrößerung der Tabellen und Indizes verursacht, da Datensätze über eine zunehmende Anzahl von Datenseiten verteilt werden. Dies kann dazu führen, dass die Anzahl der Seiten ansteigt, die während der Ausführung von Abfragen gelesen werden müssen. Eine Reorganisation der Tabellen und Indizes sorgt für eine kompaktere Speicherung Ihrer Daten und gibt auf diese Weise verschwendeten Speicherplatz frei und verbessert den Datenzugriff.

Vorgehensweise

Folgende Schritte sind für eine Tabellen- oder Indexreorganisation auszuführen:

1. Ermitteln Sie, ob und welche Tabellen oder Indizes reorganisiert werden müssen.
2. Wählen Sie eine Reorganisationsmethode.

3. Führen Sie die Reorganisation für die ermittelten Objekte aus.
4. Optional: Überwachen Sie den Fortschritt der Reorganisation.
5. Stellen Sie fest, ob die Reorganisation erfolgreich war. Bei einer Offlinetabellenreorganisation und einer beliebigen Indexreorganisation erfolgt die Operation synchron; das Ergebnis ist beim Abschluss der Operation sichtbar. Bei der Onlinetabellenreorganisation erfolgt die Operation asynchron; Details können über die Verlaufsdatei abgerufen werden.
6. Erfassen Sie Statistikdaten für reorganisierte Objekte.
7. Führen Sie einen Rebind für Anwendungen durch, die auf reorganisierte Objekte zugreifen.

Tabellenreorganisation

Nach zahlreichen Änderungen an Tabellendaten können sich logisch sequenzielle Daten auf nicht sequenziellen Datenseiten befinden, sodass der Datenbankmanager zusätzliche Leseoperationen ausführen muss, um auf die Daten zuzugreifen. Wenn zahlreiche Zeilen gelöscht wurden, fallen außerdem zusätzliche Leseoperationen an. In diesem 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 auch die Systemkatalogtabellen reorganisieren.

Da die Reorganisation einer Tabelle in der Regel mehr Zeit als das Aktualisieren von Statistikdaten beansprucht, könnten Sie den Befehl **RUNSTATS** ausführen, um die Statistiken für Ihre Daten auf den aktuellen Stand zu bringen, und anschließend einen Rebind für Ihre Anwendungen ausführen. Wenn aktualisierte Statistiken keine Leistungsverbesserungen erzielen, kann eine Reorganisation eventuell helfen.

Die folgenden Faktoren können darauf hinweisen, dass eine Tabellenreorganisation erforderlich ist:

- Es hat ein hohes Aufkommen an INSERT-, UPDATE- und DELETE-Aktivitäten an Tabellen stattgefunden, auf die durch Abfragen zugegriffen wird.
- Es sind wesentliche Änderungen in der Leistung von Abfragen aufgetreten, die einen Index mit einem hohen Clusterverhältnis verwenden.
- Die Leistung wird durch die Ausführung des Befehls **RUNSTATS** zur Aktualisierung der Tabellenstatistiken nicht besser.
- Die Ausgabe des Befehls **REORGCHK** zeigt den Bedarf an einer Tabellenreorganisation an.

Anmerkung: Bei DB2 Version 9.7 Fixpack 1 und späteren Releases wird eine höhere Datenverfügbarkeit für eine datenpartitionierte Tabelle, die nur partitionierte Indizes (abgesehen von systemgenerierten XML-Pfadindizes) hat, durch eine Reorganisation der Daten für eine bestimmte Datenpartition erzielt. Die Reorganisation auf Partitionsebene ist eine Tabellenreorganisation in einer angegebenen Datenpartition, bei der der Zugriff auf die übrigen Datenpartitionen der Tabelle bestehen bleibt. Die Ausgabe des Befehls **REORGCHK** für eine partitionierte Tabelle enthält Statistikdaten und Empfehlungen für die Ausführung von Reorganisationen auf Partitionsebene.

Die Befehle **REORG TABLE** und **REORG INDEXES ALL** können für eine datenpartitionierte Tabelle ausgeführt werden, um verschiedene Datenpartitionen bzw. partitionierte Indizes für eine Partition gleichzeitig zu reorganisieren. Wenn Datenpartitionen

oder die partitionierten Indizes für eine Partition gleichzeitig reorganisiert werden, können Benutzer auf die nicht betroffenen Partitionen zugreifen. Ein Zugriff auf die betroffenen Partitionen ist nicht möglich. Für die gleichzeitige Ausführung von REORG-Befehlen für dieselbe Tabelle müssen alle folgenden Kriterien erfüllt werden:

- Jeder REORG-Befehl muss eine andere Partition mit der Klausel **ON DATA PARTITION** angeben.
- Jeder REORG-Befehl muss mit dem Modus ALLOW NO ACCESS arbeiten, um den Zugriff auf die Datenpartitionen zu beschränken.
- Die partitionierte Tabelle darf nur partitionierte Indizes haben, wenn Befehle **REORG TABLE** ausgeführt werden. Es dürfen keine nicht partitionierten Indizes (mit Ausnahme von systemgenerierten XML-Pfadindizes) für die Tabelle definiert sein.

Auswählen einer Methode zur Tabellenreorganisation

Es stehen zwei Methoden für die Tabellenreorganisation zur Verfügung: die *klassische Reorganisation* (offline) und die *Inplace-Reorganisation* (online).

Die Offlinereorganisation ist das Standardverhalten. Zur Angabe einer Onlinereorganisation verwenden Sie die Option INPLACE im Befehl **REORG TABLE**.

Es ist eine weitere Methode für die Inplace-Reorganisation verfügbar, bei der gespeicherte Prozeduren zum Versetzen von Tabellen im Online-Modus verwendet werden. Nähere Informationen hierzu finden Sie im Abschnitt „Versetzen von Tabellen mit der Prozedur ADMIN_MOVE_TABLE im Online-Modus".

Jede Methode hat ihre Vor- und Nachteile, die nachfolgend zusammengefasst werden. Bei der Auswahl einer Reorganisationsmethode müssen Sie berücksichtigen, welche Methode für Ihre Prioritäten Vorteile bietet. Wenn zum Beispiel die Wiederherstellbarkeit im Fall eines Fehlers wichtiger als die Leistung ist, ist eine Onlinereorganisation vielleicht vorzuziehen.

Vorteile der Offlinereorganisation

Diese Methode bietet die folgenden Vorteile:

- Sie führt Tabellenreorganisationsoperationen am schnellsten aus, insbesondere wenn LOB-Daten (große Objekte) oder LONG-Daten (Langfelddaten) nicht mit eingeschlossen werden.
- Tabellen und Indizes haben nach Abschluss ein perfektes Clustering.
- Indizes werden nach der Reorganisation einer Tabelle automatisch erneut erstellt, sodass kein separater Schritt zur erneuten Erstellung von Indizes erforderlich ist.
- Sie verwendet einen Tabellenbereich für temporäre Tabellen zur Erstellung einer Spiegelkopie. Dadurch verringert sich der Speicherbedarf für den Tabellenbereich, in dem die Zieltabelle oder der Index gespeichert ist.
- Sie ermöglicht die Verwendung eines anderen Index als den Clusterindex zur erneuten Herstellung eines Datenclusterings.

Nachteile der Offlinereorganisation

Diese Methode ist durch folgende Nachteile gekennzeichnet:

- Der Tabellenzugriff ist eingeschränkt: Es ist nur ein Lesezugriff während der Sortier- und Erstellungsphase einer REORG-Operation möglich.
- Ein großer Platzbedarf für die Spiegelkopie der Tabelle, die reorganisiert wird.

- Der REORG-Prozess bietet weniger Steuerungsmöglichkeiten: Eine Offlinereorganisationsoperation lässt sich nicht anhalten und wieder starten.

Vorteile der Onlinereorganisation

Diese Methode bietet die folgenden Vorteile:

- Es besteht voller Tabellenzugriff außer während der Abschneidephase einer REORG-Operation.
- Mehr Steuerungsmöglichkeiten für den REORG-Prozess, der asynchron im Hintergrund ausgeführt wird. Der Prozess kann angehalten, fortgesetzt oder gestoppt werden. Sie können zum Beispiel eine in Bearbeitung befindliche REORG-Operation anhalten, wenn eine größere Anzahl von Aktualisierungs- oder Löschoperationen für die Tabelle ausgeführt werden.
- Der Prozess ist im Fall eines Fehlers wiederherstellbar.
- Der Bedarf an Arbeitsspeicher ist geringer, weil eine Tabelle inkrementell verarbeitet wird.
- Die Vorteile der Reorganisation sind unverzüglich verfügbar, sogar vor Abschluss einer REORG-Operation.

Nachteile der Onlinereorganisation

Diese Methode ist durch folgende Nachteile gekennzeichnet:

- Nicht optimales Daten- oder Indexclustering abhängig vom Typ der Transaktionen, die während einer REORG-Operation auf die Tabelle zugreifen.
- Geringere Leistung als bei einer offline ausgeführten REORG-Operation.
- Potenziell hoher Protokollierungsbedarf abhängig von der Anzahl der Zeilen, die versetzt werden, der Anzahl von Indizes, die für die Tabelle definiert sind, und der Größe dieser Indizes.
- Potenzielle Notwendigkeit einer nachfolgenden Indexreorganisation, da Indizes gepflegt, jedoch nicht neu erstellt werden.

Tabelle 2. Vergleich zwischen Online- und Offlinereorganisation

Merkmal	**Offlinereorganisation**	**Onlinereorganisation**
Leistung	Schnell.	Langsam.
Clusterfaktor der Daten bei Abschluss	Gut.	Nicht optimal.
Gemeinsamer Zugriff (auf die Tabelle)	Reicht von 'kein Zugriff' bis 'Lesezugriff'.	Reicht von 'Lesezugriff' bis 'Vollzugriff'.
Datenspeicherplatzbedarf	Beträchtlich.	Nicht erheblich.
Protokollspeicherplatzbedarf	Nicht erheblich.	Kann beträchtlich sein.
Benutzerkontrolle (Möglichkeit zum Anhalten und Neustarten des Prozesses)	Weniger Kontrolle.	Mehr Kontrolle.
Wiederherstellbarkeit	Nicht wiederherstellbar.	Wiederherstellbar.
Indexneuerstellung	Wird ausgeführt.	Wird nicht ausgeführt.
Unterstützung für alle Tabellentypen	Ja	Nein
Möglichkeit zur Angabe eines anderen Index als des Clusterindex	Ja	Nein

Tabelle 2. Vergleich zwischen Online- und Offlinereorganisation (Forts.)

Merkmal	Offlinereorganisation	Onlinereorganisation
Verwendung eines Tabellenbereichs für temporäre Tabellen	Ja	Nein

Tabelle 3. Unterstützte Tabellentypen für Online- und Offlinereorganisationen

Tabellentyp	Offlinereorganisation unterstützt	Onlinereorganisation unterstützt
Tabellen mit mehrdimensionalem Clustering (MDC-Tabellen)	Ja[1]	Nein
Bereichsclustertabelle (RCT-Tabelle)	Nein[2]	Nein
Tabellen im Anfügemodus	Nein	Nein[3]
Tabellen mit Langfelddaten oder LOB-Daten (LOB, große Objekte)	Ja[4]	Nein
Systemkatalogtabellen: SYSIBM.SYSDBAUTH, SYSIBM.SYSROUTINEAUTH, SYSIBM.SYSSEQUENCES, SYSIBM.SYSTABLES	Ja	Nein

Anmerkungen:

1. Da das Clustering automatisch durch MDC-Blockindizes sichergestellt wird, dient die Reorganisation einer MDC-Tabelle lediglich zur Freigabe von Speicherplatz. Es können keine Indizes angegeben werden.
2. Der Bereich einer RCT-Tabelle (Bereichsclustertabelle) bleibt stets in Clustern angeordnet.
3. Eine Onlinereorganisation kann ausgeführt werden, nachdem der Anfügemodus (APPEND) inaktiviert wurde.
4. Die Reorganisation von Langfelddaten oder LOB-Daten (LOB, Large Object) kann eine geraume Zeit in Anspruch nehmen und führt zu keiner Verbesserung der Abfrageleistung. Sie sollte nur zur Freigabe von Speicherplatz ausgeführt werden.

Überwachen des Fortschritts der Tabellenreorganisation

Informationen über den Fortschritt einer laufenden REORG-Operation für eine Tabelle werden in die Protokolldatei geschrieben. Die Protokolldatei enthält einen Datensatz für jedes Reorganisationsereignis. Zum Anzeigen dieser Datei führen Sie den Befehl **LIST HISTORY** für die Datenbank aus, in der sich die Tabelle befindet, die reorganisiert wird.

Sie können den Fortschritt von REORG-Operationen für Tabellen auch mithilfe von Tabellenmomentaufnahmen überwachen. Überwachungsdaten zur Tabellenreorganisation werden unabhängig von der Einstellung des Datenbankmonitorschalters für Tabellen aufgezeichnet.

Wenn ein Fehler auftritt, wird eine SQLCA-Nachricht in die Protokolldatei geschrieben. Bei einer Inplace-REORG-Operation für eine Tabelle wird der Status mit PAUSED ('Angehalten') aufgezeichnet.

Klassische Tabellenreorganisation (offline)

Bei der klassischen Tabellenreorganisation wird ein Spiegelkopieverfahren verwendet, bei dem eine vollständige Kopie der zu reorganisierenden Tabelle erstellt wird.

Die Operation der klassischen bzw. offline ausgeführten Tabellenreorganisation hat vier Phasen:

1. Sortierung (SORT) - Wenn ein Index im Befehl **REORG TABLE** angegeben wurde oder ein Clusterindex für die Tabelle definiert wurde, werden die Zeilen der Tabelle in dieser Phase zunächst nach diesem Index sortiert. Wenn die Option INDEXSCAN angegeben wird, wird eine Indexsuche zur Sortierung der Tabelle verwendet. Anderenfalls wird eine Tabellensuche zur Sortierung verwendet. Diese Phase wird nur bei einer REORG-Operation für eine Tabelle mit Clustering ausgeführt. REORG-Operationen zur Freigabe von Speicherplatz beginnen mit der Erstellungsphase.
2. Erstellung (BUILD) - In dieser Phase wird eine reorganisierte Kopie der gesamten Tabelle erstellt. Dies geschieht entweder im Tabellenbereich der Tabelle oder in einem Tabellenbereich für temporäre Tabellen, der im Befehl **REORG TABLE** angegeben wurde.
3. Ersetzung (REPLACE) - In dieser Phase wird das ursprüngliche Tabellenobjekt durch eine Kopie aus dem Tabellenbereich für temporäre Tabellen ersetzt oder es wird ein Verweis auf das neu erstellte Objekt im Tabellenbereich der Tabelle erstellt, die reorganisiert wird.
4. Neuerstellung aller Indizes (RECREATE ALL INDEXES) - In dieser Phase werden alle Indizes, für die Tabelle definiert waren, neu erstellt.

Mithilfe von Snapshot Monitor oder Verwaltungssichten mit Momentaufnahmen können Sie den Fortschritt der REORG-Operation überwachen und die aktuelle Phase erkennen.

Im Offlinemodus sind die Sperrbedingungen restriktiver als im Onlinemodus. Während der Erstellung der Kopie ist ein Lesezugriff auf die Tabelle verfügbar. Allerdings ist ein exklusiver Zugriff auf die Tabelle erforderlich, wenn die ursprüngliche Tabelle durch die reorganisierte Kopie ersetzt wird oder Indizes erneut erstellt werden.

Eine IX-Tabellenbereichssperre ist während des gesamten REORG-Prozesses für die Tabelle erforderlich. Während der Erstellungsphase (Build) wird eine U-Sperre für die Tabelle aktiviert und beibehalten. Eine U-Sperre ermöglicht dem Sperreneigner, die Daten in der Tabelle zu aktualisieren. Obwohl keine andere Anwendung die Daten aktualisieren kann, wird ein Lesezugriff zugelassen. Die U-Sperre wird auf eine Z-Sperre hochgestuft, wenn die Ersetzungsphase (Replace) gestartet wurde. Während dieser Phase können keine anderen Anwendungen auf die Daten zugreifen. Diese Sperre wird bis zum Abschluss der REORG-Operation beibehalten.

Durch den offline ausgeführten Reorganisationsprozess wird eine Reihe von Dateien erstellt. Diese Dateien werden in Ihrem Datenbankverzeichnis erstellt. Den Namen dieser Dateien werden die Tabellenbereichs- und Objekt-IDs vorangestellt. Zum Beispiel ist `0030002.ROR` die Statusdatei für die REORG-Operation einer Tabelle mit der Tabellenbereichs-ID 3 und der Tabellen-ID 2.

In der folgenden Liste sind die temporären Dateien aufgeführt, die bei einer offline ausgeführten REORG-Operation für eine Tabelle in einem SMS-Tabellenbereich (SMS - vom System verwalteter Speicherbereich) erstellt werden:

- .DTR - Datendatei der Spiegelkopie

- .LFR - Datei für Langfelddaten (LONG)
- .LAR - Zuordnungsdatei für Langfelddaten
- .RLB - Datei für LOB-Daten
- .RBA - Zuordnungsdatei für LOB-Daten
- .BMR - Blockobjektdatei für Tabellen mit mehrdimensionalem Clustering (MDC-Tabellen)

Die folgende temporäre Datei wird bei einer REORG-Operation für einen Index erstellt:

- .IN1 - Spiegelkopiedatei

In der folgenden Liste sind die temporären Dateien aufgeführt, die im Tabellenbereich für temporäre Systemtabellen während der Sortierungsphase (Sort) erstellt werden:

- .TDA - Datendatei
- .TIX - Indexdatei
- .TLF - Datei für Langfelddaten (LONG)
- .TLA - Zuordnungsdatei für Langfelddaten
- .TLB - Datei für LOB-Daten
- .TBA - Zuordnungsdatei für LOB-Daten
- .TBM - Blockobjektdatei

Die dem Reorganisationsprozess zugeordneten Dateien sollten nicht manuell aus dem System entfernt werden.

Ausführen von Offlinetabellenreorganisationen:

Offline ausgeführte Tabellenreorganisationen sind die schnellste Methode zur Defragmentierung von Tabellen. Die Reorganisation verringert den Speicherplatzbedarf einer Tabelle und verbessert den Datenzugriff und die Abfrageleistung.

Vorbereitende Schritte

Sie benötigen die Berechtigung SYSADM, SYSCTRL, SYSMAINT, DBADM oder SQLADM oder das Zugriffsrecht CONTROL für die Tabelle, die reorganisiert werden soll. Darüber hinaus benötigen Sie eine Datenbankverbindung zum Reorganisieren einer Tabelle.

Informationen zu diesem Vorgang

Nachdem Sie die Tabellen ermittelt haben, für die eine Reorganisation erforderlich ist, können Sie das Dienstprogramm REORG für diese Tabellen und optional für die Indizes, die für diese Tabellen definiert sind, ausführen.

Vorgehensweise

1. Zum Reorganisieren einer Tabelle mit dem Befehl **REORG TABLE** geben Sie den Namen der Tabelle an. Beispiel:

   ```
   reorg table employee
   ```

   Sie können beim Reorganisieren einer Tabelle einen bestimmten temporären Tabellenbereich verwenden. Beispiel:

   ```
   reorg table employee use mytemp
   ```

Sie können beim Reorganisieren einer Tabelle die Zeilen auf der Basis eines bestimmten Index neu sortieren. Zum Beispiel:

```
reorg table employee index myindex
```

2. Zum Durchführen einer Tabellenreorganisation mit einer SQL-Anweisung CALL führen Sie den Befehl **REORG TABLE** mit der Prozedur ADMIN_CMD aus. Beispiel:

   ```
   call sysproc.admin_cmd ('reorg table employee')
   ```

3. Zum Durchführen einer Tabellenreorganisation über die Verwaltungs-API rufen Sie die API db2Reorg auf.

Nächste Schritte

Im Anschluss an die Reorganisation einer Tabelle erfassen Sie Statistikdaten für diese Tabelle, damit dem Optimierungsprogramm die präzisesten Daten zur Auswertung von Abfragezugriffsplänen zur Verfügung stehen.

Recovery einer Offlinetabellenreorganisation:

Eine offline ausgeführte Tabellenreorganisation ist bis zum Beginn der Ersetzungsphase (Replace) ein Alles-oder-nichts-Prozess. Wenn das System während der Sortier- oder Erstellungsphase abstürzt, wird die REORG-Operation per Rollback rückgängig gemacht und bei der Recovery nach dem Systemabsturz nicht erneut ausgeführt.

Falls das System nach dem Beginn der Ersetzungsphase (Replace) abstürzt, muss die REORG-Operation abgeschlossen werden, weil alle Arbeiten erledigt wurden und die ursprüngliche Tabelle möglicherweise nicht mehr verfügbar ist. Während der Recovery nach einem Systemabsturz ist die temporäre Datei für das reorganisierte Tabellenobjekt erforderlich, nicht jedoch der Tabellenbereich für temporäre Tabellen, der für die Sortierung verwendet wird. Die Recovery startet die Ersetzungsphase erneut von Beginn an. Daher sind alle Daten im Objekt der Kopie für die Recovery erforderlich. In diesem Fall besteht ein Unterschied zwischen Tabellenbereichen, die vom System verwaltet werden (SMS), und Tabellenbereichen, die von der Datenbank verwaltet werden (DMS): Im SMS-Tabellenbereich muss das reorganisierte Tabellenobjekt von einem Objekt in das andere kopiert werden, während im DMS-Tabellenbereich nur ein Verweis auf das reorganisierte Tabellenobjekt eingefügt und die ursprüngliche Tabelle gelöscht werden muss, wenn die Reorganisation im selben Tabellenbereich ausgeführt wurde. Indizes werden nicht erneut erstellt. Jedoch werden sie während der Recovery nach dem Systemabsturz als ungültig markiert, und die Datenbank folgt den normalen Regeln zur Bestimmung, wann sie erneut erstellt werden, das heißt, entweder beim Neustart der Datenbank oder beim ersten Indexzugriff.

Wenn es während der Phase der Indexneuerstellung zu einem Systemabsturz kommt, wird keine Aktion ausgeführt, weil das neue Tabellenobjekt bereits vorhanden ist. Indizes werden wie zuvor beschrieben bearbeitet.

Bei einer aktualisierenden Recovery wird die REORG-Operation wiederholt, wenn die alte Version der Tabelle auf Platte vorhanden ist. Die aktualisierende Recovery verwendet die Satz-IDs (RIDs), die während der Erstellungsphase protokolliert wurden, um die Operationen, die zur Erstellung der reorganisierten Tabelle ausgeführt wurden, erneut anzuwenden und dadurch die Erstellungs- und die Ersetzungsphase zu wiederholen. Indizes werden wie zuvor beschrieben bearbeitet. Ein Tabellenbereich für temporäre Tabellen ist für eine Kopie des reorganisierten Objekts nur dann erforderlich, wenn ursprünglich ein Tabellenbereich für temporäre

Tabellen verwendet wurde. Bei einer aktualisierenden Recovery können mehrere REORG-Operationen gleichzeitig wiederholt werden (parallele Recovery).

Verbessern der Leistung von Offlinetabellenreorganisationen:

Die Leistung einer Offlinetabellenreorganisation wird weitgehend durch die Merkmale der Datenbankumgebung bestimmt.

Es gibt nahezu keinen Unterschied zwischen der Leistung einer REORG-Operation, die im Modus ALLOW NO ACCESS ausgeführt wird, und einer, die im Modus ALLOW READ ACCESS ausgeführt wird. Der Unterschied besteht darin, dass bei einer REORG-Operation im Modus ALLOW READ ACCESS das Dienstprogramm eventuell darauf warten muss, dass andere bereits aktive Anwendungen ihre Suchläufe abschließen und ihre Sperren freigeben, bevor es die Tabelle ersetzt. Die Tabelle ist in der Phase der Indexneuerstellung einer REORG-Operation in beiden Modi nicht verfügbar.

Tipps zur Leistungsverbesserung

- Wenn genügend Speicherplatz vorhanden ist, verwenden Sie denselben Tabellenbereich für die ursprüngliche Tabelle und die reorganisierte Kopie der Tabelle, anstatt einen Tabellenbereich für temporäre Tabellen zu verwenden. Dies spart die Zeit, die zum Kopieren der reorganisierten Tabelle aus dem Tabellenbereich für temporäre Tabellen erforderlich ist.
- Ziehen Sie in Betracht, vor der Reorganisation einer Tabelle nicht benötigte Indizes zu löschen, damit während der REORG-Operation weniger Indizes gepflegt werden müssen.
- Stellen Sie sicher, dass die Vorablesezugriffsgröße (PREFETCHSIZE) der Tabellenbereiche, in denen sich die zu reorganisierende Tabelle befindet, ordnungsgemäß definiert ist.
- Optimieren Sie die Datenbankkonfigurationsparameter **sortheap** und **sheapthres**, um den für Sortierungen verfügbaren Speicherbereich zu steuern. Da jeder Prozessor eine private Sortieroperation ausführt, sollte der Parameter **sheapthres** mindestens den Wert **sortheap** x *anzahl_prozessoren* haben.
- Passen Sie die Anzahl der Seitenlöschfunktionen an, um sicherzustellen, dass benutzte Indexseiten im Pufferpool möglichst rasch bereinigt werden.

Inplace-Tabellenreorganisation (online)

Durch eine Inplace-Tabellenreorganisation können Sie eine Tabelle reorganisieren und gleichzeitig vollen Zugriff auf die Daten der Tabelle behalten. Die Kosten dieses ununterbrochenen Zugriffs auf die Daten liegen in einer langsameren Ausführung der REORG-Operation für die Tabelle.

Bei einer inplace bzw. online ausgeführten REORG-Operation für eine Tabelle werden Teile der Tabelle sequenziell reorganisiert. Die Daten werden nicht in einen Tabellenbereich für temporäre Tabellen kopiert, sondern es werden Zeilen innerhalb des bestehenden Tabellenobjekts versetzt, um das Clustering wiederherzustellen, ungenutzte Speicherbereiche wieder freizugeben und Überlaufzeilen zu beseitigen.

Eine online ausgeführte REORG-Operation für eine Tabelle hat vier Hauptphasen:

1. Auswahl von *n* Seiten

   Während dieser Phase wählt der Datenbankmanager einen Bereich von *n* Seiten für die REORG-Verarbeitung aus, wobei *n* die Größe eines Speicherbereichs mit mindestens 32 sequenziellen Seiten ist.

2. Freimachen des Bereichs

   Das Dienstprogramm REORG versetzt alle Zeilen innerhalb dieses Bereichs, um Seiten in der Tabelle freizugeben. Jede Zeile, die versetzt wird, hinterlässt einen REORG-Tabellenverweissatz (RP - Reorg Pointer), der die Satz-ID (RID) der neuen Position der Zeile enthält. Die Zeile wird auf einer freien Seite in der Tabelle als REORG-Tabellenüberlaufsatz (RO - Reorg Overflow) platziert, der die Daten enthält. Wenn das Dienstprogramm das Versetzen einer Gruppe von Zeilen beendet hat, wartet es, bis alle Anwendungen, die auf die Daten in der Tabelle zugreifen, beendet sind. Diese „alten Suchvorgänge" verwenden die alten Satz-IDs (RIDs) beim Zugriff auf die Tabellendaten. Jeder Tabellenzugriff, der während dieser Wartephase („neuer Suchvorgang") gestartet wird, verwendet neue Satz-IDs (RIDs) für den Zugriff auf die Daten. Wenn alle alten Suchvorgänge abgeschlossen sind, bereinigt das Dienstprogramm REORG die versetzten Zeilen, indem es RP-Datensätze löscht und RO-Datensätze in reguläre Datensätze umwandelt.

3. Füllen des Bereichs

   Nachdem alle Zeilen aus einem bestimmten Bereich entfernt wurden, werden sie in einem reorganisierten Format, nach den verwendeten Indizes sortiert und gemäß den definierten PCTFREE-Einschränkungen zurückgeschrieben. Wenn alle Seiten im Bereich zurückgeschrieben wurden, werden die nächsten *n* sequenziellen Seiten in der Tabelle ausgewählt und der Prozess wiederholt.

4. Abschneiden der Tabelle

   Standardmäßig wird die Tabelle abgeschnitten, nachdem alle Seiten in der Tabelle reorganisiert wurden, um ungenutzten Speicherplatz wieder freizugeben. Wenn die Option NOTRUNCATE angegeben wurde, wird die reorganisierte Tabelle nicht abgeschnitten.

Bei einer online ausgeführten REORG-Operation für eine Tabelle erstellte Dateien

Bei einer online ausgeführten REORG-Operation für eine Tabelle wird eine Statusdatei (`.OLR`) für jede Datenbankpartition erstellt. Diese Binärdatei hat einen Namen im Format `xxxxyyyy.OLR`, wobei *xxxx* die Tabellenbereichs-ID und *yyyy* die Objekt-ID im Hexadezimalformat ist. Diese Datei enthält die folgenden Informationen, die zur Fortsetzung einer online ausgeführten REORG-Operation im angehaltenen Status erforderlich sind:

- Den Typ der REORG-Operation
- Die aktuelle Protokollfolgenummer (LSN) der Tabelle, die reorganisiert wird
- Den nächsten frei zu machenden Bereich
- Informationen dazu, ob die REORG-Operation die Daten in Clustern ordnet oder nur Speicherplatz freigibt
- Die ID des Index, der für das Clustering der Daten verwendet wird

Für die `.OLR`-Datei wird eine Kontrollsumme berechnet. Wenn die Datei beschädigt wird und Kontrollsummenfehler verursacht oder wenn die Tabellenprotokollfolgenummer (LSN) nicht mit der aktiven Protokollfolgenummer übereinstimmt, muss eine neue REORG-Operation eingeleitet werden, sodass eine neue Statusdatei erstellt wird.

Wenn die `.OLR`-Statusdatei gelöscht wird, kann der REORG-Prozess nicht fortgesetzt werden und eine Nachricht SQL2219N wird zurückgegeben. Es muss eine neue REORG-Operation eingeleitet werden.

Die dem Reorganisationsprozess zugeordneten Dateien sollten nicht manuell aus dem System entfernt werden.

Ausführen von Onlinetabellenreorganisationen:

Eine online bzw. mit der Option INPLACE ausgeführte Tabellenreorganisation ermöglicht Benutzern den Zugriff auf eine Tabelle, während diese reorganisiert wird.

Vorbereitende Schritte

Sie benötigen die Berechtigung SYSADM, SYSCTRL, SYSMAINT, DBADM oder SQLADM oder das Zugriffsrecht CONTROL für die Tabelle, die reorganisiert werden soll. Darüber hinaus benötigen Sie eine Datenbankverbindung zum Reorganisieren einer Tabelle.

Informationen zu diesem Vorgang

Nachdem Sie die Tabellen ermittelt haben, für die eine Reorganisation erforderlich ist, können Sie das Dienstprogramm REORG für diese Tabellen und optional für die Indizes, die für diese Tabellen definiert sind, ausführen.

Vorgehensweise

1. Zum Durchführen einer Onlinetabellenreorganisation mit dem Befehl **REORG TABLE** müssen Sie den Namen der Tabelle und die Option INPLACE angeben. Beispiel:

   ```
   reorg table employee inplace
   ```

2. Zum Durchführen einer Onlinetabellenreorganisation mit einer SQL-Anweisung CALL führen Sie den Befehl **REORG TABLE** mit der Prozedur ADMIN_CMD aus. Beispiel:

   ```
   call sysproc.admin_cmd ('reorg table employee inplace')
   ```

3. Zum Durchführen einer Onlinetabellenreorganisation über die Verwaltungs-API rufen Sie die API db2Reorg auf.

Ergebnisse

Nächste Schritte

Im Anschluss an die Reorganisation einer Tabelle erfassen Sie Statistikdaten für diese Tabelle, damit dem Optimierungsprogramm die präzisesten Daten zur Auswertung von Abfragezugriffsplänen zur Verfügung stehen.

Recovery einer Onlinetabellenreorganisation:

Das Fehlschlagen einer Onlinetabellenreorganisation ist häufig auf Verarbeitungsfehler zurückzuführen, wie sie zum Beispiel durch einen vollen Datenträger oder durch Protokollierungsfehler verursacht werden. Wenn eine Onlinetabellenreorganisation fehlschlägt, wird eine Nachricht des SQL-Kommunikationsbereichs (SQLCA) in die Protokolldatei geschrieben.

Wenn während der Ausführung ein Fehler auftritt, wird die online ausgeführte REORG-Operation für eine Tabelle angehalten und anschließend bei der Recovery nach Systemabsturz durch Rollback rückgängig gemacht. Nachfolgend können Sie die REORG-Operation fortsetzen, indem Sie die Option RESUME im Befehl **REORG TABLE** angeben. Da der Prozess vollständig protokolliert wird, ist eine online ausgeführte Tabellenreorganisation garantiert wiederherstellbar.

Unter bestimmten Umständen kann eine online ausgeführte REORG-Operation für eine Tabelle den Grenzwert überschreiten, der durch den Wert des Datenbankkonfigurationsparameters **num_log_span** festgelegt ist. In diesem Fall versetzt der Datenbankmanager das Dienstprogramm REORG zwangsweise in den Pausestatus (PAUSE). In der Ausgabe von Snapshot Monitor wird der Status des Dienstprogramms mit 'Angehalten' (PAUSED) angezeigt.

Das Anhalten der Onlinetabellenreorganisation erfolgt interruptgesteuert. Dies bedeutet, dass es entweder durch einen Benutzer (mit der Option PAUSE im Befehl **REORG TABLE** bzw. mit dem Befehl **FORCE APPLICATION**) oder unter bestimmten Umständen durch den Datenbankmanager ausgelöst werden kann, zum Beispiel wenn es zu einem Systemabsturz kommt.

Wenn eine oder mehrere Datenbankpartitionen in einer Umgebung mit partitionierten Datenbanken einen Fehler feststellen, wird der SQLCODE-Wert der ersten Datenbankpartition, die einen Fehler meldet, zurückgegeben.

Anhalten und erneutes Starten einer Onlinetabellenreorganisation:

Eine Onlinetabellenreorganisation, die gerade ausgeführt wird, kann durch den Benutzer angehalten und wieder gestartet werden.

Vorbereitende Schritte

Sie benötigen die Berechtigung SYSADM, SYSCTRL, SYSMAINT, DBADM oder SQLADM oder das Zugriffsrecht CONTROL für die Tabelle, deren Onlinereorganisation angehalten oder erneut gestartet werden soll. Darüber hinaus benötigen Sie eine Datenbankverbindung zum Anhalten oder erneuten Starten einer Onlinetabellenreorganisation.

Vorgehensweise

1. Zum Anhalten einer Onlinetabellenreorganisation mit dem Befehl **REORG TABLE** müssen Sie den Namen der Tabelle, die Option INPLACE und die Option PAUSE angeben. Beispiel:

   ```
   reorg table employee inplace pause
   ```

2. Zum erneuten Starten einer angehaltenen Onlinetabellenreorganisation müssen Sie die Option RESUME angeben. Beispiel:

   ```
   reorg table employee inplace resume
   ```

   Wenn eine Onlinereorganisation einer Tabelle angehalten wurde, können Sie keine neue Reorganisation dieser Tabelle starten. Sie müssen die angehaltene Reorganisation fortsetzen oder vollständig stoppen, bevor Sie einen neuen Reorganisationsprozess starten.

   Nach einer Anforderung RESUME beachtet der Reorganisationsprozess alle Optionen für das Abschneiden, die in der aktuellen Anforderung RESUME angegeben sind. Wenn beispielsweise in der aktuellen Anforderung RESUME die Option NOTRUNCATE nicht angegeben ist, wird eine Option NOTRUNCATE, die im ursprünglichen Befehl **REORG TABLE** oder einer vorhergehenden Anforderung RESUME angegeben wurde, ignoriert.

   Eine Tabellenreorgansisationsoperation kann nicht wieder aufgenommen werden, nachdem eine RESTORE- und ROLLFORWARD-Operation durchgeführt wurde.

Hinweise zu Sperren und zum gemeinsamem Zugriff bei Onlinetabellenreorganisationen:

Einer der wichtigsten Aspekte einer Onlinetabellenreorganisation ist die Steuerung von Sperren, da sie für den gemeinsamen Zugriff durch Anwendungen von entscheidender Bedeutung ist.

Eine online ausgeführte REORG-Operation für eine Tabelle kann die folgenden Sperren aktivieren:

- Zur Sicherstellung des Schreibzugriffs auf Tabellenbereiche wird eine IX-Sperre für die Tabellenbereiche aktiviert, die von der REORG-Operation betroffen sind.
- Eine Tabellensperre wird aktiviert und die gesamte REORG-Operation hindurch beibehalten. Die Sperrebene hängt von dem Zugriffsmodus ab, der für die Reorganisation gilt:
  - Wenn ALLOW WRITE ACCESS angegeben wurde, wird eine IS-Tabellensperre aktiviert.
  - Wenn ALLOW READ ACCESS angegeben wurde, wird eine S-Tabellensperre aktiviert.
- Während der Abschneidephase wird eine S-Sperre für die Tabelle angefordert. Bis die S-Sperre aktiviert wird, können Zeilen von gleichzeitig ausgeführten Transaktionen eingefügt werden. Diese eingefügten Zeilen werden vom Dienstprogramm REORG möglicherweise nicht erkannt und können verhindern, dass die Tabelle abgeschnitten wird. Nachdem die S-Tabellensperre aktiviert ist, werden Zeilen, die verhindern, dass die Tabelle abgeschnitten wird, versetzt, um die Tabelle zu komprimieren. Nach der Komprimierung wird die Tabelle abgeschnitten, jedoch erst, wenn alle Transaktionen, die auf die Tabelle zu dem Zeitpunkt zugreifen, der als Abschneidepunkt bestimmt wurde, abgeschlossen sind.
- Eine Zeilensperre kann abhängig vom Typ der Tabellensperre aktiviert werden:
  - Wenn eine S-Sperre für die Tabelle aktiviert ist, sind keine einzelnen S-Sperren auf Zeilenebene erforderlich, sodass keine weiteren Sperren benötigt werden.
  - Wenn für die Tabelle eine IS-Sperre aktiviert ist, wird eine NS-Zeilensperre aktiviert, bevor die Zeile versetzt wird, und wieder freigegeben, wenn die Versetzung ausgeführt ist.
- Bestimmte interne Sperren können bei einer online ausgeführten REORG-Operation für eine Tabelle ebenfalls aktiviert werden.

Das Sperren hat Auswirkungen auf die Leistung sowohl für online ausgeführte REORG-Operationen als auch für gleichzeitig ausgeführte Benutzeranwendungen. Mithilfe von Momentaufnahmedaten zu Sperren können Sie die Sperraktivitäten untersuchen, die während der Ausführung von Onlinetabellenreorganisationen stattfinden.

Überwachen einer Tabellenreorganisation

Mithilfe des Befehls **GET SNAPSHOT**, der Verwaltungssicht SNAPTAB_REORG oder der Tabellenfunktion SNAP_GET_TAB_REORG können Sie Informationen über den Status Ihrer Operationen zur Tabellenreorganisation abrufen.

Vorgehensweise

- Für den Zugriff auf Informationen zu Reorganisationsoperationen mithilfe von SQL verwenden Sie die Verwaltungssicht SNAPTAB_REORG. Die folgende Abfrage gibt beispielsweise Details zu Tabellenreorganisationsoperationen für alle Datenbankpartitionen der momentan verbundenen Datenbank zurück. Wenn keine Tabellen reorganisiert wurden, werden keine Zeilen zurückgegeben.

```
select
    substr(tabname, 1, 15) as tab_name,
    substr(tabschema, 1, 15) as tab_schema,
    reorg_phase,
```

```
  substr(reorg_type, 1, 20) as reorg_type,
  reorg_status,
  reorg_completion,
  dbpartitionnum
from sysibmadm.snaptab_reorg
order by dbpartitionnum
```

- Für den Zugriff auf Informationen zu Reorganisationsoperationen mithilfe von Snapshot Monitor verwenden Sie den Befehl **GET SNAPSHOT FOR TABLES** und überprüfen Sie die Werte der Monitorelemente für die Tabellenreorganisation.

Ergebnisse

Da Offlinetabellenreorganisationen synchron sind, werden Fehler an das aufrufende Programm des Dienstprogramms (d. h. an die Anwendung oder den Befehlszeilenprozessor) zurückgegeben. Da Onlinetabellenreorganisationen asynchron sind, werden Fehlernachrichten in diesem Fall nicht an den Befehlszeilenprozessor zurückgegeben. Wenn Sie SQL-Fehlernachrichten anzeigen möchten, die während einer Onlinetabellenreorganisation zurückgegeben werden, verwenden Sie den Befehl **LIST HISTORY REORG**.

Eine Onlinetabellenreorganisation wird im Hintergrund als Prozess mit dem Namen db2Reorg ausgeführt. Dieser Prozess wird weiter ausgeführt, auch wenn die aufrufende Anwendung ihre Datenbankverbindung beendet.

Indexreorganisation

Im Lauf der Aktualisierungen von Tabellen kann sich die Indexleistung verschlechtern.

Die Verschlechterung kann auf folgende Weisen auftreten:

- Blattseiten werden fragmentiert. 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, was zu einem Index mit *schlechter Clusterbildung* führt. Wenn Blattseiten eine schlechte Clusterbildung aufweisen, ist ein sequenzieller Vorablesezugriff nicht effizient und die Anzahl der Wartezeiten auf E/A-Vorgänge erhöht sich.
- Der Index entwickelt zu viele Stufen. In einem solchen Fall sollte der Index reorganisiert werden.

Eine Indexreorganisation hat folgende Voraussetzungen:

- Die Berechtigung SYSADM, SYSMAINT, SYSCTRL, DBADM oder SQLADM oder das Zugriffsrecht CONTROL für die Tabelle und ihre Indizes.
- 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 in einem großen Tabellenbereich anzulegen, wenn Sie die Anweisung CREATE TABLE ausführen.
- Zusätzlicher Protokollspeicherbereich. Das Dienstprogramm für die Indexreorganisation (REORG) protokolliert seine Aktivitäten.

Wenn Sie die Option MINPCTUSED in der Anweisung CREATE INDEX angeben, 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.
- Verwenden des Befehls **REORG TABLE** mit Optionen, die eine Offlinereorganisation der Tabelle und ihrer Indizes ermöglichen.
- Reorganisieren von Indizes online mithilfe 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.

Onlineindexreorganisation

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 weiterhin möglich ist. Während der Reorganisation werden alle Änderungen an der zugrunde liegenden Tabelle, die sich auf die Indizes auswirken würden, protokolliert. Die REORG-Operation verarbeitet diese protokollierten Änderungen bei der erneuten Erstellung der Indizes.

Änderungen an der zugrunde liegenden Tabelle, die sich auf die Indizes auswirken würden, werden auch in einen internen Hauptspeicherpuffer geschrieben, wenn ein solcher Speicherbereich zur Verwendung verfügbar ist. Der interne Puffer ist ein designierter Speicherbereich, der bei Bedarf aus dem Dienstprogrammspeicher zugeordnet wird. Durch Verwendung eines Hauptspeicherpufferbereichs kann das Dienstprogramm REORG die Änderungen verarbeiten, indem es zuerst direkt aus dem Hauptspeicher liest und anschließend, falls erforderlich, in den Protokollen, jedoch zu einem wesentlich späteren Zeitpunkt, liest. Der zugeordnete Speicher wird nach Beendigung der REORG-Operation wieder freigegeben.

Die Online-Indexreorganisation im ALLOW WRITE ACCESS-Modus (mit oder ohne die Option CLEANUP) wird für räumliche Indizes oder MDC-Tabellen (MDC, mehrdimensionales Clustering) nicht unterstützt.

Bei DB2 Version 9.7 Fixpack 1 und späteren Releases wird durch den Befehl **REORG INDEXES ALL** für eine datenpartitionierte Tabelle unter Angabe einer Partition mit der Klausel `ON DATA PARTITION` eine Reorganisation der partitionierten Indizes für eine einzelne Datenpartition ausgeführt. Während der Indexreorganisation bleibt der Zugriff auf die nicht betroffenen Partitionen bestehen. Nur der Lese- und Schreibzugriff auf die betroffene Partition wird beschränkt.

Die Befehle **REORG TABLE** und **REORG INDEXES ALL** können für eine datenpartitionierte Tabelle ausgeführt werden, um verschiedene Datenpartitionen bzw. partitionierte Indizes für eine Partition gleichzeitig zu reorganisieren. Wenn Datenpartitionen oder die partitionierten Indizes für eine Partition gleichzeitig reorganisiert werden, können Benutzer auf die nicht betroffenen Partitionen zugreifen. Für die gleichzeitige Ausführung von REORG-Befehlen für dieselbe Tabelle müssen alle folgenden Kriterien erfüllt werden:

- Jeder REORG-Befehl muss eine andere Partition mit der Klausel **ON DATA PARTITION** angeben.
- Jeder REORG-Befehl muss mit dem Modus ALLOW NO ACCESS arbeiten, um den Zugriff auf die Datenpartitionen zu beschränken.

- Die partitionierte Tabelle darf nur partitionierte Indizes haben, wenn Befehle **REORG TABLE** ausgeführt werden. Es dürfen keine nicht partitionierten Indizes (mit Ausnahme von systemgenerierten XML-Pfadindizes) für die Tabelle definiert sein.

Anmerkung: Die Ausgabe des Befehls **REORGCHK** enthält Statistikdaten und Empfehlungen für die Reorganisation von Indizes. Für eine partitionierte Tabelle enthält die Ausgabe Statistikdaten und Empfehlungen für die Reorganisation von partitionierten und nicht partitionierten Indizes.

Hinweise zu Sperren und zum gemeinsamem Zugriff bei Onlineindexreorganisationen

Der Begriff 'Onlineindexreorganisation' bezieht sich auf Indexreorganisationen, die mit der Option ALLOW READ ACCESS oder ALLOW WRITE ACCESS ausgeführt werden. Diese Optionen ermöglichen es Benutzern, auf die Tabelle zuzugreifen, während die zugehörigen Indizes reorganisiert werden. Bei der Onlineindexreorganisation werden neue Indizes als zusätzliche Kopien erstellt, während die ursprünglichen Indizes intakt bleiben. Gleichzeitig ablaufende Transaktionen verwenden die ursprünglichen Indizes, während die neuen Indizes erstellt werden. Am Ende der Reorganisationsoperation werden die ursprünglichen Indizes durch die neuen Indizes ersetzt. Transaktionen, die in der Zwischenzeit festgeschrieben werden, werden nach dem Ersetzen der ursprünglichen Indizes in den neuen Indizes reflektiert. Wenn die Reorganisationsoperation fehlschlägt und die Transaktion rückgängig gemacht wird, bleiben die ursprünglichen Indizes intakt.

Eine Onlineindexreorganisation kann die folgenden Sperren aktivieren:

- Zur Sicherstellung des Zugriffs auf Tabellenbereiche wird eine IX-Sperre für die Tabellenbereiche aktiviert, die von der Reorganisationsoperation betroffen sind. Hierzu gehören die Tabellenbereiche mit der Tabelle, der Partition und Indexobjekten.
- Um zu verhindern, dass die betroffene Tabelle während der Reorganisation geändert wird, wird eine X-Sperre für Tabellenänderungen aktiviert.
- Eine Tabellensperre wird aktiviert und während der Reorganisationsoperation gehalten. Der Sperrentyp hängt vom Tabellentyp, vom Zugriffsmodus und von der Reorganisationsoption ab:
  - Für nicht partitionierte Tabellen:
    - Bei der Angabe von ALLOW READ ACCESS wird eine U-Sperre (Aktualisierungssperre) für die Tabelle aktiviert.
    - Bei der Angabe von ALLOW WRITE ACCESS wird eine IN-Sperre für die Tabelle aktiviert.
    - Bei der Angabe von CLEANUP ONLY wird für die Tabelle eine S-Sperre für den Lesezugriff aktiviert und eine IX-Sperre für den Schreibzugriff.
  - Für partitionierte Tabellen wird die Reorganisation mit ALLOW READ oder WRITE ACCESS nur auf der Partitionsebene unterstützt:
    - Bei der Angabe von ALLOW READ ACCESS wird eine Aktualisierungssperre (U-Sperre) für die Partition aktiviert.
    - Bei der Angabe von ALLOW WRITE ACCESS wird eine IS-Sperre für die Partition aktiviert.
    - Bei der Angabe von CLEANUP ONLY wird für die Partition eine S-Sperre für den Lesezugriff aktiviert und eine IX-Sperre für den Schreibzugriff.
    - Für die Tabelle wird unabhängig vom angegebenen Zugriffsmodus und der angegebenen Option eine IS-Sperre aktiviert.

- Am Ende der Indexreorganisation wird eine exklusive Z-Sperre für die Tabelle bzw. Partition angefordert. Wenn eine partitionierte Tabelle nicht partitionierte Indizes enthält, wird eine Z-Sperre sowohl für die Tabelle als auch für die Partition aktiviert. Durch diese Sperre wird der Tabellen- und Partitionszugriff ausgesetzt, um das Ersetzen der ursprünglichen Indizes durch die neuen Indizes zu ermöglichen. Diese Sperre wird gehalten, bis die Transaktionen, die während der Reorganisation festgeschrieben werden, in den neuen Indizes reflektiert werden.
- Die IS-Tabellensperre und die NS-Zeilensperre werden für die Systemkatalogtabelle SYSIBM.SYSTABLES aktiviert.
- Bei einer Reorganisation auf Partitionsebene werden die IS-Tabellensperre und die NS-Zeilensperre auch für die Systemkatalogtabelle SYSIBM.SYSDATAPARTITIONS aktiviert.
- Bestimmte interne Sperren können bei einer Onlineindexreorganisationsoperation ebenfalls aktiviert werden.
- Die Onlineindexreorganisation kann Auswirkungen auf den gemeinsamen Zugriff haben, falls die Reorganisationsoperation fehlschlägt. Gründe für ein Fehlschlagen der Reorganisation können fehlender Hauptspeicherplatz, fehlender Plattenspeicherplatz oder eine Überschreitung Zeitlimits für Sperren sein. Die Reorganisationstransaktion führt vor dem Abbrechen bestimmte Aktualisierungen durch. Zur Durchführung von Aktualisierungen muss die Reorganisation warten, bis vorhandene Transaktionen festgeschrieben werden. Hierdurch können andere Transaktion im Prozess blockiert werden. Ab DB2 Version 9.7 Fixpack 1 fordert die Reorganisation eine spezielle DRAIN-Sperre für das Indexobjekt an. Reorganisationsoperationen warten, bis vorhandene Transaktionen abgeschlossen sind; neue Anforderungen für den Zugriff auf das Indexobjekt sind jedoch zulässig.

Ermitteln des Zeitpunkts für die Reorganisation von Tabellen und Indizes

Nach zahlreichen Änderungen an Tabellendaten können sich logisch sequenzielle Daten auf physisch nicht sequenziellen Datenseiten befinden, vor allem dann, wenn durch eine große Anzahl von Aktualisierungsoperationen Überlaufsätze entstanden sind. Wenn die Daten auf diese Weise organisiert sind, muss der Datenbankmanager zusätzliche Leseoperationen durchführen, um auf die erforderlichen Daten zuzugreifen. Darüber hinaus sind zusätzliche Leseoperationen auch erforderlich, wenn viele Zeilen gelöscht wurden.

Informationen zu diesem Vorgang

Durch die Tabellenreorganisation werden die Daten defragmentiert und eine unnötige Belegung von Speicherplatz beseitigt. Außerdem werden die Zeilen neu geordnet, sodass Überlaufsätze eingereiht und somit der Datenzugriff und letztendlich die Abfrageleistung verbessert werden. Sie können angeben, dass die Daten anhand eines bestimmten Index neu geordnet werden sollen, sodass Abfragen bei einer minimalen Anzahl von Leseoperationen auf die Daten zugreifen können.

Eine hohe Anzahl von Änderungen an Tabellendaten kann dazu führen, dass sich die Indexleistung verschlechtert. Auf Indexblattseiten kann es zu Fragmentierung und schlechterem Clustering kommen und der Index könnte mehr Stufen als für eine optimale Leistung erforderlich entwickeln. Alle diese Aspekte verursachen ein höheres E/A-Aufkommen und können sich negativ auf die Leistung auswirken.

Jeder einzelne der nachfolgend aufgeführten Faktoren kann darauf hinweisen, dass eine Tabelle oder ein Index reorganisiert werden sollte:

- Es ist ein hohes Aufkommen an Einfüge-, Aktualisierungs- und Löschaktivitäten für eine Tabelle angefallen, seit die Tabelle zuletzt reorganisiert wurde.
- Es treten wesentliche Änderungen in der Leistung von Abfragen auf, die einen Index mit einem hohen Clusterverhältnis verwenden.
- Die Leistung wird durch die Ausführung des Befehls **RUNSTATS** zur Aktualisierung der Statistikdaten nicht besser.
- Die Ausgabe zum Befehl **REORGCHK** lässt darauf schließen, dass die Leistung durch eine Reorganisation einer Tabelle oder der zugehörigen Indizes verbessert werden kann.

  In manchen Fällen empfiehlt das Dienstprogramm REORGCHK immer eine Tabellenreorganisation, selbst nachdem eine REORG-Operation ausgeführt wurde. Beispiel: Wenn eine Seitengröße von 32 KB mit einer durchschnittlichen Satzlänge von 15 Byte und maximal 253 Sätzen pro Seite verwendet wird, bedeutet dies, dass jede Seite 33.700 - (15 x 253) = 28.905 nicht verwendbare Byte enthält. Etwa 88% der Seite besteht also aus freiem Speicherbereich. Sie sollten die Empfehlungen des Dienstprogramms REORGCHK analysieren und die potenziellen Vorteile gegen den Ausführungsaufwand einer Reorganisation abwägen.

Der Befehl **REORGCHK** gibt Statistikdaten zur Datenorganisation zurück und kann Empfehlungen dazu liefern, ob bestimmte Tabellen oder Indizes reorganisiert werden müssen. Allerdings können Sie durch eine Ausführung bestimmter Abfragen auf die SYSSTAT-Sichten in regelmäßigen Intervallen oder zu bestimmten Zeitpunkten ein Verlaufsprotokoll erstellen, dass Ihnen bei der Erkennung von Trends hilft, die potenziell erhebliche Auswirkungen auf die Leistung haben können.

Zur Ermittlung, ob die Notwendigkeit besteht, Tabellen oder Indizes zu reorganisieren, fragen Sie die SYSSTAT-Sichten ab und überwachen Sie die folgenden Statistiken:

- Überlauf von Zeilen

  Fragen Sie die Spalte OVERFLOW in der Sicht SYSSTAT.TABLES ab, um den OVERFLOW-Wert zu überwachen. Der Wert stellt die Anzahl der Zeilen dar, die nicht auf ihre ursprünglichen Seiten passen. Zeilendaten können überlaufen, wenn Spalten mit variabler Länge dazu führen, dass die Datensatzlänge bis zu einem Punkt zunimmt, an dem die Zeile nicht mehr in ihre zugewiesene Position auf der Datenseite passt. Längenänderungen können auch auftreten, wenn der Tabelle eine Spalte hinzugefügt wird. In diesem Fall wird an der ursprünglichen Position der Zeile ein Verweis hinterlegt und der tatsächliche Wert an einer anderen, durch den Verweis angegebenen Position gespeichert. Dadurch kann die Leistung beeinträchtigt werden, da der Datenbankmanager dem Verweis folgen muss, um den Inhalt der Spalte zu finden. Durch diese beiden Schritte verlängert sich die Verarbeitungszeit und erhöht sich möglicherweise auch die Anzahl der erforderlichen E/A-Operationen. Durch eine Reorganisation der Tabellendaten werden sämtliche Zeilenüberläufe beseitigt.
- Abrufstatistiken (Fetch)

  Fragen Sie die Spalten in der Katalogsicht 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 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 den Wert von 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, die beim Abrufen von Tabellenzeilen über den Index zwischen sequenziellen Seitenzugriffen durch einen wahlfreien Zugriff abgerufen werden. 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 zunehmende Unordnung wird in der Regel durch Einfügungen außerhalb der Reihenfolge, das heißt, entweder am Ende der Tabelle oder auf Überlaufseiten, verursacht. Sie hat Auswirkungen auf die Abfrageleistung, wenn ein Index für den Zugriff auf einen Bereich von Werten verwendet wird.
- In der Spalte AVERAGE_SEQUENCE_FETCH_GAP wird die durchschnittliche Lücke zwischen Tabellenseitensequenzen (d. h. in der Reihenfolge vorliegende Seiten) beim Abrufen von Tabellenzeilen über den Index gespeichert. Erkannt beim Durchsuchen von Indexseiten stellt jede Lücke die durchschnittliche Anzahl von Tabellenseiten dar, die jeweils zwischen Sequenzen von Tabellenseiten wahlfrei abgerufen werden müssen. Dies ist von Bedeutung, wenn auf viele Seiten wahlfrei zugegriffen wird, wodurch die Vorablesefunktionen unterbrochen werden. Ein hoher Wert zeigt an, dass die Tabelle nicht gut organisiert ist oder bezüglich des Index eine niedrige Clusterbildung aufweist.

- Anzahl der Indexblattseiten, die Satz-IDs (RIDs) enthalten, die zwar als gelöscht markiert, jedoch noch nicht entfernt wurden

  Satz-IDs (RIDs, Record Identifiers) werden gewöhnlich nicht physisch gelöscht, wenn sie als gelöscht markiert werden. 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 Sicht SYSSTAT.INDEXES ab. Für Blattseiten, auf 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 einen Aufruf des Befehls **REORG INDEXES** mit der Option CLEANUP ALL freigegeben werden könnte. Um nur den Speicherplatz auf Seiten wieder verfügbar zu machen, auf denen alle Satz-IDs als gelöscht markiert sind, rufen Sie den Befehl **REORG INDEXES** mit der Option CLEANUP ONLY PAGES auf.

- Statistiken zum Clusterverhältnis und zum Clusterfaktor für Indizes

  Im Allgemeinen kann nur einer der Indizes für eine Tabelle einen hohen Grad an Clusterbildung aufweisen. In der Spalte CLUSTERRATIO der Katalogsicht SYSCAT.INDEXES wird ein Statistikwert für das Clusterverhältnis gespeichert. Dieser Wert (zwischen 0 und 100) stellt den Grad der Datenclusterbildung im Index dar. Wenn Sie detaillierte Indexstatistikdaten 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 kann in der Katalogsicht SYSCAT.INDEXES aufgezeichnet werden. Zum Vergleich von CLUSTERFACTOR-Werten mit CLUSTERRATIO-Werten müssen Sie jeweils den CLUSTERFACTOR-Wert mit 100 multiplizieren, um einen Prozentsatzwert zu erhalten.

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 diesen Typ von Suche, da die Wahrscheinlichkeit geringer ist, dass eine Datenseite im Pufferpool verbleibt, bis der nächste Zugriff auf sie erfolgt. Die Leistung für einen Index mit geringerer 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.

- Anzahl der Blattseiten (Leaf pages)

  Fragen Sie die Spalte NLEAF in der Sicht 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 des Index erforderlich sind.

  Ein Index sollte idealerweise möglichst wenig Speicherplatz belegen, um die Anzahl von E/A-Operationen zu reduzieren, die für eine Indexsuche erforderlich sind. Wahlfreie Aktualisierungen können dazu führen, dass Seiten geteilt werden und sich ein Index dadurch vergrößert. Bei einer Tabellenreorganisation kann jeder Index mit minimalem Speicherplatz erneut erstellt werden.

  Bei der Erstellung eines Index bleiben standardmäßig zehn Prozent des Speicherbereichs auf jeder Indexseite frei. Zur Erhöhung des Betrages an freiem Speicherbereich geben Sie die Option PCTFREE bei der Erstellung des Index an. Der angegebene PCTFREE-Wert wird bei jeder Reorganisation des Index verwendet. Ein freier Speicherbereich von mehr als zehn Prozent kann die Häufigkeit von Indexreorganisationen verringern, weil in dem zusätzlichen Speicherbereich weitere Indexeinfügungen untergebracht werden können.

- Anzahl der leeren Datenseiten

  Zur Berechnung der Anzahl leerer Seiten in einer Tabelle fragen Sie die Spalten FPAGES und NPAGES in der Sicht SYSCAT.TABLES ab und subtrahieren den NPAGES-Wert (die Anzahl der Seiten, die Zeilen enthalten) vom FPAGES-Wert (die Gesamtzahl der Seiten, die verwendet werden). 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 leere Seiten 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 Suchleistung verbessert werden.

  Wenn die Gesamtzahl der genutzten Seiten (FPAGES) in einer Tabelle keiner oder gleich (NPARTITIONS * 1 EXTENTSIZE) ist, wird keine Tabellenreorganisation empfohlen. NPARTITIONS stellt die Anzahl der Datenpartitionen dar, wenn es sich um eine partitionierte Tabelle handelt. Ansonsten hat NPARTITIONS den Wert 1. In einer Umgebung mit partitionierten Datenbanken wird eine Tabellenreorganisation nicht empfohlen, wenn FPAGES <= (Anzahl der Datenbankpartitionen in der Datenbankpartitionsgruppe der Tabelle) * (NPARTITIONS * 1 EXTENTSIZE).

Wägen Sie vor einer Reorganisation von Tabellen oder Indizes die Kosten einer zunehmenden Verschlechterung der Abfrageleistung und die Kosten einer Tabellen- oder Indexreorganisation, zu denen die Verarbeitungszeit, der Zeitaufwand und ein verringerter Grad des gemeinsamen Zugriffs gehören, gegeneinander ab.

Reorganisationsaufwand für Tabellen und Indizes

Bei der Ausführung einer Tabellen- oder Indexreorganisation entsteht ein gewisser Systemaufwand, der bei der Entscheidung, ob ein Objekt reorganisiert werden soll, zu berücksichtigen ist.

Im Hinblick auf den Aufwand, der bei der Reorganisation von Tabellen und Indizes entsteht, sind folgende Aspekte zu beachten:

- Verarbeitungszeit des ausführenden Dienstprogramms
- Eingeschränkter gemeinsamer Zugriff (aufgrund von Sperren) während der Ausführung des Dienstprogramms REORG
- Zusätzlicher Speicherbedarf
  - Die Offlinetabellenreorganisation erfordert zusätzlichen Speicherplatz zur Aufnahme einer Spiegelkopie der Tabelle
  - Die Online- oder Inplace-Tabellenreorganisation erfordert mehr Protokollspeicherbereich
  - Die Offline-Indexreorganisation benötigt weniger Protokollspeicherbereich und arbeitet ohne Spiegelkopie
  - Die Onlineindexreorganisation erfordert mehr Protokollspeicherbereich und zusätzlichen Speicherplatz zur Aufnahme einer Spiegelkopie der Tabelle

In einigen Fällen kann eine reorganisierte Tabelle größer als die ursprüngliche Tabelle sein. Eine Tabelle kann in folgenden Situationen nach der Reorganisation möglicherweise größer sein:

- Bei einer REORG-Operation für eine Tabelle mit Clustering, bei der ein Index zur Festlegung der Reihenfolge der Zeilen verwendet wird, kann mehr Speicherplatz erforderlich sein, wenn die Tabellendatensätze von variabler Länge sind, da einige Seiten in der reorganisierten Tabelle vielleicht weniger Zeilen als in der ursprünglichen Tabelle enthalten.
- Der freie Speicherbereich, der auf jeder Seite behalten wird (festgelegt durch den Wert von PCTFREE), könnte seit der letzten Reorganisation größer geworden sein.

Speicherplatzbedarf für eine Offlinetabellenreorganisation

Da bei der Offlinereorganisation ein Spiegelkopieverfahren verwendet wird, ist ausreichend zusätzlicher Speicherplatz erforderlich, um eine weitere Kopie der Tabelle aufnehmen zu können. Die Spiegelkopie wird entweder in dem Tabellenbereich, in dem sich die Originaltabelle befindet, oder in einem vom Benutzer angegebenen Tabellenbereich für temporäre Tabellen erstellt.

Zusätzlicher Speicherplatz in einem Tabellenbereich für temporäre Tabellen kann für die Verarbeitung von Sortierungen erforderlich werden, wenn eine Tabellensuchsortierung ausgeführt wird. Der erforderliche zusätzliche Speicherplatz kann bis zur Größe der Tabelle betragen, die reorganisiert wird. Wenn der Clusterindex ein SMS-Typ (SMS - vom System verwalteter Speicherbereich) oder ein eindeutiger DMS-Typ (DMS - von der Datenbank verwalteter Speicherbereich) ist, erfordert die erneute Erstellung dieses Index keine Sortierung. Stattdessen wird dieser Index durch ein Durchsuchen der neu reorganisierten Daten erstellt. Alle anderen Indizes, die neu erstellt werden, erfordern eine Sortierung, bei der potenziell ein Speicherplatzbedarf im Tabellenbereich für temporäre Tabellen anfällt, der bis zur Größe der Tabelle, die reorganisiert wird, betragen kann.

Offline ausgeführte REORG-Operationen für Tabellen generieren relativ wenige Steuerprotokolleinträge und belegen daher relativ wenig Protokollspeicherplatz. Wenn das Dienstprogramm REORG keinen Index verwendet, werden nur Protokollsätze für Tabellendaten erstellt. Wenn ein Index angegeben wird oder wenn ein Clusterindex für die Tabelle vorhanden ist, werden Satz-IDs (RIDs) in der Reihenfolge protokolliert, in der sie in die neue Version der Tabelle eingefügt werden. Jeder Satz-ID-Protokollsatz enthält maximal 8000 Satz-IDs, wobei jede Satz-ID 4 Byte belegt. Dies kann zu Protokollspeicherproblemen bei einer offline ausgeführten REORG-Operation für eine Tabelle beitragen. Beachten Sie, dass Satz-IDs nur protokolliert werden, wenn die Datenbank wiederherstellbar ist.

Protokollspeicherbedarf für eine Onlinetabellenreorganisation

Der für eine online ausgeführte REORG-Operation für eine Tabelle erforderliche Protokollspeicherbedarf ist in der Regel höher als der für eine offline ausgeführte REORG-Operation. Die Größe des erforderlichen Speicherplatzes wird durch die Anzahl der Zeilen, die reorganisiert werden, die Anzahl von Indizes, die Größe der Indexschlüssel sowie durch den anfangs bestehenden Grad an Fragmentierung der Tabelle bestimmt. Es empfiehlt sich daher, einen typischen Vergleichspunkt (Benchmark) für die Protokollspeicherbelegung zu ermitteln, die mit Ihren Tabellen verbunden ist.

Jede Zeile in einer Tabelle wird mit hoher Wahrscheinlichkeit zweimal während einer online ausgeführten REORG-Operation für die Tabelle versetzt. In jedem Index muss der Indexschlüssel für jede Tabellenzeile aktualisiert werden, um auf die neue Position der Zeile zu verweisen. Nachdem alle Zugriffe auf die alte Position abgeschlossen wurden, wird der Indexschlüssel erneut aktualisiert, um den Verweis auf die alte Satz-ID zu entfernen. Wenn die Zeile zurückversetzt wird, werden erneut Aktualisierungen am Indexschlüssel ausgeführt. Alle diese Aktivitäten werden protokolliert, um eine Onlinetabellenreorganisation vollständig wiederherstellbar zu machen. Dies erfordert mindestens zwei Datenprotokollsätze (jeweils mit den Zeilendaten) und vier Indexprotokollsätze (jeweils mit den Schlüsseldaten) für jede Zeile (bei einem Index). Clusterindizes tendieren besonders dazu, die Indexseiten zu füllen, sodass es zu Indexteilungen und -zusammenführungen kommt, die ebenfalls protokolliert werden müssen.

Da das Dienstprogramm REORG bei einer Onlinereorganisation häufig interne COMMIT-Anweisungen absetzt, erfordert es in der Regel keine große Anzahl von aktiven Protokollen. Eine Ausnahme kann während der Abschneidephase auftreten, wenn das Dienstprogramm eine S-Tabellensperre anfordert. Wenn das Dienstprogramm die Sperre nicht aktivieren kann, wartet es ab, wobei andere Transaktionen die Protokolle in der Zwischenzeit rasch füllen könnten.

Senken des Bedarfs an Tabellen- und Indexreorganisationen

Verschiedene Strategien können zur Verringerung des Bedarfs an Tabellen- und Indexreorganisationen (und des damit verbundenen Aufwands) verwendet werden.

Senken des Bedarfs an Tabellenreorganisationen

Gehen Sie wie folgt vor, um den Bedarf an Tabellenreorganisationen zu senken:

- Verwenden Sie Mehrpartitionstabellen.
- Erstellen Sie Tabellen mit mehrdimensionalem Clustering (MDC-Tabellen). Bei MDC-Tabellen bleibt die Clusterbildung für die Spalten erhalten, die Sie in der Klausel ORGANIZE BY DIMENSIONS der Anweisung CREATE TABLE angeben. Allerdings kann das Dienstprogramm REORGCHK trotzdem eine Reorgani-

sation einer MDC-Tabelle empfehlen, wenn es feststellt, dass zu viele ungenutzte Blöcke vorhanden sind oder dass Blöcke zusammengelegt werden sollten.

- Aktivieren Sie den Anfügemodus (APPEND) für Ihre Tabellen. Wenn die Indexschlüsselwerte für neue Zeilen zum Beispiel stets neue HIGHKEY-Werte (d. h. neue Höchstwerte) sind, versucht das Clustering-Attribut der Tabelle, diese Zeilen an das Ende der Tabelle zu setzen. In diesem Fall kann das Aktivieren des Anfügemodus eine bessere Wahl als die Verwendung eines Clusterindex sein.

Führen Sie nach dem Erstellen einer Tabelle die folgenden Maßnahmen durch, um den Bedarf an Tabellenreorganisationen weiter zu verringern:

- Ändern Sie die Tabelle, um den Prozentsatz an freiem Speicherbereich anzugeben, der bei einer Lade- oder Tabellenreorganisationsoperation auf jeder Seite beizubehalten ist (PCTFREE).
- Erstellen Sie einen Clusterindex, indem Sie die Option PCTFREE angeben.
- Sortieren Sie die Daten, bevor Sie sie in die Tabelle laden.

Wenn Sie diese Maßnahmen durchgeführt haben, helfen der Clusterindex und die PCTFREE-Einstellung für die Tabelle, die ursprüngliche sortierte Reihenfolge beizubehalten. Wenn genügend Speicherplatz auf den Tabellenseiten freigehalten wird, können neue Daten auf den richtigen Seiten eingefügt werden, um das Clustering in Bezug auf den Index beizubehalten. Wenn allerdings mehr Daten eingefügt werden und sich die Tabellenseiten vollständig füllen, werden Datensätze an das Ende der Tabelle angehängt, sodass das Clustering der Tabelle mit der Zeit abnimmt.

Wenn Sie eine REORG-Operation für eine Tabelle ausführen oder eine Sortier- und Ladeoperation nach der Erstellung eines Clusterindex durchführen, versucht der Index die Reihenfolge der Daten beizubehalten, wodurch sich die Werte für die Statistiken CLUSTERRATIO bzw. CLUSTERFACTOR verbessern, die durch das Dienstprogramm RUNSTATS erfasst werden.

Senken des Bedarfs an Indexreorganisationen

Gehen Sie wie folgt vor, um den Bedarf an Indexreorganisationen zu senken:

- Erstellen Sie Clusterindizes, indem Sie die Option PCTFREE oder LEVEL2 PCTFREE angeben.
- Erstellen Sie Indizes mit der Option MINPCTUSED. Ziehen Sie alternativ in Betracht, die Option CLEANUP ONLY ALL des Befehls **REORG INDEXES** zu verwenden, um Blattseiten (engl. leaf pages) zusammenzufügen.

Automatische Reorganisation

Nach zahlreichen Änderungen an Tabellendaten können die Tabelle und die zugehörigen Indizes fragmentiert werden. Logisch sequenzielle Daten können sich auf nicht sequenziellen Seiten befinden, sodass der Datenbankmanager für den Datenzugriff zusätzliche Leseoperationen ausführen muss.

Die statistischen Informationen, die vom Dienstprogramm RUNSTATS erfasst werden, beschreiben die Verteilung von Daten innerhalb einer Tabelle. Durch eine Analyse dieser Statistiken kann ermittelt werden, wann eine Reorganisation und welche Art von Reorganisation erforderlich ist.

Der Prozess der automatischen Reorganisation ermittelt die Notwendigkeit einer Tabellen- oder Indexreorganisation mithilfe von Formeln, die zum Dienstprogramm REORGCHK gehören. Er bewertet in regelmäßigen Abständen Tabellen und Indi-

zes, deren Statistiken aktualisiert wurden, um zu prüfen, ob eine Reorganisation erforderlich ist, und terminiert solche Operationen, wenn sie erforderlich sind.

Die Funktion zur automatischen Reorganisation kann durch die Datenbankkonfigurationsparameter **auto_reorg**, **auto_tbl_maint** und **auto_maint** aktiviert bzw. inaktiviert werden.

In einer Umgebung mit partitionierten Datenbanken erfolgt die Initiierung einer automatischen Reorganisation in der Katalogdatenbankpartition. Diese Konfigurationsparameter müssen daher nur in dieser Partition aktiviert werden. Die REORG-Operation wird hingegen in allen Datenbankpartitionen ausgeführt, in denen sich die Zieltabellen befinden.

Wenn Sie sich nicht sicher sind, wann und wie Ihre Tabellen und Indizes zu reorganisieren sind, können Sie die automatische Reorganisation in Ihren Gesamtplan zur Datenbankverwaltung aufnehmen.

Sie können auch MDC-Tabellen reorganisieren, um Speicherplatz freizugeben. Die Freigabe von Speicherbereichen über eine MDC-Tabelle wird nur für MDC-Tabellen in DMS-Tabellenbereichen unterstützt. Die Freigabe von Speicherbereichen über MDC-Tabellen kann Bestandteil der automatischen Verwaltungsaktivitäten für Ihre Datenbank sein.

Automatische Reorganisation datenpartitionierter Tabellen

Bei DB2 Version 9.7 Fixpack 1 und früheren Releases unterstützt die automatische Reorganisation die Reorganisation einer datenpartitionierten Tabelle für die gesamte Tabelle. Bei DB2 Version 9.7 Fixpack 1 und späteren Releases unterstützt die automatische Reorganisation die Reorganisation einer partitionierten Tabelle und die Reorganisation der partitionierten Indizes für eine Datenpartition einer partitionierten Tabelle.

Um zu vermeiden, dass eine gesamte datenpartitionierte Tabelle in den Modus `ALLOW NO ACCESS` versetzt wird, führt die automatische Reorganisation Operationen **REORG INDEXES ALL** auf der Datenpartitionsebene für partitionierte Indizes aus, die reorganisiert werden müssen. Die automatische Reorganisation führt Operationen **REORG INDEX** für jeden nicht partitionierten Index aus, der reorganisiert werden muss.

Die automatische Reorganisation führt die folgenden Operationen **REORG TABLE** für datenpartitionierte Tabellen aus:

- Wenn nicht partitionierte Indizes (mit Ausnahme von systemgenerierten XML-Pfadindizes) für die Tabelle definiert sind und nur eine Partition reorganisiert werden muss, führt die automatische Reorganisation eine Operation **REORG TABLE** mit der Klausel ON DATA PARTITION zur Angabe der Partition aus, die reorganisiert werden muss. Ansonsten führt die automatische Reorganisation eine Operation **REORG TABLE** für die gesamte Tabelle ohne die Klausel ON DATA PARTITION aus.
- Wenn keine nicht partitionierten Indizes (außer systemgenerierten XML-Pfadindizes) für die Tabelle definiert sind, führt die automatische Reorganisation eine Operation **REORG TABLE** mit der Klausel ON DATA PARTITION für jede Partition aus, die reorganisiert werden muss.

Automatische Reorganisation flüchtiger Tabellen

Seit Version 9.7 Fixpack 4 ist es möglich, die automatische Indexreorganisation in flüchtigen Tabellen zu aktivieren. Der Prozess für die automatische Reorganisation stellt fest, ob die Indexreorganisation in flüchtigen Tabellen erforderlich ist und plant die erforderlichen Operationen. Die Indexreorganisation wird für flüchtige Tabellen in regelmäßigen Zeitabständen durchgeführt und gibt den Speicherbereich frei, der von den Indizes, die für diese Tabellen definiert wurden, wiederverwendet werden kann.

Statistikdaten können in flüchtigen Tabellen nicht erfasst werden, da sie sehr häufig aktualisiert werden. Um festzustellen, welche Indizes reorganisiert werden müssen, verwendet die automatische Reorganisation anstelle der Statistiken das Attribut numInxPseudoEmptyPagesForVolatile. Dieses Attribut in der Richtlinie AUTO_REORG wurde in Version 9.7 Fixpack 4 eingeführt und gibt an, wie viele leere Indexseiten mit pseudogelöschten Schlüsseln ein Index aufweisen muss, damit eine Indexreorganisation ausgelöst wird.

Um die automatische Indexreorganisation in flüchtigen Tabellen zu aktivieren, muss die Registrierdatenbankvariable **DB2_WORKLOAD** auf die Einstellung SAP gesetzt werden. Außerdem muss die automatische Reorganisation aktiviert und das Attribut numInxPseudoEmptyPagesForVolatile muss gesetzt werden.

Aktivieren der automatischen Reorganisation von Tabellen und Indizes

Verwenden Sie die automatische Reorganisation von Tabellen und Indizes, sodass Sie sich nicht darum zu kümmern brauchen, wann und wie die Daten reorganisiert werden.

Informationen zu diesem Vorgang

Über gut organisierte Tabellen- und Indexdaten zu verfügen, ist von kritischer Bedeutung für einen effizienten Datenzugriff und eine optimale Auslastungsleistung. Nach zahlreichen Einfüge-, Aktualisierungs- und Löschoperationen können sich logisch sequenzielle Daten auf nicht sequenziellen Datenseiten befinden, sodass der Datenbankmanager zusätzliche Leseoperationen ausführen muss, um auf die Daten zuzugreifen. Darüber hinaus sind zusätzliche Leseoperationen auch erforderlich, wenn auf Daten in einer Tabelle zugegriffen wird, aus der eine beträchtliche Anzahl von Zeilen gelöscht wurde. Sie können den DB2-Server veranlassen, die Systemkatalogtabellen ebenso wie Benutzertabellen zu reorganisieren.

Vorgehensweise

Zur Aktivierung der automatischen Reorganisation in Ihrer Datenbank setzen Sie die folgenden Konfigurationsparameter auf den Wert ON:

- **auto_maint**
- **auto_tbl_maint**
- **auto_reorg**

Aktivieren der automatischen Indexreorganisation in flüchtigen Tabellen

Sie können die automatische Reorganisation aktivieren, um die Indexreorganisation in flüchtigen Tabellen auszuführen.

Informationen zu diesem Vorgang

Wenn Sie die automatische Indexreorganisation in flüchtigen Tabellen aktivieren, dann überprüft die automatische Reorganisation in jeden Aktualisierungsintervall, ob die Indizes in flüchtigen Tabellen eine Reorganisation benötigen. Die erforderliche Operation wird mithilfe des Befehls **REORG** geplant.

Vorgehensweise

Führen Sie die folgenden Schritte aus, um die automatische Indexreorganisation in flüchtigen Tabellen zu aktivieren:

1. Setzen Sie die Registrierdatenbankvariable **DB2_WORKLOAD** auf SAP. Das folgende Beispiel zeigt, wie diese Variable mit dem Befehl **db2set** gesetzt werden kann.
   ```
   db2set DB2_WORKLOAD=SAP
   ```
   Führen Sie für die Datenbank einen Neustart durch, damit diese Einstellung wirksam wird.
2. Setzen Sie den Datenbankkonfigurationsparameter **auto_reorg** auf ON. Das folgende Beispiel zeigt, wie dieser Datenbankkonfigurationsparameter über die Befehlszeilenschnittstelle des DB2-Befehlszeilenprozessors gesetzt werden kann.
   ```
   UPDATE DB CFG FOR SAMPLE USING auto_reorg ON
   ```
   Vergewissern Sie sich, dass die Datenbankkonfigurationsparameter **auto_maint** und **auto_tbl_maint** ebenfalls auf ON gesetzt sind. Die Parameter **auto_maint** und **auto_tbl_maint** sind standardmäßig auf ON gesetzt.
3. Setzen Sie das Attribut numInxPseudoEmptyPagesForVolatileTables in der Richtlinie AUTO_REORG, indem Sie die Prozedur AUTOMAINT_SET_POLICY oder AUTOMAINT_SET_POLICYFILE aufrufen. Dieses Attribut gibt die minimale Anzahl der leeren Indexseiten mit pseudogelöschten Schlüsseln an, die zur Ausführung einer Indexreorganisation erforderlich sind. Das folgende Beispiel zeigt, wie dieses Attribut gesetzt werden kann:
   ```
   CALL SYSPROC.AUTOMAINT_SET_POLICY
     ('AUTO_REORG',
     BLOB(' <?xml version="1.0" encoding="UTF-8"?>
   <DB2AutoReorgPolicy
   xmlns="http://www.ibm.com/xmlns/prod/db2/autonomic/config" >

    <ReorgOptions  dictionaryOption="Keep" indexReorgMode="Online"
       useSystemTempTableSpace="false" numInxPseudoEmptyPagesForVolatileTables="20" />

    <ReorgTableScope  maxOfflineReorgTableSize="0">
     <FilterClause>TABSCHEMA NOT LIKE 'SYS%'</FilterClause>
    </ReorgTableScope>
   </DB2AutoReorgPolicy>')
     )
   ```
   Sie können die Werte für die Spalten PSEUDO_EMPTY_PAGES, EMPTY_PAGES_DELETED und EMPTY_PAGES_REUSED überwachen, indem Sie die Tabellenfunktion MON_GET_INDEX abfragen. Auf diese Weise können Sie einen geeigneten Wert für das Attribut numInxPseudoEmptyPagesForVolatileTables einfacher ermitteln.

Anwendungsentwurf

Der Entwurf einer Datenbankanwendung ist einer der Faktoren, die sich auf die Anwendungsleistung auswirken. Lesen Sie die Informationen zu den verschiedenen Aspekten des Anwendungsentwurfs in diesem Abschnitt, die Ihnen helfen können, die Leistung von Datenbankanwendungen zu maximieren.

Anwendungsprozesse, gemeinsamer Zugriff und Recovery

Alle SQL-Programme werden als Teil eines *Anwendungsprozesses* oder eines Agenten ausgeführt. Ein Anwendungsprozess bezieht die Ausführung eines oder mehrerer Programme mit ein und stellt die Einheit dar, der der Datenbankmanager Ressourcen und Sperren zuordnet. Verschiedene Anwendungsprozesse können die Ausführung verschiedener Programme oder auch verschiedene Ausführungen desselben Programms beinhalten.

Es können mehrere Anwendungsprozesse zur gleichen Zeit einen Zugriff auf dieselben Daten anfordern. Als *Sperren* wird der Mechanismus bezeichnet, mit dessen Hilfe die Datenintegrität unter solchen Bedingungen aufrechterhalten wird. Das Sperren verhindert zum Beispiel, dass zwei Anwendungsprozesse dieselbe Zeile von Daten gleichzeitig aktualisieren.

Der Datenbankmanager fordert Sperren an, um zu verhindern, dass nicht festgeschriebene Änderungen, die von einem Anwendungsprozess ausgeführt wurden, unbeabsichtigt von einem anderen Prozess gelesen werden. Der Datenbankmanager gibt alle Sperren, die er für einen Anwendungsprozess angefordert und aktiviert hat, frei, wenn dieser Prozess endet. Ein Anwendungsprozess kann jedoch explizit anfordern, dass Sperren früher freigegeben werden sollen. Dies geschieht durch eine Commitoperation (*commit*). Diese Operation gibt Sperren frei, die während einer UOW (Unit of Work) aktiviert wurden und schreibt darüber hinaus Datenbankänderungen fest, die während der UOW ausgeführt wurden.

Eine *Unit of Work* (UOW, Arbeitseinheit) ist eine wiederherstellbare Folge von Operationen innerhalb eines Anwendungsprozesses. Eine UOW wird eingeleitet, wenn ein Anwendungsprozess gestartet wird oder wenn die vorherige UOW durch ein anderes Ereignis als die Beendigung des Anwendungsprozesses beendet wird. Eine UOW endet mit einer Commitoperation, einer Rollback-Operation oder mit dem Ende eines Anwendungsprozesses. Eine Commit- oder Rollback-Operation betrifft nur die Datenbankänderungen, die innerhalb der UOW, die beendet wird, ausgeführt wurden.

Der Datenbankmanager stellt eine Funktionalität zum Rückgängigmachen von nicht festgeschriebenen Änderungen bereit, die durch einen Anwendungsprozess ausgeführt wurden. Dies kann erforderlich werden, wenn aufseiten eines Anwendungsprozesses ein Fehler auftritt oder falls es zu einem Deadlock oder einer Überschreitung des Sperrzeitlimits kommt. Ein Anwendungsprozess kann explizit anfordern, dass seine eigenen Datenbankänderungen zurückgenommen werden. Dies geschieht mithilfe einer *Rollback-Operation*.

Solange diese Änderungen nicht festgeschrieben werden, sind sie für andere Anwendungsprozesse nicht sichtbar und können durch ein Rollback rückgängig gemacht werden. Dies gilt jedoch nicht, wenn die geltende Isolationsstufe UR (nicht festgeschriebener Lesevorgang) ist. Wenn Datenbankänderungen durch eine Commitoperation festgeschrieben wurden, sind sie für andere Anwendungsprozesse zugänglich und können nicht mehr durch eine Rollback-Operation rückgängig gemacht werden.

Sowohl DB2 Call Level Interface (CLI) als auch eingebettetes SQL lassen einen Verbindungsmodus zu, der als Modus für *gleichzeitig ablaufende Transaktionen* bezeichnet wird und der mehrere Verbindungen unterstützt, die jeweils eine unabhängige Transaktion darstellen. Eine Anwendung kann über mehrere gleichzeitig bestehende Verbindungen zur selben Datenbank verfügen.

Sperren, die vom Datenbankmanager für einen Anwendungsprozess aktiviert werden, werden bis zum Ende einer UOW beibehalten, sofern nicht die Isolationsstufe CS (Cursorstabilität, bei der die Sperre mit dem Versetzen des Cursors von Zeile zu Zeile freigegeben wird) oder UR (nicht festgeschriebener Lesevorgang) verwendet wird.

Ein Anwendungsprozess wird durch die eigenen Sperren nie an der Ausführung von Operationen gehindert. Wenn eine Anwendung jedoch gleichzeitig ablaufende Transaktionen verwendet, können sich die Sperren einer Transaktion auf die Operation einer gleichzeitig ablaufenden Transaktion auswirken.

Der Start und das Ende einer UOW definieren die *Konsistenzzustände* innerhalb eines Anwendungsprozesses. Eine Banktransaktion könnte beispielsweise die Überweisung einer Geldsumme von einem Konto auf ein anderes Konto beinhalten. Eine solche Transaktion sieht vor, dass der betreffende Betrag vom ersten Konto subtrahiert und auf dem zweiten Konto hinzuaddiert wird. Nach dem Subtraktionsschritt sind die Daten inkonsistent. Erst wenn der Betrag auf dem zweiten Konto addiert wurde, ist die Konsistenz wiederhergestellt. Wenn beide Schritte abgeschlossen sind, kann die UOW durch eine Commitoperation beendet werden, sodass die Änderungen für andere Anwendungsprozesse verfügbar werden. Wenn vor dem Ende der UOW ein Fehler auftritt, macht der Datenbankmanager alle nicht festgeschriebenen Änderungen durch eine Rollback-Operation rückgängig, um die Datenkonsistenz wiederherzustellen.

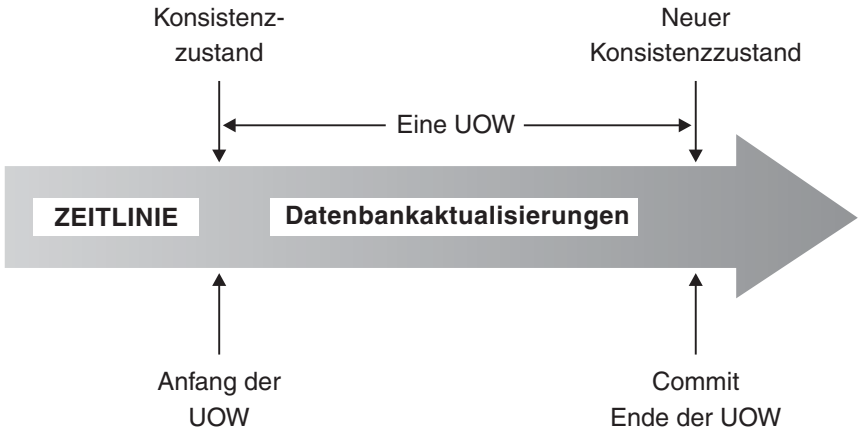


Abbildung 21. UOW mit einer COMMIT-Anweisung

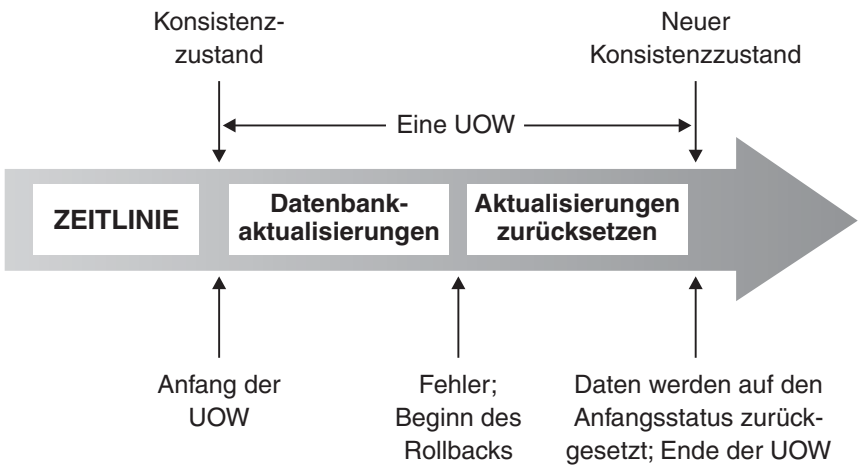


Abbildung 22. UOW mit einer ROLLBACK-Anweisung

Aspekte des gemeinsamen Zugriffs

Da viele Benutzer auf Daten in einer relationalen Datenbank zugreifen und sie ändern, muss der Datenbankmanager einerseits Benutzern diese Änderungen ermöglichen und andererseits sicherstellen, dass die Datenintegrität erhalten 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 könnten zum Beispiel sein:

- **Verlorene Aktualisierungen**. Zwei Anwendungen A und B lesen zum Beispiel dieselbe Zeile und berechnen aufgrund der Daten, die von den Anwendungen gelesen werden, neue Werte für eine der Spalten. Wenn die Anwendung A die Zeile aktualisiert und anschließend B die Zeile ebenfalls aktualisiert, geht die von A durchgeführte Aktualisierung verloren.
- **Zugriff auf nicht festgeschriebene Daten**. Anwendung A könnte einen Wert in der Datenbank aktualisieren, den Anwendung B liest, bevor er festgeschrieben wurde. Wenn dann A diese Aktualisierung rückgängig macht, basieren die von B ausgeführten Berechnungen vielleicht auf ungültigen Daten.
- **Nichtwiederholbare Lesevorgänge**. Anwendung A liest zum Beispiel eine Zeile, bevor sie andere Anforderungen verarbeitet. In der Zwischenzeit ändert oder löscht Anwendung B die Zeile und schreibt diese Änderung fest. Später, wenn Anwendung A versucht, die ursprüngliche Zeile erneut zu lesen, erhält sie die geänderte Zeile oder stellt fest, dass die ursprüngliche Zeile gelöscht wurde.
- **Lesevorgänge mit Phantomzeilen**. Anwendung A führt möglicherweise eine Abfrage aus, die eine Menge von Zeilen entsprechend einer Suchbedingung liest. Anwendung B fügt neue Daten ein oder aktualisiert vorhandene Daten, die die Abfrage von Anwendung A erfüllen würden. Anwendung A führt die Abfrage innerhalb derselben UOW erneut aus und es werden zusätzliche Werte („Phantomzeilen") zurückgegeben.

Der gemeinsame Zugriff ist für globale temporäre Tabellen kein Problem, da diese nur für die Anwendung verfügbar sind, die sie deklariert oder erstellt.

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) innerhalb einer Anweisung verweisen. Zum Verweisen auf solche Datenquellen (die jeweils aus einem DBMS und Daten bestehen) verwendet der DB2-Server Kurznamen. *Kurznamen* sind Aliasnamen für Objekte in anderen Datenbankverwaltungssystemen. In einem föderierten System verlässt sich der DB2-Server auf die Protokolle zur Steuerung des gemeinsamen Zugriffs des Datenbankmanagers, der die angeforderten Daten bereitstellt.

In einem föderierten DB2-System besteht für Datenbankobjekte *Positionstransparenz*. Wenn zum Beispiel Informationen von Tabellen und Sichten versetzt werden, können Verweise auf diese Informationen (durch Kurznamen) aktualisiert werden, ohne die Anwendungen ändern zu müssen, die diese Informationen anfordern. Wenn eine Anwendung auf Daten über Kurznamen zugreift, verlässt sich der DB2-Server darauf, dass die Protokolle zur Steuerung des gemeinsamen Zugriffs an den Datenquellen die Umsetzung der Isolationsstufen sicherstellen. Obwohl der DB2-Server versucht, die angeforderte Isolationsstufe an der Datenquelle mit einem logischen Äquivalent abzugleichen, können die Ergebnisse je nach Funktionsspektrum der Datenquelle unterschiedlich sein.

Isolationsstufen

Die *Isolationsstufe* (engl. isolation level), die einem Anwendungsprozess zugeordnet wird, bestimmt den Grad, bis zu dem die Daten, auf die von diesem Prozess zugegriffen wird, gesperrt bzw. von anderen gleichzeitig aktiven Prozessen isoliert werden. Die Isolationsstufe ist für die Dauer einer UOW (Unit of Work) wirksam.

Die Isolationsstufe eines Anwendungsprozesses gibt daher folgende Größen an:

- Den Grad, bis zu dem Zeilen, die von der Anwendung gelesen oder aktualisiert werden, für andere gleichzeitig ausgeführte Anwendungsprozesse verfügbar sind
- Den Grad, bis zu dem die Aktualisierungsaktivitäten anderer gleichzeitig ausgeführter Anwendungsprozesse Auswirkungen auf die Anwendung haben können

Die Isolationsstufe für statische SQL-Anweisungen wird als Attribut eines Pakets angegeben und gilt für die Anwendungsprozesse, die das betreffende Paket verwenden. Die Isolationsstufe wird während des Prozesses der Programmvorbereitung mithilfe der Binde- oder Vorkompilierungsoption ISOLATION angegeben. Für dynamische SQL-Anweisungen ist die Standardisolationsstufe die Isolationsstufe, die für das Paket, das die Anweisung vorbereit, angegeben wurde. Mithilfe der Anweisung SET CURRENT ISOLATION können Sie eine andere Isolationsstufe für dynamische SQL-Anweisungen angeben, die innerhalb derselben Sitzung abgesetzt werden. Weitere Informationen finden Sie in der Beschreibung zum „Sonderregister CURRENT ISOLATION". Sowohl bei statischen als auch bei dynamischen SQL-Anweisungen überschreibt die *ISOLATION-Klausel* in einer *SELECT-Anweisung* den Wert des Sonderregisters (falls definiert) und den Wert der Bindeoption. Weitere Informationen finden Sie in der Beschreibung zur „Anweisung SELECT".

Isolationsstufen werden durch Sperren umgesetzt. Der Typ der verwendeten Sperre schränkt den Zugriff auf die Daten durch gleichzeitig aktive Anwendungsprozesse ein oder unterbindet ihn völlig. Deklarierte temporäre Tabellen und ihre Zeilen können nicht gesperrt werden, weil sie nur für die Anwendung zugänglich sind, von der sie deklariert wurden.

Der Datenbankmanager unterstützt drei allgemeine Kategorien von Sperren:

Share (S)
Unter einer S-Sperre werden gleichzeitig aktive Anwendungsprozesse auf Operationen mit schreibgeschütztem Zugriff auf die Daten begrenzt.

Update (U)
Unter einer U-Sperre werden gleichzeitig aktive Anwendungsprozesse auf schreibgeschützte Operationen an den Daten begrenzt, sofern diese Prozesse nicht deklariert haben, dass sie möglicherweise eine Zeile aktualisieren. Der Datenbankmanager nimmt an, dass der Prozess, der momentan eine Zeile betrachtet, diese Zeile möglicherweise aktualisiert.

Exclusive (X)
Unter einer X-Sperre werden gleichzeitig aktive Anwendungsprozesse daran gehindert, irgendeinen Zugriff auf die Daten auszuführen. Dies gilt nicht für Anwendungsprozesse mit der Isolationsstufe UR (Uncommitted Read, nicht festgeschriebener Lesevorgang), die die Daten lesen, jedoch nicht ändern können.

Unabhängig von der Isolationsstufe belegt der Datenbankmanager jede Zeile, die eingefügt, aktualisiert oder gelöscht wird, mit einer exklusiven Sperre. Das heißt, alle Isolationsstufen stellen sicher, dass jede Zeile, die durch einen Anwendungsprozess während einer UOW geändert wird, nicht durch irgendeinen anderen Anwendungsprozess geändert wird, bis die UOW abgeschlossen ist.

Der Datenbankmanager unterstützt vier Isolationsstufen:

- „Wiederholbares Lesen (RR, Repeatable Read)"
- „Lesestabilität (RS, Read Stability)" auf Seite 159
- „Cursorstabilität (CS, Cursor Stability)" auf Seite 160
- „Nicht festgeschriebener Lesevorgang (UR, Uncommitted Read)" auf Seite 160

Anmerkung: Einige Hostdatenbankserver unterstützen die Isolationsstufe *No Commit (NC)* ('Kein Commit'). Auf anderen Datenbankservern verhält sich diese Isolationsstufe wie die Isolationsstufe UR (Uncommitted Read, nicht festgeschriebener Lesevorgang).

Im Folgenden werden die einzelnen Isolationsstufen detailliert beschrieben. Die Isolationsstufen sind in absteigender Reihenfolge bezüglich der Auswirkungen auf die Leistung, jedoch in aufsteigender Reihenfolge bezüglich der Vorsicht aufgeführt, die beim Zugriff auf Daten und beim Aktualisieren von Daten erforderlich ist.

Wiederholbares Lesen (RR, Repeatable Read)

Die Isolationsstufe *Wiederholbares Lesen* (RR) sperrt alle Zeilen, auf die eine Anwendung während einer UOW verweist. Wenn eine Anwendung eine Anweisung SELECT zweimal innerhalb derselben UOW absetzt, wird jedes Mal dasselbe Ergebnis zurückgegeben. Unter der Isolationsstufe RR sind Aktualisierungen, die verloren gehen, Zugriffe auf nicht festgeschriebene Daten, nicht wiederholbare Lesevorgänge und Phantomlesevorgänge nicht möglich.

Unter der Isolationsstufe RR kann eine Anwendung Zeilen so oft wie erforderlich abrufen und an diesen Zeilen arbeiten, bis die UOW abgeschlossen ist. Jedoch kann keine andere Anwendung eine Zeile, die die Ergebnismenge beeinflusst, aktualisieren, löschen oder einfügen, bis die UOW abgeschlossen ist. Für Anwendun-

gen, die unter der Isolationsstufe RR ausgeführt werden, sind nicht festgeschriebene Änderungen anderer Anwendungen nicht sichtbar. Diese Isolationsstufe stellt sicher, dass alle zurückgegebenen Daten bis zu dem Zeitpunkt, zu dem die Daten für die Anwendung sichtbar werden, unverändert bleiben, auch wenn temporäre Tabellen oder Zeilenblockung verwendet werden.

Alle Zeilen, auf die verwiesen wird, werden gesperrt, und nicht nur die Zeilen, die abgerufen werden. 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. Eine andere Anwendung kann keine Zeile einfügen oder aktualisieren, die der Liste von Zeilen, auf die in einer Abfrage verwiesen wird, hinzugefügt würde, wenn diese Abfrage erneut ausgeführt würde. Dadurch wird das Lesen von Phantomzeilen vermieden.

Da unter der Isolationsstufe RR eine beträchtliche Anzahl Sperren aktiviert werden kann, kann diese Anzahl die Grenzen überschreiten, die durch die Datenbankkonfigurationsparameter **locklist** und **maxlocks** angegeben werden. Um eine Sperreneskalation zu vermeiden, kann das Optimierungsprogramm eventuell eine einzige Sperre auf Tabellenebene für eine Indexsuche anfordern, wenn das Auftreten einer Sperreneskalation wahrscheinlich ist. Wenn Sie keine Sperren auf Tabellenebene wünschen, verwenden Sie die Isolationsstufe der Lesestabilität (RS).

Bei der Auswertung referenzieller Integritätsbedingungen kann der DB2-Server gelegentlich die für Suchen in der Fremdtabelle verwendete Isolationsstufe unabhängig von der zuvor vom Benutzer festgelegten Isolationsstufe auf RR heraufsetzen. Dadurch werden zusätzliche Sperren bis zur COMMIT-Operation aktiviert, sodass die Wahrscheinlichkeit eines Deadlocks oder einer Sperrzeitlimitüberschreitung zunimmt. Zur Vermeidung solcher Probleme können Sie einen Index erstellen, der nur Fremdschlüsselspalten enthält und der vom Suchlauf zur Überprüfung auf referenzielle Integrität alternativ verwendet werden kann.

Lesestabilität (RS, Read Stability)

Die Isolationsstufe *Lesestabilität* (RS) sperrt nur die Zeilen, die von einer Anwendung während einer UOW abgerufen werden. Sie stellt sicher, dass jede den Vergleichselementen entsprechende gelesene Zeile während einer UOW nicht von Prozessen anderer Anwendungen geändert werden kann, bis die UOW abgeschlossen ist, und dass keine von einem anderen Anwendungsprozess geänderte Zeile gelesen werden kann, bis die Änderung von diesem Prozess festgeschrieben wurde. Unter der Isolationsstufe RS sind Zugriffe auf nicht festgeschriebene Daten und nicht wiederholbare Lesevorgänge nicht möglich. Lesevorgänge mit Phantomzeilen sind jedoch möglich.

Diese Isolationsstufe stellt sicher, dass alle zurückgegebenen Daten bis zu dem Zeitpunkt, zu dem die Daten für die Anwendung sichtbar werden, unverändert bleiben, auch wenn temporäre Tabellen oder Zeilenblockung verwendet werden.

Die Isolationsstufe RS stellt einen hohen Grad an gemeinsamem Zugriff und eine stabile Sicht der Daten zur Verfügung. Zu diesem Zweck stellt Optimierungsprogramm sicher, dass Sperren auf Tabellenebene erst aktiviert werden, wenn eine Sperreneskalation auftritt.

Die Isolationsstufe RS eignet sich für eine Anwendung mit folgenden Merkmalen:

- Sie arbeitet in einer Umgebung mit gemeinsamem Zugriff.

- Sie erfordert, dass Zeilen, die den Vergleichselementen entsprechen, die für die Dauer der UOW stabil bleiben müssen.
- Die setzt dieselbe Abfrage nicht mehr als einmal während einer UOW ab oder benötigt keine identischen Ergebnismengen, wenn eine Abfrage mehrmals während einer UOW abgesetzt wird.

Cursorstabilität (CS, Cursor Stability)

Die Isolationsstufe *Cursorstabilität* (CS) sperrt jede Zeile, auf die während einer Transaktion zugegriffen wird, während sich der Cursor auf dieser Zeile befindet. Diese Sperre bleibt in Kraft, bis die nächste Zeile abgerufen oder die Transaktion beendet wird. Wenn jedoch Daten in der Zeile geändert wurden, wird die Sperre beibehalten, bis die Änderung durch ein Commit festgeschrieben wird.

Unter dieser Isolationsstufe kann keine andere Anwendung eine Zeile aktualisieren oder löschen, während ein Aktualisierungscursor auf dieser Zeile positioniert ist. Unter der Isolationsstufe CS ist kein Zugriff auf nicht festgeschriebene Daten anderer Anwendungen möglich. Nicht wiederholbare Lesevorgänge und Lesevorgänge mit Phantomzeilen sind jedoch möglich.

Cursorstabilität ist die Standardisolationsstufe. Sie ist geeignet, wenn Sie einen maximalen gemeinsamen Zugriff wünschen und nur festgeschriebene Daten lesen wollen.

Anmerkung: Unter der Semantik zur Verarbeitung *zurzeit festgeschriebener Daten* ('cur_commit'), die in Version 9.7 eingeführt wurde, werden nur festgeschriebene Daten zurückgegeben, wie dies auch zuvor der Fall war. Jetzt warten jedoch lesende Anwendungen nicht mehr auf die Freigabe von Zeilensperren durch aktualisierende Anwendungen. Stattdessen geben lesende Anwendungen Daten auf der Basis der zurzeit festgeschriebenen Version zurück. Das heißt, Daten, wie sie vor dem Start der Schreiboperation vorlagen.

Nicht festgeschriebener Lesevorgang (UR, Uncommitted Read)

Die Isolationsstufe *Nicht festgeschriebener Lesevorgang* (UR) erlaubt einer Anwendung den Zugriff auf nicht festgeschriebene Änderungen anderer Transaktionen. Darüber hinaus verhindert die Isolationsstufe UR nicht, dass eine andere Anwendung auf eine Zeile zugreift, die gerade gelesen wird, sofern diese nicht versucht, die Tabelle zu ändern oder zu löschen.

Unter der Isolationsstufe UR sind Zugriffe auf nicht festgeschriebene Daten, nicht wiederholbare Lesevorgänge und Phantomlesevorgänge möglich. Diese Isolationsstufe ist geeignet, wenn Sie Abfragen auf schreibgeschützte Tabelle ausführen oder wenn Sie nur SELECT-Anweisungen ausführen und die Anzeige von Daten, die von anderen Anwendungen nicht festgeschrieben wurden, kein Problem ist.

Die Isolationsstufe UR funktioniert für Nur-Lese-Cursor und für Aktualisierungscursor unterschiedlich:

- Nur-Lese-Cursor 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 nicht verfügbar. Alle anderen Änderungen durch andere Transaktionen können gelesen werden, bevor sie mit COMMIT festgeschrieben oder mit ROLLBACK rückgän-

gig gemacht wurden. Aktualisierungscursor, die unter der Isolationsstufe UR arbeiten, verhalten sich wie unter der Isolationsstufe CS.

Wenn eine Anwendung unter der Isolationsstufe UR mehrdeutige Cursor verwendet, verwendet sie bei der Ausführung möglicherweise die Isolationsstufe CS. Die mehrdeutigen Cursor können auf die Isolationsstufe CS eskaliert werden, wenn der Wert UNAMBIG (Standardwert) für die Option BLOCKING im Befehl **PREP** oder **BIND** angegeben wird. Zur Vermeidung dieser Eskalation haben Sie folgende Möglichkeiten:

- Ändern Sie die Cursor im Anwendungsprogramm in eindeutige Cursor (UNAMBIGUOUS). Ändern Sie die SELECT-Anweisungen, um die Klausel FOR READ ONLY hinzuzufügen.
- Belassen Sie die Cursor im Anwendungsprogramm mehrdeutig. Kompilieren Sie jedoch das Programm mit den Optionen BLOCKING ALL und STATICREADONLY YES vor oder binden Sie es mit diesen Optionen, damit die mehrdeutigen Cursor als Nur-Lese-Cursor behandelt werden, wenn das Programm ausgeführt wird.

Vergleich der Isolationsstufen

In Tabelle 4 werden die unterstützten Isolationsstufen zusammengefasst.

Tabelle 4. Vergleich der Isolationsstufen

	UR	**CS**	**RS**	**RR**
Sind für eine Anwendung nicht festgeschriebene Änderungen anderer Anwendungsprozesse sichtbar?	Ja	Nein	Nein	Nein
Kann eine Anwendung nicht festgeschriebene Änderungen anderer Anwendungsprozesse aktualisieren?	Nein	Nein	Nein	Nein
Kann die erneute Ausführung einer Anweisung von anderen Anwendungsprozessen beeinflusst werden? [1]	Ja	Ja	Ja	Nein[2]
Können aktualisierte Zeilen von anderen Anwendungsprozessen aktualisiert werden? [3]	Nein	Nein	Nein	Nein
Können aktualisierte Zeilen von anderen Anwendungsprozessen gelesen werden, die unter einer anderen Isolationsstufe als UR ausgeführt werden?	Nein	Nein	Nein	Nein
Können aktualisierte Zeilen von anderen Anwendungsprozessen gelesen werden, die unter der Isolationsstufe UR ausgeführt werden?	Ja	Ja	Ja	Ja
Können Zeilen, auf die zugegriffen wurde, von anderen Anwendungsprozessen aktualisiert werden? [4]	Ja	Ja	Nein	Nein
Können Zeilen, auf die zugegriffen wurde, von anderen Anwendungsprozessen gelesen werden?	Ja	Ja	Ja	Ja
Kann die aktuelle Zeile von anderen Anwendungsprozessen aktualisiert oder gelöscht werden? [5]	Ja/Nein [6]	Ja/Nein [6]	Nein	Nein

Tabelle 4. Vergleich der Isolationsstufen (Forts.)

	UR	CS	RS	RR

Anmerkung:

1. Ein Beispiel für das *Phänomen von gelesenen Phantomzeilen* ist folgendes: Eine UOW mit dem Namen UW1 liest die Gruppe von *n* Zeilen, die eine Suchbedingung erfüllen. Eine UOW mit dem Namen UW2 fügt eine oder mehrere Zeilen ein, die dieselbe Suchbedingung erfüllen und schreibt die Änderung durch Commit fest. Wenn UW1 nachfolgend den Lesevorgang mit derselben Suchbedingung wiederholt, wird eine andere Ergebnismenge zurückgegeben: Die Zeilen, die ursprünglich gelesen wurden, und dazu die Zeilen, die von UW2 eingefügt wurden.
2. Wenn Ihre Berechtigungsnachweise für die kennsatzbasierte Zugriffssteuerung (LBAC) zwischen den Lesevorgängen geändert werden, können die Ergebnisse für den zweiten Lesevorgang abweichen, weil Sie Zugriff auf andere Zeilen haben.
3. Die Isolationsstufe bietet keinen Schutz für die Anwendung, wenn die Anwendung aus einer Tabelle liest und in sie schreibt. Eine Anwendung öffnet zum Beispiel einen Cursor auf einer Tabelle und führt eine Einfüge-, Aktualisierungs- oder Löschoperation an derselben Tabelle aus. Der Anwendung werden möglicherweise inkonsistente Daten zurückgegeben, wenn mehr Zeilen aus dem geöffneten Cursor abgerufen werden.
4. Ein Beispiel für das *Phänomen nicht wiederholbarer Lesevorgänge* ist folgendes: Eine UOW mit dem Namen UW1 liest eine Zeile. Eine UOW mit dem Namen UW2 ändert diese Zeile und schreibt sie mit Commit fest. Wenn UW1 anschließend diese Zeile erneut liest, wird ihr ein anderer Wert zurückgegeben.
5. Ein Beispiel für das *Phänomen des Lesens geänderter Daten* ist folgendes: Eine UOW mit dem Namen UW1 ändert eine Zeile. Eine UOW mit dem Namen UW2 liest diese Zeile, bevor UW1 eine Commitoperation ausführt. Falls UW1 nachfolgend die Änderungen durch Rollback rückgängig macht, hat UW2 Daten gelesen, die nicht mehr vorhanden sind.
6. Unter der Isolationsstufe UR oder CS kann die aktuelle Zeile, wenn der Cursor kein Aktualisierungscursor ist, in einigen Fällen von anderen Anwendungsprozessen aktualisiert oder gelöscht werden. Zum Beispiel kann sich aufgrund der Pufferung die aktuelle Zeile auf dem Client von der aktuellen Zeile auf dem Server unterscheiden. Darüber hinaus kann bei Verwendung der Semantik für zurzeit festgeschriebene Daten unter der Isolationsstufe CS eine Zeile, die gerade gelesen wird, anstehende nicht festgeschriebene Aktualisierungen haben. In diesem Fall wird immer die zurzeit festgeschriebene Version an die Anwendung zurückgegeben.

Zusammenfassung der Isolationsstufen

In Tabelle 5 werden die Aspekte des gemeinsamen Zugriffs in Verbindung mit verschiedenen Isolationsstufen aufgeführt.

Tabelle 5. Zusammenfassung der Isolationsstufen

Isolationsstufe	Zugriff auf nicht festgeschriebene Daten	Nicht wiederholbare Lesevorgänge	Lesevorgänge mit Phantomzeilen
Wiederholbares 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

Von der Isolationsstufe sind nicht nur die verschiedenen Grade der Isolation zwischen Anwendungen, sondern auch die Leistungsmerkmale einer einzelnen An-

wendung betroffen, weil die Verarbeitungs- 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. In Tabelle 6 finden Sie einfache heuristische Anhaltpunkte, die Ihnen bei der Wahl der ersten Isolationsstufe für Ihre Anwendungen helfen können.

Tabelle 6. Richtlinien zur Wahl einer Isolationsstufe

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

Angeben der Isolationsstufe

Da die Isolationsstufe festlegt, wie Daten 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 gegeneinander abwägt.

Informationen zu diesem Vorgang

Die Isolationsstufe, die Sie angegeben, gilt für die Dauer der UOW (Unit of Work, Arbeitseinheit). Die folgenden heuristischen Methoden können verwendet werden, um die Isolationsstufe zu bestimmen, die zum Kompilieren einer SQL- oder XQuery-Anweisung verwendet wird:

- Für statisches SQL:
  - Wenn eine *isolation-klausel* in der Anweisung angegeben wird, wird der Wert dieser Klausel verwendet.
  - Wenn keine *isolation-klausel* in der Anweisung angegeben wird, wird die Isolationsstufe verwendet, die für das Paket angegeben war, als das Paket an die Datenbank gebunden (BIND) wurde.
- Für dynamisches SQL:
  - Wenn eine *isolation-klausel* in der Anweisung angegeben wird, wird der Wert dieser Klausel verwendet.
  - Wenn keine *isolation-klausel* in der Anweisung angegeben wird und die Anweisung SET CURRENT ISOLATION in der aktuellen Sitzung abgesetzt wurde, wird der Wert des Sonderregisters CURRENT ISOLATION verwendet.
  - Wenn keine *isolation-klausel* in der Anweisung angegeben wird und die Anweisung SET CURRENT ISOLATION in der aktuellen Sitzung nicht abgesetzt wurde, wird die Isolationsstufe verwendet, die für das Paket angegeben war, als das Paket an die Datenbank gebunden (BIND) wurde.
- Für statische oder dynamische XQuery-Anweisungen legt die Isolationsstufe der Umgebung die Isolationsstufe fest, die bei der Auswertung des XQuery-Ausdrucks verwendet wird.

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

Die Isolationsstufe kann auf verschiedene Arten angegeben werden.

Vorgehensweise

- **Auf Anweisungsebene:**

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

  Verwenden Sie die Klausel WITH. 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.

  Diese Isolationsstufe setzt die Isolationsstufe außer Kraft, die für das Paket angegeben ist, in dem die Anweisung enthalten ist. Für die folgenden SQL-Anweisungen können Sie eine Isolationsstufe angeben:

  – DECLARE CURSOR
  – DELETE mit Suche
  – INSERT
  – SELECT
  – SELECT INTO
  – UPDATE mit Suche

- **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. Durch die Ausführung dieser Anweisung wird das Sonderregister CURRENT ISOLATION auf einen Wert gesetzt, der die Isolationsstufe für alle dynamischen SQL-Anweisungen angibt, die innerhalb der aktuellen Sitzung abgesetzt werden. Sobald es festgelegt ist, gibt das Sonderregister CURRENT ISOLATION die Isolationsstufe für alle nachfolgenden dynamischen SQL-Anweisungen an, die innerhalb der Sitzung kompiliert werden, unabhängig davon, welches Paket die Anweisung abgesetzt hat. Diese Isolationsstufe bleibt in Kraft, bis die Sitzung endet oder bis die Anweisung SET CURRENT ISOLATION...RESET ausgeführt wird.

- **Beim Vorkompilieren oder Binden:**

  Für eine Anwendung, die in einer unterstützten kompilierten Sprache geschrieben ist, verwenden Sie die Option ISOLATION der Befehle **PREP** oder **BIND**. Sie können die Isolationsstufe auch über die API sqlaprep bzw. sqlabndx 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.

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

  ```
  select isolation from syscat.packages
    where pkgname = 'paketname'
      and pkgschema = 'paketschema'
  ```

  Dabei ist *paketname* der nicht qualifizierte Name des Pakets und *paketschema* der Schemaname des Pakets. Beide Namen müssen vollständig in Großbuchstaben angegeben werden.

- **Über JDBC oder SQLJ bei der Ausführung:**

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

Zur Erstellung eines Pakets (und zur Angabe der zugehörigen Isolationsstufe) in SQLJ verwenden Sie die SQLJ-Profilanpassungsfunktion (Befehl **db2sqljcustomize**).

- **Über CLI oder ODBC bei der Ausführung:**

  Verwenden Sie den Befehl **CHANGE ISOLATION LEVEL**. Mit DB2 Call-level Interface (CLI) können Sie die Isolationsstufe bei der CLI-Konfiguration ändern. Verwenden Sie während der Ausführung die Funktion SQLSetConnectAttr mit dem Attribut SQL_ATTR_TXN_ISOLATION, um die Transaktionsisolationsstufe für die aktuelle Verbindung zu festzulegen, auf die durch das Argument *ConnectionHandle* verwiesen wird. Sie können auch das Schlüsselwort TXNISOLATION in der Datei `db2cli.ini` verwenden.

- **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 (CLP-Pakete) werden ebenfalls an die Datenbank gebunden, wenn eine Datenbank erstellt wird.

  REXX und der Befehlszeilenprozessor (CLP) stellen eine Verbindung zu einer Datenbank unter Verwendung der Standardisolationsstufe 'Cursorstabilität' (CS) her. Durch eine Änderung dieser Isolationsstufe wird der Status der Verbindung nicht geändert.

  Zur Ermittlung der Isolationsstufe, die von einer REXX-Anwendung verwendet wird, prüfen Sie den Wert der vordefinierten REXX-Variablen SQLISL. Der Wert wird jedes Mal aktualisiert, wenn der Befehl **CHANGE ISOLATION LEVEL** ausgeführt wird.

Ergebnisse

Semantik für zurzeit festgeschriebene Daten verbessert den gemeinsamen Zugriff

Überschreitungen des Sperrzeitlimits und Deadlocks können unter der Isolationsstufe 'Cursorstabilität' (CS) für Sperren auf Zeilenebene auftreten, insbesondere bei Anwendungen, die nicht darauf ausgerichtet wurden, solche Probleme zu vermeiden. Einige Datenbankanwendungen mit hohem Durchsatz können kein Warten auf Sperren tolerieren, die während der Transaktionsverarbeitung angefordert werden, und einige Anwendungen können keine Verarbeitung nicht festgeschriebener Daten tolerieren, erfordern jedoch trotzdem ein blockierungsfreies Verhalten für Lesetransaktionen.

Unter der neuen Semantik zur Verarbeitung nur *zurzeit festgeschriebener Daten* werden nur festgeschriebene Daten zurückgegeben, wie dies auch zuvor der Fall war. Jetzt warten jedoch lesende Anwendungen nicht mehr auf die Freigabe von Zeilensperren durch Ausgabeprogramme. Stattdessen geben lesende Anwendungen Daten auf der Basis der zurzeit festgeschriebenen Version zurück. Das heißt, Daten, wie sie vor dem Start der Schreiboperation vorlagen.

Die Semantik zur Verarbeitung nur der zurzeit festgeschriebenen Daten wird für neue Datenbanken standardmäßig aktiviert. Dadurch kann jede Anwendung von dem neuen Verhalten profitieren und es sind keine Änderungen an der Anwendung selbst erforderlich. Der neue Datenbankkonfigurationsparameter **cur_commit** kann zum Überschreiben dieses Verhaltens verwendet werden. Dies könnte zum

Beispiel im Fall von Anwendungen nützlich sein, für die eine Blockierung bei schreibenden Anwendungen erforderlich ist, um interne Logik zu synchronisieren.

Dementsprechend wird bei Datenbanken, für die ein Upgrade durchgeführt wird, der Parameter **cur_commit** standardmäßig für den Fall inaktiviert, dass Anwendungen eine Blockierung von schreibenden Anwendungen erfordern, um ihre interne Logik zu synchronisieren. Dieser Parameter kann später aktiviert werden, wenn dies gewünscht wird.

Die Semantik für zurzeit festgeschriebene Daten gilt nur für reine Lesesuchläufe, die keine Katalogtabellen oder die internen Suchläufe betreffen, die zum Auswerten oder Erzwingen von Integritätsbedingungen verwendet werden. Beachten Sie, dass der Zugriffsplan einer schreibenden Anwendung Suchläufe auf zurzeit festgeschriebenen Daten enthalten kann, weil der Status von zurzeit festgeschriebenen Daten auf der Suchlaufebene festgestellt wird. Zum Beispiel kann der Suchlauf für eine reine Leseunterabfrage die Semantik für zurzeit festgeschriebene Daten mit einbeziehen. Da die Semantik für zurzeit festgeschriebene Daten der Isolationsstufensemantik untergeordnet ist, beachten Anwendungen, die unter der Semantik für zurzeit festgeschriebene Daten ausgeführt werden, weiterhin die Isolationsstufen.

Die Semantik für zurzeit festgeschriebene Daten erfordert einen größeren Protokollspeicherbereich für Ausgabeprogramme. Zusätzlicher Speicherbereich ist für die Protokollierung der ersten Aktualisierung einer Datenzeile während einer Transaktion erforderlich. Diese Daten werden zum Abrufen des zurzeit festgeschriebenen Abbilds der Zeile benötigt. Abhängig von der Auslastung kann dies eine unerhebliche oder messbare Auswirkung auf den insgesamt belegten Protokollspeicherbereich haben. Der Bedarf an zusätzlichem Protokollspeicher fällt nicht an, wenn der Parameter **cur_commit** inaktiviert ist.

Einschränkungen

Die folgenden Einschränkungen gelten für die Semantik für zurzeit festgeschriebene Daten:

- Das Zieltabellenobjekt in einem Abschnitt, der für Datenaktualisierungs- oder Datenlöschoperationen verwendet werden soll, verwendet keine Semantik für zurzeit festgeschriebene Daten. Zeilen, die modifiziert werden sollen, müssen durch Sperren geschützt werden, um sicherzustellen, dass sie nach dem Erfüllen von Abfragevergleichselementen, die Teil der Aktualisierungsoperation sind, nicht geändert werden.
- Eine Transaktion, die eine nicht festgeschriebene Modifikation an einer Zeile vorgenommen hat, erzwingt den Zugriff auf die entsprechenden Protokollsätze durch die lesende Anwendung der zurzeit festgeschriebenen Daten, um die zurzeit festgeschriebene Version der Zeile zu ermitteln. Protokollsätze, die sich nicht mehr im Protokollpuffer befinden, können zwar physisch gelesen werden, die Semantik für zurzeit festgeschriebene Daten unterstützt jedoch das Abrufen der Protokolldateien aus dem Protokollarchiv nicht. Dies betrifft nur Datenbanken, die für die Endlosprotokollierung konfiguriert sind.
- Die folgenden Suchen verwenden keine Semantik für zurzeit festgeschriebene Daten:
  - Katalogtabellesuchen
  - Suchen, die zum Erzwingen referenzieller Integritätsbedingungen verwendet werden
  - Suchen, die LONG VARCHAR- oder LONG VARGRAPHIC-Spalten refrenzieren

– RCT-Suchen (RCT = Range-Clustered Table, Bereichsclustertabelle)
– Suchen, die räumliche oder erweiterte Indizes verwenden

Beispiel

Betrachten Sie das folgende Szenario, in dem Deadlocks unter der Semantik für zurzeit festgeschriebene Daten vermieden werden. In diesem Szenario aktualisieren zwei Anwendungen zwei separate Tabellen, führen jedoch noch keine Commitoperation aus. Jede Anwendung versucht dann, Daten (mit einem Nur-Lese-Cursor) aus der Tabelle zu lesen, die von der anderen Anwendung aktualisiert wurde.

Schritt	Anwendung A	Anwendung B
1	update T1 set col1 = ? where col2 = ?	update T2 set col1 = ? where col2 = ?
2	select col1, col3, col4 from T2 where col2 >= ?	select col1, col5, from T1 where col5 = ? and col2 = ?
3	Commit	Commit

Ohne die Semantik für zurzeit festgeschriebene Daten könnten diese Anwendung unter der Isolationsstufe 'Cursorstabilität' einen Deadlock verursachen, sodass die Ausführung einer der Anwendungen fehlschlagen würde. Dies geschieht, wenn jede der beiden Anwendungen Daten lesen muss, die von der anderen Anwendung gerade aktualisiert werden.

Wenn unter der Semantik für zurzeit festgeschriebene Daten die Abfrage in Schritt 2 (für beide Anwendungen) zufällig die Daten benötigen würde, die zurzeit von der anderen Anwendung aktualisiert werden, würde diese Anwendung nicht auf die Freigabe der Sperre warten, sodass ein Deadlock nicht möglich wäre. Stattdessen würde die zuvor festgeschriebene Version der Daten lokalisiert und verwendet.

Option zum Ignorieren nicht festgeschriebener Einfügungen

Die Registrierdatenbankvariable **DB2_SKIPINSERTED** steuert, ob nicht festgeschriebene Dateneinfügungen für Anweisungen ignoriert werden können, die mit der Isolationsstufe der Cursorstabilität (CS) oder der Lesestabilität (RS) arbeiten.

Nicht festgeschriebene Einfügungen werden abhängig vom Wert der Registrierdatenbankvariablen **DB2_SKIPINSERTED** auf eine von zwei Arten behandelt.

- Wenn der Wert ON ist, ignoriert der DB2-Server nicht festgeschriebene Einfügungen. Dies kann in vielen Fällen den gemeinsamen Zugriff verbessern und stellt das bevorzugte Verhalten für die meisten Anwendungen dar. Nicht festgeschriebene Einfügungen werden so behandelt, als hätten sie noch nicht stattgefunden.
- Wenn der Wert OFF (Standardwert) ist, wartet der DB2-Server, bis die Einfügeoperation (entweder durch Commit oder Rollback) abgeschlossen wird und verarbeitet anschließend die Daten entsprechend. Dies ist in bestimmten Fällen ein sinnvolles Verhalten. Beispiele:
  – Nehmen Sie an, dass zwei Anwendungen eine Tabelle verwenden, 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 übergebenen Reihenfolge verarbeitet werden. Das heißt, wenn die nächste zu lesende Zeile von der ersten Anwendung eingefügt wird, muss die zweite Anwendung warten, bis die Einfügeoperation durch Commit festgeschrieben wird.
  – Eine Anwendung vermeidet UPDATE-Anweisungen, indem sie Daten löscht und anschließend ein neues Image der Daten einfügt.

Auswerten nicht festgeschriebener Daten durch Sperrenverzögerung

Zur Verbesserung des gemeinsamen Zugriffs lässt der Datenbankmanager in einigen Fällen eine Verzögerung von Zeilensperren für Suchen unter der Isolationsstufe CS oder RS zu, bis für eine Zeile festgestellt wird, dass sie die Vergleichselemente einer Abfrage erfüllt.

Standardmäßig sperrt der Datenbankmanager bei Verwendung der Zeilensperrung während einer Tabellen- oder Indexsuche jede durchsuchte Zeile, deren Festschreibestatus (COMMIT-Status) unbekannt ist, bevor festgestellt wird, ob die Zeile die Vergleichselemente der Abfrage erfüllt.

Zur Verbesserung des gemeinsamen Zugriffs bei solchen Suchläufen können Sie die Registrierdatenbankvariable **DB2_EVALUNCOMMITTED** aktivieren, sodass die Auswertung von Vergleichselementen für nicht festgeschriebene Daten erfolgen kann. Eine Zeile, die eine nicht festgeschriebene Aktualisierung enthält, erfüllt die Abfrage vielleicht nicht. Wenn jedoch die Auswertung von Vergleichselementen bis nach dem Abschluss der Transaktion verzögert wird, erfüllt die Zeile möglicherweise die Abfrage.

Wenn die Registrierdatenbankvariable **DB2_SKIPDELETED** aktiviert ist, werden nicht festgeschriebene gelöschte Zeilen bei Tabellensuchen übersprungen und der Datenbankmanager überspringt gelöschte Schlüssel bei Indexsuchen.

Die Einstellung der Registrierdatenbankvariablen **DB2_EVALUNCOMMITTED** gilt bei der Kompilierung für dynamische SQL- oder XQuery-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 implementiert 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 angewendet. Für statische SQL- oder XQuery-Anweisungen gilt beim Rebind eines Pakets die Einstellung der Registrierdatenbankvariablen, die zum Zeitpunkt des Bindens in Kraft ist. 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 7. Zugriff nur über Satz-ID-Index

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

Tabelle 8. Reiner Datenzugriff (relational oder mit verzögerter Satz-ID-Liste)

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

Tabelle 9. 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 10. 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 zwischen der Standardfunktionsweise von Sperren und der Funktionsweise mit Auswertung nicht festgeschriebener Daten. Die 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 finden unter der Standardisolationsstufe der Cursorstabilität (CS) statt.

Tabelle 11. Transaktionen an der Tabelle ORG unter der Isolationsstufe CS

SITZUNG 1	SITZUNG 2
`connect to sample`	`connect to sample`
`+c update org set  deptnumb=5 where manager=160`	

Tabelle 11. Transaktionen an der Tabelle ORG unter der Isolationsstufe CS (Forts.)

SITZUNG 1	SITZUNG 2
	`select * from org where deptnumb >= 10`

Die nicht festgeschriebene UPDATE-Anweisung in Sitzung 1 aktiviert eine exklusive Sperre für die erste Zeile in der Tabelle, die verhindert, dass die Abfrage in Sitzung 2 eine Ergebnismenge zurückgibt, obwohl die Zeile, die in Sitzung 1 aktualisiert wird, momentan die Abfrage in Sitzung 2 nicht erfüllt. Die Isolationsstufe der Cursorstabilität (CS) legt fest, 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.

Das Warten auf eine Sperre in Sitzung 2 kann durch die Verwendung der Funktion zur Auswertung nicht festgeschriebener Daten vermieden werden. Diese Funktion wertet zunächst das Vergleichselement aus und sperrt dann die Zeile. In diesem Fall würde die Abfrage in Sitzung 2 nicht versuchen, die erste Zeile in der Tabelle zu sperren, sodass der gemeinsame Zugriff durch Anwendungen verbessert wird. Beachten Sie, dass dies auch bedeutet, dass die Auswertung des Vergleichselements in Sitzung 2 im Hinblick auf den nicht festgeschriebenen Wert `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, könnte mit der Funktion zur Auswertung nicht festgeschriebener Daten trotzdem eine Verbesserung für den gemeinsamen Zugriff erzielt werden. Bei der Standardfunktionsweise von Sperren würde Sitzung 2 zunächst eine Zeilensperre anfordern, die die Ausführung der UPDATE-Anweisung mit Suche in Sitzung 1 verhindern würde, obwohl die UPDATE-Anweisung in Sitzung 1 die von der Abfrage in Sitzung 2 gesperrte Zeile nicht ändern würde. Wenn die UPDATE-Anweisung 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 keine Blockierung verursachen.

Einschränkungen

- Die Registrierdatenbankvariable **DB2_EVALUNCOMMITTED** muss aktiviert sein.
- Die Isolationsstufe muss CS oder RS sein.
- Sperren auf Zeilenebene werden aktiviert.
- Als Suchargument verwendbare Vergleichselemente sind vorhanden.
- Die Auswertung nicht festgeschriebener Daten gilt nicht für Suchen in den Katalogtabellen.
- Für MDC-Tabellen (Tabellen mit mehrdimensionalem Clustering) können Sperren auf Blockebene für eine Indexsuche verzögert werden. Bei Tabellensuchen können Sperren auf Blockebene jedoch nicht verzögert werden.
- Die Sperrverzögerung erfolgt nicht für eine Tabelle, die eine Inplace-Tabellenreorganisation ausführt.
- Für Iscan-Fetch-Pläne werden Sperren auf Zeilenebene 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 Indexschlüssel nur übersprungen werden, wenn die Registrierdatenbankvariable **DB2_SKIPDELETED** aktiviert ist.

Schreiben und Optimieren von Abfragen für eine optimale Leistung

Es stehen verschiedene Möglichkeiten zur Verfügung, die Auswirkungen von SQL-Anweisungen auf die DB2-Datenbankleistung zu minimieren.

Sie können diese Auswirkungen durch folgende Maßnahmen minieren:

- Schreiben von SQL-Anweisungen, die das DB2-Optimierungsprogramm leichter optimieren kann. Das DB2-Optimierungsprogramm ist möglicherweise nicht in der Lage, SQL-Anweisungen effizient auszuführen, die Joinvergleichselemente mit Ungleichheitsoperatoren, Datentypabweichungen in Joinspalten, unnötige Outer Joins und andere komplexe Suchbedingungen enthalten.
- Korrektes Konfigurieren der DB2-Datenbank, um die Vorteile der DB2-Optimierungsfunktionalität zu nutzen. Das DB2-Optimierungsprogramm kann den optimalen Abfragezugriffsplan auswählen, wenn präzise Katalogstatistiken vorhanden sind und die beste Optimierungsklasse für die jeweilige Auslastung ausgewählt ist.
- Verwenden der DB2-EXPLAIN-Funktionalität, um potenzielle Abfragezugriffspläne zu prüfen und zu ermitteln, wie Abfragen auf beste Leistung optimiert werden.

Empfohlene Methoden beziehen sich auf allgemeine Auslastungen, Data-Warehouse-Auslastungen und SAP-Auslastungen.

Obwohl es eine Reihe von Möglichkeiten gibt, bestimmte Probleme der Abfrageleistung zu behandeln, nachdem eine Anwendung geschrieben wurde, können gute und grundlegende Schreib- und Optimierungsverfahren schon früh und weitreichend angewendet werden, um eine Verbesserung der DB2-Datenbankleistung zu erleichtern.

Die Abfrageleistung ist keine einmalige Überlegung. Sie müssen sie während der gesamten Entwurfs-, Entwicklungs- und Produktionsphasen des Anwendungsentwicklungszyklus im Blick behalten.

SQL ist eine sehr flexible Sprache. Dies bedeutet, dass sie zahlreiche Möglichkeiten bietet, zum selben korrekten Ergebnis zu gelangen. Diese Flexibilität bedeutet außerdem, dass einige Abfragen besser als andere die vom DB2-Optimierungsprogramm gebotenen Vorteile nutzen können.

Während der Ausführung von Abfragen wählt das DB2-Optimierungsprogramm einen Abfragezugriffsplan für jede SQL-Anweisung aus. Das Optimierungsprogramm modelliert den Ausführungsaufwand zahlreicher alternativer Zugriffspläne und wählt den mit dem geringsten geschätzten Aufwand aus. Wenn eine Abfrage zahlreiche komplexe Suchbedingungen enthält, kann das DB2-Optimierungsprogramm das Vergleichselement in einigen Fällen umschreiben. Es gibt jedoch auch Fälle, in denen dies nicht möglich ist.

Die Zeit zum Vorbereiten und Kompilieren einer SQL-Anweisung kann bei komplexen Abfragen, wie zum Beispiel Abfragen in Business-Intelligence-Anwendungen, geraume Zeit in Anspruch nehmen. Sie können eine Minimierung der Zeit, die für die Anweisungskompilierung aufgewendet wird, unterstützen, indem Sie

Ihre Datenbank ordnungsgemäß entwerfen und konfigurieren. Dazu gehört auch die Auswahl der geeigneten Optimierungsklasse und die korrekte Einstellung weiterer Registrierdatenbankvariablen.

Das Optimierungsprogramm erfordert zudem präzise Eingaben, um korrekte Zugriffsplanentscheidungen treffen zu können. Dies bedeutet, dass Sie präzise Statistikdaten erfassen und möglicherweise sogar erweiterte Statistikmerkmale wie Statistiksichten und Spaltengruppenstatistiken verwenden müssen.

Sie können Abfragen mithilfe der DB2-Tools, insbesondere mithilfe der DB2-EXPLAIN-Funktion, optimieren. Der DB2-Compiler kann Informationen zu den Zugriffsplänen und Umgebungen von statischen oder dynamischen Abfragen erfassen. Aus diesen erfassten Informationen können Sie Erkenntnisse darüber gewinnen, wie einzelne Anweisungen ausgeführt werden, sodass Sie diese Anweisungen und die Konfiguration Ihres Datenbankmanagers zur Realisierung einer besseren Leistung optimieren können.

Schreiben von SQL-Anweisungen

SQL ist eine leistungsfähige Sprache, bei der Sie relationale Ausdrücke auf syntaktisch unterschiedliche, jedoch semantisch äquivalente Weise angeben können. Allerdings lassen sich einige semantisch äquivalente Varianten einfacher optimieren als andere. Das DB2-Optimierungsprogramm verfügt zwar über eine leistungsfähige Funktionalität zum Umschreiben von Abfragen, ist jedoch vielleicht nicht in jedem Fall in der Lage, eine SQL-Anweisung in die optimale Form umzuschreiben.

Bestimmte SQL-Konstrukte können die Palette an Zugriffsplänen, die vom Abfrageoptimierungsprogramm berücksichtigt wird, einschränken. Diese Konstrukte sollten vermieden bzw., soweit möglich, ersetzt werden.

Vermeiden komplexer Ausdrücke in Suchbedingungen:

Vermeiden Sie die Verwendung komplexer Ausdrücke in Suchbedingungen, bei denen die Ausdrücke verhindern, dass das Optimierungsprogramm die Katalogstatistiken zur präzisen Abschätzung einer Selektivität verwendet.

Die Ausdrücke begrenzen möglicherweise auch die Auswahlmöglichkeiten von Zugriffsplänen, die zur Anwendung des Vergleichselements verwendet werden können. In der Optimierungsphase der Abfrageumschreibung kann das Optimierungsprogramm eine Reihe von Ausdrücken umschreiben, sodass es die Selektivität präzise abschätzen kann. Es kann jedoch nicht alle Möglichkeiten berücksichtigen.

Vermeiden von Joinvergleichselementen in Ausdrücken:

Die Verwendung von Joinvergleichselementen in Ausdrücken beschränkt die Joinmethode auf Joins mit Verschachtelungsschleife (Nested Loop Join).

Darüber hinaus kann die Kardinalitätsschätzung ungenau sein. Die folgenden Beispiele zeigen Joins mit Ausdrücken:

```
WHERE SALES.PRICE * SALES.DISCOUNT = TRANS.FINAL_PRICE
WHERE UPPER(CUST.LASTNAME) = TRANS.NAME
```

Vermeiden von Ausdrücken für Spalten in lokalen Vergleichselementen:

Statt einen Ausdruck über Spalten in einem lokalen Vergleichselement anzuwenden, sollten Sie die Umkehrung des Ausdrucks verwenden.

Betrachten Sie die folgenden Beispiele:

```
AUSDRUCK(C) = 'konstante'
INTEGER(TRANS_DATE)/100 = 200802
```

Diese Anweisungen können wie folgt umgeschrieben werden:

```
C = UMKEHRAUSDRUCK('konstante')
TRANS_DATE BETWEEN 20080201 AND 20080229
```

Die Anwendung von Ausdrücken über Spalten verhindert die Verwendung von Start- und Stoppschlüsseln in Indizes, führt zu unpräzisen Selektivitätsschätzungen und erfordert einen zusätzlichen Verarbeitungsaufwand bei der Abfrageausführung.

Solche Ausdrücke können auch Abfrageoptimierungen durch Umschreiben verhindern, die zum Beispiel darin bestünden, dass erkannt wird, wenn Spalten äquivalent sind, dass Spalten durch Konstanten ersetzt werden oder dass ermittelt wird, dass höchstens eine Zeile zurückgegeben wird. Wenn nachgewiesen wird, dass höchstens eine Zeile zurückgegeben wird, können normalerweise weitere Optimierungen angewendet werden. Daher wird der Verlust an Optimierungsmöglichkeiten noch größer. Betrachten Sie die folgende Abfrage:

```
SELECT LASTNAME, CUST_ID, CUST_CODE FROM CUST
WHERE (CUST_ID * 100) + INT(CUST_CODE) = 123456 ORDER BY 1,2,3
```

Diese Anweisung kann wie folgt umgeschrieben werden:

```
SELECT LASTNAME, CUST_ID, CUST_CODE FROM CUST
WHERE CUST_ID = 1234 AND CUST_CODE = '56' ORDER BY 1,2,3
```

Wenn für die Spalte CUST_ID ein eindeutiger Index definiert ist, bietet die umgeschriebene Version der Abfrage dem Abfrageoptimierungsprogramm die Möglichkeit zu erkennen, dass höchstens eine Zeile zurückgegeben wird. Dadurch wird eine unnötige Sortieroperation vermieden. Darüber hinaus können die Spalten CUST_ID und CUST_CODE durch die Werte 1234 und '56' ersetzt werden, sodass ein Kopieren von Werten aus den Daten- oder Indexseiten vermieden wird. Und schließlich kann das Vergleichselement für die Spalte CUST_ID als Start- oder Stoppschlüssel für den Index angewendet werden.

Es ist möglicherweise nicht immer offensichtlich, dass ein Ausdruck in einem Vergleichselement enthalten ist. Dies ist häufig der Fall bei Abfragen, die auf Sichten verweisen, deren Spalten durch Ausdrücke definiert sind. Betrachten Sie zum Beispiel die folgende Sichtdefinition und Abfrage:

```
CREATE VIEW CUST_V AS
  (SELECT LASTNAME, (CUST_ID * 100) + INT(CUST_CODE) AS CUST_KEY
    FROM CUST)

SELECT LASTNAME FROM CUST_V WHERE CUST_KEY = 123456
```

Das Abfrageoptimierungsprogramm fügt die Abfrage mit der Sichtdefinition zusammen, sodass sich die folgende Abfrage ergibt:

```
SELECT LASTNAME FROM CUST
WHERE (CUST_ID * 100) + INT(CUST_CODE) = 123456
```

Dieses Vergleichselement ist das gleiche problematische Vergleichselement wie im vorherigen Beispiel. Sie können das Ergebnis einer Sichtzusammenfügung mithilfe der EXPLAIN-Funktion zum Anzeigen des optimierten SQL prüfen.

Wenn die Umkehrfunktion schwierig auszudrücken ist, ziehen Sie die Verwendung einer generierten Spalte in Betracht. Wenn Sie zum Beispiel einen Nachnamen ermitteln wollen, der die Bedingungen im Ausdruck `LASTNAME IN ('Woo', 'woo', 'WOO', 'WOo',...)` erfüllt, können Sie eine generierte Spalte `UCASE(LASTNAME) = 'WOO'` wie folgt erstellen:

```
CREATE TABLE CUSTOMER (
  LASTNAME VARCHAR(100),
  U_LASTNAME VARCHAR(100) GENERATED ALWAYS AS (UCASE(LASTNAME))
)

CREATE INDEX CUST_U_LASTNAME ON CUSTOMER(U_LASTNAME)
```

Die Unterstützung für von der Groß-/Kleinschreibung unabhängige Suchoperationen in DB2 Database für Linux, UNIX und Windows Version 9.5 Fixpack 1 ist darauf ausgerichtet, die Situation in diesem besonderen Beispiel zu lösen. Sie können das Attribut _Sx für den Namen der Sortierfolge UCA500R1 verwenden, um die Stärke von Sortierfolgen zu steuern. Zum Beispiel ist UCA500R1_LFR_S1 eine französische Sortierfolge, bei der die Groß-/Kleinschreibung und Akzente ignoriert werden.

Vermeiden nicht übereinstimmender Datentypen in Joinspalten:

In einigen Fällen verhindern nicht übereinstimmende Datentypen die Verwendung von Hash-Joins.

Für einen Hash-Join gelten im Vergleich zu anderen Joinmethoden noch zusätzliche Einschränkungen für die Joinvergleichselemente. Insbesondere müssen die Datentypen in den Joinspalten exakt übereinstimmen. Wenn zum Beispiel die eine Joinspalte den Typ FLOAT und die andere den Typ REAL hat, wird der Hash-Join nicht unterstützt. Darüber hinaus müssen bei den Datentypen CHAR, GRAPHIC, DECIMAL oder DECFLOAT in Joinspalten die Längen identisch sein.

Vermeiden von Ausdrücken mit Nulloperationen in Vergleichselementen zum Ändern der Schätzung des Optimierungsprogramms:

Ein Nulloperationsvergleichselement mit 'coalesce()' der Form `COALESCE(X, X) = X` führt bei allen Abfragen, die es verwenden, zu einem Schätzfehler in der Planung. Gegenwärtig verfügt der DB2-Abfragecompiler über keine Funktion, um dieses Vergleichselement zu analysieren und festzustellen, dass es tatsächlich von allen Zeilen erfüllt wird.

Infolgedessen reduziert das Vergleichselement die geschätzte Anzahl von Zeilen, die aus einem Teil eines Abfrageplans kommen, auf künstliche Weise. Diese kleinere Zeilenschätzung verringert in der Regel die Zeilen- und Aufwandsschätzwerte für den Rest der Abfrageplanung. Manchmal führt sie zur Auswahl eines anderen Plans, weil sich die relativen Schätzwerte zwischen unterschiedlichen Kandidatenplänen geändert haben.

Wie ist es möglich, dass Vergleichselemente mit Nulloperationen manchmal die Abfrageleistung verbessern? Durch das Hinzufügen des Nulloperationsvergleichselement mit 'coalesce()' wird ein Fehler eingeführt, der einen anderen Umstand verdeckt, der eine optimale Leistung verhindert.

Einige Tools zur Leistungsverbesserung wenden ein simples Wiederholungstestverfahren an: Das Tool baut das Vergleichselement wiederholt an verschiedenen Stellen für verschiedene Spalten in eine Abfrage ein und versucht einen Fall zu finden, in dem durch die Einführung eines Fehlers die Abfrage auf einen Plan mit besse-

rer Leistung stößt. Ähnliches gilt auch für einen Abfrageentwickler, der das Nulloperationsvergleichselement manuell in eine Abfrage kopiert. In der Regel verfügt der Entwickler über einige Kenntnisse über die Daten, die die Platzierung des Vergleichselements nahe legen.

Diese Methode zur Verbesserung der Abfrageleistung ist eine kurzfristige Lösung, die nicht auf die eigentliche Ursache eingeht und die folgenden Auswirkungen haben könnte:

- Potenzielle Bereiche für Leistungsverbesserungen werden verdeckt.
- Es gibt keine Garantien, dass diese Ausweichlösung permanente Leistungsverbesserungen bringt, weil der DB2-Abfragecompiler das Vergleichselement schließlich besser verarbeiten kann oder Zufallsfaktoren eine Rolle spielen könnten.
- Es könnten andere Abfragen vorhanden sein, die von derselben Ursache betroffen sind, sodass infolgedessen die Leistung Ihres Systems insgesamt beeinträchtigt werden kann.

Wenn Sie empfohlene Methoden befolgt haben, jedoch glauben, dass weiterhin eine nicht optimale Leistung erzielt wird, können Sie explizite Optimierungsrichtlinien für das DB2-Optimierungsprogramm angeben. Dies ist sinnvoller, als ein Vergleichselement mit Nulloperationen einzuführen. Nähere Informationen hierzu finden Sie im Abschnitt „Optimierungsprofile und Optimierungsrichtlinien".

Vermeiden von Joinvergleichselementen mit Ungleichheitsoperatoren:

Joinvergleichselemente, die andere Vergleichsoperatoren als Gleichheitsvergleichsoperatoren verwenden, sollten vermieden werden, weil die Joinmethode auf einen Join mit Verschachtelungsschleife begrenzt ist.

Darüber hinaus ist das Optimierungsprogramm möglicherweise nicht in der Lage, eine präzise Selektivitätsschätzung für das Joinvergleichselement zu berechnen. Allerdings können Joinvergleichselemente mit Ungleichheitsoperatoren nicht immer vermieden werden. Wenn sie erforderlich sind, stellen Sie sicher, dass ein geeigneter Index für beide Tabellen vorhanden ist, da die Joinvergleichselemente auf die innere Tabelle des Joins mit Verschachtelungsschleife angewendet werden.

Ein gängiges Beispiel für Joinvergleichselemente mit Ungleichheitsoperatoren ist der Fall, in dem Dimensionsdaten in einem Sternschema versionsgesteuert sein müssen, um den Status einer Dimension zu verschiedenen Zeitpunkten genau wiederzugeben. Dies wird häufig als eine *Slowly Changing Dimension* (sich langsam ändernde Dimension) bezeichnet. Ein Typ einer Slowly Changing Dimension ist mit der Einfügung effektiver Start- und Enddatumswerte für jede Dimensionszeile verbunden. Ein Join zwischen der Fakttabelle und der Dimensionstabelle erfordert nicht nur den Join über den Primärschlüssel, sondern auch eine Überprüfung, ob ein Datum, das mit dem Fakt verbunden ist, in den Zeitraum zwischen Start- und Enddatum der Dimension fällt. Dies wird häufig als *Typ 6 von Slowly Changing Dimension* bezeichnet. Der Bereichsjoin zurück zur Fakttabelle, um die Dimensionsversion durch ein Fakttransaktionsdatum näher zu qualifizieren, kann aufwendig sein. Beispiel:

```
SELECT...
  FROM PRODUCT P, SALES F
  WHERE
    P.PROD_KEY = F.PROD_KEY AND
    F.SALE_DATE BETWEEN P.START_DATE AND
    P.END_DATE
```

Stellen Sie in diesem Fall sicher, dass ein Index für die Spalten (F.PROD_KEY, F.SALE_DATE) vorhanden ist.

Ziehen Sie in Betracht, eine Statistiksicht zu erstellen, um dem Optimierungsprogramm zu helfen, eine bessere Selektivitätsschätzung für dieses Szenario zu berechnen. Beispiel:

```
CREATE VIEW V_PROD_FACT AS
  SELECT P.*
    FROM PRODUCT P, SALES F
    WHERE
      P.PROD_KEY = F.PROD_KEY AND
      F.SALE_DATE BETWEEN P.START_DATE AND
      P.END_DATE

ALTER VIEW V_PROD_FACT ENABLE QUERY OPTIMIZATION

RUNSTATS ON TABLE DB2USER.V_PROD_FACT WITH DISTRIBUTION
```

Spezialisierte Sternschemajoins, wie zum Beispiel Sternjoins mit logischem Verknüpfen von Indizes über AND (Index ANDing) und Hub-Joins werden nicht in Betracht gezogen, wenn im Abfrageblock Joinvergleichselemente mit Ungleichheitsoperatoren enthalten sind. Nähere Informationen hierzu finden Sie in dem Abschnitt „Überprüfen von Abfragen auf erforderliche Kriterien für Sternschemajoins".

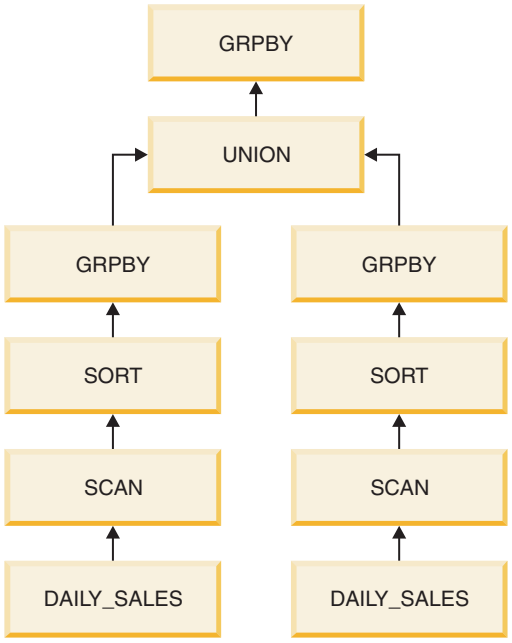
Vermeiden mehrfacher Spaltenberechnungen mit dem Schlüsselwort DISTINCT:

Vermeiden Sie die Verwendung von Abfragen, die mehrere DISTINCT-Spaltenberechnungen in demselben Subselect ausführen, da deren Ausführung aufwendig ist.

Betrachten Sie das folgende Beispiel:

```
SELECT SUM(DISTINCT REBATE), AVG(DISTINCT DISCOUNT)
  FROM DAILY_SALES
  GROUP BY PROD_KEY
```

Zur Ermittlung der Menge von unterschiedlichen (distinkten) REBATE- und DISCOUNT-Werten muss der Eingabedatenstrom aus der PROD_KEY-Tabelle möglicherweise zweimal sortiert werden. Der Abfragezugriffsplan für diese Abfrage könnte folgendermaßen aussehen:



Das Optimierungsprogramm schreibt die ursprüngliche Abfrage in separate Spaltenberechnungen um, indem es das Schlüsselwort DISTINCT für beide angibt, und kombiniert die mehrfachen Spaltenberechnungen anschließend mit einem Schlüsselwort UNION. Die intern umgeschriebene Fassung der Anweisung sieht wie folgt aus:

```
SELECT Q8.MAXC0, (Q8.MAXC1 / Q8.MAXC2)
FROM
  (SELECT MAX(Q7.C0) AS MAXC0, MAX(Q7.C1) AS MAXC1, MAX(Q7.C2) AS MAXC2
    FROM
      (SELECT SUM(DISTINCT Q2.REBATE) AS C0, CAST(NULL AS INTEGER) AS C1,
        0 AS C2, Q2.PROD_KEY
      FROM
        (SELECT Q1.PROD_KEY, Q1.REBATE
          FROM DB2USER.DAILY_SALES AS Q1) AS Q2
      GROUP BY Q2.PROD_KEY
      UNION ALL
      SELECT CAST(NULL AS INTEGER) AS C0, SUM(DISTINCT Q5.DISCOUNT) AS C1,
          COUNT(DISTINCT Q5.DISCOUNT) AS C2, Q5.PROD_KEY
        FROM
          (SELECT Q4.PROD_KEY, Q4.DISCOUNT
            FROM DB2USER.DAILY_SALES AS Q4) AS Q5
        GROUP BY Q5.PROD_KEY) AS Q7
      GROUP BY Q7.PROD_KEY) AS Q8
```

Wenn Sie mehrfache DISTINCT-Spaltenberechnungen nicht vermeiden können, ziehen Sie in Betracht, die Registrierdatenbankvariable **DB2_EXTENDED_OPTIMIZATION** mit der Option ENHANCED_MULTIPLE_DISTINCT zu verwenden. Diese Option bewirkt, dass der Eingabedatenstrom an die mehrfachen DISTINCT-Spaltenberechnungen einmal gelesen und anschließend für jeden Zweig der UNION-Operation wiederverwendet wird. Diese Option könnte die Leistung solcher Typen von Abfragen verbessern, wenn das Verhältnis von Prozessoren zur Anzahl der Datenbankpartitionen niedrig ist (z. B. kleiner oder gleich 1). Diese Einstellung muss in Umgebungen mit partitionierten Datenbanken ohne symmetrische Mehrprozessoren (SMPs) verwendet werden. Diese Optimierungserweiterung verbessert die Abfrageleistung möglicherweise nicht in allen Umgebungen. Einzelne Verbesserungen in der Abfrageleistung sollten durch Tests ermittelt werden.

Vermeiden unnötiger Outer Joins:

Die Semantik bestimmter Abfragen erfordert Outer Joins (entweder linke, rechte oder vollständige Outer Joins). Wenn die Abfragesemantik jedoch keinen Outer Join erfordert und die Abfrage zur Behandlung inkonsistenter Daten verwendet wird, ist es am besten, die Probleme mit inkonsistenten Daten an der eigentlichen Ursache zu behandeln.

In einem Datamart mit einem Sternschema könnte die Fakttabelle zum Beispiel Zeilen für Transaktionen enthalten, jedoch aufgrund von Datenkonsistenzproblemen keine entsprechenden übergeordneten Dimensionszeilen für einige Dimensionen. Dies könnte geschehen, weil der Extraktions-, Transformations-und Ladeprozess (ETL-Prozess) einige Geschäftsschlüssel aus irgendeinem Grund nicht abgleichen konnte. In diesem Szenario werden die Zeilen der Fakttabelle als linker Outer Join mit den Dimensionen verknüpft, um sicherzustellen, dass sie zurückgegeben werden, selbst wenn sie keine übergeordnete Zeile haben. Beispiel:

```
SELECT...
  FROM DAILY_SALES F
    LEFT OUTER JOIN CUSTOMER C ON F.CUST_KEY = C.CUST_KEY
    LEFT OUTER JOIN STORE S ON F.STORE_KEY = S.STORE_KEY
  WHERE
    C.CUST_NAME = 'SMITH'
```

Der linke Outer Join kann eine Reihe von Optimierungen verhindern, wie zum Beispiel die Verwendung von spezialisierten Zugriffsmethoden mit Sternschemajoins. Jedoch kann der linke Outer Join in einigen Fällen durch das Abfrageoptimierungsprogramm automatisch in einen Inner Join umgeschrieben werden. In diesem Beispiel kann der linke Outer Join zwischen CUSTOMER und DAILY_SALES in einen Inner Join umgewandelt werden, weil das Vergleichselement `C.CUST_NAME = 'SMITH'` alle Zeilen mit Nullwerten in dieser Spalte entfernt, sodass ein linker Outer Join semantisch unnötig wird. Auf diese Weise ist es möglich, dass der Verlust einiger Optimierungen aufgrund von Outer Joins nicht alle Abfragen negativ beeinflusst. Es ist jedoch wichtig, sich dieser Einschränkungen bewusst zu sein und Outer Joins zu vermeiden, sofern sie nicht absolut notwendig sind.

Verwenden der Klausel OPTIMIZE FOR N ROWS mit der Klausel FETCH FIRST N ROWS ONLY:

Die Klausel OPTIMIZE FOR *n* ROWS teilt dem Optimierungsprogramm mit, dass die Anwendung nur *n* Zeilen abzurufen beabsichtigt, die Abfrage jedoch die vollständige Ergebnismenge zurückgeben soll. Die Klausel FETCH FIRST *n* ROWS ONLY gibt an, dass die Abfrage nur *n* Zeilen zurückgeben soll.

Der DB2-Datenserver nimmt nicht automatisch die Klausel OPTIMIZE FOR *n* ROWS an, wenn die Klausel FETCH FIRST *n* ROWS ONLY für den übergeordneten Subselect (Outer Subselect) angegeben wird. Versuchen Sie die Angabe der Klausel OPTIMIZE FOR *n* ROWS zusammen mit der Klausel FETCH FIRST *n* ROWS ONLY, um Abfragezugriffspläne zu fördern, die Zeilen direkt aus den Tabellen, auf die verwiesen wird, zurückgeben, ohne zunächst eine Pufferoperation wie das Einfügen in eine temporäre Tabelle, das Sortieren oder das Einfügen in die Hash-Tabelle eines Hash-Joins auszuführen.

Anwendungen, die die Klausel OPTIMIZE FOR *n* ROWS angeben, um Abfragezugriffspläne zu fördern, die Pufferoperationen vermeiden, jedoch trotzdem die gesamte Ergebnismenge abrufen, zeigen möglicherweise eine schlechte Leistung. Dies

liegt daran, dass der Abfrageplan, der die ersten *n* Zeilen am schnellsten zurückgibt, vielleicht nicht der beste Abfragezugriffsplan ist, wenn die gesamte Ergebnismenge abgerufen wird.

Überprüfen von Abfragen auf erforderliche Kriterien für Sternschemajoins:

Das Optimierungsprogramm zieht zwei spezialisierte Joinmethoden für Sternschemata in Betracht, die als Sternjoin oder Hub-Join bezeichnet werden und die Leistung erheblich verbessern können.

Allerdings muss die Abfrage den folgenden Kriterien entsprechen.

- Für jeden Abfrageblock gilt:
  - Es müssen mindestens drei verschiedene Tabellen durch Join verknüpft werden.
  - Alle Joinvergleichselemente müssen Gleichheitsvergleichselemente sein.
  - Es dürfen keine Unterabfragen vorhanden sein.
  - Es dürfen keine Korrelationen oder Abhängigkeiten zwischen Tabellen oder außerhalb des Abfrageblocks vorhanden sein.
  - Für das logische Verknüpfen von Indizes über AND (Index ANDing) müssen nicht deterministische Funktionen vorhanden sein, weil Vergleichselemente für die Fakttabelle durch Indizes angewendet werden müssen, um Semi-Joins zu ermöglichen.
  - Für eine Fakttabelle gilt:
    - Sie ist die größte Tabelle im Abfrageblock.
    - Sie enthält mindestens 10.000 Zeilen.
    - Sie wird als nur eine einzige Tabelle betrachtet.
    - Sie muss mit mindestens zwei Dimensionen oder Gruppen, die als Snowflakes (Schneeflocken) bezeichnet werden, durch Join verknüpft werden.
  - Für eine Dimensionstabelle gilt:
    - Sie ist nicht die Fakttabelle.
    - Sie kann einzeln mit der Fakttabelle oder in Snowflake-Schemata durch Join verknüpft werden.
  - Für eine Dimensionstabelle oder eine Snowflake gilt:
    - Sie muss die Fakttabelle filtern (die Filterung basiert auf den Schätzungen des Optimierungsprogramms).
    - Sie muss ein Joinvergleichselement zum Vergleich mit der Fakttabelle haben, das eine führende Spalte in einem Index der Fakttabelle verwendet. Dieses Kriterium muss erfüllt sein, damit entweder ein Sternjoin oder ein Hub-Join in Betracht gezogen werden kann, obwohl ein Hub-Join nur einen Fakttabellenindex verwenden muss.

Ein Abfrageblock, der einen linken oder rechten Outer Join darstellt, kann auf zwei Tabellen verweisen, sodass ein Sternschemajoin in diesem Fall nicht infrage kommt.

Eine explizite Deklaration der referenziellen Integrität ist nicht erforderlich, damit das Optimierungsprogramm einen Sternschemajoin erkennt.

Vermeiden redundanter Vergleichselemente:

Vermeiden Sie redundante Vergleichselemente, insbesondere wenn diese über verschiedene Tabellen hinweg auftreten. In einigen Fällen kann das Optimierungspro-

gramm nicht erkennen, dass die Vergleichselemente redundant sind. Dies kann zu Unterschätzungen von Kardinalitäten führen.

Zum Beispiel wird in SAP-BI-Anwendungen (BI = Business-Intelligence) das Snowflake-Schema mit Fakttabelle und Dimensionstabellen als abfrageoptimierte Datenstruktur genutzt. In einigen Fällen ist eine redundante Zeitmerkmalsspalte ("SID_0CALMONTH" für Monat oder "SID_0FISCPER" für Jahr) für die Fakttabelle und die Dimensionstabellen definiert.

Der SAP-BI-OLAP-Prozessor (OPAP, Online Analytical Processing) generiert redundante Vergleichselemente für die Zeitmerkmalsspalten der Dimensionstabellen und der Fakttabelle.

Diese redundanten Vergleichselemente können zu längeren Abfrageausführungszeiten führen.

Der folgende Abschnitt zeigt ein Beispiel mit zwei redundanten Vergleichselementen, die in der WHERE-Bedingung einer SAP-BI-Abfrage definiert sind. Es sind identische Vergleichselemente für die Zeitdimensionstabelle (DT) und die Fakttabelle (F) definiert:

```
     AND (     "DT"."SID_0CALMONTH" = 199605
           AND "F". "SID_0CALMONTH" = 199605
            OR "DT"."SID_0CALMONTH" = 199705
           AND "F". "SID_0CALMONTH" = 199705 )
 AND NOT (     "DT"."SID_0CALMONTH" = 199803
           AND "F". "SID_0CALMONTH" = 199803 )
```

Das DB2-Optimierungsprogramm erkennt nicht, dass die Vergleichselemente identisch sind und behandelt sie wie unabhängige Vergleichselemente. Dies führt zu Unterschätzungen von Kardinalitäten, nicht optimalen Abfragezugriffsplänen und längeren Abfrageausführungszeiten.

Aus diesem Grund werden die redundanten Vergleichselemente von der für die DB2-Datenbankplattform spezifischen Softwareschicht entfernt.

Die oben gezeigten Vergleichselemente werden in die nachfolgend gezeigten übertragen. Nur die Vergleichselemente für die Fakttabellenspalte "SID_0CALMONTH" bleiben erhalten:

```
    AND (     "F". "SID_0CALMONTH" = 199605
          OR "F". "SID_0CALMONTH" = 199705 )
AND NOT (     "F". "SID_0CALMONTH" = 199803 )
```

Befolgen Sie die Anweisungen in den SAP-Hinweisen (Notes) 957070 und 1144883, um die redundanten Vergleichselemente zu entfernen.

Verbessern der Abfrageoptimierung durch Integritätsbedingungen

Ziehen Sie in Betracht, eindeutige Integritätsbedingungen, Prüfungen auf Integritätsbedingungen und referenzielle Integritätsbedingungen zu definieren. Solche Integritätsbedingungen stellen semantische Informationen bereit, die es dem DB2-Optimierungsprogramm ermöglichen, Abfragen umzuschreiben, um Joins zu eliminieren, Spaltenberechnungen (Aggregation) durch Joins zu verschieben, die Operation für die Klausel FETCH FIRST *n* ROWS durch Joins zu verschieben, unnötige DISTINCT-Operationen zu entfernen und eine Reihe weiterer Optimierungsmaßnahmen durchzuführen.

Es können auch informative Integrationsbedingungen anstelle von Prüfungen auf Integritätsbedingungen und referenziellen Integritätsbedingungen verwendet werden, wenn die Anwendung selbst die Beziehungen garantieren kann. In diesem Fall sind dieselben Optimierungsmaßnahmen möglich. Integritätsbedingungen, die der Datenbankmanager beim Einfügen, Aktualisieren oder Löschen von Zeilen umsetzt, können zu einem hohen Systemaufwand führen, insbesondere wenn eine große Anzahl von Zeilen aktualisiert wird, für die referenzielle Integritätsbedingungen gelten. Wenn eine Anwendung Informationen bereits vor dem Aktualisieren einer Zeile überprüft hat, kann es effizienter sein, informative Integrationsbedingungen anstelle von regulären Integritätsbedingungen zu verwenden.

Betrachten Sie zum Beispiel die beiden Tabellen DAILY_SALES und CUSTOMER. Jede Zeile in der Tabelle CUSTOMER hat einen eindeutigen Kundenschlüssel (CUST_KEY). Die Tabelle DAILY_SALES enthält eine Spalte CUST_KEY und jede Zeile verweist auf einen Kundenschlüssel in der Tabelle CUSTOMER. Es könnte eine referenzielle Integritätsbedingung erstellt werden, um diese 1:N-Beziehung zwischen der CUSTOMER und der Tabelle DAILY_SALES darzustellen. Wenn die Anwendung jedoch für die Einhaltung der Beziehung sorgen würde, könnte die Integritätsbedingung als informative Integritätsbedingung definiert werden. Die folgende Abfrage könnte in diesem Fall die Ausführung des Joins zwischen den Tabellen CUSTOMER und DAILY_SALES vermeiden, weil keine Zeilen aus der Tabelle CUSTOMER abgerufen werden und jede Zeile in der Tabelle DAILY_SALES eine Entsprechung in der Tabelle CUSTOMER findet. Das Abfrageoptimierungsprogramm entfernt den Join automatisch.

```
SELECT AMT_SOLD, SALE PRICE, PROD_DESC
  FROM DAILY_SALES, PRODUCT, CUSTOMER
  WHERE
    DAILY_SALES.PROD_KEY = PRODUCT.PRODKEY AND
    DAILY_SALES.CUST_KEY = CUSTOMER.CUST_KEY
```

Die Anwendung muss selbst für die Umsetzung der informativen Integrationsbedingungen sorgen. Ansonsten könnten Abfragen falsche Ergebnisse liefern. Wenn in diesem Beispiel irgendwelche Zeilen in der Tabelle DAILY_SALES keinen entsprechenden Kundenschlüssel in der Tabelle CUSTOMER haben, könnte die Abfrage diese Zeilen fälschlicherweise zurückgeben.

Verwenden der Bindeoption REOPT mit Eingabevariablen in komplexen Abfragen

Eingabevariablen spielen eine wesentliche Rolle bei der Erzielung guter Anweisungsvorbereitungszeiten in einer OLTP-Umgebung (OLTP, Onlinetransaktionsverarbeitung), in der Anweisungen in der Regel einfacher und die Auswahl von Abfragezugriffsplänen unaufwendiger sind.

Mehrere Ausführungen derselben Abfrage mit verschiedenen Werten für Eingabevariablen können den kompilierten Zugriffsabschnitt im Cache für dynamische Anweisungen wiederverwenden, sodass aufwendige Kompilierungen von SQL-Anweisungen bei jeder Änderung von Eingabewerten vermieden werden.

Eingabevariablen können jedoch für komplexe Abfrageauslastungen problematisch sein, bei denen die Auswahl von Abfragezugriffsplänen komplizierter ist und das Optimierungsprogramm mehr Informationen benötigt, um gute Entscheidungen zu treffen. Darüber hinaus ist die Anweisungskompilierungszeit in der Regel nur eine kleine Komponente in der Gesamtausführungszeit und Business-Intelligence-Abfragen (BI-Abfragen), die zumeist nicht wiederholt werden, ziehen keinen Vorteil aus dem Cache für dynamische Anweisungen.

Wenn Eingabevariablen in einer komplexen Abfrageauslastung benötigt werden, ziehen Sie eine Verwendung der Bindeoption REOPT(ALWAYS) in Betracht. Die Bindeoption REOPT verzögert die Anweisungskompilierung vom PREPARE-Zeitpunkt zum OPEN- oder EXECUTE-Zeitpunkt, zu dem die Werte der Eingabevariablen bekannt sind. Die Werte werden an den SQL-Compiler übergeben, sodass das Optimierungsprogramm die Werte zur Berechnung einer präziseren Selektivitätsschätzung verwenden kann. Die Bindeoption REOPT(ALWAYS) gibt an, dass die Anweisung für jede Ausführung erneut kompiliert werden soll. Die Bindeoption REOPT(ALWAYS) kann außerdem für komplexe Abfragen verwendet werden, die auf Sonderregister zum Beispiel in der Form `WHERE TRANS_DATE = CURRENT DATE - 30 DAYS` verweisen. Wenn Eingabevariablen zu einer ungünstigen Zugriffsplanauswahl für OLTP-Auslastungen führen und die Bindeoption REOPT(ALWAYS) einen übermäßigen Systemaufwand aufgrund von Anweisungskompilierungen verursacht, ziehen Sie die Verwendung der Bindeoption REOPT(ONCE) für ausgewählte Abfragen in Betracht. Die Bindeoption REOPT(ONCE) verzögert die Anweisungskompilierung, bis der erste Eingabevariablenwert gebunden wird. Die SQL-Anweisung wird mit diesem ersten Eingabevariablenwert kompiliert und optimiert. Nachfolgende Ausführungen der Anweisung mit anderen Werten verwenden den Zugriffsabschnitt wieder, der auf der Basis des ersten Eingabewerts kompiliert wurde. Dies kann eine gute Lösung sein, wenn der erste Eingabevariablenwert für nachfolgende Werte repräsentativ ist und dadurch ein besserer Abfragezugriffsplan ausgewählt wird als einer, der auf Standardwerten basiert, wenn die Werte der Eingabevariablen unbekannt sind.

Die Option REOPT kann auf verschiedene Weise angegeben werden:

- Für eingebettetes SQL in C/C++-Anwendungen verwenden Sie die Bindeoption REOPT. Diese Bindeoption betrifft das Reoptimierungsverhalten für statisches und dynamisches SQL.
- Für CLP-Pakete binden Sie das CLP-Paket mit der Bindeoption REOPT erneut. Beispiel: Zum erneuten Binden des CLP-Pakets, das für die Isolationsstufe CS verwendet wird, mit der Option REOPT ALWAYS geben Sie den folgenden Befehl an:

  ```
  rebind nullid.SQLC2G13 reopt always
  ```

- Für CLI-Anwendungen oder JDBC-Anwendungen, die DB2 JDBC Type 2 Driver for Linux, UNIX and Windows verwenden, legen Sie den Wert für REOPT auf eine der folgenden Arten fest:
  - Verwenden Sie die REOPT-Schlüsselworteinstellungen in der Konfigurationsdatei 'db2cli.ini'. Die möglichen Werte und die entsprechenden Optionen sind folgende:
    - 2 = SQL_REOPT_NONE
    - 3 = SQL_REOPT_ONCE
    - 4 = SQL_REOPT_ALWAYS
  - Verwenden Sie das Verbindungs- oder Anweisungsattribut SQL_ATTR_REOPT.
  - Verwenden Sie das Verbindungs- oder Anweisungsattribut SQL_ATTR_CURRENT_PACKAGE_SET, um die Paketgruppen NULLID, NULLIDR1 oder NULLIDRA anzugeben. NULLIDR1 und NULLIDRA sind reservierte Paketgruppennamen. REOPT ONCE bzw. REOPT ALWAYS sind bei der Verwendung stets impliziert. Diese Paketgruppen müssen mit den folgenden Befehlen explizit erstellt werden:

    ```
    db2 bind db2clipk.bnd collection NULLIDR1
    db2 bind db2clipk.bnd collection NULLIDRA
    ```

- Für JDBC-Anwendungen, die IBM Data Server Driver for JDBC and SQLJ verwenden, geben Sie den Wert für '-reopt' an, wenn Sie das Dienstprogramm 'DB2Binder' ausführen.
- Für SQL PL-Prozeduren verwenden Sie eine der folgenden Methoden:
  - Verwenden Sie die gespeicherte Prozedur SET_ROUTINE_OPTS, um die Bindeoption festzulegen, die zur Erstellung von SQL PL-Prozeduren innerhalb der aktuellen Sitzung zu verwenden sind. Geben Sie zum Beispiel den folgenden Aufruf an:

    ```
    sysproc.set_routine_opts('reopt always')
    ```

  - Verwenden Sie die Registrierdatenbankvariable **DB2_SQLROUTINE_PREPOPTS**, um die Optionen für SQL PL-Prozeduren auf der Instanzebene festzulegen. Werte, die mit der gespeicherten Prozedur SET_ROUTINE_OPTS festgelegt werden, überschreiben die Werte, die mit **DB2_SQLROUTINE_PREPOPTS** festgelegt werden.

Sie können auch Optimierungsprofile verwenden, um die Option REOPT für statische und dynamische Anweisungen wie im folgenden Beispiel festzulegen:

```
<STMTPROFILE ID="REOPT example ">
     <STMTKEY>
       <![CDATA[select acct_no from customer where name = ? ]]>
     </STMTKEY>
     <OPTGUIDELINES>
        <REOPT VALUE='ALWAYS'/>
    </OPTGUIDELINES>
</STMTPROFILE>
```

Verwenden von Parametermarken zur Reduzierung der Kompilierzeit für dynamische Abfragen

Der DB2-Datenserver kann eine erneute Kompilierung einer dynamischen SQL-Anweisung, die zuvor ausgeführt wurde, vermeiden, indem er den Zugriffsabschnitt und den Anweisungstext im Cache für dynamische Anweisungen speichert.

Eine nachfolgende Vorbereitungsanforderung (PREPARE) für diese Anweisung versucht, den Zugriffsabschnitt im Cache für dynamische Anweisungen zu finden und eine Kompilierung zu vermeiden. Allerdings werden Anweisungen, die sich nur in den Literalen, die in Vergleichselementen verwendet werden, unterscheiden, nicht als übereinstimmend erkannt. Zum Beispiel werden die beiden folgenden Anweisungen im Cache für dynamische Anweisungen als unterschiedlich betrachtet:

```
SELECT AGE FROM EMPLOYEE WHERE EMP_ID = 26790
SELECT AGE FROM EMPLOYEE WHERE EMP_ID = 77543
```

Selbst relativ einfache SQL-Anweisungen können aufgrund der Anweisungskompilierung eine übermäßige CPU-Belastung zur Folge haben, wenn sie sehr häufig ausgeführt werden. Wenn Ihr System diesen Typ von Leistungsproblem zeigt, ziehen Sie in Betracht, die Anwendung zu ändern und Parametermarken zu verwenden, um Vergleichselementwerte an den DB2-Compiler zu übergeben, anstatt sie in die SQL-Anweisung mit einzuschließen. Allerdings ist der Zugriffsplan für komplexe Abfragen, die Parametermarken in Vergleichselementen verwenden, möglicherweise nicht optimal. Weitere Informationen finden Sie in dem Abschnitt „Verwenden der Bindeoption REOPT mit Eingabevariablen in komplexen Abfragen”.

Definieren der Registrierdatenbankvariablen DB2_REDUCED_OPTIMIZATION

Wenn die Festlegung der Optimierungsklasse die Kompilierzeit für Ihre Anwendung nicht ausreichend reduziert, versuchen Sie es, indem Sie die Registrierdatenbankvariable **DB2_REDUCED_OPTIMIZATION** definieren.

Diese Registrierdatenbankvariable bietet mehr Kontrolle über den Suchbereich des Optimierungsprogramms als die Festlegung der Optimierungsklasse. 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.

Die Optimierungszeit und die Ressourcenbelastung wird zwar reduziert, jedoch erhöht sich das Risiko, dass ein nicht optimaler Abfragezugriffsplan generiert wird.

Versuchen Sie es zunächst, indem Sie die Registrierdatenbankvariable auf den Wert YES setzen. 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 nicht zur Generierung eines besseren Abfragezugriffsplans 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 das Risiko steigt, dass ein nicht optimaler Abfragezugriffsplan 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 die Einstellung YES keine ausreichende Reduzierung der Kompilierzeit bewirkt, versuchen Sie es, indem Sie die Registrierdatenbankvariable auf einen ganzzahligen Wert (Integer) setzen. 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 Gesamtzahl von Joins in einem beliebigen Abfrageblock die Einstellung überschreitet, wechselt das Optimierungsprogramm zur schnellen Joinaufzählung (Greedy Join Enumeration), anstatt zusätzliche Optimierungstechniken zu inaktivieren. Infolgedessen wird die Abfrage mit einem Grad optimiert, der dem der Optimierungsklasse 2 ähnlich ist.

Verbessern der Einfügeleistung

Bevor Daten in eine Tabelle eingefügt werden, untersucht ein Einfügesuchalgorithmus die Steuersätze für freie Speicherbereiche (FSCR, Free Space Control Records), um eine Seite mit genügend Speicherplatz ausfindig zu machen.

Es ist jedoch möglich, dass, selbst wenn ein FSCR-Satz angibt, dass auf einer Seite genügend freier Speicherbereich vorhanden ist, dieser Speicherplatz nicht nutzbar ist, wenn er durch eine nicht festgeschriebene Löschoperation von einer anderen Transaktion reserviert ist.

Die Registrierdatenbankvariable **DB2MAXFSCRSEARCH** gibt die Anzahl von FSCR-Sätzen an, die beim Einfügen eines Datensatzes in eine Tabelle zu durchsuchen sind. Standardmäßig werden fünf Steuersätze für freien Speicherbereich durchsucht. Eine Änderung dieses Werts bietet Ihnen die Möglichkeit, eine Gewichtung zwischen der Einfügegeschwindigkeit und der Wiederverwendung von Speicherplatz anzugeben. Mit hohen Werten optimieren Sie die Speicherwiederverwendung. Mit niedrigen Werten optimieren Sie die Einfügegeschwindigkeit. Der Wert -1 veranlasst den Datenbankmanager, alle FSCR-Sätze zu durchsuchen. Wenn beim Durchsuchen der FSCR-Sätze kein ausreichender Speicherbereich gefunden wird, werden die Daten an das Ende der Tabelle angehängt.

Die Option APPEND ON in der Anweisung ALTER TABLE gibt an, dass Tabellendaten angehängt werden und dass keine Informationen zu freiem Speicherbereich auf Seiten verwaltet werden. Solche Tabellen dürfen keinen Clusterindex haben. Diese Option kann die Leistung für Tabellen verbessern, die nur wachsen.

Wenn ein Clusterindex für die Tabelle definiert ist, versucht der Datenbankmanager, Datensätze auf derselben Seite wie andere Datensätze mit ähnlichen Indexschlüsselwerten einzufügen. Wenn auf der betreffenden Seite kein Platz ist, werden benachbarte Seite in Betracht gezogen. Wenn diese Seiten nicht geeignet sind, werden die FSCR-Sätze wie oben beschrieben durchsucht. In diesem Fall wird jedoch nach der Methode des „am wenigsten passenden" Speicherplatzes und nicht nach der des „ersten passenden" Speicherplatzes verfahren. Die Methode des am wenigsten passenden Speicherplatzes wählt meist Seiten mit mehr freiem Speicherbereich aus. Diese Methode richtet einen neuen Clustering-Bereich für Zeilen mit ähnlichen Schlüsselwerten ein.

Wenn Sie einen Clusterindex für eine Tabelle definiert haben, verwenden Sie die Klausel PCTFREE in der Anweisung ALTER TABLE, bevor Sie die Tabelle mit Daten laden oder reorganisieren. Die Klausel PCTFREE gibt den Prozentsatz an freiem Speicherplatz an, der auf einer Datenseite nach einer LOAD- oder REORG-Operation verbleiben soll. Dadurch steigt die Wahrscheinlichkeit, dass bei der Clusterindexoperation auf der richtigen Seite freier Speicherplatz gefunden wird.

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 Sie die folgenden Richtlinien für die Erstellung effizienter SELECT-Anweisungen:

- Geben Sie nur Spalten an, die Sie benötigen. Die Angabe aller Spalten durch einen Stern (*) verursacht unnötigen Verarbeitungsaufwand.
- Verwenden Sie Vergleichselemente, die die Ergebnismenge nur auf die Zeilen einschränken, die Sie benötigen.
- Wenn Sie wesentlich weniger als die Gesamtzahl der Zeilen, die möglicherweise zurückgegeben wird, benötigen, geben Sie die Klausel OPTIMIZE FOR an. Diese Klausel wirkt sich sowohl auf die Auswahl des Zugriffsplans als auch auf die Anzahl der im Kommunikationspuffer geblockten Zeilen aus.
- Zur Verwendung der Zeilenblockung und zur Verbesserung der Leistung geben Sie die Klausel FOR READ ONLY oder FOR FETCH ONLY an. Dadurch verbessert sich zudem der gleichzeitige Zugriff, weil für die abgerufenen Spalten zu keiner Zeit exklusive Sperren aktiviert werden. Zusätzliches Umschreiben der Abfrage kann ebenfalls stattfinden. Durch die Angabe dieser Klauseln sowie der Bindeoption BLOCKING ALL lässt sich die Leistung von Abfragen, die auf Kurznamen in einem System föderierter Datenbanken ausgeführt werden, in ähnlicher Weise verbessern.
- Geben Sie für Cursor, die mit positionierten Aktualisierungen verwendet werden, die Klausel FOR UPDATE OF an, um dem Datenbankmanager zu ermöglichen, gleich zu Anfang angemessenere Sperrstufen auszuwählen und potenzielle Deadlocks zu vermeiden. Beachten Sie, dass FOR UPDATE-Cursor die Zeilenblockung nicht nutzen können.
- Für Cursor, die bei Aktualisierungen mit Suche verwendet werden, geben Sie die Klauseln FOR READ ONLY und USE AND KEEP UPDATE LOCKS an, um

Deadlocks zu vermeiden und trotzdem eine Zeilenblockung ermöglichen, indem Sie U-Sperren für die betroffenen Zeilen erzwingen.

- Vermeiden Sie nach Möglichkeit Umsetzungen numerischer Datentypen. Versuchen Sie beim Vergleichen von Werten Elemente zu verwenden, die denselben Datentyp besitzen. 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), Dezimalzahlen (DECIMAL) oder dezimaler Gleitkommazahlen (DECFLOAT)
  – DECFLOAT anstatt Dezimalzahlen
  – Datum/Uhrzeit (Datetime) anstatt Zeichen
  – Numerische Typen anstatt Zeichen
- Lassen Sie zur Verringerung der Wahrscheinlichkeit von Sortieroperationen Klauseln wie DISTINCT oder ORDER BY weg, wenn solche Operationen nicht erforderlich sind.
- Wählen Sie eine einzelne Zeile aus, wenn Sie das Vorhandensein von Zeilen prüfen wollen. Öffnen Sie entweder einen Cursor und rufen Sie eine Zeile ab (FETCH) oder führen Sie eine SELECT INTO-Operation für eine Zeile aus. 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 <tabellenname>
  ```

  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 Vergleichselementen verwendet wird. Bei großen IN-Listen, die mit Hostvariablen verwendet werden, können Verarbeitungsschleifen für Untergruppen der Hostvariablen die Leistung verbessern.

Die folgenden Vorschläge gelten speziell 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 einem 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 OR-Klauseln oder Ausdrücke mit Joinvergleichselementen.
- Für eine Umgebung mit partitionierten Datenbanken wird empfohlen, dass Tabellen, die durch einen Join verknüpft werden sollen, über die Joinspalte partitioniert werden.

Richtlinien zur Einschränkung von SELECT-Anweisungen

Das Optimierungsprogramm geht von der Annahme aus, dass eine Anwendung sämtliche Zeilen abrufen muss, die durch die SELECT-Anweisung angegeben werden. Diese Annahme ist in OLTP-Umgebungen (Onlinetransaktionsverarbeitung) und in Stapelumgebungen in der Regel zutreffend.

Bei reinen Such- bzw. Anzeigeanwendungen hingegen definieren Abfragen häufig potenziell sehr umfangreiche Ergebnismengen, rufen jedoch nur die ersten wenigen Zeilen, in der Regel die Anzahl von Zeilen, die für ein bestimmtes Anzeigeformat erforderlich sind, ab.

Zur Verbesserung der Leistung solcher Anwendungen können Sie die SELECT-Anweisung auf folgende Weisen modifizieren:

- Verwenden Sie die Klausel FOR UPDATE, um die Spalten anzugeben, die von einer nachfolgenden positionierten Anweisung UPDATE aktualisiert werden könnten.
- Verwenden Sie die Klausel FOR READ oder FETCH ONLY, um die angegebenen Spalten im Lesezugriff zurückzugeben.
- Verwenden Sie die Klausel OPTIMIZE FOR *n* ROWS, um dem Abruf der ersten *n* Zeilen aus der Gesamtergebnismenge Priorität geben.
- Verwenden Sie die Klausel FETCH FIRST *n* ROWS ONLY, um nur eine angegebene Anzahl von Zeilen abzurufen.
- Verwenden Sie die Anweisung DECLARE CURSOR WITH HOLD, um jeweils nur eine Zeile pro Mal abzurufen.

In den folgenden Abschnitten werden die Leistungsvorteile der einzelnen Methoden erläutert.

Klausel FOR UPDATE

Die Klausel FOR UPDATE grenzt die Ergebnismenge ein, indem sie nur die Spalten berücksichtigt, die durch eine nachfolgende positionierte UPDATE-Anweisung aktualisiert werden können. Bei der Angabe der Klausel FOR UPDATE ohne Spaltennamen werden alle Spalten, die aktualisiert werden können, in der Tabelle oder Sicht mit eingeschlossen. Bei Angabe von Spaltennamen muss jeder Name unqualifiziert angegeben werden und eine Spalte der Tabelle oder Sicht bezeichnen.

In folgenden Fällen kann die Klausel FOR UPDATE nicht verwendet werden:

- Der Cursor, der der der SELECT-Anweisung zugeordnet ist, kann nicht gelöscht werden.
- Mindestens eine der durch SELECT ausgewählten Spalten ist eine Spalte, die in einer Katalogtabelle nicht aktualisiert werden kann und in der Klausel FOR UPDATE nicht ausgeschlossen wurde.

In CLI-Anwendungen können Sie das CLI-Verbindungsattribut SQL_ATTR_ACCESS_MODE zum selben Zweck verwenden.

Klausel FOR READ oder FETCH ONLY

Die Klausel FOR READ ONLY oder FOR FETCH ONLY stellt sicher, dass Ergebnisse nur im Lesezugriff zurückgegeben werden. Bei Tabellen, für die Aktualisierungen und Löschungen zulässig sind, kann die Angabe der Klausel FOR READ ONLY die Leistung von Abrufoperationen (FETCH) verbessern, wenn der Datenbankmanager Blöcke von Daten abrufen kann, statt exklusive Sperren zu aktivie-

ren. Geben Sie die Klausel FOR READ ONLY nicht in Abfragen an, die in positionierten UPDATE- oder DELETE-Anweisungen verwendet werden.

In CLI-Anwendungen können Sie das CLI-Verbindungsattribut SQL_ATTR_ACCESS_MODE zum selben Zweck verwenden.

Klausel OPTIMIZE FOR *n* ROWS

Mit der Klausel OPTIMIZE FOR wird die Absicht deklariert, nur eine Teilmenge des Ergebnisses abzurufen oder dem Abruf nur der ersten wenigen Zeilen Priorität zu geben. Das Optimierungsprogramm kann in diesem Fall Zugriffspläne auswählen, die die Antwortzeiten für den Abruf der ersten wenigen Zeilen minimieren. Darüber hinaus wird die Anzahl der Zeilen, die als ein Block an den Client gesendet werden, durch den Wert *n* begrenzt. Daher beeinflusst die Klausel OPTIMIZE FOR, wie der Server Zeilen, die den Bedingungen entsprechen, aus der Datenbank abruft und wie er diese Zeilen an den Client zurückgibt.

Nehmen Sie zum Beispiel an, dass Sie die Tabelle EMPLOYEE regelmäßig abfragen, um zu ermitteln, welche Mitarbeiter das höchste Gehalt ('salary') haben:

```
select lastname, firstnme, empno, salary
  from employee
  order by salary desc
```

Obwohl Sie zuvor einen absteigenden Index für die Spalte SALARY definiert haben, weist dieser Index wahrscheinlich nur eine geringe Clusterbildung auf, weil die Mitarbeiter nach Personalnummer (EMPNO) geordnet sind. Zur Vermeidung zahlreicher synchroner E/A-Operationen wählt das Optimierungsprogramm wahrscheinlich die Methode des Vorablesezugriffs über Listen aus, für die eine Sortierung der Satz-IDs (RIDs) aller qualifizierten Zeilen erforderlich ist. Diese Sortierung führt zu einer Verzögerung, bevor die ersten Ergebniszeilen an die Anwendung zurückgegeben werden. Zur Vermeidung dieser Verzögerung können Sie der Anweisung die Klausel OPTIMIZE FOR wie folgt hinzufügen:

```
select lastname, firstnme, empno, salary
  from employee
  order by salary desc
  optimize for 20 rows
```

In diesem Fall wählt das Optimierungsprogramm wahrscheinlich eine direkte Verwendung des Index für SALARY aus, weil nur die 20 Mitarbeiter mit den höchsten Gehältern abgerufen werden. Unabhängig davon, wie viele Zeilen geblockt werden könnten, wird nun alle zwanzig Zeilen ein Zeilenblock an den Client zurückgegeben.

Mit der Klausel OPTIMIZE FOR bevorzugt das Optimierungsprogramm Zugriffspläne, die Massenoperationen oder Datenflussunterbrechungen, zum Beispiel durch Sortierungen, vermeiden. Am wahrscheinlichsten wird ein Zugriffspfad durch die Klausel OPTIMIZE FOR 1 ROW beeinflusst. Die Verwendung dieser Klausel kann folgende Auswirkungen haben:

- Joinsequenzen mit zusammengesetzten inneren Tabellen treten mit geringerer Wahrscheinlichkeit auf, da für sie eine temporäre Tabelle erforderlich ist.
- Die Joinmethode könnte sich ändern. Ein Join mit Verschachtelungsschleife (Nested Loop Join) wird mit größter Wahrscheinlichkeit ausgewählt, da diese Methode relativ geringen Aufwand verursacht und in der Regel bei Abruf einiger weniger Zeilen effizienter ist.
- Ein Index, der der Klausel ORDER BY entspricht, wird mit größerer Wahrscheinlichkeit genutzt, weil dadurch die Sortierung für die Klausel ORDER BY entfällt.

- Ein Vorablesezugriff über Listen wird mit geringerer Wahrscheinlichkeit verwendet, da diese Zugriffsmethode eine Sortierung erforderlich macht.
- Ein sequenzieller Vorablesezugriff ist weniger wahrscheinlich, weil nur eine kleine Anzahl von Zeilen erforderlich ist.
- Bei einer Joinabfrage wird wahrscheinlich die Tabelle mit Spalten in der Klausel ORDER BY als äußere Tabelle gewählt, wenn ein Index für die äußere Tabelle die Reihenfolge bietet, die für die Klausel ORDER BY erforderlich ist.

Obwohl die Klausel OPTIMIZE FOR für alle Optimierungsklassen gültig ist, zeigt sie für Klassen ab Optimierungsklasse 3 die besten Ergebnisse, weil die Klassen unter 3 mit der Suchstrategie der schnellen Joinaufzählung (*Greedy Join Enumeration*) arbeiten. Diese Methode führt manchmal zu Zugriffsplänen für Joins mehrerer Tabellen, die für ein schnelles Abrufen der ersten Zeilen nicht geeignet sind.

Wenn eine Paketanwendung mit Call Level Interface (CLI oder ODBC) arbeitet, können Sie das Schlüsselwort **OPTIMIZEFORNROWS** in der Konfigurationsdatei `db2cli.ini` verwenden, damit CLI automatisch eine Klausel OPTIMIZE FOR am Ende jeder Abfrageanweisung anfügt.

Beim Auswählen von Daten aus Kurznamen können die Ergebnisse abhängig von der Unterstützung durch die Datenquelle unterschiedlich ausfallen. Wenn die durch einen Kurznamen angegebene Datenquelle die Klausel OPTIMIZE FOR unterstützt und das DB2-Optimierungsprogramm die gesamte Abfrage an die Datenquelle verschiebt (Pushdown), wird die Klausel im fernen SQL generiert, das an die Datenquelle gesendet wird. Wenn die Datenquelle diese Klausel nicht unterstützt oder das Optimierungsprogramm entscheidet, das der Plan des geringsten Aufwands eine lokale Ausführung ist, wird die Klausel OPTIMIZE FOR lokal angewendet. In diesem Fall bevorzugt das DB2-Optimierungsprogramm Zugriffspläne, die die Antwortzeit für das Abrufen der ersten wenigen Zeilen einer Abfrage minimieren. Allerdings sind die dem Optimierungsprogramm zur Verfügung stehenden Optionen zur Generierung von Plänen etwas begrenzt und die Leistungsgewinne aus der Klausel OPTIMIZE FOR sind möglicherweise vernachlässigbar.

Wenn sowohl die Klausel OPTIMIZE FOR als auch die Klausel FETCH FIRST angegeben werden, wirkt sich der niedrigere der beiden Werte *n* auf die Größe des Kommunikationspuffers aus. Die beiden Werte werden hinsichtlich der Optimierung als unabhängig voneinander betrachtet.

Klausel FETCH FIRST *n* ROWS ONLY

Die Klausel FETCH FIRST *n* ROWS ONLY legt die maximale Anzahl von Zeilen fest, die abgerufen werden können. Die Beschränkung der Ergebnistabelle auf die ersten wenigen Zeilen kann die Leistung erhöhen. Es werden nur *n* Zeilen abgerufen, ungeachtet der Anzahl von Zeilen, die ansonsten in der Ergebnismenge enthalten wären.

Wenn sowohl die Klausel FETCH FIRST als auch die Klausel OPTIMIZE FOR angegeben werden, wirkt sich der niedrigere der beiden Werte *n* auf die Größe des Kommunikationspuffers aus. Die beiden Werte werden hinsichtlich der Optimierung als unabhängig voneinander betrachtet.

Anweisung DECLARE CURSOR WITH HOLD

Wenn Sie einen Cursor mit einer Anweisung DECLARE CURSOR deklarieren, die die Klausel WITH HOLD enthält, bleiben geöffnete Cursor geöffnet, wenn die

Transaktion eine Festschreibung mit COMMIT ausführt, und alle Sperren werden freigegeben, mit Ausnahme derer, die die aktuelle Cursorposition schützen. Wenn die Transaktion mit ROLLBACK rückgängig gemacht wird, werden alle geöffneten Cursor geschlossen und alle Sperren und LOB-Querverweise freigegeben.

In CLI-Anwendungen können Sie das CLI-Verbindungsattribut SQL_ATTR_CURSOR_HOLD zum selben Zweck verwenden. Wenn eine Paketanwendung mit Call Level Interface (CLI oder ODBC) arbeitet, verwenden Sie das Schlüsselwort **CURSORHOLD** in der Konfigurationsdatei `db2cli.ini`, damit CLI automatisch die Klausel WITH HOLD für jeden deklarierten Cursor verwendet.

Angeben von Zeilenblockung zur Verringerung des Systemaufwands

Die Zeilenblockung, die für alle Anweisungen und Datentypen (einschließlich LOB-Datentypen) unterstützt wird, verringert den Systemaufwand des Datenbankmanagers für Cursor durch Abrufen eines Blocks von Zeilen in einer einzigen Operation.

Informationen zu diesem Vorgang

Dieser Block von Zeilen stellt eine Anzahl von Seiten im Speicher dar. Dabei handelt es sich nicht um einen Block einer MDC-Tabelle (MDC, mehrdimensionales Clustering), der physisch einem EXTENTSIZE großen Speicherbereich auf dem Datenträger zugeordnet ist.

Die Zeilenblockung wird durch die folgenden Optionen im Befehl **BIND** oder **PREP** angegeben:

BLOCKING ALL
Cursor, die mit der Klausel FOR READ ONLY deklariert oder nicht mit FOR UPDATE angegeben sind, werden mit Blockung ausgeführt.

BLOCKING NO
Cursor werden nicht mit Blockung ausgeführt.

BLOCKING UNAMBIG
Cursor, die mit der Klausel FOR READ ONLY deklariert sind, werden mit Blockung ausgeführt. Cursor, die nicht mit der Klausel FOR READ ONLY oder FOR UPDATE deklariert sind und die nicht mehrdeutig sind oder die nur einen Lesezugriff ausführen, werden mit Blockung ausgeführt. Mehrdeutige Cursor werden nicht mit Blockung ausgeführt.

Die folgenden Konfigurationsparameter des Datenbankmanagers werden bei Blockgrößenberechnungen verwendet.

- Der Parameter **aslheapsz** gibt die Größe des Zwischenspeichers für die Anwendungsunterstützungsebene für lokale Anwendungen an. Dieser Parameter wird verwendet, um die E/A-Blockgröße festzulegen, wenn ein Blockcursor geöffnet wird.
- Der Parameter **rqrioblk** gibt die Größe des Kommunikationspuffers zwischen fernen Anwendungen und ihren Datenbankagenten auf dem Datenbankserver an. Dieser Parameter wird auch verwendet, um die E/A-Blockgröße auf dem Data Server Runtime Client festzulegen, wenn ein Blockcursor geöffnet wird.

Bevor Sie die Blockung von Zeilendaten für LOB-Datentypen aktivieren, müssen Sie verstehen, welche Auswirkungen dies auf die Systemressourcen hat. Es wird mehr gemeinsam genutzter Speicher auf dem Server benötigt, um die Verweise auf

LOB-Werte in jedem Block von Daten zu speichern, wenn LOB-Spalten zurückgegeben werden. Die Anzahl solcher Verweise variiert entsprechend dem Wert des Konfigurationsparameters **rqrioblk**.

Zur Erhöhung der Speicherkapazität, die dem Zwischenspeicher zugeordnet wird, ändern Sie den Datenbankkonfigurationsparameter **database_memory** wie folgt:

- Setzen Sie den Wert des Parameters auf AUTOMATIC.
- Erhöhen Sie den Wert um 256 Seiten, wenn der Parameter gegenwärtig auf einen benutzerdefinierten numerischen Wert eingestellt ist.

Zur Erhöhung der Leistung einer vorhandenen Anwendung mit eingebettetem SQL, die auf LOB-Werte zugreift, binden Sie die Anwendung erneut, indem Sie im Befehl **BIND** entweder die Klausel BLOCKING ALL oder die Klausel BLOCKING UNAMBIG zur Anforderung der Zeilenblockung angeben. Anwendungen mit eingebettetem SQL rufen LOB-Werte jeweils zeilenweise ab, nachdem ein Block von Zeilen vom Server abgerufen wurde. Benutzerdefinierte Funktionen (UDFs), die große LOB-Ergebnisse zurückgeben, können den DB2-Server veranlassen, zum einzeiligen Abruf von LOB-Daten zurückzukehren, wenn große Mengen Speicher auf dem Server belegt werden.

Gehen Sie wie folgt vor, um die Zeilenblockung anzugeben:

Vorgehensweise

1. Schätzen Sie mithilfe 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 in Byte.
   - Verwenden Sie die folgende Formel für lokale Anwendungen:
     ```
     Zeilen pro Block = aslheapsz * 4096 / azl
     ```
     Die Anzahl von Byte pro Seite beträgt 4.096.
   - Verwenden Sie die folgende Formel für ferne Anwendungen:
     ```
     Zeilen pro Block = rqrioblk / azl
     ```
2. Zur Aktivierung der Zeilenblockung geben Sie einen entsprechenden Wert für die Option BLOCKING im Befehl **BIND** oder **PREP** an.

   Wenn Sie die Option BLOCKING nicht angeben, wird standardmäßig der Zeilenblockungstyp UNAMBIG verwendet. Bei Verwendung des Befehlszeilenprozessors (CLP) und von Call Level Interface (CLI) ist der Standardtyp der Zeilenblockung ALL.

Datenstichproben in Abfragen

Es ist häufig unpraktisch und manchmal auch unnötig, auf sämtliche für eine Abfrage relevanten Daten zuzugreifen. In einigen Fällen reicht eine Ermittlung von allgemeinen Trends oder Mustern in einer Teilmenge der Daten aus. Eine Methode bei dieser Vorgehensweise besteht darin, eine Abfrage auf eine Zufallsstichprobe aus der Datenbank auszuführen.

Das DB2-Produkt bietet Ihnen die Möglichkeit, effiziente Datenstichproben für SQL- und XQuery-Abfragen zu erfassen, durch die eine potenzielle Leistungsverbesserung für umfangreiche Abfragen um mehrere Größenordnungen erzielt und gleichzeitig ein hoher Grad an Genauigkeit aufrechterhalten werden kann.

Stichproben werden häufig für Abfragen mit Spaltenfunktionen wie AVG, COUNT und SUM verwendet, bei denen angemessen genaue Ergebniswerte aus einer Stichprobe der Daten gewonnen werden können. Eine Sichprobenentnahme kann auch dazu dienen, eine zufällige Teilmenge der Zeilen einer Tabelle zu Prüfungszwecken abzurufen oder Data Mining- und Analyseoperationen zu beschleunigen.

Es sind zwei Methoden zur Stichprobenentnahme verfügbar: die Stichprobenentnahme auf Zeilenebene und die Stichprobenentnahme auf Seitenebene.

Stichprobenentnahme auf Zeilenebene nach Bernoulli

Die Bernoulli-Stichprobe auf Zeilenebene ruft eine Stichprobe von *P* Prozent der Tabellenzeilen mithilfe 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 vom Grad der Datenwerthäufung (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 angewendet werden muss. Wenn kein Index vorhanden ist, ergeben sich keine E/A-Einsparungen gegenüber der Ausführung der Abfrage ohne Stichprobenerstellung. Wenn ein Index vorhanden ist, wird eine bessere Leistung erzielt, weil das Vergleichselement der Stichprobe auf die Satz-IDs (RIDs) innerhalb der Indexblattseiten angewendet wird. Im Normalfall erfordert dies nur eine E/A-Operation pro ausgewählter RID und eine E/A-Operation pro Indexblattseite.

Stichprobenentnahme auf Systemseitenebene

Die Stichprobenentnahme auf Systemseitenebene ist der Stichprobenentnahme auf Zeilenebene ähnlich, jedoch werden bei ihr Stichproben der Seiten (nicht der Zeilen) erstellt. Die Wahrscheinlichkeit, dass eine Seite in die Stichprobe aufgenommen wird, ist *P*/100. Die Aufnahme 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 aufgenommen wird. Im Vergleich zu keiner Stichprobenentnahme verbessert die Stichprobe auf Seitenebene die Leistung um Größenordnungen. Allerdings ist die Genauigkeit kumulierter Schätzungen bei Seitenstichproben tendenziell geringer als bei Zeilenstichproben. Dieser Unterschied wird am deutlichsten, wenn viele Zeilen pro Seite vorhanden sind oder die in der Abfrage angegebenen Zeilen einen hohen Grad der Clusterbildung innerhalb von Seiten aufweisen.

Angeben der Stichprobenmethode

Verwenden Sie die Klausel TABLESAMPLE, um eine Abfrage auf eine Zufallsstichprobe von Daten aus einer Tabelle auszuführen. Die Klausel TABLESAMPLE BERNOULLI gibt an, dass eine Bernoulli-Stichprobe auf Zeilenebene ausgeführt werden soll. Die Klausel TABLESAMPLE 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.

Parallelverarbeitung für Anwendungen

Das DB2-Produkt unterstützt parallele Umgebungen auf symmetrischen Multiprozessormaschinen (SMP-Maschinen).

In SMP-Maschinen kann mehr als ein Prozessor auf die Datenbank zugreifen, sodass die Ausführung komplexer SQL-Anforderungen unter den Prozessoren aufgeteilt werden kann. Diese *partitionsinterne Parallelität* ist die Unterteilung einer einzelnen Datenbankoperation (z. B. einer Indexerstellung) in mehrere Teile, die anschließend parallel innerhalb einer Datenbankpartition ausgeführt werden.

Zur Angabe des Grades der Parallelität bei der Kompilierung einer Anwendung verwenden Sie das Sonderregister CURRENT DEGREE oder die Bindeoption DEGREE. *Grad* bezieht sich in diesem Zusammenhang auf die Anzahl von Abfrageteilen, die gleichzeitig ausgeführt werden können. Es gibt keine strenge Beziehung zwischen der Anzahl der Prozessoren und dem Wert, den Sie für den Grad der Parallelität auswählen. Sie können einen Wert angeben, der größer oder kleiner als die Anzahl von Prozessoren auf der Maschine ist. Selbst für Einzelprozessormaschinen können Sie einen höheren Grad als 1 definieren, um die Leistung zu verbessern. Beachten Sie jedoch, das jeder weitere Grad an Parallelität den Hauptspeicherbedarf und die Prozessorbelastung im System erhöht.

Einige Konfigurationsparameter müssen modifiziert werden, um die Leistung zu optimieren, wenn Abfragen parallel ausgeführt werden. In einer Umgebung mit einem hohen Grad an Parallelität sollten Sie die Konfigurationsparameter prüfen und modifizieren, über die die Größen des gemeinsamen Speichers und des Vorablesezugriffs gesteuert werden.

Die folgenden Konfigurationsparameter steuern und verwalten die Parallelverarbeitung:

- Der Konfigurationsparameter **intra_parallel** des Datenbankmanagers aktiviert oder inaktiviert die Parallelverarbeitung.
- Der Konfigurationsparameter **max_querydegree** des Datenbankmanagers definiert eine obere Grenze für den Grad der Parallelität für alle Abfragen in der Datenbank. Dieser Wert hat Vorrang vor dem Wert des Sonderregisters CURRENT DEGREE und der Bindeoption DEGREE.
- Der Datenbankkonfigurationsparameter **dft_degree** definiert den Standardwert für das Sonderregister CURRENT DEGREE und die Bindeoption DEGREE.

Wenn eine Abfrage mit der Definition DEGREE = ANY kompiliert wird, wählt der Datenbankmanager den Grad der partitionsinternen Parallelität anhand einer Reihe von Faktoren aus, zu denen die Anzahl der Prozessoren und die Merkmale der Abfrage gehören. Der tatsächlich bei der Ausführung verwendete Grad kann aufgrund dieser Faktoren und der Aktivitätsauslastung im System niedriger als die Anzahl der Prozessoren sein. Der Grad der Parallelität kann vor der Abfrageausführung verringert werden, wenn das System ausgelastet ist.

Verwenden Sie die DB2-EXPLAIN-Funktion zum Anzeigen von Informationen zum Grad der Parallelität, der vom Optimierungsprogramm ausgewählt wurde. Zum Anzeigen von Informationen über den tatsächlich bei der Ausführung genutzten Grad der Parallelität verwenden Sie den Datenbanksystemmonitor.

Parallelität in Umgebungen ohne SMP-Maschinen

Sie können einen Grad an Parallelität angeben, ohne eine SMP-Maschine zu haben. Zum Beispiel könnten von der Ein-/Ausgabeleistung abhängige Abfragen auf einer

Einzelprozessormaschine von der Deklaration des Grades 2 oder eines höheren Grades profitieren. In diesem Fall braucht der Prozessor vielleicht nicht auf die Beendigung von Ein-/Ausgabefunktionen zu warten, bevor er mit der Verarbeitung einer neuen Abfrage beginnt. Dienstprogramme wie LOAD können die E/A-Parallelität unabhängig steuern.

Sperrenverwaltung

Die Sperrenverwaltung ist einer der Faktoren, die sich auf die Anwendungsleistung auswirken. Lesen Sie die Informationen zu den verschiedenen Aspekten der Sperrenverwaltung in diesem Abschnitt, die Ihnen helfen können, die Leistung von Datenbankanwendungen zu maximieren.

Sperren und Steuerung des gemeinsamen Zugriffs

Zur Steuerung des gemeinsamen Zugriffs sowie zur Verhinderung eines unkontrollierten Datenzugriffs aktiviert der Datenbankmanager Sperren für Pufferpools, Tabellen, Datenpartitionen, Tabellenblöcke oder Tabellenzeilen.

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, Uncommitted Read)
  - Cursorstabilität (CS, Cursor Stability)
  - Lesestabilität (RS, Read Stability)
  - Wiederholbares Lesen (RR, Repeatable Read)

  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 Zeilensperren, TABLE für Tabellensperren oder BLOCKINSERT für lediglich Blocksperren bei Tabellen mit multidimensionalem Clustering (MDC-Tabellen). Wenn die Klausel BLOCKINSERT für eine MDC-Tabelle verwendet wird, erfolgt das Sperren auf Zeilenebene, abgesehen von Einfügeoperationen (INSERT), bei denen das Sperren auf Blockebene erfolgt. Verwenden Sie für MDC-Tabellen die Anweisung ALTER TABLE...LOCKSIZE BLOCKINSERT, wenn Transaktionen umfangreiche Einfügeoperationen in getrennten Zellen durchführen. Verwenden Sie die Anweisung ALTER TABLE...LOCKSIZE TABLE für schreibgeschützte Tabellen. Dadurch wird die Zahl der Sperren verringert, die für Datenbankaktivitäten erforderlich werden. Bei partitionierten Tabellen wer-

den zuerst Tabellensperren und anschließend Datenpartitionssperren angefordert, wie dies aufgrund der Daten, auf die zugegriffen wird, erforderlich ist.

- Der Speicherplatz, der für das Sperren aufgewendet wird und der durch den Konfigurationsparameter **locklist** (Sperrenliste) der Datenbank gesteuert wird. Wenn sich die Sperrenliste gänzlich füllt, kann sich die Leistung aufgrund von Sperreneskalationen und verringertem gemeinsamen Zugriff unter gemeinsam verwendeten Objekten 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 zur Verringerung der Anzahl von Sperren, die gleichzeitig aktiviert sind, sicher, dass Transaktionen häufig Commits durchführen.

Eine Pufferpoolsperre (exklusive Sperre) wird aktiviert, wenn ein Pufferpool erstellt, geändert oder gelöscht wird. Beim Erfassen von Systemüberwachungsdaten ist es möglich, dass Sie auf diesen Sperrentyp treffen. Der Name der Sperre ist die Kennung (ID) für den Pufferpool selbst.

Im Allgemeinen werden Sperren auf Zeilenebene verwendet, sofern nicht eine der folgenden Bedingungen zutrifft:

- Die Isolationsstufe ist 'Nicht festgeschriebenes Lesen' (UR).
- Die Isolationsstufe ist 'Wiederholbares Lesen' (RR) und der Zugriffsplan erfordert eine Suche ohne Vergleichselemente für den Indexbereich.
- Das Tabellenattribut LOCKSIZE hat den Wert TABLE.
- Die Sperrenliste wird vollständig gefüllt und verursacht Sperreneskalation.
- Eine explizite Tabellensperre wurde durch die Anweisung LOCK TABLE aktiviert, wodurch verhindert wird, dass gleichzeitig aktive Anwendungsprozesse eine Tabelle ändern oder verwenden.

Bei einer MDC-Tabelle werden in folgenden Fällen Sperren auf Blockebene anstelle von Sperren auf Zeilenebene verwendet:

- Das Tabellenattribut LOCKSIZE hat den Wert BLOCKINSERT.
- Die Isolationsstufe ist 'Wiederholbares Lesen' (RR) und der Zugriffsplan beinhaltet Vergleichselemente.
- Eine Aktualisierungs- oder Löschoperation mit Suche, die nur Vergleichselemente für Dimensionsspalten umfasst.

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 im Allgemeinen nur für die Zeit aktiviert, in der sich der Cursor auf der Zeile befindet. Dabei ist zu beachten, dass bei einer CS-Suche möglicherweise gar keine Sperren aktiviert sind.
- RS-Suchen: Zeilensperren für Zeilen, die den Suchbedingungen entsprechen, bleiben nur für die Dauer der Transaktion aktiviert.
- RR-Suchen: Alle Zeilensperren bleiben für die Dauer der Transaktion aktiviert.

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, sodass eine einzelne Tabellensperre den Sperrenaufwand minimiert.

Die Klausel LOCKSIZE der Anweisung ALTER TABLE legt den Granularität (Unterteilung) von Sperren auf die Zeilen-, Datenpartitions-, Block- oder Tabellenebene fest. Standardmäßig werden Zeilensperren verwendet. Die Verwendung dieser Option in der Tabellendefinition verhindert nicht, dass eine normale Sperreneskalation auftritt.

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.

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 Wartungsmaßnahmen auf sie zu, und dieser Benutzer benötigt eine X-Sperre. Während das Wartungsprogramm ausgeführt wird, werden die reinen Leseanwendungen ausgesperrt. Unter anderen Umständen können reine Leseanwendungen jedoch auf die Tabelle unter minimalem Sperrenaufwand gleichzeitig zugreifen.

Bei einer Tabelle mit mehrdimensionalem Clustering (MDC-Tabelle) können Sie den Wert BLOCKINSERT für die Klausel LOCKSIZE angeben, um das Sperren auf Blockebene nur bei INSERT-Operationen zu verwenden. Wenn BLOCKINSERT angegeben wird, wird das Sperren auf Zeilenebene für alle anderen Operationen durchgeführt, jedoch nur minimal für INSERT-Operationen. Das heißt, das Sperren auf Blockebene wird beim Einfügen von Zeilen verwendet, während das Sperren auf Zeilenebene für das Sperren des nächsten Schlüssels verwendet wird, wenn bei der Aktualisierung von Satz-ID-Indizes in diesen Indizes RR-Suchen angetroffen werden. Das Sperren mit BLOCKINSERT kann in den folgenden Fällen nützlich sein:

- Es gibt mehrere Transaktionen, bei denen umfangreiche Einfügungen in getrennte Zellen ausgeführt werden.
- Normalerweise werden keine Einfügeoperationen von mehreren Transaktionen in dieselbe Zelle ausgeführt oder es werden von jeder Transaktion genügend Daten pro Zelle eingefügt, sodass der Benutzer keinen Zweifel hat, dass jede Transaktion die Einfügung in getrennte Blöcke ausführt.

Sperrenattribute

Die Sperren des Datenbankmanagers verfügen über eine Reihe von Basisattributen.

Diese Attribute sind folgende:

Modus
Der Typ des Zugriffs, der dem Sperreneigner gewährt wird, sowie der Typ

des Zugriffs, der 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 aktiviert Sperren auch für andere Typen von Ressourcen, wie Zeilen und Tabellenbereiche. Für Tabellen mit mehrdimensionalem Clustering (MDC-Tabellen) können außerdem Blocksperren aktiviert werden und Datenpartitionssperren können für partitionierte Tabellen aktiviert werden. Das Objekt, das gesperrt wird, bestimmt die *Granularität* der Sperre.

Zähler für Sperre
Der Zeitraum, während dessen eine Sperre aktiviert ist. Die Isolationsstufe, unter der eine Abfrage ausgeführt wird, beeinflusst den Zähler.

In Tabelle 12 sind die Sperrmodi mit ihren Auswirkungen in der Reihenfolge der zunehmenden Ressourcenbeschränkung aufgeführt.

Tabelle 12. Zusammenfassung der Sperrmodi

Sperrmodus	Relevanter Objekttyp	Beschreibung
IN (Intent None)	Tabellenbereiche, Blöcke, Tabellen, Datenpartitionen	Der Sperreneigner 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 Sperreneigner kann Daten in der gesperrten Tabelle lesen, aber nicht aktualisieren. Andere Anwendungen können die Tabelle lesen oder aktualisieren.
IX (Intent Exclusive)	Tabellenbereiche, Blöcke, Tabellen, Datenpartitionen	Der Sperreneigner und gleichzeitig ausgeführte Anwendungen können Daten lesen und aktualisieren. Andere gleichzeitig ausgeführte Anwendungen können die Tabelle sowohl lesen als auch aktualisieren.
NS (Scan Share)	Zeilen	Der Sperreneigner 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.
NW (Next Key Weak Exclusive)	Zeilen	Wenn eine Zeile in einen Index eingefügt wird, wird eine NW-Sperre für die nächste Zeile aktiviert. Dies geschieht nur, wenn die nächste Zeile momentan von einer RR-Suche gesperrt wird. Der Sperreneigner kann die gesperrte Zeile lesen, aber nicht aktualisieren. Dieser Sperrmodus ist dem Sperrmodus X ähnlich, abgesehen davon, dass er auch mit NS-Sperren kompatibel ist.
S (Share)	Zeilen, Blöcke, Tabellen, Datenpartitionen	Der Sperreneigner und jede gleichzeitig ausgeführte Anwendung kann die gesperrten Daten lesen, aber nicht aktualisieren.
SIX (Share with Intent Exclusive)	Tabellen, Blöcke, Datenpartitionen	Der Sperreneigner kann Daten lesen und aktualisieren. Andere gleichzeitig ausgeführte Anwendungen können die Tabelle lesen.
U (Update)	Zeilen, Blöcke, Tabellen, Datenpartitionen	Der Sperreneigner kann Daten aktualisieren. Andere UOWs (Units of Work) können die Daten im gesperrten Objekt lesen, jedoch nicht aktualisieren.
X (Exclusive)	Zeilen, Blöcke, Tabellen, Pufferpools, Datenpartitionen	Der Sperreneigner kann Daten im gesperrten Objekt lesen und auch aktualisieren. Nur Anwendungen mit der Isolationsstufe UR (Nicht festgeschriebenes Lesen) können auf das gesperrte Objekt zugreifen.

Tabelle 12. Zusammenfassung der Sperrmodi (Forts.)

Sperrmodus	Relevanter Objekttyp	Beschreibung
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 Typen von Tabellenreorganisation durchgeführt werden. Keine anderen gleichzeitig ausgeführten Anwendungen können die Tabelle lesen oder aktualisieren.

Faktoren mit Auswirkungen auf Sperren

Eine Reihe von Faktoren beeinflusst den Modus und die Granularität von Sperren des Datenbankmanagers:

Zu diesem Faktoren gehört Folgendes:

- Die Art der Verarbeitung, die von der Anwendung ausgeführt wird
- Die Datenzugriffsmethode
- Die Werte verschiedener Konfigurationsparameter

Sperren und Typen der Anwendungsverarbeitung

Zum Zweck der Bestimmung von Sperrenattributen kann die Anwendungsverarbeitung einem der folgenden Typen zugeordnet werden: mit Lesezugriff, mit Änderungsabsicht, mit Änderung und cursorgesteuert.

- Anwendungsverarbeitung mit Lesezugriff

  Dieser Verarbeitungstyp umfasst alle SELECT-Anweisungen, die inhärent 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 Typ sind nur gemeinsam nutzbare Sperren (IS, NS oder S) erforderlich.
- Anwendungsverarbeitung mit Änderungsabsicht

  Dieser Verarbeitungstyp umfasst alle SELECT-Anweisungen mit der Klausel FOR UPDATE, der Klausel USE AND KEEP UPDATE LOCKS, der Klausel USE AND KEEP EXCLUSIVE LOCKS oder Anweisungen, die mehrdeutig sind, jedoch vom Abfragecompiler als Anweisung mit Änderungsabsicht eingestuft werden. Für diesen Typ werden gemeinsam nutzbare Sperren und Aktualisierungssperren (S, U oder X für Zeilen; IX, S, U oder X für Blöcke und IX, U oder X für Tabellen) verwendet.
- Anwendungsverarbeitung mit Änderung

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

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

Eine Anweisung, die an einer Zieltabelle Einfüge-, Aktualisierungs- oder Löschoperationen (INSERT, UPDATE oder DELETE) auf der Basis des Ergebnisses einer Subselect-Anweisung vornimmt, führt zwei Typen von Verarbeitung aus. Die Regeln für die Verarbeitung mit Lesezugriff bestimmen die Sperren für die Tabellen,

die Daten in der Subselect-Anweisung zurückgeben. Die Regeln für die Verarbeitung mit Änderung bestimmen die Sperren für die Zieltabelle.

Sperren 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 eine Indexsuche zum Auffinden einer bestimmten Zeile verwendet wird, wählt das Optimierungsprogramm gewöhnlich Sperren auf Zeilenebene (IS) für die Tabelle aus. Wenn die Tabelle EMPLOYEE zum Beispiel einen Index für die Spalte EMPNO (Personalnummer) hat, könnte ein Zugriff über diesen Index ausgeführt werden, um Informationen zu einem einzelnen Mitarbeiter auszuwählen:

```
select * from employee
  where empno = '000310'
```

Wenn kein Index verwendet wird, muss die gesamte Tabelle der Reihe nach durchsucht werden, um die erforderlichen Zeilen zu finden. In diesem Fall wählt das Optimierungsprogramm wahrscheinlich eine einzelne Sperre auf Tabellenebene (S) aus. Wenn zum Beispiel kein Index für die Spalte SEX vorhanden ist, könnte eine Tabellensuche verwendet werden, um alle männlichen Mitarbeiter wie folgt auszuwählen:

```
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 des Cursors immer eine exklusive Sperre aktiviert, um die Aktualisierungs- oder Löschoperation auszuführen.

Das Sperren in Bereichsclustertabellen funktioniert etwas anders als das standardmäßig ausgeführte Sperren von 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 standardmäßig ausgeführten Sperren von Schlüsseln werden nur Zeilen mit vorhandenen Daten gesperrt.

Der verzögerte Zugriff auf Datenseiten impliziert, dass der Zugriff auf eine Zeile in zwei Schritten erfolgt, wodurch es zu komplexeren Sperrszenarios kommt. Die Zeitpunkte der Aktivierung von Sperren und die Dauer von Sperren hängen von der Isolationsstufe ab. Da bei der Isolationsstufe 'Wiederholbares Lesen' (RR) alle Sperren bis zum Ende einer 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' (RS) und 'Cursorstabilität' (CS) müssen Sperren im zweiten Schritt aktiviert werden. Um den gemeinsamen Zugriff zu maximieren, werden im ersten Schritt keine Sperren aktiviert und die erneute Anwendung aller Vergleichselemente stellt sicher, dass nur Zeilen zurückgegeben werden, die den Auswahlbedingungen entsprechen.

Sperrtypenkompatibilität

Die Sperrenkompatibilität wird zu einem Problem, wenn eine Anwendung eine Sperre für ein Objekt aktiviert hat und eine weitere Anwendung eine Sperre für dasselbe Objekt anfordert. Wenn die beiden Sperrmodi kompatibel sind, kann die Anforderung für eine zweite Sperre für das Objekt zugelassen werden.

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

In Tabelle 13 ist aufgelistet, welche Sperrtypen kompatibel (**Ja**) und welche Typen nicht kompatibel (**Nein**) sind. Beachten Sie, dass es zu einer Zeitüberschreitung kommen kann, wenn ein Anforderer auf eine Sperre wartet.

Tabelle 13. Kompatibilität der Sperrmodi

	Status der aktiven Sperre										
Angeforderter Status	**None**	**IN**	**IS**	**NS**	**S**	**IX**	**SIX**	**U**	**X**	**Z**	**NW**
None	Ja	Ja	Ja	Ja	Ja	Ja	Ja	Ja	Ja	Ja	Ja
IN (Intent None)	Ja	Ja	Ja	Ja	Ja	Ja	Ja	Ja	Ja	Nein	Ja
IS (Intent Share)	Ja	Ja	Ja	Ja	Ja	Ja	Ja	Ja	Nein	Nein	Nein
NS (Scan Share)	Ja	Ja	Ja	Ja	Ja	Nein	Nein	Ja	Nein	Nein	Ja
S (Share)	Ja	Ja	Ja	Ja	Ja	Nein	Nein	Ja	Nein	Nein	Nein
IX (Intent Exclusive)	Ja	Ja	Ja	Nein	Nein	Ja	Nein	Nein	Nein	Nein	Nein
SIX (Share with Intent Exclusive)	Ja	Ja	Ja	Nein	Nein	Nein	Nein	Nein	Nein	Nein	Nein
U (Update)	Ja	Ja	Ja	Ja	Ja	Nein	Nein	Nein	Nein	Nein	Nein
X (Exclusive)	Ja	Ja	Nein	Nein	Nein	Nein	Nein	Nein	Nein	Nein	Nein
Z (Super Exclusive)	Ja	Nein	Nein	Nein	Nein	Nein	Nein	Nein	Nein	Nein	Nein
NW (Next Key Weak Exclusive)	Ja	Ja	Nein	Ja	Nein	Nein	Nein	Nein	Nein	Nein	Nein

Sperren der nächsten Schlüssel

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 Indexsuche unter der Isolationsstufe 'Wiederholbares Lesen' (RR) gesperrt ist. Wenn dies geschieht, wird die Einfügung des neuen Indexschlüssels ausgesetzt, bis die Transaktion, die die RR-Suche ausgeführt hat, abgeschlossen wird.

Der Sperrmodus, der für die Sperre des nächsten Schlüssels verwendet wird, ist NW (Next Key Weak Exclusive). Diese Sperre des nächsten Schlüssels wird freigegeben, bevor die Einfügung des Schlüssels erfolgt, das heißt, bevor eine Zeile in die Tabelle eingefügt wird.

Das Einfügen eines Schlüssels erfolgt auch, wenn Aktualisierungen an einer Zeile eine Änderung am Wert des Indexschlüssels für diese Zeile zur Folge haben, 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. Bei partitionierten Indizes für partitionierte Tabellen werden anstel-

le einer einzigen Sperre für das Indexende Sperren für das Ende der einzelnen Indexpartitionen abgerufen. Ist der Schlüssel, der auf das Ende des Suchbereichs folgt, als gelöscht markiert, erfolgt eine der folgenden Aktionen:

- Bei der Suche werden die entsprechenden Zeilen weiter gesperrt, bis ein Schlüssel gefunden wird, der nicht als gelöscht markiert ist.
- Die Zeile für den betreffenden Schlüssel wird im Rahmen der Suche gesperrt.
- Das Ende des Index wird im Rahmen der Suche gesperrt.

Sperrmodi und Zugriffspläne für Standardtabellen

Der Typ von Sperre, den eine Standardtabelle aktiviert, hängt von der geltenden Isolationsstufe sowie vom verwendeten Datenzugriffsplan ab.

In den folgenden Tabellen werden die Typen von Sperren aufführt, die für Standardtabellen unter der jeweiligen Isolationsstufe für verschiedene Zugriffspläne aktiviert werden. Jeder Eintrag hat zwei Teile: die Tabellensperre und die Zeilensperre. Ein Bindestrich gibt an, dass eine bestimmte Sperrgranularität nicht verfügbar ist.

Die Tabellen 7 - 12 zeigen die Typen von Sperren, die aktiviert werden, wenn das Lesen von Datenseiten verzögert wird, um zu ermöglichen, dass die Liste der Zeilen unter Verwendung mehrerer Indizes weiter qualifiziert oder für einen effizienten Vorablesezugriff sortiert wird.

- Tabelle 1. Sperrmodi für Tabellensuchen ohne Vergleichselemente
- Tabelle 2. Sperrmodi für Tabellensuchen mit Vergleichselementen
- Tabelle 3. Sperrmodi für Satz-ID-Indexsuchen ohne Vergleichselemente
- Tabelle 4. Sperrmodi für Satz-ID-Indexsuchen mit einer einzigen qualifizierten Zeile
- Tabelle 5. Sperrmodi für Satz-ID-Indexsuchen nur mit Start- und Stoppvergleichselementen
- Tabelle 6. Sperrmodi für Satz-ID-Indexsuchen nur mit Index und anderen Vergleichselementen (sargs, resids)
- Tabelle 7. Sperrmodi für Indexsuchen, die für verzögerten Datenseitenzugriff verwendet werden: Satz-ID-Indexsuche ohne Vergleichselemente
- Tabelle 8. Sperrmodi für Indexsuchen, die für verzögerten Datenseitenzugriff verwendet werden: Nach einer Satz-ID-Indexsuche ohne Vergleichselemente
- Tabelle 9. Sperrmodi für Indexsuchen, die für verzögerten Datenseitenzugriff verwendet werden: Satz-ID-Indexsuche mit Vergleichselementen (sargs, resids)
- Tabelle 10. Sperrmodi für Indexsuchen, die für verzögerten Datenseitenzugriff verwendet werden: Nach einer Satz-ID-Indexsuche mit Vergleichselementen (sargs, resids)
- Tabelle 11. Sperrmodi für Indexsuchen, die für verzögerten Datenseitenzugriff verwendet werden: Satz-ID-Indexsuche nur mit Start- und Stoppvergleichselementen
- Tabelle 12. Sperrmodi für Indexsuchen, die für verzögerten Datenseitenzugriff verwendet werden: Nach einer Satz-ID-Indexsuche nur mit Start- und Stoppvergleichselementen

Anmerkung:

1. Für MDC-Tabellen (MDC, mehrdimensionales Clustering) sind auch Sperren auf Blockebene verfügbar.

2. Sperrmodi können explizit mit einer *Sperranforderungsklausel* (LOCK REQUEST) einer SELECT-Anweisung geändert werden.

Tabelle 14. 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 15. 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 unter der Isolationsstufe UR Vergleichselemente für INCLUDE-Spalten im Index angewendet werden, wird die Isolationsstufe auf CS erhöht und die Sperren werden auf eine IS-Tabellensperre oder auf NS-Zeilensperren hochgestuft.

Tabelle 16. 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	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 17. Sperrmodi für Satz-ID-Indexsuchen (RID) mit einer einzigen qualifizierten Zeile

Isolationsstufe	Suchen: mehrdeutig und Lesezugriff	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

Tabelle 17. Sperrmodi für Satz-ID-Indexsuchen (RID) mit einer einzigen qualifizierten Zeile (Forts.)

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

Tabelle 18. Sperrmodi für Satz-ID-Indexsuchen (RID) 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	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 19. Sperrmodi für Satz-ID-Indexsuchen (RID) nur mit Index und anderen Vergleichselementen (sargs, resids)

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	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

Tabelle 20. Sperrmodi für Indexsuchen, die für verzögerten Datenseitenzugriff verwendet werden: Satz-ID-Indexsuche (RID) ohne Vergleichselemente

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		X/-	
RS	IN/-	IN/-		IN/-	
CS	IN/-	IN/-		IN/-	
UR	IN/-	IN/-		IN/-	

Tabelle 21. Sperrmodi für Indexsuchen, die für verzögerten Datenseitenzugriff verwendet werden: Nach einer Satz-ID-Indexsuche (RID) ohne Vergleichselemente

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	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 22. Sperrmodi für Indexsuchen, die für verzögerten Datenseitenzugriff verwendet werden: Satz-ID-Indexsuche (RID) mit Vergleichselementen (sargs, resids)

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/S	
RS	IN/-	IN/-		IN/-	
CS	IN/-	IN/-		IN/-	
UR	IN/-	IN/-		IN/-	

Tabelle 23. Sperrmodi für Indexsuchen, die für verzögerten Datenseitenzugriff verwendet werden: Nach einer Satz-ID-Indexsuche (RID) mit Vergleichselementen (sargs, resids)

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

Tabelle 24. Sperrmodi für Indexsuchen, die für verzögerten Datenseitenzugriff verwendet werden: Satz-ID-Indexsuche (RID) 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 25. Sperrmodi für Indexsuchen, die für verzögerten Datenseitenzugriff verwendet werden: Nach einer Satz-ID-Indexsuche (RID) 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

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

Der Typ von Sperre, den eine Tabelle mit mehrdimensionalem Clustering (MDC-Tabelle) bei einer Tabellen- oder Satz-ID-Indexsuche aktiviert, hängt von der geltenden Isolationsstufe sowie vom verwendeten Datenzugriffsplan ab.

In den folgenden Tabellen werden die Typen von Sperren aufführt, die für MDC-Tabellen unter der jeweiligen Isolationsstufe für verschiedene Zugriffspläne aktiviert werden. Jeder Eintrag hat drei Teile: die Tabellensperre, die Blocksperre und die Zeilensperre. Ein Bindestrich gibt an, dass eine bestimmte Sperrgranularität nicht verfügbar ist.

Die Tabellen 9 - 14 zeigen die Typen von Sperren, die für Satz-ID-Indexsuchen aktiviert werden, wenn das Lesen von Datenseiten verzögert wird. Wenn unter der Isolationsstufe UR Vergleichselemente für INCLUDE-Spalten im Index angewendet werden, wird die Isolationsstufe auf CS erhöht und die Sperren werden auf eine IS-Tabellensperre, eine IS-Blocksperre oder auf NS-Zeilensperren hochgestuft.

- Tabelle 1. Sperrmodi für Tabellensuchen ohne Vergleichselemente
- Tabelle 2. Sperrmodi für Tabellensuchen mit Vergleichselementen nur für Dimensionsspalten
- Tabelle 3. Sperrmodi für Tabellensuchen mit anderen Vergleichselementen (sargs, resids)
- Tabelle 4. Sperrmodi für Satz-ID-Indexsuchen ohne Vergleichselemente
- Tabelle 5. Sperrmodi für Satz-ID-Indexsuchen mit einer einzigen qualifizierten Zeile
- Tabelle 6. Sperrmodi für Satz-ID-Indexsuchen nur mit Start- und Stoppvergleichselementen
- Tabelle 7. Sperrmodi für Satz-ID-Indexsuchen nur mit Indexvergleichselementen
- Tabelle 8. Sperrmodi für Satz-ID-Indexsuchen mit anderen Vergleichselementen (sargs, resids)
- Tabelle 9. Sperrmodi für Indexsuchen, die für verzögerten Datenseitenzugriff verwendet werden: Satz-ID-Indexsuche ohne Vergleichselemente
- Tabelle 10. Sperrmodi für Indexsuchen, die für verzögerten Datenseitenzugriff verwendet werden: Nach einer Satz-ID-Indexsuche (RID) ohne Vergleichselemente
- Tabelle 11. Sperrmodi für Indexsuchen, die für verzögerten Datenseitenzugriff verwendet werden: Satz-ID-Indexsuche (RID) mit Vergleichselementen (sargs, resids)

- Tabelle 12. Sperrmodi für Indexsuchen, die für verzögerten Datenseitenzugriff verwendet werden: Nach einer Satz-ID-Indexsuche mit Vergleichselementen (sargs, resids)
- Tabelle 13. Sperrmodi für Indexsuchen, die für verzögerten Datenseitenzugriff verwendet werden: Satz-ID-Indexsuche nur mit Start- und Stoppvergleichselementen
- Tabelle 14. Sperrmodi für Indexsuchen, die für verzögerten Datenseitenzugriff verwendet werden: Nach einer Satz-ID-Indexsuche nur mit Start- und Stoppvergleichselementen

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

Tabelle 26. 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/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 27. Sperrmodi für Tabellensuchen mit Vergleichselementen nur für Dimensionsspalten

Isolationsstufe	Suchen: mehrdeutig und Lesezugriff	Cursorgesteuerte Operationen		UPDATE oder DELETE mit Suche	
		Suche	WHERE CURRENT OF	Suche	UPDATE oder DELETE
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 28. Sperrmodi für Tabellensuchen mit anderen Vergleichselementen (sargs, resids)

Isolationsstufe	Suchen: mehrdeutig und Lesezugriff	Cursorgesteuerte Operationen		UPDATE oder DELETE mit Suche	
		Suche	WHERE CURRENT OF	Suche	UPDATE oder DELETE
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

Tabelle 29. 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	UPDATE oder DELETE
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 30. Sperrmodi für Satz-ID-Indexsuchen (RID) mit einer einzigen qualifizierten Zeile

Isolationsstufe	Suchen: mehrdeutig und Lesezugriff	Cursorgesteuerte Operationen		UPDATE oder DELETE mit Suche	
		Suche	WHERE CURRENT OF	Suche	UPDATE oder DELETE
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 31. Sperrmodi für Satz-ID-Indexsuchen (RID) 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/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 32. Sperrmodi für Satz-ID-Indexsuchen (RID) nur mit Indexvergleichselementen

Isolationsstufe	Suchen: mehrdeutig und Lesezugriff	Cursorgesteuerte Operationen		UPDATE oder DELETE mit Suche	
		Suche	WHERE CURRENT OF	Suche	UPDATE oder DELETE
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 33. Sperrmodi für Satz-ID-Indexsuchen (RID) mit anderen Vergleichselementen (sargs, resids)

Isolationsstufe	Suchen: mehrdeutig und Lesezugriff	Cursorgesteuerte Operationen		UPDATE oder DELETE mit Suche	
		Suche	WHERE CURRENT OF	Suche	UPDATE oder DELETE
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 34. Sperrmodi für Indexsuchen, die für verzögerten Datenseitenzugriff verwendet werden: Satz-ID-Indexsuche (RID) ohne Vergleichselemente

Isolationsstufe	Suchen: mehrdeutig und Lesezugriff	Cursorgesteuerte Operationen		UPDATE oder DELETE mit Suche	
		Suche	WHERE CURRENT OF	Suche	UPDATE oder DELETE
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 35. Sperrmodi für Indexsuchen, die für verzögerten Datenseitenzugriff verwendet werden: Nach einer Satz-ID-Indexsuche (RID) ohne Vergleichselemente

Isolationsstufe	Suchen: mehrdeutig und Lesezugriff	Cursorgesteuerte Operationen		UPDATE oder DELETE mit Suche	
		Suche	WHERE CURRENT OF	Suche	UPDATE oder DELETE
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 36. Sperrmodi für Indexsuchen, die für verzögerten Datenseitenzugriff verwendet werden: Satz-ID-Indexsuche (RID) mit Vergleichselementen (sargs, resids)

Isolationsstufe	Suchen: mehrdeutig und Lesezugriff	Cursorgesteuerte Operationen		UPDATE oder DELETE mit Suche	
		Suche	WHERE CURRENT OF	Suche	UPDATE oder DELETE
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 37. Sperrmodi für Indexsuchen, die für verzögerten Datenseitenzugriff verwendet werden: Nach einer Satz-ID-Indexsuche (RID) mit Vergleichselementen (sargs, resids)

Isolationsstufe	Suchen: mehrdeutig und Lesezugriff	Cursorgesteuerte Operationen		UPDATE oder DELETE mit Suche	
		Suche	WHERE CURRENT OF	Suche	UPDATE oder DELETE
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

Tabelle 38. Sperrmodi für Indexsuchen, die für verzögerten Datenseitenzugriff verwendet werden: Satz-ID-Indexsuche (RID) 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/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 39. Sperrmodi für Indexsuchen, die für verzögerten Datenseitenzugriff verwendet werden: Nach einer Satz-ID-Indexsuche (RID) 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/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

Sperrmodi für MDC-Blockindexsuchen

Der Typ von Sperre, den eine Tabelle mit mehrdimensionalem Clustering (MDC-Tabelle) bei einer Blockindexsuche aktiviert, hängt von der geltenden Isolationsstufe sowie vom verwendeten Datenzugriffsplan ab.

In den folgenden Tabellen werden die Typen von Sperren aufführt, die für MDC-Tabellen unter der jeweiligen Isolationsstufe für verschiedene Zugriffspläne aktiviert werden. Jeder Eintrag hat drei Teile: die Tabellensperre, die Blocksperre und die Zeilensperre. Ein Bindestrich gibt an, dass eine bestimmte Sperrgranularität nicht verfügbar ist.

Die Tabellen 5 - 12 zeigen die Typen von Sperren, die für Blockindexsuchen aktiviert werden, wenn das Lesen von Datenseiten verzögert wird.

- Tabelle 1. Sperrmodi für Indexsuchen ohne Vergleichselemente
- Tabelle 2. Sperrmodi für Indexsuchen mit Vergleichselementen nur für Dimensionsspalten
- Tabelle 3. Sperrmodi für Indexsuchen nur mit Start- und Stoppvergleichselementen
- Tabelle 4. Sperrmodi für Indexsuchen mit Vergleichselementen
- Tabelle 5. Sperrmodi für Indexsuchen, die für verzögerten Datenseitenzugriff verwendet werden: Blockindexsuche ohne Vergleichselemente
- Tabelle 6. Sperrmodi für Indexsuchen, die für verzögerten Datenseitenzugriff verwendet werden: Nach einer Blockindexsuche ohne Vergleichselemente
- Tabelle 7. Sperrmodi für Indexsuchen, die für verzögerten Datenseitenzugriff verwendet werden: Blockindexsuche mit Vergleichselementen nur für Dimensionsspalten
- Tabelle 8. Sperrmodi für Indexsuchen, die für verzögerten Datenseitenzugriff verwendet werden: Nach einer Blockindexsuche Vergleichselementen nur für Dimensionsspalten
- Tabelle 9. Sperrmodi für Indexsuchen, die für verzögerten Datenseitenzugriff verwendet werden: Blockindexsuche nur mit Start- und Stoppvergleichselementen
- Tabelle 10. Sperrmodi für Indexsuchen, die für verzögerten Datenseitenzugriff verwendet werden: Nach einer Blockindexsuche nur mit Start- und Stoppvergleichselementen
- Tabelle 11. Sperrmodi für Indexsuchen, die für verzögerten Datenseitenzugriff verwendet werden: Blockindexsuche mit anderen Vergleichselementen (sargs, resids)
- Tabelle 12. Sperrmodi für Indexsuchen, die für verzögerten Datenseitenzugriff verwendet werden: Nach einer Blockindexsuche mit anderen Vergleichselementen (sargs, resids)

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

Tabelle 40. Sperrmodi für Indexsuchen ohne Vergleichselemente

Isolationsstufe	Suchen: mehrdeutig und Lesezugriff	Cursorgesteuerte Operationen		UPDATE oder DELETE mit Suche	
		Suche	WHERE CURRENT OF	Suche	UPDATE oder DELETE
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 41. Sperrmodi für Indexsuchen mit Vergleichselementen nur für Dimensionsspalten

Isolationsstufe	Suchen: mehrdeutig und Lesezugriff	Cursorgesteuerte Operationen		UPDATE oder DELETE mit Suche	
		Suche	WHERE CURRENT OF	Suche	UPDATE oder DELETE
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 42. 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	UPDATE oder DELETE
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 43. Sperrmodi für Indexsuchen mit Vergleichselementen

Isolationsstufe	Suchen: mehrdeutig und Lesezugriff	Cursorgesteuerte Operationen		UPDATE oder DELETE mit Suche	
		Suche	WHERE CURRENT OF	Suche	UPDATE oder DELETE
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

Tabelle 44. Sperrmodi für Indexsuchen, die für verzögerten Datenseitenzugriff verwendet werden: Blockindexsuche ohne Vergleichselemente

Isolationsstufe	Suchen: mehrdeutig und Lesezugriff	Cursorgesteuerte Operationen		UPDATE oder DELETE mit Suche	
		Suche	WHERE CURRENT OF	Suche	UPDATE oder DELETE
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 45. Sperrmodi für Indexsuchen, die für verzögerten Datenseitenzugriff verwendet werden: Nach einer Blockindexsuche ohne Vergleichselemente

Isolationsstufe	Suchen: mehrdeutig und Lesezugriff	Cursorgesteuerte Operationen		UPDATE oder DELETE mit Suche	
		Suche	WHERE CURRENT OF	Suche	UPDATE oder DELETE
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 46. Sperrmodi für Indexsuchen, die für verzögerten Datenseitenzugriff verwendet werden: Blockindexsuche mit Vergleichselementen nur für Dimensionsspalten

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

Tabelle 47. Sperrmodi für Indexsuchen, die für verzögerten Datenseitenzugriff verwendet werden: Nach einer Blockindexsuche Vergleichselementen nur für Dimensionsspalten

Isolationsstufe	Suchen: mehrdeutig und Lesezugriff	Cursorgesteuerte Operationen		UPDATE oder DELETE mit Suche	
		Suche	WHERE CURRENT OF	Suche	UPDATE oder DELETE
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 48. Sperrmodi für Indexsuchen, die für verzögerten Datenseitenzugriff verwendet werden: Blockindexsuche 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/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 49. Sperrmodi für Indexsuchen, die für verzögerten Datenseitenzugriff verwendet werden: 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	UPDATE oder DELETE
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 50. Sperrmodi für Indexsuchen, die für verzögerten Datenseitenzugriff verwendet werden: Blockindexsuche mit anderen Vergleichselementen (sargs, resids)

Isolationsstufe	Suchen: mehrdeutig und Lesezugriff	Cursorgesteuerte Operationen		UPDATE oder DELETE mit Suche	
		Suche	WHERE CURRENT OF	Suche	UPDATE oder DELETE
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 51. Sperrmodi für Indexsuchen, die für verzögerten Datenseitenzugriff verwendet werden: 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	UPDATE oder DELETE
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

Sperrverhalten für partitionierte Tabellen

Zusätzlich zu einer Sperre für die gesamte Tabelle wird eine Sperre für jede Datenpartition einer partitionierten Tabelle aktiviert.

Auf diese Weise können die zu sperrenden Bereiche besser differenziert und der gemeinsame Zugriff im Vergleich zu einer nicht partitionierten Tabelle erhöht werden. Die Datenpartitionssperre wird in der Ausgabe des Befehls **db2pd**, von Ereignismonitoren, von Verwaltungssichten und von Tabellenfunktionen ausgewiesen.

Wenn auf eine Tabelle zugegriffen wird, wird zunächst eine Tabellensperre aktiviert. Anschließend werden Datenpartitionssperren nach Bedarf aktiviert. Aufgrund der verwendeten Zugriffsmethoden und Isolationsstufen ist es möglich, dass Da-

tenpartitionen gesperrt werden müssen, die nicht in der Ergebnismenge vertreten sind. Wenn diese Datenpartitionssperren aktiviert wurden, können sie möglicherweise ebenso lange wie die Tabellensperre beibehalten werden. Bei einer Suche in einem Index unter der Isolationsstufe 'Cursorstabilität' (CS) können beispielsweise die Sperren für Datenpartitionen, auf die zuvor zugegriffen wurde, beibehalten werden, um den späteren Aufwand für eine erneute Aktivierung von Datenpartitionssperren zu verringern.

Datenpartitionssperren beinhalten auch den Aufwand für die Sicherstellung des Zugriffs auf Tabellenbereiche. Bei nicht partitionierten Tabellen wird der Zugriff auf Tabellenbereiche durch Tabellensperren gesteuert. Datenpartitionssperren werden aktiviert, auch wenn eine exklusive oder den gemeinsamen Zugriff zulassende Sperre (Share) auf Tabellenebene aktiviert ist.

Durch die feinere Differenzierung (Granularität) kann eine Transaktion exklusiven Zugriff auf eine bestimmte Datenpartition haben und Zeilensperren vermeiden, während andere Transaktionen auf andere Datenpartitionen zugreifen. Dies kann das Ergebnis des für eine Massenaktualisierung ausgewählten Zugriffsplans oder der Eskalation von Sperren auf die Datenpartitionsebene sein. Als Tabellensperre für zahlreiche Zugriffsmethoden wird normalerweise eine Intent-Sperre verwendet. Dies gilt auch dann, wenn für die Datenpartitionen eine den gemeinsamen Zugriff zulassende oder eine exklusive Sperre aktiviert 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 unter Umständen auf alle Datenpartitionen zugegriffen wird, wird möglicherweise auf Tabellenebene eine Nicht-Intent-Sperre ausgewählt, um Datenpartitionsdeadlocks zwischen gleichzeitig zugreifenden Transaktionen zu vermeiden.

Anweisungen LOCK TABLE

Bei partitionierten Tabellen wird durch die Anweisung LOCK TABLE nur eine Sperre auf Tabellenebene angefordert. Dadurch werden Zeilensperren durch nachfolgende DML-Anweisungen (DML, Data Manipulation Language - Datenbearbeitungssprache) verhindert und Deadlocks auf Zeilen-, Block- oder Datenpartitionsebene vermieden. Mithilfe der Option IN EXCLUSIVE MODE kann ein exklusiver Zugriff bei der Aktualisierung von Indizes sichergestellt werden. Dies ist sinnvoll, um das Anwachsen von Indizes während einer umfangreichen Aktualisierung zu begrenzen.

Auswirkung der Option LOCKSIZE TABLE der Anweisung ALTER TABLE

Die Option LOCKSIZE TABLE stellt sicher, dass eine Tabelle im Modus für gemeinsamen Zugriff (Share) oder im Exklusivmodus ohne Intent-Sperren gesperrt wird. Bei einer partitionierten Tabelle wird diese Sperrenstrategie sowohl auf die Tabellensperre als auch auf Datenpartitionssperren angewendet.

Sperreneskalation auf Zeilen- und Blockebene

Sperren auf Zeilen- und Blockebene in partitionierten Tabellen können auf die Partitionsebene eskaliert werden. Wenn dies geschieht, können andere Transaktionen besser auf die Tabelle zugreifen, selbst wenn die Sperre für eine Datenpartition auf den gemeinsamen, exklusiven oder superexklusiven Modus (Share, Exclusive oder Super Exclusive) eskaliert wird, weil andere Datenpartitionen davon unberührt bleiben. Der Benachrichtigungsprotokolleintrag für eine Eskalation enthält die betroffene Datenpartition und den Namen der Tabelle.

Ein exklusiver Zugriff auf einen nicht partitionierten Index kann durch eine Sperreneskalation nicht sichergestellt werden. Für einen exklusiven Zugriff müssen folgende Bedingungen erfüllt sein:

- Die Anweisung muss eine exklusive Sperre auf Tabellenebene verwenden.
- Es muss die explizite Anweisung LOCK TABLE IN EXCLUSIVE MODE abgesetzt werden.
- Die Tabelle muss über das Attribut LOCKSIZE TABLE verfügen.

Bei partitionierten Indizes wird der exklusive Zugriff auf eine Indexpartition durch eine Sperreneskalation zu einem exklusiven Zugriffsmodus bzw. einem Zugriff im Modus "Z" (Super Exclusive) sichergestellt.

Interpretieren von Informationen über Sperren

Die Verwaltungssicht SNAPLOCK kann Sie bei der Interpretation von Sperreninformationen unterstützen, die für eine partitionierte Tabelle zurückgegeben werden. Die folgende Verwaltungssicht SNAPLOCK wurde während einer Offline-Indexreorganisation erfasst.

```
SELECT SUBSTR(TABNAME, 1, 15) TABNAME, TAB_FILE_ID, SUBSTR(TBSP_NAME, 1, 15) TBSP_NAME, DATA_PARTITION_ID, LOCK_OBJECT_TYPE,
LOCK_MODE, LOCK_ESCALATION FROM SYSIBMADM.SNAPLOCK where TABNAME like 'TP1' and LOCK_OBJECT_TYPE like 'TABLE_%'
ORDER BY TABNAME, DATA_PARTITION_ID, LOCK_OBJECT_TYPE, TAB_FILE_ID, LOCK_MODE

TABNAME         TAB_FILE_ID          TBSP_NAME       DATA_PARTITION_ID LOCK_OBJECT_TYPE   LOCK_MODE  LOCK_ESCALATION
--------------- -------------------- --------------- ----------------- ------------------ ---------- ---------------
TP1                           32768 -                              -1 TABLE_LOCK         Z                       0
TP1                               4 USERSPACE1                     0 TABLE_PART_LOCK    Z                       0
TP1                               5 USERSPACE1                     1 TABLE_PART_LOCK    Z                       0
TP1                               6 USERSPACE1                     2 TABLE_PART_LOCK    Z                       0
TP1                               7 USERSPACE1                     3 TABLE_PART_LOCK    Z                       0
TP1                               8 USERSPACE1                     4 TABLE_PART_LOCK    Z                       0
TP1                               9 USERSPACE1                     5 TABLE_PART_LOCK    Z                       0
TP1                              10 USERSPACE1                     6 TABLE_PART_LOCK    Z                       0
TP1                              11 USERSPACE1                     7 TABLE_PART_LOCK    Z                       0
TP1                              12 USERSPACE1                     8 TABLE_PART_LOCK    Z                       0
TP1                              13 USERSPACE1                     9 TABLE_PART_LOCK    Z                       0
TP1                              14 USERSPACE1                    10 TABLE_PART_LOCK    Z                       0
TP1                              15 USERSPACE1                    11 TABLE_PART_LOCK    Z                       0
TP1                               4 USERSPACE1                     - TABLE_LOCK         Z                       0
TP1                               5 USERSPACE1                     - TABLE_LOCK         Z                       0
TP1                               6 USERSPACE1                     - TABLE_LOCK         Z                       0
TP1                               7 USERSPACE1                     - TABLE_LOCK         Z                       0
TP1                               8 USERSPACE1                     - TABLE_LOCK         Z                       0
TP1                               9 USERSPACE1                     - TABLE_LOCK         Z                       0
TP1                              10 USERSPACE1                     - TABLE_LOCK         Z                       0
TP1                              11 USERSPACE1                     - TABLE_LOCK         Z                       0
TP1                              12 USERSPACE1                     - TABLE_LOCK         Z                       0
TP1                              13 USERSPACE1                     - TABLE_LOCK         Z                       0
TP1                              14 USERSPACE1                     - TABLE_LOCK         Z                       0
TP1                              15 USERSPACE1                     - TABLE_LOCK         Z                       0
TP1                              16 USERSPACE1                     - TABLE_LOCK         Z                       0

  26 Satz/Sätze ausgewählt.
```

In diesem Beispiel werden der Sperrobjekttyp TABLE_LOCK (Tabellensperre) und die Datenpartitions-ID (DATA_PARTITION_ID) mit dem Wert -1 verwendet, um den Zugriff auf die partitionierte Tabelle TP1 und die gleichzeitige Verwendbarkeit dieser Tabelle zu steuern. Die Sperrobjekte des Typs TABLE_PART_LOCK dienen zur Steuerung des größtmöglichen Zugriffs auf jede Datenpartition und ihrer größtmöglichen gemeinsamen Nutzbarkeit.

Darüber hinaus werden in dieser Ausgabe weitere Sperrobjekte des Typs TABLE_LOCK erfasst (mit TAB_FILE_ID 4 bis 16), die keinen Wert für DATA_PARTITION_ID haben. Eine Sperre dieses Typs, bei der ein Objekt mit einer TAB_FILE_ID und einem TBSP_NAME einer Datenpartition oder einem Index für die partitio-

nierte Tabelle entsprechen, kann verwendet werden, um den gleichzeitigen Zugriff mit dem Online-Backup-Dienstprogramm zu steuern.

Sperrenumwandlung

Die Änderung des Modus einer Sperre, die bereits aktiviert ist, wird als *Sperrenumwandlung* 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. Ein Prozess kann immer nur eine Sperre für ein Datenobjekt gleichzeitig aktiviert haben, obwohl er eine Sperre für dasselbe Datenobjekt mehrfach indirekt durch eine Abfrage anfordern kann.

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

IX- und S-Sperren sind Spezialfälle im Hinblick auf die Sperrenumwandlung. Keiner der Modi wird als stärker einschränkend angesehen als der jeweils andere. Wenn eine Sperre des einen dieser Modi aktiv ist und eine Sperre des anderen angefordert wird, erfolgt eine Sperrenumwandlung in eine SIX-Sperre (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. Die Systemmonitorelemente **lock_current_mode** und **lock_mode** können Informationen zu Sperrenumwandlungen bereitstellen, die in Ihrer Datenbank stattfinden.

Wartestatus und Zeitlimitüberschreitungen für Sperren

Das Erkennen von Überschreitungen der Sperrzeit ist eine Datenbankmanagerfunktion, die verhindert, dass Anwendungen unendlich lange auf die Freigabe einer Sperre 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 Datenbankkonfigurationsparameter **locktimeout** auf die maximale Zeitdauer, die eine beliebige Anwendung auf eine Sperre warten müssen sollte.

Durch die Einstellung dieses Parameters können globale Deadlocks besser vermieden werden, insbesondere in Anwendungen mit DUOWs (Distributed Units of Work, verteilte Arbeitseinheiten). Wenn die Zeit, während deren eine Sperrenanforderung ansteht, länger ist als die durch den Wert des Parameters **locktimeout** defi-

nierte Zeit, wird ein Fehler an die anfordernde Anwendung zurückgegeben und die zugehörige Transaktion mit ROLLBACK rückgängig gemacht. Beispiel: Wenn Anwendung APPL1 versucht, eine Sperre zu erhalten, die bereits für Anwendung APPL2 aktiv ist, empfängt Anwendung APPL1 den SQLCODE-Wert -911 (SQLSTATE-Wert 40001) mit dem Ursachencode 68, wenn das Zeitlimit abläuft. Der Standardwert für den Parameter **locktimeout** ist -1, d. h., dass die Erkennung von Überschreitungen der Sperrzeit ausgeschaltet ist.

Für Tabellen-, Zeilen-, Datenpartitions- und MDC-Blocksperren (MDC, mehrdimensionales Clustering) kann eine Anwendung den Wert des Parameters **locktimeout** überschreiben, indem sie den Wert des Sonderregisters CURRENT LOCK TIMEOUT ändert.

Wenn eine Berichtsdatei über Überschreitungen der Sperrzeit generiert werden soll, setzen Sie die Registrierdatenbankvariable **DB2_CAPTURE_LOCKTIMEOUT** auf den Wert ON. Der Bericht zu Zeitüberschreitungen enthält Informationen zu den Hauptanwendungen, die an Sperrenkonflikten beteiligt waren, die zu Sperrzeitüberschreitungen führten. Darüber hinaus enthält er Details über die Sperren, wie zum Beispiel den Namen und den Typ der Sperren, die Zeilen-ID, die Tabellenbereichs-ID und die Tabellen-ID. Dabei ist zu beachten, dass diese Variable veraltet ist und in einem zukünftigen Release möglicherweise entfernt wird, da neue Methoden zur Erfassung von Ereignissen durch Überschreitung der Sperrzeit unter Verwendung der Anweisung CREATE EVENT MONITOR FOR LOCKING zur Verfügung stehen.

Wenn Sie mehr Informationen zu Zeitüberschreitungen bei Sperrenanforderungen in den **db2diag**-Protokolldateien protokollieren wollen, setzen Sie den Konfigurationsparameter **diaglevel** des Datenbankmanagers auf den Wert 4. Die protokollierten Informationen geben den Namen des gesperrten Objekts, 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 der Parameter **diaglevel** auf den Wert 4 gesetzt ist.

Zusätzliche Informationen über Wartezeiten für Sperren und Überschreitungen der Sperrzeit können Sie mithilfe der Systemmonitorelemente zum Wartestatus von Sperren ('lock_wait') oder über den Diagnoseanzeiger **db.apps_waiting_locks** abrufen.

Angabe einer Strategie für den Modus 'Wartestatus für Sperre'

Eine Sitzung kann eine Strategie für den Modus 'Wartestatus für Sperre' angeben, die verwendet wird, wenn für die Sitzung eine Sperre erforderlich ist, die nicht unverzüglich aktiviert werden kann.

Die Strategie gibt an, ob die Sitzung folgende Aktionen ausführt:

- Zurückgeben eines SQLCODE- und SQLSTATE-Werts, wenn keine Sperre abgerufen werden kann
- Unbegrenztes 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 Anweisung SET CURRENT LOCK TIMEOUT angegeben, durch die der Wert des Sonderregisters CURRENT LOCK TIMEOUT geändert wird. Dieses Sonderregister gibt die Anzahl

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 dazu führen, dass sich Anwendungen gegenseitig blockieren. Dies geschieht, wenn eine Anwendung darauf warten muss, dass eine andere Anwendung die Sperre freigibt. Strategien für den Umgang mit den Auswirkungen solcher Blockierungen stellen in der Regel einen Mechanismus bereit, um die annehmbare Höchstdauer der Blockierung anzugeben. Dies ist die Dauer, die eine Anwendung wartet, bevor sie ohne Sperre zurückkehrt. Zuvor 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 von der Strategie für den Modus 'Wartestatus für Sperre' betroffen werden, sind Zeilen- und Tabellensperren sowie Sperren für Indexschlüssel und MDC-Blöcke (MDC, mehrdimensionales Clustering).

Deadlocks

Ein Deadlock entsteht, 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 ihre Ausführung fortsetzen kann.

Beispiel: In Abb. 23 auf Seite 219 werden zwei Anwendungen gleichzeitig ausgeführt: Anwendung A und Anwendung B. Die erste Transaktion von Anwendung A ist die Aktualisierung der ersten Zeile von Tabelle 1. Die zweite Transaktion 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 T1 sperrt Anwendung A die erste Zeile in Tabelle 1. Gleichzeitig sperrt Anwendung B die zweite Zeile in Tabelle 2. Zu einem Zeitpunkt T2 fordert Anwendung A eine Sperre für die zweite Zeile in Tabelle 2 an, während Anwendung B gleichzeitig versucht, die erste Zeile in Tabelle 1 zu sperren. Da Anwendung A ihre Sperre für die erste Zeile von Tabelle 1 erst freigibt, wenn sie eine Aktualisierung an der zweiten Zeile in Tabelle 2 ausführen kann, und Anwendung B ihre Sperre für die zweite Zeile in Tabelle 2 erst freigibt, wenn sie eine Aktualisierung an der ersten Zeile in Tabelle 1 ausführen kann, kommt es zu einem Deadlock. Die Anwendungen warten, bis eine von ihnen ihre Sperre der Daten freigibt.

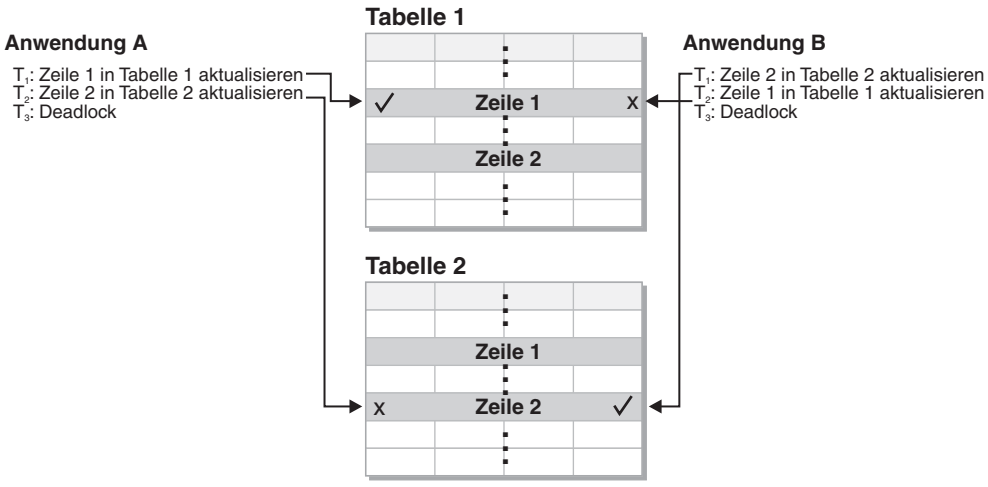


Abbildung 23. Deadlock zwischen Anwendungen

Da Anwendungen Sperren für Daten, die sie benötigen, nicht von sich aus freigeben, ist ein Detektorprozess erforderlich, der Deadlocks auflöst. Der Deadlock-Detektor überwacht Informationen zu Agenten, die auf Sperren warten, und wird in Intervallen aktiv, die durch den Datenbankkonfigurationsparameter **dlchktime** angegeben werden.

Wenn der Deadlock-Detektor einen Deadlock ermittelt, definiert er einen der Prozesse im Deadlock nach dem Zufallsprinzip als *ausgewählten Prozess*, für den ein Rollback durchgeführt werden muss. Der ausgewählte Prozess wird aktiviert und gibt den SQLCODE -911 (SQLSTATE 40001) mit Ursachencode 2 an die aufrufende Anwendung zurück. Der Datenbankmanager macht nicht festgeschriebene Transaktionen aus dem ausgewählten Prozess automatisch rückgängig (ROLLBACK). Wenn die Rollback-Operation beendet ist, werden Sperren, die zu dem ausgewählten Prozess gehörten, freigegeben und die anderen am Deadlock beteiligten Prozesse können fortfahren.

Zur Gewährleistung einer guten Leistung müssen Sie einen geeigneten Wert für den Konfigurationsparameter **dlchktime** auswählen. Ein zu kurzes Intervall verursacht unnötigen Systemaufwand, ein zu langes Intervall lässt zu, dass Deadlocks eine Weile bestehen bleiben.

In einer Umgebung mit partitionierten Datenbanken wird der Wert für den Konfigurationsparameter **dlchktime** nur in der Katalogdatenbankpartition angewendet. Wenn eine hohe Zahl Deadlocks auftritt, erhöhen Sie den Wert des Parameters **dlchktime**, um den Wartezeiten für Sperren und Übertragungen Rechnung zu tragen.

Gehen Sie wie folgt vor, um Deadlocks zu vermeiden, wenn Anwendungen Daten lesen, die sie nachfolgend zu aktualisieren beabsichtigen:

- Verwenden der Klausel FOR UPDATE bei der Ausführung einer SELECT-Anweisung. Diese Klausel stellt sicher, dass eine U-Sperre aktiviert wird, wenn ein Prozess versucht, Daten zu lesen, und sie lässt Zeilenblockung nicht zu.
- Verwenden der Klauseln WITH RR oder WITH RS und USE AND KEEP UPDATE LOCKS in Abfragen. Diese Klauseln stellen sicher, dass eine U-Sperre aktiviert wird, wenn ein Prozess versucht, Daten zu lesen, und sie lassen Zeilenblockung nicht zu.

In einem System föderierter Datenbanken ist es möglich, dass die von einer Anwendung angeforderten Daten aufgrund eines Deadlocks an einer Datenquelle nicht verfügbar sind. In diesem Fall ist der DB2-Server von den Einrichtungen zur Behandlung von Deadlocks an der Datenquelle abhängig. Wenn Deadlocks in mehreren Datenquellen auftreten, ist der DB2-Server für die Auflösung der Deadlocks auf die Zeitlimitmechanismen der Datenquellen angewiesen.

Wenn Sie mehr Informationen zu Deadlocks protokollieren wollen, setzen Sie den Konfigurationsparameter **diaglevel** des Datenbankmanagers auf den Wert 4. Die protokollierten Informationen geben den Namen des gesperrten Objekts, den Sperrmodus und die Anwendung an, die die Sperre aktiviert hat. Der Name der aktuellen dynamischen SQL- und XQuery-Anweisung oder des statischen Pakets kann ebenfalls protokolliert werden.

Abfrageoptimierung

Die Abfrageoptimierung ist einer der Faktoren, die sich auf die Anwendungsleistung auswirken. Lesen Sie die Informationen zu den verschiedenen Aspekten der Abfrageoptimierung in diesem Abschnitt, die Ihnen helfen können, die Leistung von Datenbankanwendungen zu maximieren.

Der SQL- und XQuery-Compilerprozess

Der SQL- und XQuery-Compiler führt mehrere Schritte aus, um einen Zugriffsplan zu erstellen, der ausgeführt werden kann.

Das *Abfragediagrammmodell* ist eine interne, im Speicher befindliche Datenbank, welche die Abfrage im Verlauf der in Abb. 24 auf Seite 221 gezeigten und nachfolgend beschriebenen Schritte darstellt. Beachten Sie, dass einige Schritte nur für Abfragen stattfinden, die in einer föderierten Datenbank ausgeführt werden.

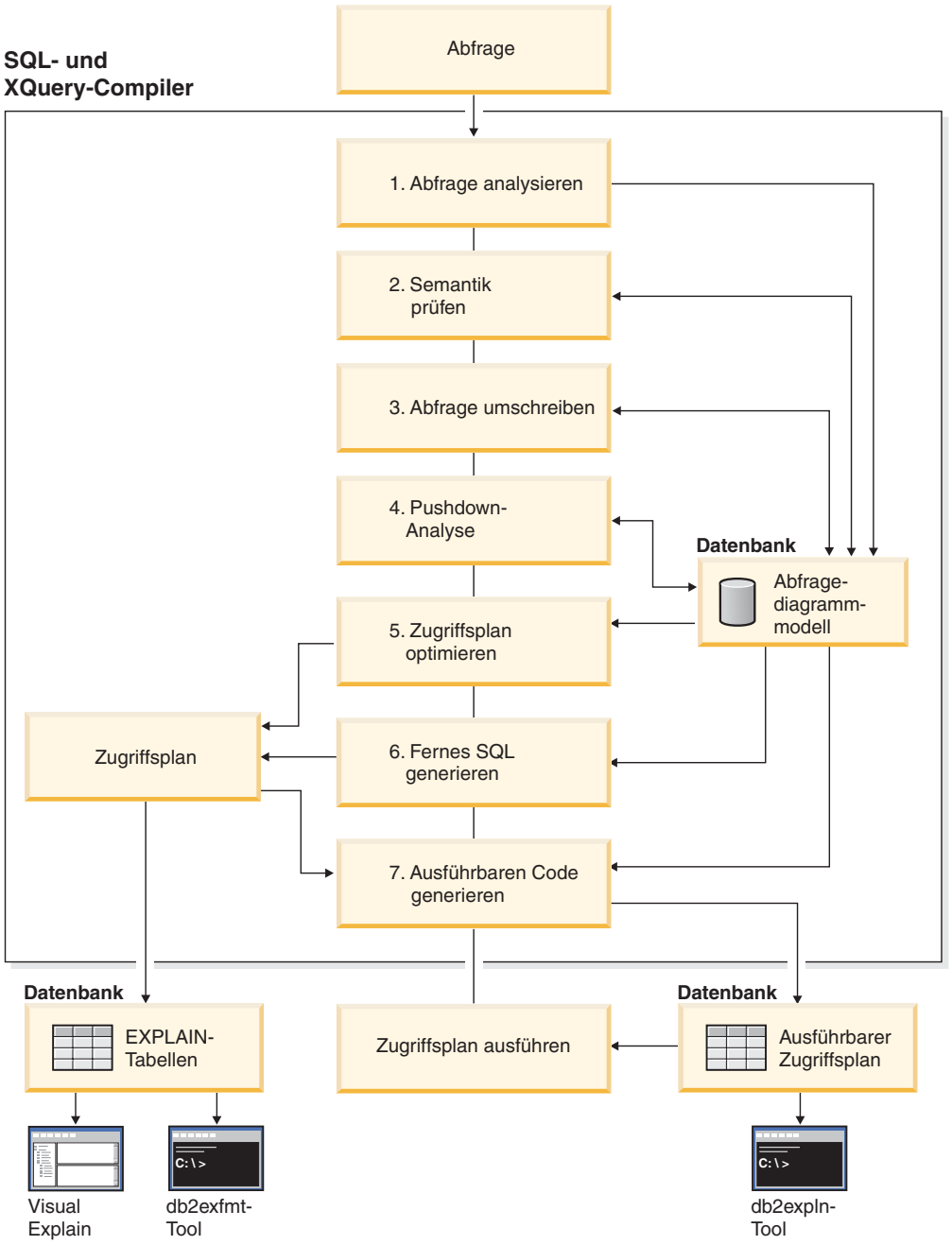


Abbildung 24. Vom SQL- und XQuery-Compiler ausgeführte Schritte

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 einen 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 Beispiel 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 für die 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. Anschließend speichert er das Ergebnis im Abfragediagrammmodell.

   Zum Beispiel kann der Compiler ein Vergleichselement verschieben und somit die Ebene ändern, auf der es angewendet wird, um dadurch potenziell die Abfrageleistung zu erhöhen. Diese Art der Verschiebung einer Operation wird als *allgemeine Vergleichselementverschiebung* (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 in 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 (nur föderierte 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 stellt eine Erweiterung zur allgemeinen Vergleichselementverschiebung dar.

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 für jeden der Pläne mithilfe der Statistiken für Tabellen, Indizes, Spalten und Funktionen ab. Anschließend wählt es den Plan mit dem geringsten geschätzten Ausführungsaufwand aus. Das Optimierungsprogramm verwendet das Abfragediagrammmodel, 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 einen anderen Typ 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 (DMS, Data Management Services) verschoben werden kann.

   Das Optimierungsprogramm berücksichtigt auch, ob es Pufferpools verschiedener Größen gibt, wenn es die Auswahl der Seitengröße festlegt. Es berücksichtigt, ob die Datenbank partitioniert ist oder ob eine abfrageinterne Parallelität in einer symmetrischen Multiprozessorumgebung (SMP-Umgebung) eine Option

ist. Diese Informationen werden vom Optimierungsprogramm zur Auswahl des am besten geeigneten Zugriffsplans für die Abfrage verwendet.

Die Ausgabe dieses Schritts ist ein Zugriffsplan. Detaillierte Informationen zu diesem Zugriffsplan werden in den EXPLAIN-Tabellen erfasst. Die Informationen, die zur Generierung eines Zugriffsplans verwendet werden, können durch eine EXPLAIN-Momentaufnahme erfasst werden.

6. Fernes SQL generieren (nur föderierte Datenbanken)

   Der endgültige Plan, der vom Optimierungsprogramm gewählt wird, kann aus einer Reihe von Schritten bestehen, die an einer fernen Datenquelle ausgeführt werden. Der Schritt zur Generierung von fernem SQL erstellt auf der Basis der SQL-Version der jeweiligen Datenquelle eine effiziente SQL-Anweisung für Operationen, die von den einzelnen Datenquellen auszuführen sind.

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 nur einmal berechnet werden müssen. Dieser Typ von Optimierung ist für Codepagekonvertierungen sowie bei der Verwendung von Hostvariablen möglich.

   Damit eine Abfrageoptimierung bzw. Reoptimierung für statische oder dynamische SQL- bzw. XQuery-Anweisungen, die Hostvariablen, Sonderregister oder Parametermarken enthalten, ausgeführt werden kann, binden Sie das Paket mit der Bindeoption REOPT. Der Zugriffspfad für eine Anweisung, die zu einem solchen Paket gehört und die Hostvariablen, Sonderregister oder Parametermarken enthält, wird 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 ein Paket ausgeführt wird, verwendet der Datenbankmanager die im Systemkatalog gespeicherten Informationen, um festzulegen, wie auf die Daten zugegriffen und Ergebnisse für eine Abfrage bereitgestellt werden sollen. Diese Informationen werden vom Tool **db2expln** verwendet.

Anmerkung: Führen Sie den Befehl **RUNSTATS** in geeigneten Intervallen für Tabellen aus, die häufig geändert werden. Das Optimierungsprogramm benötigt aktuelle Statistikinformationen zu 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 der Befehl **RUNSTATS** nicht ausgeführt wird oder das Optimierungsprogramm annimmt, dass dieser Befehl für leere oder fast leere Tabellen ausgeführt wurde, kann das Optimierungsprogramm Standardwerte verwenden oder versuchen, bestimmte Statistikdaten auf der Basis der Anzahl von Dateiseiten abzuleiten, die zum Speichern der Tabelle auf dem Plattenspeicher verwendet werden. Weitere Informationen hierzu enthält der Abschnitt „Automatische Statistikerfassung".

Methoden zum Umschreiben von Abfragen und Beispiele

Während der Phase des Umschreibens der Abfrage setzt der Abfragecompiler SQL- und XQuery-Anweisungen in Formen um, die leichter optimiert werden können. Dies kann die möglichen Zugriffspläne verbessern. 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 mithilfe der EXPLAIN-Funktion oder über Visual Explain anzeigen.

Abfragen können durch die folgenden Methoden umgeschrieben werden:

- Zusammenfügen von Operationen

  Zur Generierung einer 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 einer Abfrage mit der kleinstmöglichen Anzahl an Operationen und Vergleichselementen schreibt der Compiler die Abfrage 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 manchen Fällen kann die Operation DISTINCT vollständig entfernt werden.

  – Beispiel - Allgemeine Vergleichselementverschiebung (Pushdown)

    Während des Umschreibens von Abfragen kann das Optimierungsprogramm die Reihenfolge der Anwendung der Vergleichselemente ändern, sodass die Vergleichselemente, die die Auswahl in größerem Maße einschränken, zum frühstmöglichen 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.

- Beispiel - Umsetzen 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 umgesetzt werden können:

– Beispiel - Hinzufügen implizierter Vergleichselemente

  Während des Umschreibens können Vergleichselemente zu einer 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.

Beispiel für das Umschreiben durch den Compiler: 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.

Nehmen Sie zum Beispiel an, Sie haben Zugriff auf die beiden folgenden Sichten, die auf der Tabelle EMPLOYEE basieren. Die eine zeigt die Mitarbeiter, die einen hohen Ausbildungsgrad (EDLEVEL) besitzen, während die andere die Mitarbeiter zeigt, deren Gehalt (SALARY) über 35.000 Dollar liegt:

```
create view emp_education (empno, firstnme, lastname, edlevel) as
  select empno, firstnme, lastname, edlevel
    from employee
    where edlevel > 17

create view emp_salaries (empno, firstname, lastname, salary) as
  select empno, firstnme, lastname, salary
    from employee
    where salary > 35000
```

Die folgende benutzerdefinierte Abfrage listet die Mitarbeiter auf, die einen hohen Ausbildungsgrad besitzen und über 35.000 Dollar pro Jahr verdienen:

```
select e1.empno, e1.firstnme, 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.firstnme, 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 zusammengefügten Sichten dieselbe Basistabelle verwenden.

Beispiel - Umsetzungen von Unterabfragen in Joins

Angenommen, der SQL- und XQuery-Compiler erhält eine Abfrage mit einer Unterabfrage wie die folgende:

```
select empno, firstnme, 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, firstnme, 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

Abfragen können gelegentlich unnötige Joins enthalten.

```
select e1.empno, e1.firstnme, e1.lastname, e1.edlevel, e2.salary
  from employee e1,
       employee e2
 where e1.empno = e2.empno
   and e1.edlevel > 17
   and e2.salary  > 35000
```

Der SQL- und XQuery-Compiler kann den Join eliminieren und die Abfrage wie folgt vereinfachen:

```
select empno, firstnme, lastname, edlevel, salary
  from employee
  where
    edlevel > 17 and
    salary  > 35000
```

Im folgenden Beispiel wird davon ausgegangen, dass zwischen den Tabellen EMPLOYEE und DEPARTMENT eine referenzielle Integritätsbedingung über die Abteilung (WORKDEPT/DEPNO) vorhanden ist. Zuerst wird eine Sicht erstellt.

```
create view peplview as
  select firstnme, 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 wie folgt umgeschrieben:

```
select lastname, salary
  from employee
  where workdept not null
```

Beachten Sie in diesem Fall, dass Sie die Abfrage eventuell nicht umschreiben können, selbst wenn Sie wissen, dass dies möglich ist, weil Sie keinen Zugriff auf die zugrunde liegenden Tabellen haben. Sie haben möglicherweise nur Zugriff auf die (oben gezeigte) Sicht. Daher muss dieser Typ von Optimierung vom Datenbankmanager ausgeführt werden.

Redundanz in Joins mit referenzieller Integrität tritt wahrscheinlich unter folgenden Bedingungen auf:

- Sichten sind mit Joins definiert.
- Abfragen werden automatisch generiert.

Beispiel - Gemeinsame Spaltenberechnungen

Bei Verwendung mehrerer Funktionen in einer Abfrage können zahlreiche Berechnungen entstehen, die zeitaufwendig sind. Durch Verringern der Anzahl von erforderlichen Berechnungen lässt sich der Plan verbessern. Der Compiler nimmt eine Abfrage entgegen, die mehrere Funktionen wie die folgenden enthält:

```
select sum(salary+bonus+comm) as osum,
    avg(salary+bonus+comm) as oavg,
    count(*) as ocount
  from employee
```

Diese Abfrage wird wie folgt umgeformt:

```
select osum, osum/ocount ocount
  from (
    select sum(salary+bonus+comm) as osum,
        count(*) as ocount
      from employee
  ) as shared_agg
```

Durch dieses Umschreiben wird die Anzahl der erforderlichen Summierungen und Zählungen halbiert.

Beispiel für das Umschreiben durch den Compiler: Eliminierung von DISTINCT:

Während des Umschreibens einer Abfrage kann das Optimierungsprogramm den Zeitpunkt verschieben, zu dem die DISTINCT-Operation ausgeführt wird, um den Aufwand für diese Operation zu verringern. In manchen Fällen kann die Operation DISTINCT vollständig entfernt werden.

Betrachten Sie zum Beispiel die folgende Abfrage für den Fall, dass die Spalte EMPNO der Tabelle EMPLOYEE als Primärschlüssel definiert ist:

```
select distinct empno, firstnme, lastname
  from employee
```

Diese Abfrage kann so umgeschrieben werden, dass die Klausel DISTINCT entfernt wird:

```
select empno, firstnme, lastname
  from employee
```

Da in diesem Beispiel der Primärschlüssel ausgewählt wird, weiß der Compiler, dass jede zurückgegebene Zeile bereits eindeutig ist. In diesem Fall wird das Schlüsselwort DISTINCT nicht benötigt. Wenn die Abfrage nicht umgeschrieben wird, muss das Optimierungsprogramm einen Plan mit den nötigen Verarbeitungsschritten (z. B. einer Sortierung) generieren, um sicherzustellen, dass die Spaltenwerte eindeutig sind.

Beispiel - Allgemeine Vergleichselementverschiebung (Pushdown)

Durch das Ändern der Ebene, auf der Vergleichselemente normalerweise angewendet werden, lässt sich eventuell eine Leistungsverbesserung erreichen. Zum Beispiel wird in der folgenden Sicht eine Liste aller Mitarbeiter der Abteilung D11 zusammengestellt:

```
create view d11_employee
  (empno, firstnme, lastname, phoneno, salary, bonus, comm) as
    select empno, firstnme, lastname, phoneno, salary, bonus, comm
      from employee
      where workdept = 'D11'
```

Die folgende Abfrage auf diese Sicht ist nicht so effizient, wie sie sein könnte:

```
select firstnme, phoneno
  from d11_employee
  where lastname = 'BROWN'
```

Beim Umschreiben der Abfrage verschiebt der Compiler das Vergleichselement `lastname = 'BROWN'` in die Sicht D11_EMPLOYEE. Dadurch kann das Vergleichselement früher und möglicherweise effektiver angewendet werden. Die tatsächliche Abfrage, die in diesem Beispiel möglicherweise ausgeführt wird, sieht wie folgt aus:

```
select firstnme, phoneno
  from employee
  where
    lastname = 'BROWN' and
    workdept = 'D11'
```

Der Pushdown von Vergleichselementen ist nicht auf Sichten beschränkt. Beispiele für andere Situationen, in denen Vergleichselemente verschoben werden können, sind Klauseln UNION und GROUP BY sowie abgeleitete Tabellen (verschachtelte Tabellenausdrücke oder allgemeine Tabellenausdrücke).

Beispiel - Dekorrelierung

In einer Umgebung mit partitionierten Datenbanken kann der Compiler die folgende Abfrage umschreiben, mit der alle Mitarbeiter ermittelt werden sollen, die an Programmierungsprojekten 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, ist die Übermittlung jedes Projekts an jede Datenbankpartition per Broadcast möglich. Außerdem müsste die Unterabfrage viele Male ausgewertet werden.

Der 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, um sicherzustellen, 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%')
```

- Join der Liste DIST_PROJS mit der Tabelle EMPLOYEE um die durchschnittliche Vergütung 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 umgeschriebene Abfrage sieht wie folgt aus:

```
select p.projno, e.empno, e.lastname, e.firstname,
    e.salary+e.bonus+e.comm as compensation
  from project p, employee e, avg_per_prog a
  where
    p.empno = e.empno and
    p.projname like '%PROGRAMMING%' and
    p.projno = a.projno and
    e.salary+e.bonus+e.comm < a.avg_comp
```

Diese Abfrage berechnet die durchschnittliche Vergütung (`avg_comp`) pro Projekt (`avg_per_proj`). Das Ergebnis kann anschließend per Broadcast an alle Datenbankpartitionen gesendet werden, in denen die Tabelle EMPLOYEE enthalten ist.

Beispiel für das Umschreiben durch den Compiler: Pushdown von Vergleichselementen für kombinierte SQL/XQuery-Anweisungen:

Ein grundlegendes Verfahren zur Optimierung von relationalen SQL-Abfragen besteht darin, Vergleichselemente in der Klausel WHERE eines umschießenden Abfrageblocks in einen umschlossenen Abfrageblock auf niedrigerer Ebene (z. B. eine Sicht) zu verschieben, um so eine möglichst frühe Datenfilterung und potenziell eine bessere Indexnutzung zu erreichen.

Dies ist für Umgebungen mit partitionierten Datenbanken noch wichtiger, da eine Filterung in einem frühen Stadium potenziell das Volumen der Daten verringert, die zwischen Datenbankpartitionen übertragen werden müssen.

Ähnliche Verfahren können dazu verwendet werden, Vergleichselemente oder XPath-Filter innerhalb einer XQuery-Abfrage zu verschieben. Die Grundstrategie besteht immer darin, Filterausdrücke so nah wie möglich an die Datenquelle zu verschieben. Dieses Optimierungsverfahren wird als *Pushdown von Vergleichselementen* in SQL und als *Extraktionspushdown* (bei Filtern und XPath-Extraktionen) in XQuery bezeichnet.

Aufgrund der Unterschiede zwischen den beiden Datenmodellen, die von SQL und XQuery verwendet werden, müssen Vergleichselemente, Filter oder Extraktionen zwischen den beiden Sprachen verschoben werden. Regeln für die Datenkonvertierung und Datentypumwandlung müssen berücksichtigt werden, wenn ein SQL-Vergleichselement in einen semantisch äquivalenten Filter transformiert und in eine XPath-Extraktion verschoben wird. Die folgenden Beispiele behandeln das Pushdownverfahren zum Verschieben von relationalen Vergleichselementen in XQuery-Abfrageblöcke.

Betrachten Sie die folgenden beiden XML-Dokumente mit Kundeninformationen:

```
Dokument 1                  Dokument 2

<customer>                  <customer>
  <name>John</name>           <name>Michael</name>
  <lastname>Doe</lastname>    <lastname>Miller </lastname>
  <date_of_birth>             <date_of_birth>
   1976-10-10                 1975-01-01
  </date_of_birth>            </date_of_birth>
  <address>                   <address>
    <zip>95141.0</zip>          <zip>95142.0</zip>
  </address>                  </address>
<volume>80000.0</volume>    <volume>100000.00</volume>
</customer>                 </customer>
<customer>                  <customer>
  <name>Jane</name>           <name>Michaela</name>
  <lastname>Doe</lastname>    <lastname>Miller</lastname>
  <date_of_birth>             <date_of_birth>
    1975-01-01                 1980-12-23
  </date_of_birth>            </date_of_birth>
  <address>                   <address>
    <zip>95141.4</zip>          <zip>95140.5</zip>
  </address>                  </address>
<volume>50000.00</volume>   <volume>100000</volume>
</customer>                 </customer>
```

Beispiel - Verschieben von INTEGER-Vergleichselementen

Betrachten Sie die folgende Abfrage:

```
select temp.name, temp.zip
  from xmltable('db2-fn:xmlcolumn("T.XMLDOC")'
    columns name varchar(20) path 'customer/name',
    zip integer path 'customer/zip'
    ) as temp
  where zip = 95141
```

Zur Verwendung möglicher Indizes für T.XMLDOC und zum Herausfiltern unerwünschter Personendaten in einem frühen Stadium wird das Vergleichselement `zip = 95141` intern in den folgenden äquivalenten XPATH-Filterausdruck konvertiert:

```
T.XMLCOL/customer/zip[. >= 95141.0 and . < 95142.0]
```

Da Schemainformationen für XML-Fragmente vom Compiler nicht verwendet werden, kann nicht angenommen werden, dass das Element 'ZIP' nur ganze Zahlen (INTEGER-Werte) enthält. Es ist möglich, dass andere numerische Werte mit Nachkommastellen und einem entsprechenden XML-DOUBLE-Index für diese bestimmte XPath-Extraktion vorhanden sind. Die XML2SQL-Umsetzung würde diese Konvertierung in der Weise behandeln, dass die Nachkommastellen abgeschnitten würden, bevor der Wert in INTEGER umgesetzt würde. Dieses Verhalten muss in der Pushdownprozedur berücksichtigt werden und das Vergleichselement muss so geändert werden, dass es semantisch korrekt bleibt.

Beispiel - Verschieben von VARCHAR(*n*)-Vergleichselementen

Betrachten Sie die folgende Abfrage:

```
select temp.name, temp.lastname
  from xmltable('db2-fn:xmlcolumn("T.XMLDOC")'
    columns name varchar(20) path 'customer/name',
    lastname varchar(20) path 'customer/lastname'
    ) as temp
  where lastname = 'Miller'
```

Zur Verwendung möglicher Indizes für T.XMLDOC und zum Herausfiltern unerwünschter Personendaten in einem frühen Stadium wird das Vergleichselement `lastname = 'Miller'` intern in den äquivalenten XPATH-Filterausdruck konvertiert:

```
T.XMLCOL/customer/lastname[. > rtrim("Miller") and . < blank_padd("Miller",
max(20,length("Miller"))]
```

Folgende Leerzeichen werden bei SQL anders behandelt als bei XPath und XQuery. Das ursprüngliche SQL-Vergleichselement unterscheidet nicht zwischen den beiden Kunden, deren Nachname „Miller" ist, obwohl einer der Nachnamen (von Michael) ein folgendes Leerzeichen enthält. Infolgedessen werden beide Kunden zurückgegeben, was nicht der Fall wäre, wenn ein unverändertes Vergleichselement verschoben würde.

Die Lösung ist, das Vergleichselement in einen Bereichsfilter umzuwandeln.

- Die erste Grenze wird durch Abschneiden aller folgenden Leerzeichen im Vergleichswert mithilfe der Funktion RTRIM() erstellt.
- Die zweite Grenze muss größer oder gleich allen möglichen Zeichenfolgen mit „Miller" sein, die folgende Leerzeichen enthalten. Die ursprüngliche Zeichenfolge wird mit Leerzeichen bis zur maximalen Spaltenlänge bzw., falls diese länger ist, bis zur Länge der Vergleichszeichenfolge aufgefüllt.

Beispiel - Verschieben von DECIMAL(*x*,*y*)-Vergleichselementen

Betrachten Sie die folgende Abfrage:

```
select temp.name, temp.volume
  from xmltable('db2-fn:xmlcolumn("T.XMLDOC")'
    columns name varchar(20) path 'customer/name',
    volume decimal(10,2) path 'customer/volume'
    ) as temp
  where volume = 100000.00
```

Zur Verwendung möglicher DOUBLE-Indizes für T.XMLDOC und zum Herausfiltern unerwünschter Personendaten in einem frühen Stadium wird das Vergleichselement `volume = 100000.00` intern in den folgenden äquivalenten XPATH-Filterausdruck konvertiert:

```
T.XMLCOL/customer/volume[.=100000.00]
```

Das Vergleichselement braucht nicht in einen Bereichsfilter umgewandelt zu werden, weil Typumwandlungseinschränkungen festlegen, dass XML-Werte dieselbe Genauigkeit und Länge der Nachkommastellen wie der Ziel-SQL-Typ haben. Jeder Verstoß gegen diese Bedingung führt zu einem Fehler. Die Genauigkeit wird nicht verringert, wenn DOUBLE-Werte in DECIMAL(*x*,*y*) umgesetzt werden. Ein Auf-/Abrunden oder ein Abschneiden des Vergleichswerts ist nicht erforderlich.

Beispiel für das Umschreiben durch den Compiler: implizierte Vergleichselemente:

Während des Umschreibens können Vergleichselemente zu einer 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.

Die folgende Abfrage gibt eine Liste der Manager zurück, 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
```

Diese Abfrage kann mit dem folgenden implizierten Vergleichselement umgeschrieben werden, das als *Vergleichselement für transitiven Schluss* (engl. predicate for transitive closure) bezeichnet wird:

```
dept.mgrno = proj.respemp
```

Das Optimierungsprogramm kann nun zusätzliche Joins in Betracht ziehen, wenn es versucht, den günstigsten Zugriffsplan für die Abfrage auszuwählen.

In der Umschreibephase einer Abfrage werden zusätzliche lokale Vergleichselemente auf der Basis der Transitivität abgeleitet, die durch Gleichheitsvergleichselemente impliziert wird. Zum Beispiel gibt die folgende Abfrage die Namen der Abteilungen (deren Abteilungsnummer größer als E00 ist) und der Mitarbeiter zurück, die in diesen Abteilungen arbeiten.

```
select empno, lastname, firstname, deptno, deptname
  from employee emp, department dept
  where
    emp.workdept = dept.deptno and
    dept.deptno > 'E00'
```

Diese Abfrage kann mit dem folgenden implizierten Vergleichselement umgeschrieben werden:

```
emp.workdept > 'E00'
```

Dieses Umschreiben verringert die Anzahl der Zeilen, die durch einen Join verknüpft werden müssen.

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, führt die Verwendung eines IN-Vergleichselements anstelle der OR-Klausel zu einer effizienteren Verarbeitung der Abfrage:

```
select *
  from employee
  where deptno in ('D11', 'D21', 'E21')
```

In einigen Fällen kann der Datenbankmanager ein IN-Vergleichselement in eine Gruppe von Klauseln OR umwandeln, sodass ein logisches Verknüpfen von Indizes über OR (Index ORing) durchgeführt werden kann.

Vergleichselementtypologie und Zugriffspläne

Ein *Vergleichselement* ist ein Element einer Suchbedingung, das eine Vergleichsoperation definiert oder impliziert. Vergleichselemente, die in der Regel in der WHE-

RE-Klausel einer Abfrage angewendet werden, dienen zur Verringerung des Umfangs der Ergebnismenge, die von der Abfrage zurückgegeben wird.

Vergleichselemente lassen sich je nachdem, wie und wann sie im Auswertungsprozess verwendet werden, in vier Kategorien einteilen. Diese Kategorien sind nachfolgend in der Reihenfolge von der höchsten bis zur niedrigsten Leistung aufgeführt:

1. Bereichsbegrenzende Vergleichselemente
2. Bei Indexsuchen als Suchargument verwendbare Vergleichselemente (Index SARGable)
3. Bei Datensuchen als Suchargument verwendbare Vergleichselemente (Data SARGable)
4. Restvergleichselemente

Die englische Bezeichnung *SARGable* ist aus dem Begriff *s*earch *arg*ument abgeleitet.

In Tabelle 52 werden die Merkmale dieser Vergleichselementkategorien zusammengefasst.

Tabelle 52. Zusammenfassung der Merkmale der Vergleichselementkategorien

Merkmal	**Vergleichselementkategorie**			
	Bereichs-begrenzend	**Bei Indexsuchen als Suchargument verwendbar**	**Bei Datensuchen als Suchargument verwendbar**	**Rest-vergleichs-element**
Verringern der Index-Ein-/Ausgabe	Ja	Nein	Nein	Nein
Verringern der Datenseitenein-/ausgabe	Ja	Ja	Nein	Nein
Verringern der Anzahl von Zeilen, die intern übergeben werden	Ja	Ja	Ja	Nein
Verringern der Anzahl von Zeilen, die den Bedingungen entsprechen	Ja	Ja	Ja	Ja

Bereichsbegrenzende und bei Indexsuchen als Suchargument 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 mithilfe 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 and dept = :hv2`) sind bereichsbegrenzende Vergleichselemente, während `years > :hv5` ein bei Indexsuchen als Suchargument verwendbares Vergleichselement ist.

Zur Auswertung dieser Vergleichselemente verwendet das Optimierungsprogramm Indexdaten, anstatt die Basistabelle zu lesen. Bei Indexsuchen als Suchargumente verwendbare Vergleichselemente verringern die Anzahl von Zeilen, die aus der Tabelle gelesen werden müssen, sie beeinflussen jedoch nicht die Anzahl von Indexseiten, auf die zugegriffen wird.

Bei Datensuchen als Suchargument verwendbare Vergleichselemente (Data SARGable)

Vergleichselemente, die nicht vom Indexmanager ausgewertet werden können, sondern nur von den Data Management Services (DMS, 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 DMS die zur Auswertung des Vergleichselements benötigten Spalten sowie alle anderen Spalten ab, die für die SELECT-Liste benötigt werden, jedoch nicht aus dem Index abgerufen werden konnten.

Betrachten Sie zum Beispiel den folgenden Index, der für die Tabelle PROJECT definiert ist:

```
INDEX IX0:  PROJNO ASC
```

In der folgenden Abfrage ist `deptno = 'D11'` dementsprechend als ein bei Datensuchen verwendbares Vergleichselement zu betrachten:

```
select projno, projname, respemp
  from project
  where deptno = 'D11'
  order by projno
```

Restvergleichselemente

Restvergleichselemente (engl. Residual Predicates) sind im Hinblick auf den Ein-/Ausgabeaufwand ungünstiger als der Zugriff auf eine Tabelle. Solche Vergleichselemente können durch folgende Merkmale gekennzeichnet sein:

- 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 (RDS, Relational Data Services) ausgewertet.

Einige Vergleichselemente, die nur auf einen Index angewendet wurden, müssen erneut angewendet werden, wenn auf die Datenseite zugegriffen wird. Zum Beispiel wenden Zugriffspläne mit logischem Verknüpfen von Indizes über AND (Index ANDing) oder OR (Index ORing) die Vergleichselemente immer ein weiteres Mal als Restvergleichselemente an, wenn auf die Datenseite zugegriffen wird.

Compilerphasen für Abfragen auf föderierte Datenbanken

Pushdown-Analyse für föderierte Datenbanken:

Für Abfragen, die in föderierten Datenbanken ausgeführt werden sollen, führt das Optimierungsprogramm eine Pushdown-Analyse durch, um zu ermitteln, ob eine bestimmte Operation an einer fernen Datenquelle ausgeführt werden kann.

Eine solche Operation kann eine Funktion, zum Beispiel ein relationaler Operator, oder eine System- oder Benutzerfunktion sein. Sie kann auch ein SQL-Operator wie zum Beispiel ORDER BY oder GROUP BY sein.

Stellen Sie sicher, dass die lokalen Kataloginformationen regelmäßig aktualisiert werden, damit der DB2-Abfragecompiler Zugriff auf präzise Informationen über die SQL-Unterstützung an fernen Datenquellen hat. Verwenden Sie DDL-Anweisungen (DDL, Data Definition Language) für DB2 (z. B. CREATE FUNCTION MAPPING oder 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 den DB2-Server zwingen, die gesamte Tabelle von der fernen Datenquelle abzurufen und anschließend lokal über das Vergleichselement zu filtern. Netzbegrenzungen und eine große Tabelle könnten zu einer Beeinträchtigung der Leistung führen.

Operatoren, die nicht an die Datenquelle verschoben werden, können sich ebenfalls erheblich auf die Abfrageleistung auswirken. Wenn zum Beispiel ein Operator GROUP BY ferne Daten lokal zusammenfassen soll, muss der DB2-Server möglicherweise ebenfalls eine gesamte Tabelle von der fernen Datenquelle abrufen.

Betrachten Sie zum Beispiel einen Kurznamen N1, der auf eine Datenquellentabelle mit dem Namen EMPLOYEE an einer DB2 für z/OS-Datenquelle verweist. Die Tabelle besitzt 10.000 Zeilen. Eine der Spalten enthält die Familiennamen ('lastname') von Mitarbeitern und eine der Spalten enthält Gehälter ('salary'). Bei der Verarbeitung der folgenden Anweisung hat das Optimierungsprogramm mehrere Optionen, die davon abhängen, ob die lokale und die ferne Sortierfolge übereinstimmen:

```
select lastname, count(*)  from n1
  where
    lastname > 'B' and
    salary > 50000
  group by lastname
```

- Wenn die Sortierfolgen übereinstimmen, kann das Abfragevergleichselement wahrscheinlich an die DB2 für z/OS-Datenquelle verschoben werden (Pushdown). Eine Filterung und Gruppierung der Ergebnisse an der Datenquelle ist in der Regel effizienter, als die gesamte Tabelle zu kopieren und die Operationen lokal auszuführen. Bei dieser Abfrage können die Operationen für das Vergleichselement und für den Operator GROUP BY an der Datenquelle stattfinden.

- Wenn die Sortierfolgen nicht übereinstimmen, können beide Vergleichselemente nicht an der Datenquelle ausgewertet werden. Allerdings kann das Optimierungsprogramm entscheiden, das Vergleichselement `salary > 50000` an die Datenquelle zu verschieben. Der Vergleich des Wertebereichs muss trotzdem lokal 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 GROUP BY-Operation das günstigste Verfahren ist. Das Vergleichselement wird an der Datenquelle ausgewertet. 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 fernen Datenquellen auswertet. Zahlreiche Faktoren haben Einfluss darauf, ob eine Funktion oder ein SQL-Operator an einer fernen Datenquelle ausgewertet werden kann, wie zum Beispiel:

- 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 das DB2-Produkt eine sehr umfassende SQL-Version unterstützt. Der DB2-Datenserver kann das Fehlen von Funktionalität auf einem anderen Datenserver kompensieren, allerdings kann dies zur Folge haben, dass die Operation auf dem DB2-Server ausgeführt werden muss.

- SQL-Leistungsspektrum

  Jede Datenquelle unterstützt eine Variante der SQL-Version und verschiedene Funktionalitätsebenen. Die meisten Datenquellen unterstützen zum Beispiel 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 an der fernen Datenquelle gibt, muss der DB2-Server eine GROUP BY-Operation möglicherweise lokal ausführen.
- SQL-Einschränkungen

  Jede Datenquelle kann unterschiedliche SQL-Einschränkungen haben. Zum Beispiel erfordern einige Datenquellen, dass Parametermarken Werte an ferne SQL-Anweisungen binden. Daher müssen die Einschränkungen für Parametermarken überprüft werden, um sicherzustellen, dass die jeweilige Datenquelle solche Bindeverfahren unterstützen kann. Wenn der DB2-Server keine gute Methode zum Binden eines Werts für eine Funktion ermitteln kann, muss diese Funktion lokal ausgewertet werden.
- SQL-Begrenzungen

  Obwohl der DB2-Server möglicherweise größere ganzzahlige Werte (Integer) zulässt als ferne Datenquellen, können Werte, die die fernen Begrenzungen überschreiten, nicht in Anweisungen eingebettet werden, die an Datenquellen gesendet werden. Alle Funktionen bzw. Operatoren, die von solchen Werten betroffen sind, müssen lokal ausgewertet werden.
- Serverspezifische Faktoren

In diese Kategorie fallen verschiedene Faktoren. Wenn zum Beispiel Nullwerte an einer Datenquelle anders als vom DB2-Server sortiert würden, können ORDER BY-Operationen für einen Ausdruck, der Nullwerte enthalten kann, nicht fern ausgewertet werden.

- Sortierfolge

  Das Abrufen von Daten für lokale Sortierungen und Vergleiche setzt in der Regel die Leistung herab. 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. 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. Vergleiche können unerwartete Ergebnisse liefern, wenn Sie Anweisungen an eine Datenquelle übergeben, an der die Groß-/Kleinschreibung nicht unterschieden wird. Die Wertigkeiten der Zeichen „I" und „i" sind bei einer Datenquelle ohne Unterscheidung der Groß-/Kleinschreibung identisch. Der DB2-Server beachtet standardmäßig die Groß-/Kleinschreibung und weist diesen Zeichen unterschiedliche Wertigkeiten zu.

  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 z/OS Sortierungen, die durch Klauseln ORDER BY definiert werden, mithilfe einer Sortierfolge implementiert, die auf einer EBCDIC-Codepage (Extended Binary-Coded Decimal Interchange Code) basiert. Um den Server der föderierten Datenbank zum Abrufen von DB2 für z/OS-Daten zu verwenden, die durch Klauseln ORDER BY sortiert werden, konfigurieren Sie die föderierte Datenbank so, dass sie eine vordefinierte, auf der EBCDIC-Codepage basierende Sortierfolge verwendet.

  Wenn die Sortierfolgen der föderierten Datenbank und der Datenquelle voneinander abweichen, ruft der DB2-Server die Daten für die föderierte Datenbank ab. Da Benutzer Abfrageergebnisse nach der für den Server der föderierten Datenbank definierten Sortierfolge geordnet erwarten, stellt 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 Datenquellensicht, wenn Sie die Daten nach der Sortierfolge der Datenquelle geordnet abrufen müssen.

- Serveroptionen

  Verschiedene Serveroptionen können Einfluss auf die Pushdown-Möglichkeiten haben, wie zum Beispiel COLLATING_SEQUENCE, VARCHAR_NO_TRAILING_BLANKS und PUSHDOWN.

- Faktoren der DB2-Typzuordnung und -Funktionszuordnung

  Die lokalen Standardtypenzuordnungen auf dem DB2-Server sind dazu vorgesehen, genügend Pufferspeicherplatz für jeden Datentyp einer Datenquelle bereitzustellen, wodurch Datenverlust vermieden wird. Sie können die Typenzuordnung für eine bestimmte Datenquelle an die Anforderungen bestimmter Anwendungen anpassen. 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 der DB2-Server 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. Der Vergleichsoperator IS NULL ist ein Beispiel für diesen Fall. Die meisten Datenquellen unterstützen ihn, aber einige haben möglicherweise Einschränkungen, wie zum Beispiel, dass ein Spaltenname nur 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 den DB2-Server.

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 an einer 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 der Bereich einer Ganzzahl in DB2 die Werte von 2**31 bis (-2**31)-1 umfasst, was in etwa mit dem Typ NUMBER(9) äquivalent ist. In diesem Fall können Joins zwischen einer DB2-Spalte des Typs INTEGER und einer Oracle-Spalte des Typs INTEGER nicht an der DB2-Datenquelle stattfinden (wegen der kürzeren Joinspalte). Wenn jedoch der Wertebereich dieser Oracle-Spalte des Typs INTEGER in den DB2-Datentyp INTEGER passt, 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 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, die für solche Spalten ausgeführt werden. Der DB2-Server generiert möglicherweise eine andere, aber äquivalente Form eines Vergleichselements, das in einer SQL-Anweisung zu verwenden ist, die an eine Datenquelle gesendet wird. Sie stellen vielleicht fest, 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 Tabelle 53 auf Seite 239 werden diese Optionen beschrieben.

Tabelle 53. Spaltenoptionen und zugehörige Einstellungen

Option	Gültige Einstellungen	Standardeinstellung
NUMERIC_STRING	Y: Gibt an, dass diese Spalte nur Zeichenfolgen aus numerischen Daten enthält. Sie enthält keine Leerzeichen, die eine Sortierung der Spaltendaten beeinflussen könnten. Diese Option ist nützlich, wenn sich die Sortierfolge einer Datenquelle von der Sortierfolge des DB2-Servers unterscheidet. Spalten, die mit dieser Option markiert sind, werden nicht von der lokalen Auswertung (der Datenquelle) aufgrund unterschiedlicher Sortierfolgen ausgeschlossen. Wenn die Spalte nur numerische Zeichenfolgen enthält, auf die Leerzeichen folgen, geben Sie 'Y' nicht an. N: Gibt an, dass diese Spalte nicht auf Zeichenfolgen mit numerischen Daten beschränkt ist.	N
VARCHAR_NO_ TRAILING_BLANKS	Y: Gibt an, dass diese Datenquelle ähnlich wie der DB2-Datenserver mit einer Vergleichssemantik für VARCHAR-Datentypen arbeitet, bei der nicht mit Leerzeichen aufgefüllt wird. Bei Zeichenfolgen variabler Länge, die keine folgenden Leerzeichen enthalten, liefert die Vergleichssemantik ohne Auffüllung mit Leerzeichen einiger Datenserver die gleichen Ergebnisse wie die DB2-Vergleichssemantik. Geben Sie diesen Wert an, wenn Sie sicher sind, dass alle VARCHAR-Spalten in Tabellen oder Sichten an einer Datenquelle keine folgenden Leerzeichen enthalten. N: Gibt an, dass diese Datenquelle die VARCHAR-Vergleichssemantik ohne Auffüllung mit Leerzeichen, die der Vergleichssemantik des DB2-Datenservers ähnlich ist, nicht verwendet.	N

Abfragemerkmale mit Auswirkung auf die Pushdown-Möglichkeiten

Eine Abfrage kann einen SQL-Operator enthalten, der Kurznamen aus verschiedenen Datenquellen einbezieht. Die Operation muss auf dem DB2-Server 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.

Richtlinien zur Ermittlung, wo eine Abfrage für föderierte Datenbanken ausgewertet wird:

Die DB2-EXPLAIN-Funktion, das durch Aufrufen des Befehls **db2expln** gestartet werden kann, zeigt an, wo Abfragen ausgewertet werden. Die Ausführungsposition für jeden Operator ist der Ausgabe des Befehls zu entnehmen.

- Wenn eine Abfrage an eine Datenquelle verschoben wird (Pushdown), sollte ein Operator RETURN angezeigt werden, bei dem es sich um einen DB2-Standardoperator handelt. Wenn eine SELECT-Anweisung Daten aus einem Kurznamen abruft, wird außerdem ein Operator SHIP angezeigt, der sich ausschließlich auf Operationen in föderierten Datenbanken bezieht: Er ändert das Servermerkmal

des Datenflusses und trennt lokale Operatoren von fernen Operatoren. Die SELECT-Anweisung wird mithilfe der SQL-Version generiert, die von der Datenquelle unterstützt wird.

- Wenn eine INSERT-, UPDATE- oder DELETE-Anweisung vollständig an die ferne Datenquelle verschoben werden kann (Pushdown), wird eventuell kein Operator SHIP im Zugriffsplan ausgewiesen. Alle fern ausgeführten INSERT-, UPDATE- oder DELETE-Anweisungen werden jedoch für den Operator RETURN angezeigt. Wenn eine Abfrage jedoch nicht in ihrer Gesamtheit verschoben werden kann, gibt der Operator SHIP an, welche Operationen fern durchgeführt wurden.

Verstehen, warum eine Abfrage an einer Datenquelle und nicht vom DB2-Server 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 sehr selektives Vergleichselement zum Filtern von Zeilen verwendet werden und den Netzverkehr 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 mit 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 wird, das Einbinden eines bestimmten Werts? Würde dies 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?

  Zu untersuchende Bereiche sind:

  – 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 für kombinierte GROUP BY-Elemente
    - Spaltenspezifikationen 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 Klausel GROUP BY einen Zeichenausdruck? Ist dies der Fall, überprüfen Sie, ob die ferne Datenquelle und der DB2-Server die gleichen Einstellungen für die Beachtung der Groß-/Kleinschreibung verwenden.
- Warum wird der Gruppenoperator nicht fern ausgewertet?

Zu untersuchende Bereiche sind:

- Werden beide Operanden in ihrer Gesamtheit 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 (LOBs) oder Langfelddaten (LONG) gültige Eingaben für diesen speziellen Gruppenoperator?

- Warum wird die Operation ORDER BY nicht fern ausgeführt?

  Zu untersuchende Bereiche sind:

  - 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? Ist dies der Fall, überprüfen Sie, ob die ferne Datenquelle und der DB2-Server unterschiedliche Sortierfolgen oder Einstellungen für die Beachtung der Groß-/Kleinschreibung haben.
  - Hat die ferne Datenquelle Einschränkungen für diesen Operator? Gibt es beispielsweise eine Begrenzung für die Anzahl von ORDER BY-Elementen? Schränkt die ferne Datenquelle die Spaltenangaben für die ORDER BY-Liste ein?

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. Eine solche Generierung von fernem SQL hilft bei der Erstellung einer global optimierten Zugriffsstrategie für eine Abfrage.

Das Optimierungsprogramm verwendet die Ausgabe der Pushdown-Analyse zur Entscheidung, ob die jeweilige Operation lokal auf dem DB2-Server oder fern an einer Datenquelle ausgewertet werden soll. 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 und Nachrichten zwischen dem DB2-Server und der fernen Datenquelle berücksichtigt.

Obwohl das Ziel die Erstellung einer optimierten Abfrage ist, haben die folgenden Faktoren erhebliche Auswirkungen auf die globale Optimierung und somit auch die Abfrageleistung:

- Servermerkmale
- Kurznamenmerkmale

Serveroptionen mit Auswirkung auf die globale Optimierung

Die folgenden Serveroptionen einer Datenquelle können sich auf die globale Optimierung auswirken:

- Relatives Verhältnis der Verarbeitungsgeschwindigkeiten

  Geben Sie mithilfe der Serveroption CPU_RATIO an, wie schnell oder langsam die Verarbeitungsgeschwindigkeit an der Datenquelle im Verhältnis zur Verarbeitungsgeschwindigkeit auf dem DB2-Server sein sollte. Ein niedriger Wert für das Verhältnis gibt an, dass die Verarbeitungsgeschwindigkeit an der Datenquelle schneller als die Verarbeitungsgeschwindigkeit auf dem DB2-Server ist. In die-

sem Fall zieht das DB2-Optimierungsprogramm mit größerer Wahrscheinlichkeit in Betracht, CPU-intensive Operationen an die Datenquelle zu verschieben (Pushdown).

- Relatives Verhältnis der E/A-Geschwindigkeiten

  Geben Sie mithilfe der Serveroption IO_RATIO an, wie schnell oder langsam die System-E/A-Geschwindigkeit an der Datenquelle im Verhältnis zur System-E/A-Geschwindigkeit auf dem DB2-Server sein sollte. Ein niedriger Wert für das Verhältnis gibt an, dass die E/A-Geschwindigkeit an der Datenquelle schneller als die E/A-Geschwindigkeit auf dem DB2-Server ist. In diesem Fall zieht das DB2-Optimierungsprogramm mit größerer Wahrscheinlichkeit in Betracht, E/A-intensive Operationen an die Datenquelle zu verschieben.

- Übertragungsgeschwindigkeit zwischen dem DB2-Server und der Datenquelle

  Geben Sie mithilfe der Serveroption COMM_RATE die Netzkapazität an. Niedrige Werte, die eine langsame Netzkommunikation zwischen dem DB2-Server und einer Datenquelle angeben, 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 Netzverkehr erfordert.

- Sortierfolge der Datenquelle

  Geben Sie mithilfe 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 DB2-Optimierungsprogramm alle von dieser Datenquelle abgerufenen Daten als unsortiert.

- Ferne Planhinweise

  Geben Sie mithilfe der Serveroption PLAN_HINTS an, dass Planhinweise generiert oder an einer Datenquelle verwendet werden sollen. Standardmäßig sendet der DB2-Server keine Planhinweise an die Datenquelle.

  Planhinweise sind Anweisungsfragmente, die zusätzliche Informationen für das Optimierungsprogramm an einer Datenquelle bereitstellen. Für bestimmte Abfragen können diese Informationen die Leistung verbessern. Die Planhinweise können das Optimierungsprogramm an einer 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 mit Planhinweisen, die an ein Oracle-Optimierungsprogramm gesendet wird, folgendermaßen aussehen:

  ```
  select /*+ INDEX (table1, t1index)*/
      col1
    from table1
  ```

  Der Planhinweis ist die folgende Zeichenfolge: `/*+ INDEX (table1, t1index)*/`

- Informationen in der Wissensbasis des DB2-Optimierungsprogramms

  Der DB2-Server 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. Das heißt, der DB2-Server 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

  Der DB2-Server kann Informationen über Indizes an Datenquellen zur Optimierung von Abfragen heranziehen. Daher ist es wichtig, dass die verfügbaren Indexinformationen aktuell sind. Indexinformationen für einen Kurznamen werden zu Anfang bei der Erstellung des Kurznamens erfasst. Für Sichtkurznamen werden keine Indexinformationen erfasst.

- 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:

  – Wenn der DB2-Server während der Kurznamenerstellung keine Indexinformationen von einer Datenquelle abrufen kann.
  – Wenn Sie einen Index für einen Sichtkurznamen wünschen.
  – Wenn Sie das DB2-Optimierungsprogramm zur Verwendung eines bestimmten Kurznamens als innere Tabelle eines Joins mit Verschachtelungsschleife veranlassen wollen. Sie können 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 eine Sicht erstellt werden, die keine einfache SELECT-Anweisung darstellt, 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) auswählen. Die Abfrage wird eine geringe Leistung erzielen, weil in diesem Fall der Join mehrere Male ausgewertet wird. Eine alternative Lösung wäre, Kurznamen für jede einzelne Tabelle, auf die in der Sicht der Datenquelle verwiesen wird, zu erstellen und anschließend eine lokale Sicht auf dem DB2-Server zu erstellen, die auf beide Kurznamen verweist.

- Katalogstatistiken

  Systemkatalogstatistiken 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 der DB2-Server die in einer Datenquelle gespeicherten Statistikdaten abrufen kann, kann er Aktualisierungen an diesen Daten nicht automatisch erkennen. Darüber hinaus kann der DB2-Server auch Änderungen an der Definition

von Objekten an einer Datenquelle nicht automatisch erkennen. Wenn die statistischen Daten für ein Objekt bzw. die Definition für ein Objekt geändert wurden, haben Sie folgende Möglichkeiten:

– Sie können das entsprechende Gegenstück zu einem Befehl **RUNSTATS** an der Datenquelle ausführen, den aktuellen Kurznamen löschen und ihn anschließend erneut erstellen. Verwenden Sie diese Methode, wenn die Definition eines Objekts geändert wurde.
– Sie können die Statistikdaten in der Katalogsicht SYSSTAT.TABLES manuell aktualisieren. Diese Methode erfordert zwar weniger Schritte, jedoch funktioniert sie nicht, wenn die Definition eines Objekts geändert wurde.

Globale Analyse von Abfragen auf föderierte Datenbanken:

Das DB2-Dienstprogramm EXPLAIN, das durch Aufrufen des Befehls **db2expln** gestartet werden kann, zeigt den Zugriffsplan an, der vom fernen Optimierungsprogramm für die Datenquellen generiert wurde, die von der fernen EXPLAIN-Funktion unterstützt werden. Die Ausführungsposition für jeden Operator ist der Ausgabe des Befehls zu entnehmen.

Sie können darüber hinaus die ferne SQL-Anweisung, die für die jeweilige Datenquelle generiert wurde, je nach Abfragetyp 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.

Verstehen der DB2-Optimierungsentscheidungen

Berücksichtigen Sie die folgenden Schlüsselfragen, wenn Sie Möglichkeiten zur Leistungsverbesserung untersuchen:

- Warum wird ein Join zwischen zwei Kurznamen derselben Datenquelle nicht fern an der Datenquelle ausgewertet?

  Zu untersuchende Bereiche sind:

  – Joinoperationen. Werden sie von der fernen Datenquelle unterstützt?
  – Joinvergleichselemente. Können die Joinvergleichselemente an der fernen Datenquelle ausgewertet werden? Wenn dies zu verneinen ist, untersuchen Sie das Joinvergleichselement.
- Warum wird der Operator GROUP BY nicht fern ausgewertet?

  Untersuchen Sie die Operatorsyntax und stellen Sie sicher, dass der Operator an der fernen Datenquelle ausgewertet werden kann.
- 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 globalen Optimierungsplans auf die Aufwandsschätzung. 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, jedoch die Verarbeitungsgeschwindigkeit der Datenquelle wesentlich langsamer ist als die Verarbeitungsgeschwindigkeit des DB2-Servers, sodass es sich doch als vorteilhafter erweist, die Operationen auf dem DB2-Server auszuführen. Wenn die Ergebnisse nicht zufrieden stellend sind, überprüfen Sie die Serverstatistikdaten in der Katalogsicht 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 diese Anweisung mit der Originalabfrage identisch ist. Suchen Sie nach Änderungen in der Reihenfolge der Vergleichselemente. Ein gutes Abfrageoptimierungsprogramm sollte sich von der Reihenfolge der Vergleichselemente in einer Abfrage nicht negativ beeinflussen lassen. Das Optimierungsprogramm an der fernen Datenquelle generiert auf der Basis der Reihenfolge von Eingabevergleichselementen möglicherweise einen anderen Plan. Sie können in diesem Fall entweder eine Änderung der Reihenfolge der Vergleichselemente in der Eingabe für den DB2-Server in Betracht ziehen oder die Serviceorganisation der fernen Datenquelle um Unterstützung bitten.

    Sie können darüber hinaus auch auf Ersetzungen von Vergleichselementen prüfen. Ein gutes Abfrageoptimierungsprogramm sollte sich durch eine Ersetzung äquivalenter Vergleichselemente nicht negativ beeinflussen lassen. Das Optimierungsprogramm an der fernen Datenquelle generiert auf der Basis der Eingabevergleichselemente möglicherweise einen anderen Plan. Zum Beispiel können einige Optimierungsprogramme keine Anweisungen mit Vergleichselementen unter Verwendung der Transitivität generieren.
  – Zusätzliche Funktionen. Enthält die ferne SQL-Anweisung Funktionen, die in der Originalabfrage nicht enthalten sind? Einige dieser Funktionen werden möglicherweise zur Konvertierung von Datentypen verwendet. Stellen Sie sicher, dass sie notwendig sind.

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. Zugriffspläne für dynamische SQL- und XQuery-Anweisungen werden während der Laufzeit erstellt.

Es gibt drei Methoden für den Zugriff auf Daten in einer Tabelle:

- Sequenzielles Durchsuchen der gesamten Tabelle
- Lokalisieren bestimmter Zeilen durch Zugriff auf einen Index für die Tabelle
- Durch Scan-Sharing

Zeilen können nach Bedingungen gefiltert werden, die in Vergleichselementen definiert sind, die in der Regel in einer WHERE-Klausel angegeben werden. Die ausgewählten Zeilen in den Tabellen, auf die zugegriffen wird, werden durch einen Join verknüpft, um die Ergebnismenge zu erstellen, und diese Daten können unter Umständen weiter verarbeitet werden, indem die Ausgabe gruppiert oder sortiert wird.

Ab DB2 Version 9.7 ist das *Scan-Sharing*, d. h. die Möglichkeit, dass ein Suchlauf die Pufferpoolseiten eines anderen Suchlaufs verwendet, das Standardverhalten. Das Scan-Sharing erhöht den gleichzeitigen Zugriff und die Leistung für Auslas-

tungen. Das Scan-Sharing bietet verschiedene Vorteile: Das System kann eine höhere Anzahl gleichzeitig ausgeführter Anwendungen unterstützen, Abfragen können eine bessere Leistung erzielen und der Systemdurchsatz kann sich erhöhen, sodass sogar Abfragen davon profitieren, die selbst nicht am Scan-Sharing beteiligt sind. Das Scan-Sharing ist insbesondere in Umgebungen mit Anwendungen effektiv, die Suchläufe als Tabellensuchen oder MDC-Blockindexsuchläufe (MDC, mehrdimensionales Clustering) in großen Tabellen ausführen. Der Compiler ermittelt auf der Basis des Typs eines Suchlaufs, des Zweckes des Suchlaufs, der gültigen Isolationsstufe, des pro Datensatz auszuführenden Arbeitsvolumens usw., ob ein Suchlauf für eine Beteiligung am Scan-Sharing infrage kommt.

Datenzugriff über Indexsuchen

Eine *Indexsuche* findet statt, wenn der Datenbankmanager auf einen Index zugreift, um die Menge der den Suchbedingungen entsprechenden Zeilen (durch Durchsuchen der Zeilen in einem angegebenen Bereich des Index) einzugrenzen, bevor er auf die Basistabelle zugreift, um die Ausgabe in einer angeforderten Reihenfolge anzuordnen, oder um die angeforderten Spaltendaten direkt abzurufen (*reiner Indexzugriff*).

Wenn die Zeilen in einem angegebenen Bereich des Index durchsucht werden, wird der *Indexsuchbereich* (der Startpunkt und der Endpunkt der Suche) durch die Werte in der Abfrage bestimmt, mit denen Indexspalten verglichen werden. Im Fall eines reinen Indexzugriffs ist kein Zugriff auf die indexierte Tabelle erforderlich, da sich alle angeforderten Daten im Index befinden.

Wenn Indizes mit der Option ALLOW REVERSE SCANS erstellt wurden, können Indexsuchen auch in entgegengesetzter Richtung zu der Richtung durchgeführt werden, mit der die Indizes definiert wurden.

Das Optimierungsprogramm wählt eine Tabellensuche, wenn kein geeigneter Index erstellt wurde oder eine Indexsuche aufwendiger wäre. Eine Indexsuche kann aufwendiger sein, wenn die jeweilige Tabelle klein ist, das Clusterbildungsverhältnis in Bezug auf den Index niedrig ist, die Abfrage die Mehrzahl der Tabellenzeilen erfordert oder zusätzliche Sortiervorgänge erforderlich sind, wenn ein partitionierter Index (der die Reihenfolge in bestimmten Fällen nicht beibehalten kann) verwendet wird. Ob der Zugriffsplan eine Tabellensuche oder eine Indexsuche verwendet, lässt sich mithilfe der DB2-EXPLAIN-Funktion 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 IS NULL oder IS NOT NULL.
- Testen auf strenge oder einschließende Ungleichheit.
- Testen auf Gleichheit mit einer Konstante, einer Hostvariablen, einem Ausdruck, der zu einer Konstanten ausgewertet wird, oder einem Schlüsselwort.
- Testen auf Gleichheit mit einer einfachen Unterabfrage, d. h. mit einer Unterabfrage, die weder ANY, ALL oder SOME enthält. Diese Unterfrage darf keinen

Verweis über eine korrelierte Spalte auf den unmittelbar übergeordneten Abfrageblock (d. h. auf die SELECT-Anweisung, für die diese Unterabfrage ein Subselect ist) enthalten.

Die folgenden Beispiele zeigen, wie ein Index zur Begrenzung des Bereichs eines Suchlaufs verwendet werden könnte.

- Betrachten Sie einen Index mit der folgenden Definition:

```
INDEX IX1:  NAME    ASC,
            DEPT    ASC,
            MGR     DESC,
            SALARY  DESC,
            YEARS   ASC
```

  Die folgenden Vergleichselemente könnten zur Begrenzung des Bereichs eines Suchlaufs verwendet werden, der den Index IX1 verwendet:

```
where
  name = :hv1 and
  dept = :hv2
```

  oder

```
where
  mgr = :hv1 and
  name = :hv2 and
  dept = :hv3
```

  Die zweite WHERE-Klausel demonstriert, dass die Vergleichselemente nicht in derselben Reihenfolge angegeben werden müssen, in der die Schlüsselspalten im Index enthalten sind. In den Beispielen werden zwar Hostvariablen (hv = Hostvariable) verwendet, jedoch könnten stattdessen auch andere Variablen, wie Parametermarken, Ausdrücke oder Konstanten verwendet werden.

  In der folgenden WHERE-Klausel würden nur die Vergleichselemente, die auf NAME und DEPT verweisen, zur Begrenzung des Bereichs der Indexsuche verwendet werden:

```
where
  name = :hv1 and
  dept = :hv2 and
  salary = :hv4 and
  years = :hv5
```

  Da sich eine Schlüsselspalte (MGR) zwischen diesen Spalten und den beiden letzten Indexschlüsselspalten befindet, wäre die Reihenfolge nicht gesichert. Wenn aber der Bereich durch die Vergleichselemente `name = :hv1` und `dept = :hv2` festgelegt ist, können die anderen Vergleichselemente an den verbleibenden Indexschlüsselspalten ausgewertet werden.

- Betrachten Sie einen Index, der mit der Option ALLOW REVERSE SCANS erstellt wurde:

```
create index iname on tname (cname desc) allow reverse scans
```

  In diesem Fall basiert der Index (INAME) auf absteigenden (DESC) Werten in der Spalte CNAME. Obwohl der Index für Suchläufe in absteigender Reihenfolge definiert ist, kann ein Suchlauf auch in aufsteigender Reihenfolge erfolgen. Die Verwendung des Index wird vom Optimierungsprogramm gesteuert, wenn es Zugriffspläne erstellt und beurteilt.

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 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 einer Indexsuche berücksichtigt. Im folgenden Beispiel können Vergleichselemente für die Spalten NAME und DEPT verwendet werden, um den Bereich einzugrenzen, das Vergleichselement für die Spalte MGR kann zu diesem Zweck jedoch nicht verwendet werden.

  ```
  where
    name = :hv1 and
    dept > :hv2 and
    dept < :hv3 and
    mgr < :hv4
  ```

- Vergleichselemente für einschließende Ungleichheit

  Die folgenden Operatoren für einschließende Ungleichheit werden für bereichsbegrenzende Vergleichselemente verwendet:

  – >= und <=
  – BETWEEN
  – LIKE

  Mehrere Spalten mit Vergleichselementen für einschließende Ungleichheit können zur Begrenzung des Bereichs einer Indexsuche berücksichtigt werden. Im folgenden Beispiel können alle Vergleichselemente zur Begrenzung des Bereichs verwendet werden:

  ```
  where
    name = :hv1 and
    dept >= :hv2 and
    dept <= :hv3 and
    mgr <= :hv4
  ```

  Nehmen Sie folgende Werte an: `:hv2 = 404`, `:hv3 = 406` und `:hv4 = 12345`. Der Datenbankmanager durchsucht in diesem Fall den Index nach den Abteilungen (DEPT) 404 und 405, bricht jedoch 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" sowie INTERSECT oder EXCEPT und UNION resultieren. Es gibt folgende Ausnahmen:

- Wenn der Index partitioniert ist, kann er nur dazu verwendet werden, die Daten zu sortieren, wenn den Indexschlüsselspalten die Tabellenpartitionierungsschlüsselspalten vorangestellt sind oder beim Partitionsausschluss alle Partitionen bis auf eine Partition ausgeschlossen werden.
- Beim Sortieren der Spalten kann von den ersten Indexschlüsselspalten abgewichen werden, wenn die Indexschlüsselspalten auf Gleichheit mit „konstanten Werten" getestet werden, bzw. mit einem beliebigen Ausdruck, der zu einer Konstanten ausgewertet wird.

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 daher geordnet sind. Das heißt, die gezeigten Klauseln WHERE und ORDER BY sind mit folgenden Klauseln äquivalent:

```
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: Es gibt nur einen PROJNAME-Wert für jeden PROJNO-Wert.

Arten des Indexzugriffs

In einigen Fällen stellt das Optimierungsprogramm möglicherweise fest, dass alle Daten, die eine Abfrage 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 einem Index abgerufen werden, ohne auf die Tabelle zuzugreifen. Diese Methode wird als *reiner Indexzugriff* bezeichnet. Betrachten Sie zum Beispiel die folgende Indexdefinition:

```
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 aus diesen Spalten abzurufen, müssen Tabellenzeilen gelesen werden. Damit das Optimierungsprogramm die Möglichkeit hat, den 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'
```

Vergewissern Sie sich, dass der zusätzliche Speicher- und Pflegebedarf von INCLUDE-Spalten gerechtfertigt ist. Wenn Abfragen, die INCLUDE-Spalten nutzen, nur selten ausgeführt werden, ist der Aufwand 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 dieser beiden Indizes erfüllt werden:

```
where
  dept = :hv1 or
  (job = :hv2 and
  years >= :hv3)
```

Das Durchsuchen des Index IX2 liefert eine Liste von Satz-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 mit RIDs werden kombiniert und doppelte Werte werden entfernt, bevor auf die Tabelle zugegriffen wird. Diese Methode wird als *logisches Verknüpfen über OR von Indizes* (Index ORing) bezeichnet.

Das logische Verknüpfen von Indizes über OR kann auch für Vergleichselemente verwendet werden, die von einer IN-Klausel angegeben werden, wie im folgenden Beispiel dargestellt:

```
where
  dept in (:hv1, :hv2, :hv3)
```

Während der Zweck des logischen Verknüpfens von Indizes über OR in der Eliminierung doppelter RIDs liegt, besteht das Ziel des *logischen Verknüpfens von Indizes über AND* (Index ANDing) darin, gemeinsame RIDs zu finden. Das logische Verknüpfen von Indizes über AND kann auftreten, wenn Anwendungen, die mehrere Indizes für entsprechende Spalten innerhalb derselben Tabellen erstellen, eine Abfrage mit mehreren AND-Vergleichselementen für die Tabelle ausführen. Mehrere Indexsuchen für alle mit indexierten Spalten generieren Werte, mit denen durch ein Hashverfahren Bitzuordnungen erstellt werden. Die zweite Bitzuordnung wird zur Prüfung der ersten Bitzuordnung verwendet, um die gesuchten Zeilen für die endgültige Ergebnismenge zu generieren. Betrachten Sie zum Beispiel die folgenden Indizes:

```
INDEX IX4: SALARY   ASC
INDEX IX5: COMM     ASC
```

Diese Indizes können zur Auflösung der folgenden Vergleichselemente verwendet 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 qualifizierter Satz-IDs, die beide Vergleichselemente erfüllen. Dies wird als *dynamisches AND-Verknüpfen von Indizes über Bitzuordnungen* (engl. dynamic bitmap ANDing) bezeichnet. Diese Methode wird nur angewendet, sofern die Tabelle ausreichend Kardinalität hat, die Spalten genügend Werte innerhalb des gesuchten Bereichs enthält oder genügend Duplizität vorhanden ist, wenn Gleichheitsvergleichselemente verwendet werden.

Zur Realisierung der Leistungsvorteile dynamischer Bitzuordnungen beim Durchsuchen mehrerer Indizes kann es notwendig sein, den Wert des Datenbankkonfigurationsparameters **sortheap** und den Wert des Konfigurationsparameter **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 auf 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.

Das Optimierungsprogramm verwendet beim Zugriff auf eine einzelne Tabelle keine kombinierten AND- und OR-Verknüpfungen von Indizes.

Indexzugriff in Bereichsclustertabellen

Im Gegensatz zu Standardtabellen benötigt eine Bereichsclustertabelle keinen physischen Index (wie einen traditionellen B-Baumstrukturindex), der einer Zeile einen Schlüsselwert 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 generieren, 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.

Indexzugriff und Clusterverhältnisse

Bei der Auswahl des Zugriffsplans schätzt das Optimierungsprogramm die Anzahl der E/A-Operationen, die zum Einlesen der Seiten von der Platte in den Pufferpool erforderlich sind. Diese Schätzung schließt eine Voraussage über die Nutzung des Pufferpools ein, da zum Lesen von Zeilen einer Seite, die sich bereits im Pufferpool befindet, keine zusätzlichen E/A-Operationen anfallen.

Für Indexsuchen wird das Optimierungsprogramm durch Informationen aus dem Systemkatalog bei der Abschätzung des Ein-/Ausgabeaufwands zum Lesen von Datenseiten in einen Pufferpool unterstützt. Dabei werden Informationen aus den folgenden Spalten der Sicht 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, falls 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 Einlesen der Datenseiten in Pufferpools verschiedener Größen angeben. Für diese Spalten werden Daten nur erfasst, wenn Sie den Befehl **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 der Clusterbildung der Daten kann bedeutende Auswirkungen auf die Leistung haben, sodass einer der Indizes, die für die Tabelle definiert sind, 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 für einen Index eine Obermenge der Schlüssel für den Clusterindex darstellen oder in denen es eine tatsächliche 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 diese Clusterbildung bei der Verarbeitung von Einfügungen beibehalten wird. Da Aktualisierungs- und Einfügeoperationen die Clusterbildung in Bezug auf den Index verringern können, müssen Sie die Tabelle eventuell in regelmäßigen Abständen reorganisieren. Zur Verringerung der Anzahl von Reorganisationen für eine Tabelle, an der häufig Einfüge-, Aktualisierungs- oder Löschoperationen ausgeführt werden, geben Sie die Klausel PCTFREE in der Anweisung ALTER TABLE an.

Scan-Sharing

Scan-Sharing bezeichnet die Möglichkeit einer Suchoperation, die von einer anderen Suchoperation ausgeführten Arbeitsschritte für sich zu nutzen. Beispiele für solche gemeinsam genutzten Arbeitsschritte sind Lesevorgänge für Plattenseiten, Plattensuchvorgänge, wiederverwendete Inhalte von Pufferpools, Dekomprimierung usw.

Für aufwendige Suchläufe, wie zum Beispiel Tabellensuchen oder Suchen in MDC-Indexblöcken (MDC - mehrdimensionales Clustering) umfangreicher Tabellen kann es manchmal infrage kommen, Seitenlesevorgänge mit anderen Suchläufen gemeinsam zu nutzen. Solche gemeinsam genutzten Suchläufe können an einem beliebigen Punkt in der Tabelle beginnen, um Seiten, die bereits in den Pufferpool eingelesen wurden, vorteilhaft zu nutzen. Wenn ein Suchlauf mit gemeinsamer Nutzung das Ende der Tabelle erreicht, fährt er am Anfang fort und endet an dem Punkt, an dem er begonnen hat. Dies wird als *Suche mit Umlauf* (engl. 'wrapping scan') bezeichnet. Abb. 25 auf Seite 253 zeigt den Unterschied zwischen regulären Suchläufen und Suchläufen mit Umlauf für Tabellen und Indizes.

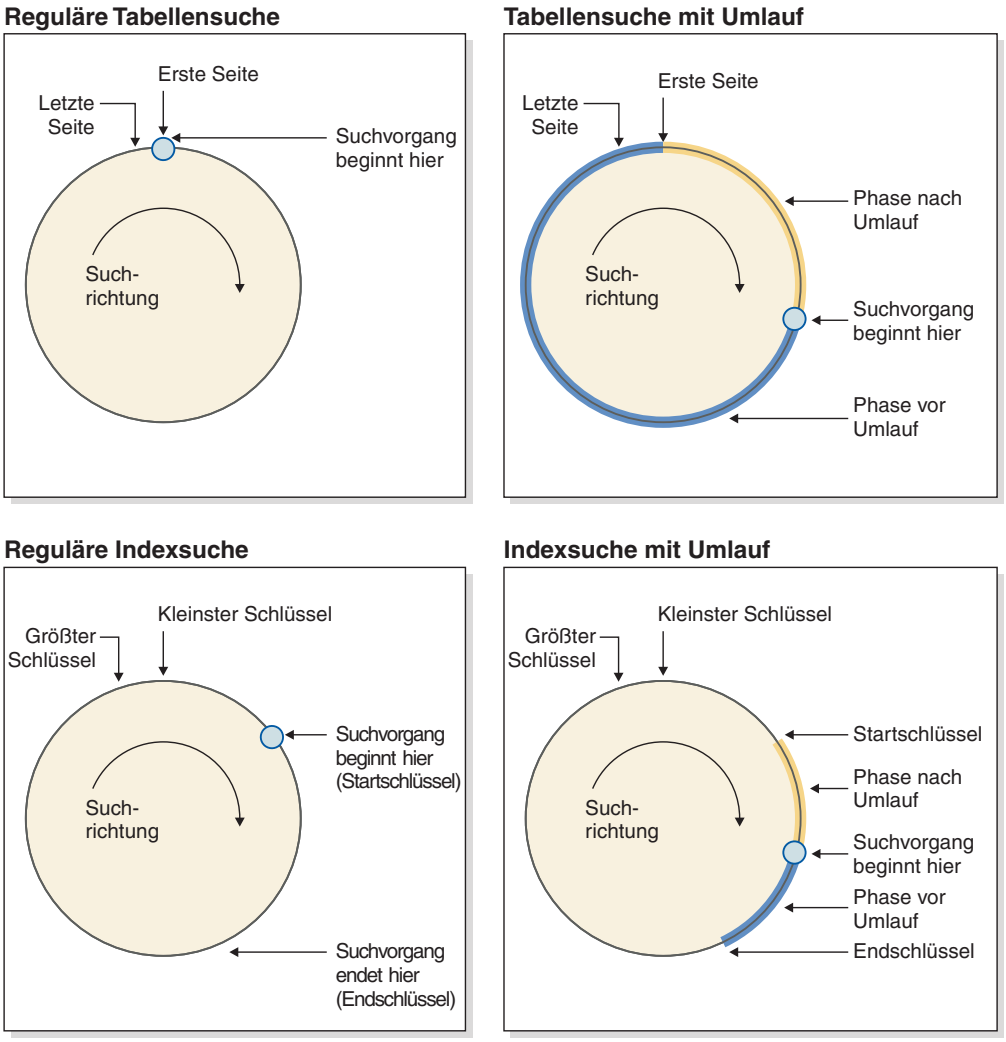


Abbildung 25. Konzeptionelle Sicht von regulären und umlaufenden Suchläufen

Die Funktion der gemeinsamen Nutzung von Suchoperationen ist standardmäßig aktiviert. Ob eine Suche für die gemeinsame Nutzung und die Suche mit Umlauf infrage kommt, wird vom SQL-Compiler automatisch bestimmt. Während der Ausführung kann eine infrage kommende Suche an der gemeinsamen Nutzung oder der Umlauffunktion teilnehmen oder nicht. Dies entscheidet sich nach Faktoren, die bei der Kompilierung nicht bekannt waren.

Gemeinsam genutzte Suchvorgänge werden in *Gruppen für gemeinsame Nutzung* (engl. 'share groups') verwaltet. Diese Gruppen halten ihre Elemente so lange wie möglich zusammen, sodass sich maximale Vorteile aus der gemeinsamen Nutzung erzielen lassen. Wenn eine Suche schneller als eine andere Suche ist, können die Vorteile der gemeinsamen Nutzung von Seiten eingebüßt werden. In diesem Fall ist es möglich, dass Seiten im Pufferpool, auf die die erste Suche zugegriffen hat, aus dem Pufferpool entfernt werden, bevor eine andere Suche in der Gruppe für gemeinsame Nutzung auf sie zugreifen kann. Der Datenserver misst den Abstand zwischen zwei Suchläufen in derselben Gruppe für gemeinsame Nutzung an der Anzahl von Pufferpoolseiten, die zwischen ihnen liegen. Der Datenserver überwacht außerdem die Geschwindigkeit der Suchläufe. Wenn der Abstand zwischen zwei Suchläufen in derselben Gruppe für gemeinsame Nutzung zu groß wird, können diese Suchläufe Pufferpoolseiten möglicherweise nicht gemeinsam nutzen. Um diesen Effekt zu verringern, können schnellere Suchläufe gedrosselt werden, sodass

langsamere Suchläufe auf die Datenseiten zugreifen können, bevor diese bereinigt werden. Abb. 26 zeigt zwei Sets für gemeinsame Nutzung, eines für eine Tabelle und eines für einen Blockindex. Ein *Set für gemeinsame Nutzung* (engl. 'sharing set') ist eine Sammlung von Gruppen für gemeinsame Nutzung, die auf dasselbe Objekt (z. B. eine Tabelle) durch denselben Zugriffsmechanismus (z. B. Tabellensuche oder Blockindexsuche) zugreifen. Bei Tabellensuchen erhöht sich die Reihenfolge für Seitenlesevorgänge nach Seiten-ID, bei Blockindexsuchen nach Schlüsselwert.

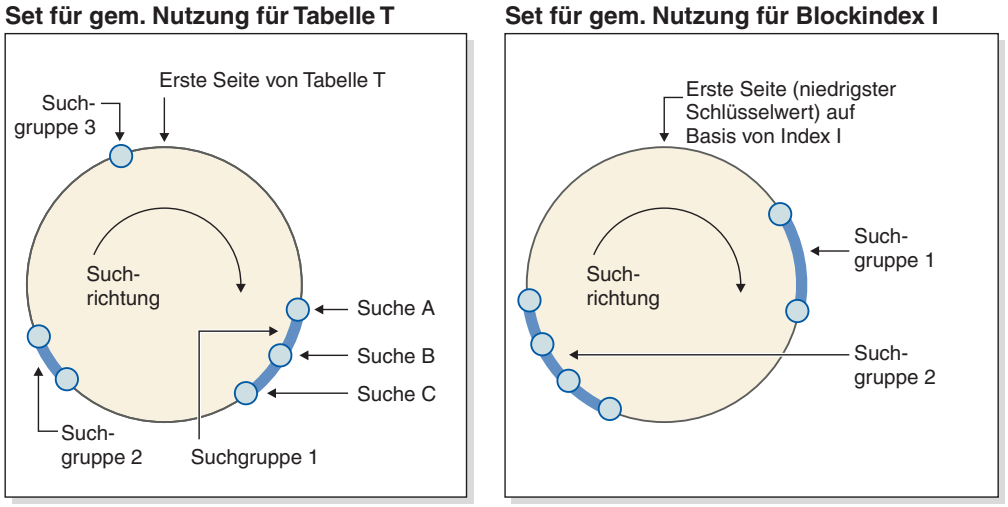


Abbildung 26. Sets für gemeinsame Nutzung beim Scan-Sharing für Tabellen und Blockindizes

Die Abbildung zeigt außerdem, wie der Inhalt von Pufferpools innerhalb von Gruppen wiederverwendet wird. Betrachten Sie 'Suche C, die die führende Suche von Gruppe 1 ist. Die nachfolgenden Suchen (A und B) sind zusammen mit C in einer Gruppe, weil sie nah beieinander liegen und mit hoher Wahrscheinlichkeit die Seiten, die von C in den Pufferpool eingelesen wurden, wiederverwenden können.

Ein Suchvorgang mit hoher Priorität wird in keinem Fall durch einen Suchvorgang mit niedrigerer Priorität gedrosselt, sondern stattdessen möglicherweise in eine andere Gruppe für gemeinsame Nutzung versetzt. Ein Suchvorgang mit hoher Priorität kann in eine Gruppe gestellt werden, in der er von der Arbeit, die von anderen Suchvorgängen mit niedrigerer Priorität in der Gruppe ausgeführt wird, profitieren kann. Er verbleibt so lange in dieser Gruppe, wie der Vorteil verfügbar ist. Durch Drosseln des schnelleren Suchvorgangs oder durch versetzen des schnelleren Suchvorgangs in eine schnellere Gruppe für gemeinsame Nutzung (sofern der Suchvorgang eine solche findet) passt der Datenserver die Gruppen für gemeinsame Nutzung an, um eine optimale Suchleistung sicherzustellen.

Mithilfe des Befehls **db2pd** können Sie Informationen zum Scan-Sharing anzeigen. Für eine einzelne gemeinsam genutzte Suche zeigt die Ausgabe des Befehls **db2pd** Daten wie zum Beispiel die Suchgeschwindigkeit und die Drosselungsdauer der Suche an. Für eine Gruppe für gemeinsame Nutzung zeigt die Befehlsausgabe die Anzahl der Suchläufe in der Gruppe und die Anzahl der Seiten an, die von der Gruppe gemeinsam genutzt werden.

Die Tabelle EXPLAIN_ARGUMENT besitzt neue Zeilen, die Scan-Sharing-Informationen zu Tabellensuchen und Indexsuchen enthalten. (Mithilfe des Befehls **db2exfmt** können Sie den Inhalt dieser Tabelle formatieren und anzeigen.)

Mithilfe von Profilen des Optimierungsprogramms können Sie Entscheidungen überschreiben, die der Compiler in Bezug auf das Scan-Sharing trifft (siehe „Zugriffstypen"). Solche Überschreibungen sind ausschließlich zur Verwendung unter besonderen Bedarfsbedingungen gedacht. Der Hinweis für das Umlaufen kann nützlich sein, wenn eine wiederholbare Reihenfolge von Datensätzen in einer Ergebnismenge benötigt wird, jedoch die Klausel ORDER BY (die eine Sortierung zur Folge haben könnte) vermieden werden soll. Ansonsten wird empfohlen, diese Optimierungsprofile nicht zu verwenden, sofern Sie nicht vom DB2-Service dazu aufgefordert werden.

Joins

Ein *Join* ist der Prozess der Kombination von Daten 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 verknüpft, wenn die Informationen in den entsprechenden Zeilen die Joinbedingung (das *Joinvergleichselement*) erfüllen.

Betrachten Sie zum Beispiel die beiden folgenden Tabellen:

TABLE1

PROJ	PROJ_ID
A	1
B	2
C	3
D	4

TABLE2

PROJ_ID	NAME
1	Sam
3	Joe
4	Mary
1	Sue
2	Mike

Für den Join von TABLE1 und TABLE2, sodass die Spalten PROJ_ID die gleichen Werte haben, verwenden Sie die folgende SQL-Anweisung:

```
select proj, x.proj_id, name
  from table1 x, table2 y
  where x.proj_id = y.proj_id
```

In diesem Fall sieht das entsprechende Joinvergleichselement wie folgt aus: `where x.proj_id = y.proj_id`.

Die Abfrage liefert die folgende Ergebnismenge:

PROJ	PROJ_ID	NAME
A	1	Sam
A	1	Sue
B	2	Mike
C	3	Joe
D	4	Mary

Abhängig von der Spezifik der Joinvergleichselemente sowie von Aufwänden, die auf der Basis von 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 ausgewählt und die andere als innere Tabelle des Joins betrachtet. Auf die äußere Tabelle wird zuerst zugegriffen und sie wird nur einmal durchsucht. Ob die innere Tabelle mehrere Male durchsucht wird, hängt vom Typ des Joins und den verfügbaren Indizes ab. Auch wenn in einer Abfrage mehr als zwei Tabellen durch einen Join 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, und anschließend die Abfrageleistung analysieren, um zu entscheiden, ob Joinoperatoren hinzugefügt werden sollen.

Joinmethoden

Das Optimierungsprogramm kann eine von drei Grundstrategien auswählen, wenn Abfragen das Verknüpfen (Join) von Tabellen erfordern: Join mit Verschachtelungsschleife, Mischjoin oder Hash-Join.

Join mit Verschachtelungsschleife (Nested Loop Join)

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

  Beispiel: Spalte A in Tabelle T1 und Spalte A in Tabelle T2 enthalten die folgenden Werte:

Äußere Tabelle T1: Spalte A	Innere Tabelle T2: Spalte A
2	3
3	2
3	2
	3
	1

  Zur Ausführung eines Joins mit Verschachtelungsschleife zwischen den Tabellen T1 und T2 geht der Datenbankmanager in folgenden Schritten vor:

  1. Lesen der ersten Zeile in 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. Wiederholen von Schritt 2, bis das Ende der Tabelle erreicht wird.
  4. Zurückkehren zu T1 und Lesen der nächsten Zeile (3).
  5. Durchsuchen von T2 (von der ersten Zeile an) bis ein übereinstimmender Wert (3) gefunden wird, und anschließender Join der beiden Zeilen.
  6. Wiederholen von Schritt 5, bis das Ende der Tabelle erreicht wird.
  7. Zurückkehren zu T1 und Lesen der nächsten Zeile (3).
  8. Durchsuchen von T2 wie zuvor 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 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. Beispiele:

```
äußere_tabelle.c1 + äußere_tabelle.c2 <= innere_tabelle.c1
äußere_tabelle.c4 < innere_tabelle.c3
```

Diese Methode kann die Anzahl der Zeilen, auf die in der inneren Tabelle für jeden Zugriff auf die äußere Tabelle zugegriffen wird, erheblich verringern. Der erzielbare Vorteil hängt von einer Reihe von Faktoren ab, zu denen u. a. die Selektivität des Joinvergleichselements gehört.

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 für die innere Tabelle für den Zugriff auf Seiten auf der Platte 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 die äußere Tabelle sortiert wurde, kann die Anzahl von Indexseiten, auf die zugegriffen wird, minimiert werden.

Wenn 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 eine Operation für die Klauseln GROUP BY, DISTINCT, ORDER BY oder einen Mischjoin zu unterstützen.

Mischjoin (Merge Join)

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 durch einen Indexzugriff oder durch eine Sortierung geordnete Eingaben für die Joinspalten. Ein Mischjoin kann nicht verwendet werden, wenn die Joinspalte eine Langfeldspalte (LONG-Spalte) 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 ebenfalls nur einmal durchsucht, sofern in der äußeren Tabelle keine Werte mehrfach auftreten. Wenn Werte mehrfach auftreten, kann eine Gruppe von Zeilen der inneren Tabelle noch einmal durchsucht werden.

Beispiel: Spalte A in Tabelle T1 und Spalte A in Tabelle T2 enthalten die folgenden Werte:

Äußere Tabelle T1: Spalte A	Innere Tabelle T2: Spalte A
2	1
3	2
3	2
	3
	3

Zur Ausführung eines Mischjoins zwischen den Tabellen T1 und T2 geht der Datenbankmanager in folgenden Schritten vor:

1. Lesen der ersten Zeile in 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. 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. Wenn das Ende von T2 erreicht ist, Zurückkehren zu T1 und Lesen der nächsten Zeile. Beachten Sie, dass der nächste Wert in T1 derselbe ist wie der vorige Wert aus T1, sodass T2 noch einmal ab dem ersten Wert 3 in T2 durchsucht wird. Der Datenbankmanager speichert diese Position.

Hash-Join

Eine Hash-Join erfordert ein oder mehrere Vergleichselemente der Form `tabelle1.spalteX = 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. Spalten des Typs DECFLOAT müssen die gleiche Genauigkeit (Stellenanzahl) aufweisen. Die Spalte kann keine Langfeldspalte (LONG-Spalte) oder LOB-Spalte sein.

Zunächst wird die als innere Tabelle vorgesehene Tabelle durchsucht und Zeilen werden in Speicherpuffer kopiert, die dem Sortierspeicher entnommen werden, der durch den Datenbankkonfigurationsparameter **sortheap** angegeben ist. Die Speicherpuffer werden auf der Basis eines Hashwerts, der auf den Spalten der Joinvergleichselemente berechnet wird, in Abschnitte unterteilt. Wenn die Größe der inneren Tabelle die Größe des Sortierspeichers überschreitet, werden Puffer aus ausgewählten Abschnitten in temporäre Tabellen geschrieben.

Wenn die innere Tabelle verarbeitet wurde, wird die zweite (äußere) Tabelle durchsucht und ihre Zeilen werden mit den Zeilen aus der inneren Tabelle abgeglichen, indem zuerst der Hashwert, der für die Spalten der Joinvergleichselemente berechnet wurde, verglichen wird. Wenn der Hashwert für die Spalte der äußeren Zeilen mit dem Hashwert für die Spalte der inneren Zeile übereinstimmt, werden die tatsächlichen Werte der Joinvergleichselementspalten verglichen.

Zeilen der äußeren Tabelle, 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 äußere Zeile ebenfalls in eine temporäre Tabelle geschrieben. Schließlich werden übereinstimmende Paare von Tabellenteilen aus den temporären Tabellen gelesen, die Hashwerte ihrer Zeilen abgeglichen und die Joinvergleichselemente geprüft.

Zur Erzielung des vollen Leistungsvorteils von Hash-Joins müssen Sie möglicherweise den Wert des Konfigurationsparameters **sortheap** der Datenbank und des Konfigurationsparameters **sheapthres** des Datenbankmanagers ändern.

Die Leistung von 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 den Konfigurationsparameter **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.

Strategien zur Auswahl optimaler Joins

Das Optimierungsprogramm nutzt verschiedene Methoden zur Auswahl einer optimalen Joinstrategie für eine Abfrage. Zu diesen Methoden, die durch die Optimierungsklasse der Abfrage bestimmt werden, zählen verschiedene Strategien, wie zum Beispiel Sternschemajoins, Joins mit frühem Verlassen (Early Out Join) und zusammengesetzte Tabellen.

Der Algorithmus für die Joinaufzählung spielt die entscheidende Rolle bei der Bestimmung der Anzahl von Zugriffsplankombinationen, die das Optimierungsprogramm prüft.

- Schnelle Joinaufzählung (Greedy Join Enumeration)
  - Ist effizient im Hinblick auf Speicher- und Zeitbedarf.
  - Verwendet eine Aufzählung in einer Richtung, d. h., wenn eine Joinmethode für zwei Tabellen ausgewählt ist, wird sie im Laufe der weiteren Optimierung nicht mehr geändert.
  - Findet eventuell nicht den optimalen Zugriffsplan, wenn viele Tabellen verknüpft werden. Wenn die 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 Anwendung der Transitivität generierte Joinvergleichselemente für dieselbe Spalte hat.
- Dynamisch programmierte Joinaufzählung (Dynamic Programming Join Enumeration)
  - Ist nicht effizient im Hinblick auf den Speicher- und Zeitbedarf, der mit zunehmender Anzahl der verknüpften Tabellen exponenziell ansteigt.
  - Ist effizient und erschöpfend bei der Suche nach dem besten Zugriffsplan.
  - Ist der Strategie ähnlich, die von DB2 für z/OS verwendet wird.

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. Dadurch wird eine Ergebnistabelle erstellt, die in der Regel sehr groß ist, weil ihre Größe das Kreuzprodukt der Größen der beiden Quellentabellen darstellt. Da ein solcher Plan wahrscheinlich keine gute Leistung zulässt, vermeidet das Optimierungsprogramm sogar die Aufwandsabschätzung für einen Plan dieses Typs.

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 *Fakttabelle* 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 Fakttabelle verbunden werden. Jede Dimensionstabelle enthält zusätzliche Werte, die die Informationen über eine bestimmte Spalte in der Fakttabelle erweitern. Eine typische Abfrage besteht aus mehreren lokalen Vergleichselementen, die auf Werte in den Dimensionstabellen verweisen, und enthält Joinvergleichselemente, die die Dimensionstabellen mit der Fakttabelle verbinden. Für solche Abfragen kann es vorteilhaft sein, das kartesische Produkt mehrerer kleiner Dimensionstabellen zu berechnen und erst anschließend auf die umfangreiche Fakttabelle zuzugreifen. Diese Technik ist dann nützlich, wenn mehrere Joinvergleichselemente einem mehrspaltigen Index entsprechen.

Der DB2-Datenserver kann Abfragen erkennen, die auf Datenbanken ausgeführt werden, die mit Sternschemata aufgebaut wurden und die über mindestens zwei Dimensionstabellen verfügen, und kann den Suchbereich vergrößern, um mögliche Pläne zur Berechnung des kartesischen Produkts von Dimensionstabellen zu berücksichtigen. Wenn der Plan mit dem kartesischen Produkt den niedrigsten geschätzten Aufwand verursacht, wird er vom Optimierungsprogramm ausgewählt.

Diese Strategie für Sternschemajoins basiert auf der Annahme, dass Primärschlüsselindizes im Join verwendet werden. Ein anderes Szenario liegt vor, wenn Fremdschlüsselindizes verwendet werden. Wenn die Fremdschlüsselspalten in der Fakttabelle einspaltige Indizes sind und es eine relativ hohe Selektivität über alle Dimensionstabellen hinweg gibt, kann die folgende Methode eines Sternschemajoins zur Anwendung kommen:

1. Verarbeiten jeder Dimensionstabelle wie folgt:
   - Durchführen eines Semi-Joins zwischen der Dimensionstabelle und dem Fremdschlüsselindex für die Fakttabelle
   - Dynamisches Erstellen einer Bitzuordnung durch ein Hashverfahren für die Werte der Satz-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 Fakttabelle 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 der Fakttabelle und den Dimensionstabellen sind nicht erforderlich, werden jedoch empfohlen.

Die dynamischen Bitzuordnungen, die von Verfahren für Sternschemajoins erstellt und verwendet werden, erfordern Sortierspeicher, dessen Größe durch den Datenbankkonfigurationsparameter **sortheap** definiert wird.

Joins mit frühem Verlassen (Early Out Joins)

Das Optimierungsprogramm kann einen Join mit frühem Verlassen (Early Out Join) auswählen, wenn es erkennt, dass jede Zeile aus einer der Tabellen maximal mit einer Zeile aus der anderen Tabelle verknüpft werden muss.

Ein Join mit frühem Verlassen ist möglich, wenn ein Joinvergleichselement für die Schlüsselspalte (bzw. Schlüsselspalten) einer der Tabellen vorhanden ist. Betrachten Sie zum Beispiel die folgende Abfrage, die die Namen von Mitarbeitern (Employee) und ihren direkten Vorgesetzten (Manager) zurückgibt.

```
select employee.name as employee_name,
    manager.name as manager_name
  from employee as employee, employee as manager
  where employee.manager_id = manager.id
```

Unter der Voraussetzung, dass die Spalte ID eine Schlüsselspalte in der Tabelle EMPLOYEE ist und das jeder Mitarbeiter höchstens einen Manager hat, vermeidet dieser Join, nach einer weiteren übereinstimmenden Zeile in der Tabelle MANAGER suchen zu müssen.

Ein Join mit frühem Verlassen ist auch möglich, wenn die Abfrage eine Klausel DISTINCT enthält. Betrachten Sie zum Beispiel die folgende Abfrage, die die Namen von Automobilherstellern (MAKE) zurückgibt, die Modelle anbieten, die für über 30.000 $ verkauft werden.

```
select distinct make.name
  from make, model
  where
    make.make_id = model.make_id and
    model.price > 30000
```

Für jeden Hersteller (MAKE) muss lediglich festgestellt werden, ob eines der von ihm hergestellten Modelle für über 30.000 $ verkauft wird. Eine Verknüpfung eines Herstellers mit allen von ihm hergestellten Modellen, die für über 30.000 $ verkauft werden, ist nicht erforderlich, da dies nicht zur Richtigkeit des Abfrageergebnisses beiträgt.

Ein Join mit frühem Verlassen ist auch möglich, wenn der Join Daten für eine Klausel GROUP BY mit einer Spaltenfunktion MIN oder MAX liefert. Betrachten Sie zum Beispiel die folgende Abfrage, die Börsenkürzel (STOCK SYMBOL) mit dem jüngsten Datum vor dem Jahr 2000 zurückgibt, für die der Schlusspreis (CLOSE) einer bestimmten Aktie mindestens 10 % höher als der Eröffnungspreis (OPEN) ist:

```
select dailystockdata.symbol, max(dailystockdata.date) as date
  from sp500, dailystockdata
  where
    sp500.symbol = dailystockdata.symbol and
    dailystockdata.date < '01/01/2000' and
    dailystockdata.close / dailystockdata.open >= 1.1
  group by dailystockdata.symbol
```

Die *qualifizierte Menge* ist als die Menge von Zeilen aus der Tabelle DAILYSTOCKDATA definiert, die die Datums- und Preisvorgaben erfüllt und mit einem bestimmten Börsenkürzel aus der Tabelle SP500 durch einen Join verknüpft wird. Wenn die qualifizierte Menge aus der Tabelle DAILYSTOCKDATA (für jede Börsenkürzelzeile aus der Tabelle SP500) über das Datum (DATE) absteigend sortiert ist, muss nur die erste Zeile aus der qualifizierten Menge für jedes Kürzel zurückgegeben werden, da diese erste Zeile das jüngste Datum für ein bestimmtes Börsenkürzel enthält. Die übrigen Zeilen in der qualifizierten Menge sind nicht erforderlich.

Zusammengesetzte Tabellen

Wenn das Ergebnis des Joins zweier Tabellen eine neue Tabelle ist (die auch als *zusammengesetzte Tabelle* bezeichnet wird), wird diese Tabelle in der Regel zur äußeren Tabelle eines Joins mit einer weiteren inneren Tabelle. Ein solcher Join wird als *Composite-Outer-Join* bezeichnet, da die äußere Tabelle die zusammengesetzte Tabelle ist. 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* 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)) wird das Ergebnis des Joins (T3xT4) als *Composite-Inner-Join* bezeichnet. 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.

Replizierte MQTs in Umgebungen mit partitionierten Datenbanken

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.

Beachten Sie, dass die Replikation einer replizierten MQT in diesem Kontext eine datenbankinterne Replikation ist. Bei der datenbankübergreifenden Replikation spielen Subskriptionen, Steuertabellen und Daten, die sich in verschiedenen Datenbanken und auf verschiedenen Betriebssystemen befinden, eine Rolle.

Betrachten Sie das folgende Beispiel:

- Die Tabelle SALES (Verkauf) befindet sich im Mehrpartitionstabellenbereich mit dem Namen REGIONTABLESPACE und wird über die Spalte REGION unterteilt.
- Die Tabellen EMPLOYEE (Mitarbeiter) und DEPARTMENT (Abteilung) sind in einer Datenbankpartitionsgruppe mit einer Einzelpartition.

Sie erstellen eine replizierte MQT auf der Basis der Informationen in der Tabelle EMPLOYEE.

```
create table r_employee as (
  select empno, firstnme, midinit, lastname, workdept
    from employee
  )
  data initially deferred refresh immediate
  in regiontablespace
  replicated
```

Sie aktualisieren den Inhalt der replizierten MQT:

```
refresh table r_employee
```

Nach der Verwendung der Anweisung REFRESH sollten Sie das Dienstprogramm RUNSTATS für die replizierte Tabelle in gleicher Weise wie für andere Tabellen ausführen.

Die folgende Abfrage berechnet den Verkauf nach Mitarbeiter, die Summe für die Abteilung und die Gesamtsumme:

```
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 sich nur in einer Datenbankpartition befindet, verwendet der Datenbankmanager R_EMPLOYEE, d. h. die MQT, die in jede der Datenbankpartitionen repliziert wird, in der die Tabelle SALES gespeichert ist. Der Leistungsvorteil ergibt sich daraus, dass die Mitarbeiterinformationen zur Ausführung des Joins nicht an jede Datenbankpartition über das Netz gesendet werden müssen.

Replizierte MQTs in zusammengefassten Joins

Replizierte MQTs können auch bei der Zusammenfassung (Kollokation) von Joins helfen. Wenn beispielsweise ein Sternschema eine große Fakttabelle enthält, die sich über zwanzig Datenbankpartitionen erstreckt, sind die Joins zwischen der Fakttabelle und den Dimensionstabellen am effizientesten, wenn diese Tabellen durch Kollokation zusammenfasst 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 Fakttabelle nicht dem Verteilungsschlüssel der Fakttabelle entsprechen.

Betrachten Sie zum Beispiel eine Tabelle mit dem Namen FACT (C1, C2, C3, ...), die über Spalte C1 unterteilt ist, eine Tabelle mit dem Namen DIM1 (C1, dim1a, dim1b, ...), die über C1 unterteilt ist, eine Tabelle mit dem Namen DIM2 (C2, dim2a, dim2b, ...), die über C2 unterteilt ist, usw. In diesem Fall ist der Join zwischen FACT und DIM1 perfekt, da das Vergleichselement `dim1.c1 = fact.c1` von den durch Kollokation zusammengefasst Daten profitiert. Beide Tabellen sind über die Spalte C1 unterteilt.

Der Join mit der Tabelle DIM2 und dem Vergleichselement `dim2.c2 = fact.c2` kann jedoch nicht mit durch Kollokation zusammengefassten Daten realisiert werden, da FACT über die Spalte C1 unterteilt ist und nicht über die Spalte C2. In diesem Fall könnten Sie die Tabelle DIM2 in der Datenbankpartitionsgruppe der Fakttabelle replizieren, sodass der Join lokal in jeder Datenbankpartition erfolgen kann.

Wenn Sie eine replizierte MQT erstellen, kann die Quellentabelle eine Einzelpartitionstabelle oder eine Mehrpartitionstabelle in einer Datenbankpartitionsgruppe sein. In den meisten Fällen ist die replizierte Tabelle klein und kann in einer Datenbankpartitionsgruppe mit einer Einzelpartition untergebracht werden. Sie können die zu

replizierende Datenmenge einschränken, indem Sie nur einen Teil der Spalten der Tabelle angeben oder indem Sie die Anzahl der qualifizierten Zeilen durch Vergleichselemente begrenzen.

Eine replizierte MQT kann auch in einer Datenbankpartitionsgruppe mit mehreren Datenbankpartitionen erstellt werden, sodass Kopien der Quellentabelle in allen Datenbankpartitionen erstellt werden. Joins zwischen einer großen Fakttabelle 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 in der Quellentabelle nicht vorhanden sind, können Sie jedoch keine eindeutigen Indizes erstellen oder Integritätsbedingungen für replizierte Tabellen definieren, selbst wenn diese Integritätsbedingungen für die Quellentabelle gelten.

Auf replizierte Tabellen kann in einer Abfrage direkt verwiesen werden, jedoch können Sie nicht die Skalarfunktion DBPARTITIONNUM für eine replizierte Tabelle verwenden, um die Tabellendaten in einer bestimmten Datenbankpartition abzurufen.

Verwenden Sie die DB2-EXPLAIN-Funktion, um festzustellen, ob eine replizierte MQT vom Zugriffsplan für eine Abfrage genutzt wurde. Ob der Zugriffsplan, der vom Optimierungsprogramm ausgewählt wird, eine replizierte MQT verwendet, hängt von den Daten ab, die durch einen Join verknüpft werden müssen. Eine replizierte MQT könnte zum Beispiel dann nicht verwendet werden, wenn das Optimierungsprogramm feststellt, dass es günstiger wäre, die ursprüngliche Quellentabelle an die anderen Datenbankpartitionen der Datenbankpartitionsgruppe per Broadcast rundzusenden.

Joinstrategien für partitionierte Datenbanken

Joinstrategien für eine Umgebung mit partitionierten Datenbanken können sich von Strategien für eine nicht partitionierte Datenbankumgebung unterscheiden. Zur Verbesserung der Leistung können zusätzliche Techniken auf die Standardjoinmethoden angewendet werden.

Für Tabellen, die häufig durch einen Join verknüpft werden, sollte eine Tabellenkollokation in Betracht gezogen werden. In einer Umgebung mit partitionierten Datenbanken bezeichnet der Begriff *Tabellenkollokation* einen Zustand, der vorliegt, wenn zwei Tabellen, die dieselbe Anzahl kompatibler Patitionierungsschlüssel haben, in derselben Datenbankpartitionsgruppe gespeichert werden. Wenn dies der Fall ist, kann die Joinverarbeitung in der Datenbankpartition ausgeführt werden, in der die Daten gespeichert sind, und nur die Ergebnismenge muss an die Koordinatordatenbankpartition übertragen werden.

Tabellenwarteschlangen

In Beschreibungen von Joinmethoden in einer Umgebung mit partitionierten Datenbanken wird die folgende Terminologie verwendet:

- Eine *Tabellenwarteschlange* (auch als TQ (Table Queue) bezeichnet) ist ein Mechanismus zur Übertragung von Zeilen zwischen Datenbankpartitionen oder zwischen Prozessoren in einer Einzelpartitionsdatenbank.

- Eine *gezielt übertragene Tabellenwarteschlange* (auch als DTQ (Directed Table Queue) bezeichnet) ist eine Tabellenwarteschlange, in der Zeilen durch ein Hashverfahren an eine der empfangenden Datenbankpartitionen gesendet werden.
- Eine *Broadcast-Tabellenwarteschlange* (auch als BTQ (Broadcast Table Queue) bezeichnet) ist eine Tabellenwarteschlange, in der Zeilen an alle empfangenden Datenbankpartitionen, jedoch ohne Hashverfahren, gesendet werden.

Eine Tabellenwarteschlangen dient zur Übergabe von Tabellendaten zwischen folgenden Komponenten:

- Von einer Datenbankpartition zu einer anderen bei Verwendung partitionsübergreifender Parallelität
- Innerhalb einer Datenbankpartition bei Verwendung partitionsinterner Parallelität
- Innerhalb einer Datenbankpartition bei Verwendung einer Einzelpartitionsdatenbank

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 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 Abrufanweisung (FETCH) von einer Anwendung vorablesen. 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 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 Mischjoins

  Bei dieser Art von Tabellenwarteschlange bleibt die Reihenfolge gewahrt.
- Reguläre Tabellenwarteschlangen

  Reguläre Tabellenwarteschlangen wahren nicht die Reihenfolge.
- Empfangende Tabellenwarteschlange (manchmal auch als LTQ (Listener Table Queue) bezeichnet)

  Diese Tabellenwarteschlangen werden mit korrelierten Unterabfragen verwendet. Die Korrelationswerte werden an die Unterabfrage übergeben und die Ergebnisse mithilfe dieses Typs von Tabellenwarteschlange an den übergeordneten Abfrageblock zurückgeliefert.

Joinmethoden für partitionierte Datenbanken

Für Umgebungen mit partitionierten Datenbanken stehen verschiedene Joinmethoden zur Verfügung. Dazu gehören: zusammengefasste Joins, Broadcast-Outer-Table-Joins, Directed-Outer-Table-Joins, Directed-Inner-Table- und Directed-Outer-Table-Joins, Broadcast-Inner-Table-Joins und Directed-Inner-Table-Joins.

In den folgenden Diagrammen bezeichnen q1, q2 und q3 ('Queue') Tabellenwarteschlangen. Die gezeigten Tabellen sind auf zwei Datenbankpartitionen verteilt, und die Pfeile geben die Richtung an, in der die Tabellenwarteschlangen übertragen bzw. gesendet werden. Die Koordinatordatenbankpartition ist 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 der entsprechenden Verteilungsschlüssel müssen in den Gleichheitsvergleichselementen enthalten sein. Abb. 27 zeigt ein Beispiel.

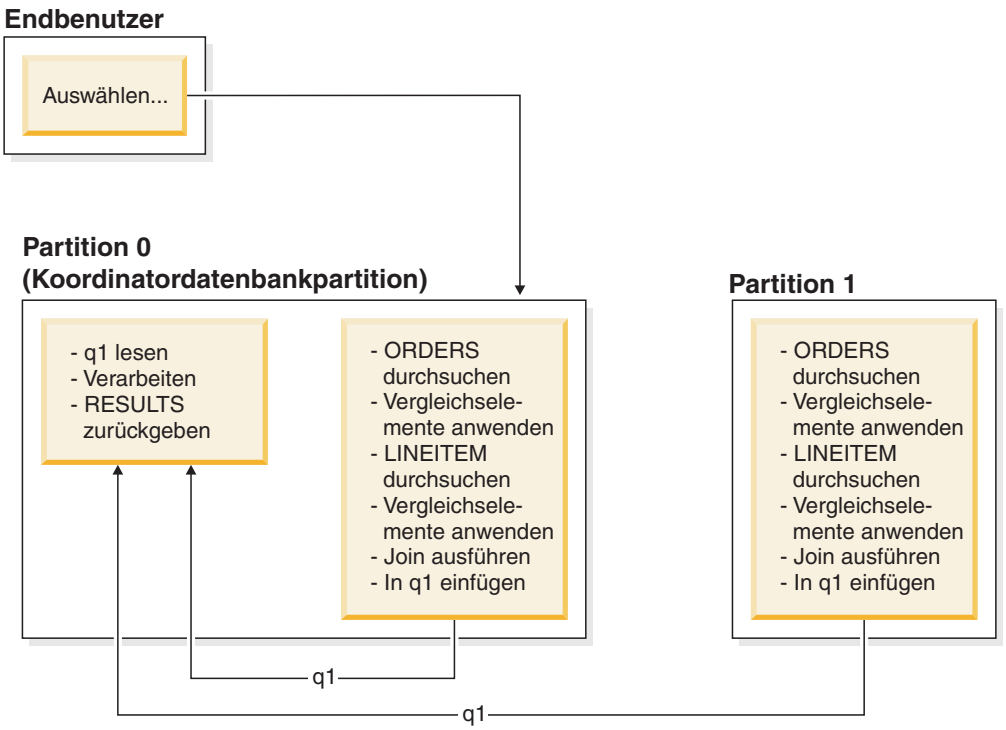


Abbildung 27. Beispiel für einen zusammengefassten Join

Die Tabellen LINEITEM und ORDERS werden beide über die Spalte ORDERKEY partitioniert. Der Join wird lokal in jeder Datenbankpartition ausgeführt. In diesem Beispiel wird folgendes Vergleichselement für den Join angenommen: `orders.orderkey = lineitem.orderkey`.

Replizierte MQTs (Materialized Query Tables) erhöhen die Wahrscheinlichkeit zusammengefasster Joins.

Broadcast-Outer-Table-Joins

Broadcast-Outer-Table-Joins (Joins mit rundgesendeter äußerer Tabelle) stellen eine parallele Joinstrategie dar, die angewendet werden kann, wenn zwischen den zu verknüpfenden Tabellen keine Equijoin-Vergleichselemente vorhanden sind. Sie kann außerdem in solchen Fällen verwendet werden, in denen sie sich als die Methode mit dem geringsten Aufwand herausstellt. 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 über die Spalten verteilt ist, die im Joinvergleichselement verwendet werden. Anstatt beide Tabellen zu teilen, kann es günstiger sein, die kleinere Tabelle an alle Datenbankpartitionen mit der größeren Tabelle per Broadcast rundzusenden. Abb. 28 zeigt ein Beispiel.

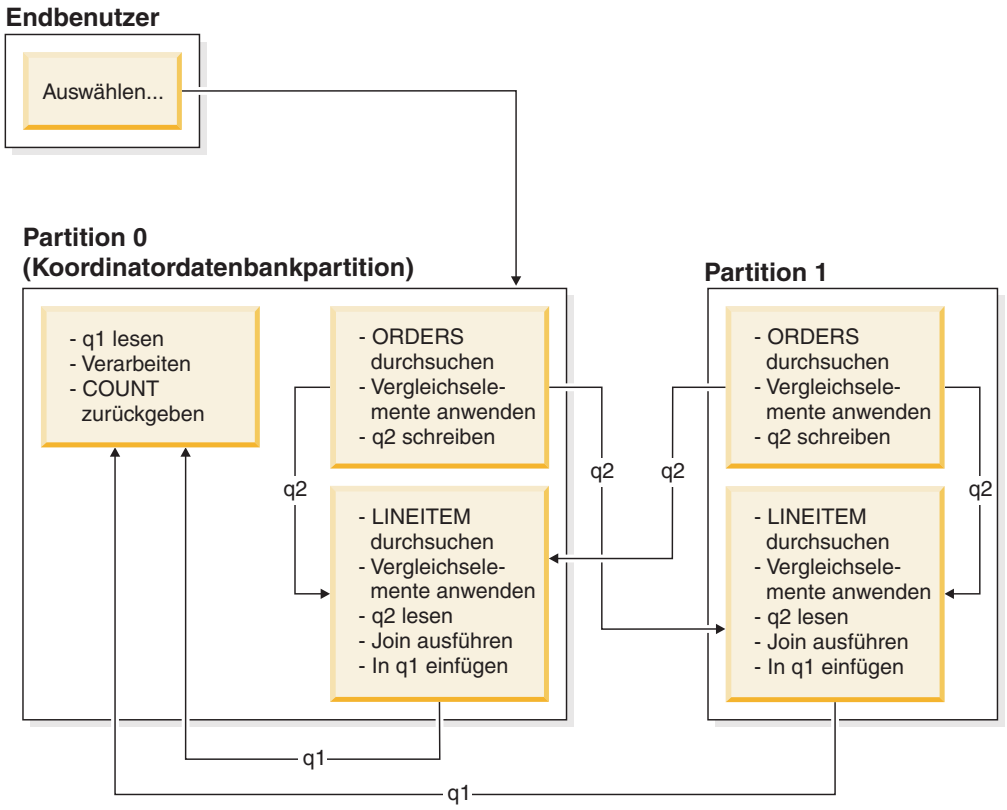
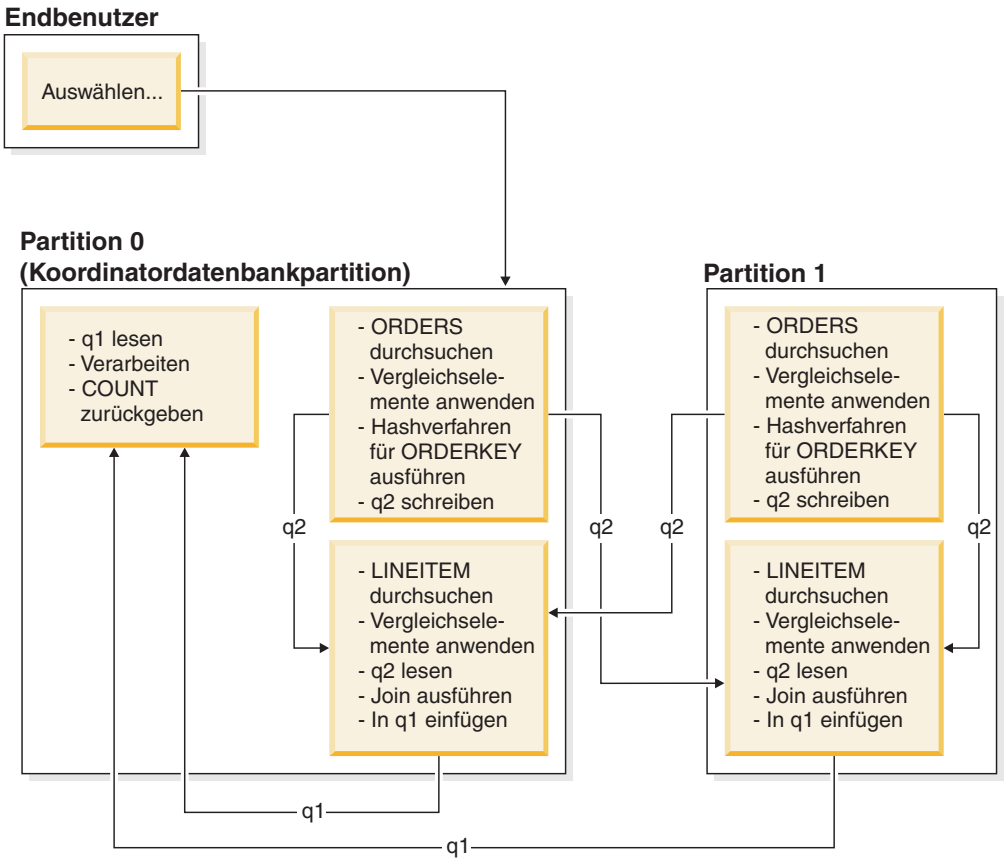


Abbildung 28. Beispiel für einen Broadcast-Outer-Table-Join

Die Tabelle ORDERS wird an alle Datenbankpartitionen mit der Tabelle LINEITEM gesendet. Tabellenwarteschlange q2 wird per Broadcast an alle Datenbankpartitionen der inneren Tabelle gesendet.

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. Abb. 29 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 29. 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. Abb. 30 zeigt ein Beispiel.

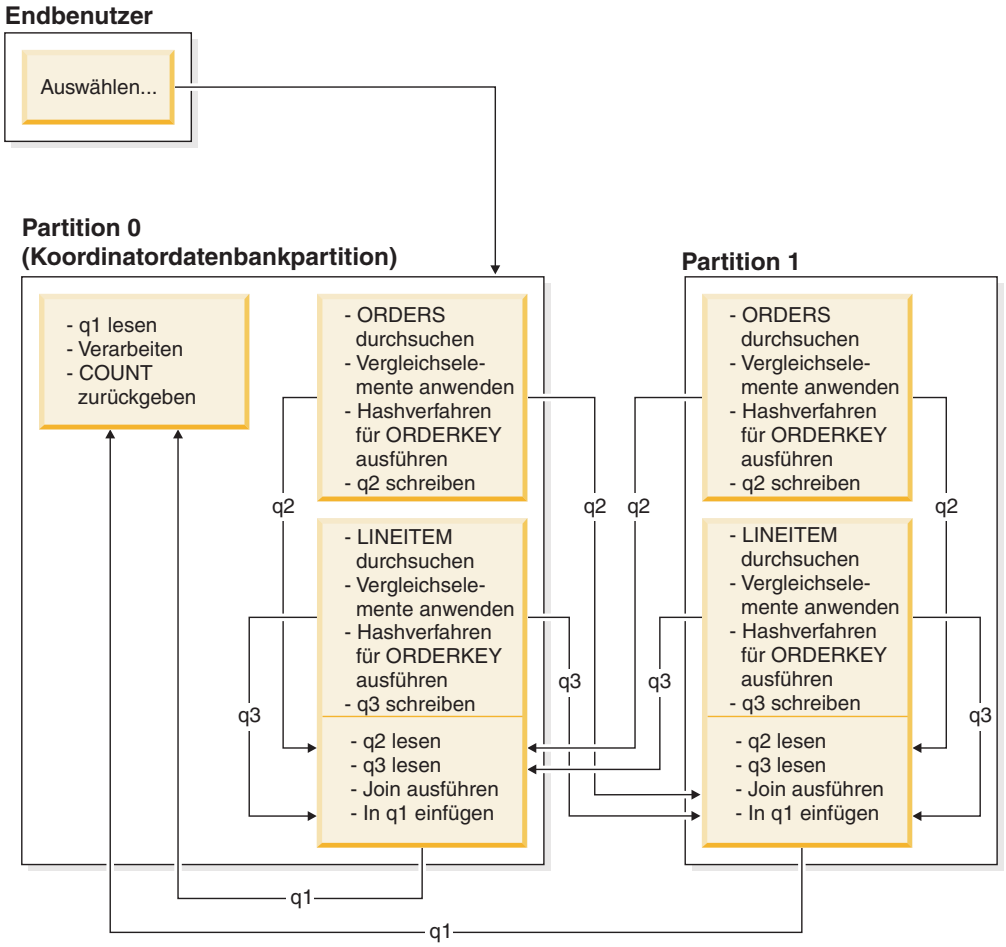
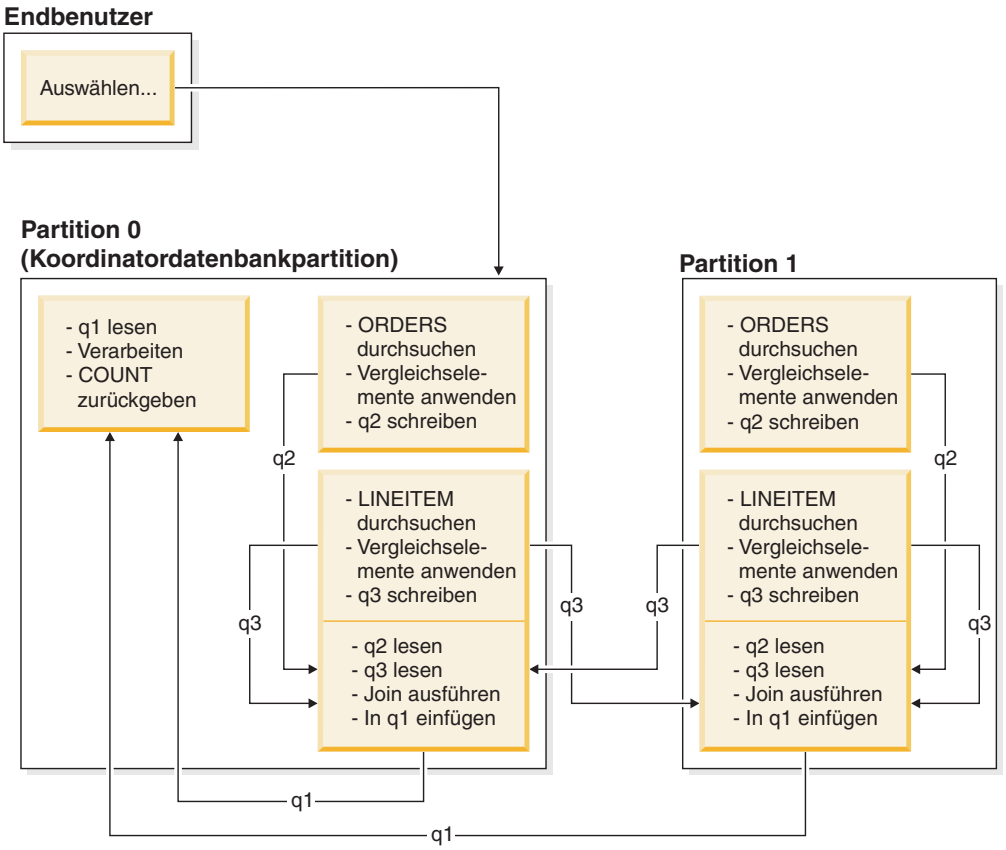


Abbildung 30. Beispiel für einen Directed-Inner-Table- und Directed-Outer-Table-Join

Es wird keine Tabelle über die Spalte ORDERKEY partitioniert. Für beide Tabellen wird das Hashverfahren ausgeführt und sie werden an neue Datenbankpartitionen gesendet, wo sie durch einen Join verknüpft werden. Beide Tabellenwarteschlangen q2 und q3 werden gezielt übertragen. In diesem Beispiel wird folgendes Vergleichselement für den Join angenommen: `orders.orderkey = lineitem.orderkey`.

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 Tabelle gesendet. Abb. 31 zeigt ein Beispiel.

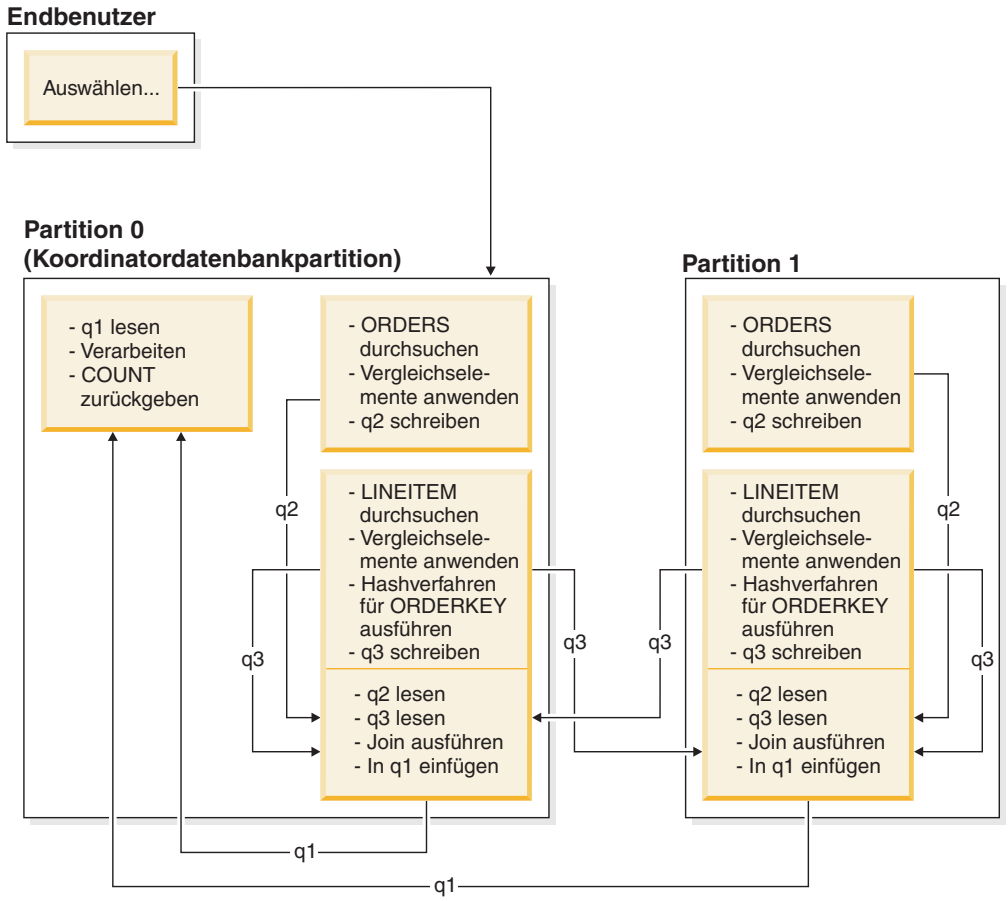


Die Tabelle LINEITEM wird an alle Datenbankpartitionen mit der Tabelle ORDERS gesendet. Tabellenwarteschlange q3 wird per Broadcast an alle Datenbankpartitionen der äußeren Tabelle gesendet.

Abbildung 31. 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. Abb. 32 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 32. Beispiel für einen Directed-Inner-Table-Join

Auswirkungen des Sortierens und Gruppierens auf die Abfrageoptimierung

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 handhabt sortierte Daten auf eine der folgenden Arten:

- Es leitet die Ergebnisse der Sortierung über eine Pipe weiter, wenn die Abfrage ausgeführt wird.
- Es lässt den Datenbankmanager die Sortierung intern verarbeiten.

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 einer 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 Sortierspeicher verfügbaren Speicherbereich in Anspruch nehmen, werden mehrere Sortierarbeitsgänge durchgeführt. In jedem Arbeitsgang wird eine Untermenge der Gesamtmenge von Zeilen sortiert. Jeder Arbeitsgang 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, werden die sortierten Untermengen zu einer einzigen sortierten Menge von Zeilen zusammengefügt. Wenn die Sortierung über eine Pipe geleitet wird, werden die Zeilen beim Zusammenfügen direkt an die Relational Data Services (RDS, Services für relationale Daten) übergegeben. (RDS sind die DB2-Komponente, die Anforderungen für den Zugriff auf den Inhalt einer Datenbank bzw. für die Bearbeitung dieses Inhalts verarbeitet.)

Pushdown von Gruppier- und Sortieroperatoren

In einigen Fällen kann das Optimierungsprogramm entscheiden, eine Sortieroperation oder eine Spaltenberechnung (Aggregation) von den RDS (Relational Data Services) an die Data Management Services (DMS, Datenverwaltungsservices) zu verschieben, was als Pushdown bezeichnet wird. (DMS sind die DB2-Komponente, die das Erstellen, Entfernen und Verwalten der Tabellen und Tabellendaten sowie den Zugriff darauf in einer Datenbank steuert.) Ein solcher Pushdown dieser Operationen verbessert die Leistung, da nun die Data Management Services Daten direkt an eine Sortier- oder Spaltenberechnungsroutine übergeben können. Ohne Pushdown übergeben die Data Management Services diese Daten zunächst an die Relational Data Services (RDS, Services für relationale Daten), die anschließend ihrerseits mit den Sortier- bzw. Spaltenberechnungsroutinen kommunizieren. Die folgende Abfrage kann zum Beispiel von dieser Art der Optimierung profitieren:

```
select workdept, avg(salary) as avg_dept_salary
  from employee
  group by workdept
```

Gruppierungsoperationen in Sortierungen

Wenn eine Sortierung 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. Der Vorteil wird sogar noch größer, wenn die Durchführung eines Teils der Gruppierung während des Sortierens die Notwendigkeit, dass die Sortierung einen Überlauf auf die Festplatte verursacht, verringert oder ausschließt.

Spaltenberechnungen während des Sortierns erfordern eine oder mehrere der drei folgenden Phasen der Spaltenberechnung, um sicherzustellen, dass die richtigen Ergebnisse zurückgegeben werden.

- In der ersten Phase der Spaltenberechnung, der *partiellen Spaltenberechnung*, werden die Ergebniswerte ermittelt, 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.
- In der zweiten Phase der Spaltenberechnung, der *Zwischenberechnung*, werden alle übergelaufenen Sortierdurchläufe aufgenommen und die Spaltenberechnungen für die Gruppierschlüssel fortgesetzt. Die Spaltenberechnung kann nicht abgeschlossen werden, weil die Gruppierschlüsselspalten eine Untergruppe der Verteilungsschlüsselspalten sind. Die Zwischenberechnung erstellt aus vorhandenen partiellen Spaltenberechnungen neue partielle Spaltenberechnungen. Diese Phase findet nicht immer statt. Sie wird sowohl für partitionsinterne als auch für partitionsübergreifende Parallelität verwendet. Bei der partitionsinternen Parallelität wird die Gruppierung beendet, wenn ein globaler Gruppierschlüssel verfügbar ist. Bei partitionsübergreifender Parallelität findet sie statt, wenn der Gruppierschlüssel 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.

- In der letzten Phase der Spaltenberechnung, der *Endberechnung*, werden alle partiellen Berechnungsergebnisse aufgenommen und die endgültigen Ergebniswerte erstellt. Dieser Schritt findet immer in einem Operator GROUP BY statt. Eine Sortierung kann keine vollständige Spaltenberechnung durchführen, weil es keine Garantie gibt, dass die Sortierung nicht geteilt wird. Die Komplettberechnung nimmt nicht berechnete Daten auf und generiert Endergebnisse. Diese Methode der Spaltenberechnung wird in der Regel dazu verwendet, Daten zu gruppieren, die sich bereits in der richtigen Reihenfolge befinden.

Optimierungsstrategien

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 der Parallelität, der bei der Kompilierung der SQL-Anweisung angegeben wurde.

Zur Parallelisierung eines Zugriffsplans 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 Tabellenwarteschlangen 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 im selben Index ausgeführt werden. Für parallele Tabellensuchen wird die Tabelle in Seiten- oder Zeilenbereiche unterteilt, die Subagenten zugewiesen werden. 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 eine 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 werden, werden die Maschinenressourcen effektiv genutzt.

Andere Zugriffsplanstrategien können eine unausgewogene Datenverteilung bei der Ausführung der Abfrage verursachen. Das Optimierungsprogramm wählt Par-

allelstrategien 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, um die Daten effizient auf alle Subagenten möglichst gleichmäßig 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 der Reihumsortierung 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 mithilfe 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 Sortierergebnis. Wenn die Anzahl von Zeilen gering ist, kann diese Art der Sortierung zur Neuverteilung des Datenstroms verwendet werden.
- Gemeinsame Sortierung

  Diese Art der Sortierung entspricht der replizierten Sortierung, abgesehen davon, dass 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 *gemeinsam genutzte temporäre Tabelle*. Die Subagenten können private oder parallele Suchoperationen über die gemeinsam genutzte temporäre Tabelle öffnen, je nachdem, ob der Datenstrom repliziert oder geteilt werden muss.

Strategien zur partitionsinternen parallelen Spaltenberechnung

Spaltenberechnungen (Aggregation) können von Subagenten parallel ausgeführt 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 ausführen. Dies ist möglich, wenn der Strom bereits aufgrund einer früheren partitionierten Sortierung über die Gruppierungsspalten geteilt ist.

Andernfalls kann der Subagent eine partielle Spaltenberechnung ausführen 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 ausgeführt werden. Parallele Joinsrategien 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 (Nested Loop Join) 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.

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 beruhen in erster Linie auf der verbesserten Effizienz von Blockindizes. Jedoch bietet das Clustering in mehreren Dimensionen auch den Vorteil eines schnelleren Datenabrufs.

Optimierungsstrategien für MDC-Tabellen können auch die Leistungsvorteile der partitionsinternen und partitionsübergreifenden Parallelität ausnutzen. MDC-Tabellen bieten die folgenden besonderen Vorteile:

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

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 diese Blöcke durchsuchen, um die Ergebnismenge schnell abzurufen.

Rolloutlöschung

Wenn Bedingungen das Löschen durch einen Rollout zulassen, wird dieses effizientere Verfahren zum Löschen von Zeilen aus MDC-Tabellen verwendet. Die folgenden Bedingungen müssen erfüllt sein:

- Bei der DELETE-Anweisung handelt es sich um eine DELETE-Anweisung mit Suche, nicht um eine positionierte DELETE-Anweisung (die Anweisung verwendet keine Klausel WHERE CURRENT OF).
- Es gibt keine WHERE-Klausel (alle Zeilen sind zu löschen) oder die einzigen Bedingungen in der WHERE-Klausel gelten für Dimensionen.
- Die Tabelle wurde nicht mit der Klausel DATA CAPTURE CHANGES definiert.
- Die Tabelle ist nicht die übergeordnete Tabelle in einer referenziellen Integritätsbeziehung.
- Für die Tabelle sind keine ON DELETE-Trigger definiert.
- Die Tabelle wird in keinen MQTs (Materialized Query Tables) verwendet, die sofort aktualisiert werden.
- Eine kaskadierende Löschoperation kommt für einen Rollout infrage, wenn der Fremdschlüssel eine Untermenge der Dimensionsspalten der Tabelle ist.
- Die Anweisung DELETE kann nicht in einer Anweisung SELECT enthalten sein, die an der temporären Tabelle ausgeführt wird, die die Menge der betroffenen Zeilen vor einer auslösenden SQL-Operation (durch die Klausel OLD TABLE AS in der Anweisung CREATE TRIGGER angegeben) angibt.

Bei einer Rolloutlöschung werden die gelöschten Datensätze nicht protokolliert. Stattdessen werden die Seiten, die die Datensätze enthalten, durch eine Neuformatierung von Teilen der Seiten optisch geleert. Die Änderungen an den neu formatierten Teilen werden protokolliert, die Datensätze selbst werden jedoch nicht protokolliert.

Das Standardverhalten *Rollout mit sofortiger Bereinigung* sieht vor, dass Satz-IDs (RIDs) beim Löschen bereinigt werden. Dieser Modus kann auch durch Setzen der Registrierdatenbankvariablen **DB2_MDC_ROLLOUT** auf den Wert IMMEDIATE oder durch Angeben von IMMEDIATE in der Anweisung SET CURRENT MDC ROLLOUT MODE angegeben werden. Es gibt keine Änderung bei der Protokollierung von Indexaktualisierungen im Vergleich zu einer normalen Löschoperation. Die Leistungsverbesserung hängt also davon ab, wie viele Satz-ID-Indizes vorhanden sind. Je weniger Satz-ID-Indizes vorhanden sind, desto größer ist die Verbesserung (als Prozentsatz von der Gesamtzeit und vom gesamten Protokollspeicher).

Ein Schätzwert für den eingesparten Platz im Protokoll kann anhand der folgenden Formel ermittelt werden:

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

Dabei ist *N* die Anzahl der gelöschten Datensätze, *S* die Gesamtgröße der gelöschten Datensätze, einschließlich Systemaufwand (z. B. Nullanzeiger und VARCHAR-Längen), und *P* die Anzahl der Seiten in den Blöcken, die die gelöschten Datensätze enthalten. Dieser Wert stellt die Verkleinerung in den tatsächlichen Protokolldaten dar. Die Einsparungen an erforderlichem aktiven Protokollspeicher betragen das Doppelte dieses Werts, da auch der Speicherbereich, der für ein Rollback reserviert war, eingespart wird.

Alternativ können Sie die Satz-ID-Indizes auch aktualisieren lassen, nachdem die Transaktion festgeschrieben wurde, indem Sie ein *Rollout mit verzögerter Bereinigung* verwenden. Dieser Modus kann auch durch Setzen der Registrierdatenbankvariablen **DB2_MDC_ROLLOUT** auf den Wert DEFER oder durch Angeben von DEFERRED in der Anweisung SET CURRENT MDC ROLLOUT MODE angegeben werden. Bei einem Rollout mit verzögerter Bereinigung werden Satz-ID-Indizes asynchron im Hintergrund bereinigt, nachdem die Löschoperation festgeschrieben wurde. Diese Rolloutmethode kann zu erheblich schnelleren Löschzeiten bei sehr umfangreichen Löschungen oder bei einer Tabelle mit mehreren Satz-ID-Indizes führen. Die Geschwindigkeit der gesamten Bereinigungsoperation erhöht sich, weil die Indizes bei einer verzögerten Indexbereinigung parallel bereinigt werden, während bei einer sofortigen Indexbereinigung jede Zeile des Index einzeln bereinigt wird. Darüber hinaus verringert sich der Platzbedarf des Transaktionsprotokolls für die Anweisung DELETE beträchtlich, weil die asynchrone Indexbereinigung die Indexaktualisierungen pro Indexseite und nicht pro Indexschlüssel protokolliert.

Anmerkung: Ein Rollout mit verzögerter Bereinigung erfordert zusätzliche Speicherressourcen, die aus dem Datenbankzwischenspeicher zugeordnet werden. Wenn der Datenbankmanager die erforderlichen Speicherstrukturen nicht zuordnen kann, schlägt der Rollout mit verzögerter Bereinigung fehl und eine Nachricht wird in das Protokoll mit Benachrichtigungen für die Systemverwaltung geschrieben.

Empfehlungen für die Verwendung eines Rollouts mit verzögerter Bereinigung

Wenn die Löschleistung der wichtigste Faktor ist und Satz-ID-Indizes für die Tabelle definiert sind, sollte ein Rollout mit verzögerter Bereinigung verwendet werden. Beachten Sie, dass vor der Indexbereinigung indexbasierte Suchoperationen in den durch Rollout gelöschten Blöcken je nach Umfang der gelöschten Daten eine leichte Leistungseinbuße erfahren. Darüber hinaus sollten auch folgende Aspekte bei der Entscheidung zwischen sofortiger und verzögerter Indexbereinigung berücksichtigt werden:

- Umfang der Löschoperation

  Wählen Sie einen Rollout mit verzögerter Bereinigung für sehr umfangreiche Löschoperationen aus. In Fällen, in denen DELETE-Anweisungen für Dimensionen häufig in vielen kleinen MDC-Tabellen ausgeführt werden, kann der Aufwand für die asynchrone Bereinigung von Indexobjekten den Vorteil der Zeiteinsparung während der Löschoperationen übersteigen.
- Anzahl und Typ von Indizes

  Wenn die Tabelle eine Reihe von Satz-ID-Indizes hat, die eine Verarbeitung auf Zeilenebene erfordern, sollte ein Rollout mit verzögerter Bereinigung verwendet werden.
- Blockverfügbarkeit

Wenn Sie wünschen, dass der Blockspeicherplatz durch die Löschoperation freigegeben wird, sodass er sofort nach dem Festschreiben der DELETE-Anweisung verfügbar ist, verwenden Sie einen Rollout mit sofortiger Bereinigung.

- Protokollspeicherbereich

  Wenn der Protokollspeicherbereich begrenzt ist, sollte bei umfangreichen Löschoperationen ein Rollout mit verzögerter Bereinigung verwendet werden.

- Speicherbeschränkungen

  Ein Rollout mit verzögerter Bereinigung benötigt zusätzlichen Speicherplatz im Datenbankzwischenspeicher für alle Tabellen, für die eine verzögerte Bereinigung ansteht.

Wenn Sie die Rolloutfunktionalität bei Löschoperationen inaktivieren wollen, setzen Sie die Registrierdatenbankvariable **DB2_MDC_ROLLOUT** auf den Wert OFF oder geben NONE in der Anweisung SET CURRENT MDC ROLLOUT MODE an.

Anmerkung: In DB2 Version 9.7 und späteren Releases wird ein Rollout mit verzögerter Bereinigung für eine datenpartitionierte MDC-Tabelle mit partitionierten Satz-ID-Indizes nicht unterstützt. Es werden nur die Modi NONE und IMMEDIATE unterstützt. Der Modus des Rollouts mit Bereinigung ist IMMEDIATE, wenn die Registrierdatenbankvariable **DB2_MDC_ROLLOUT** auf den Wert DEFER gesetzt ist oder wenn das Sonderregister CURRENT MDC ROLLOUT MODE auf den Wert DEFERRED gesetzt ist, um die Einstellung der Variablen **DB2_MDC_ROLLOUT** zu überschreiben.

Wenn nur nicht partitionierte Satz-ID-Indizes für die MDC-Tabelle vorhanden sind, wird ein Rollout mit verzögerter Indexbereinigung unterstützt.

Optimierungsstrategien für partitionierte Tabellen

Die Bezeichnung *Ausschluss von Datenpartitionen* bezieht sich auf die Fähigkeit des Datenbankservers, auf der Grundlage von Abfragevergleichselementen 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 Tabelle werden in mehrere Speicherobjekte auf der Basis von Spezifikationen partitioniert, die in der Klausel PARTITION BY der Anweisung CREATE TABLE angegeben werden. Diese Speicherobjekte können sich in verschiedenen Tabellenbereichen, im selben Tabellenbereich oder in einer Kombination solcher Tabellenbereiche befinden.

Das folgende Beispiel veranschaulicht die Leistungsvorteile des Ausschlusses von Datenpartitionen.

```
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 nur für Kundeninformationen des Jahres 2000.

```
select * from custlist
  where subsdate between '1/1/2000' and '12/31/2000'
```

Wie Abb. 33 zeigt, stellt der Datenbankserver fest, dass nur auf eine Datenpartition in Tabellenbereich TS4 zugegriffen werden muss, um diese Abfrage zu erfüllen.

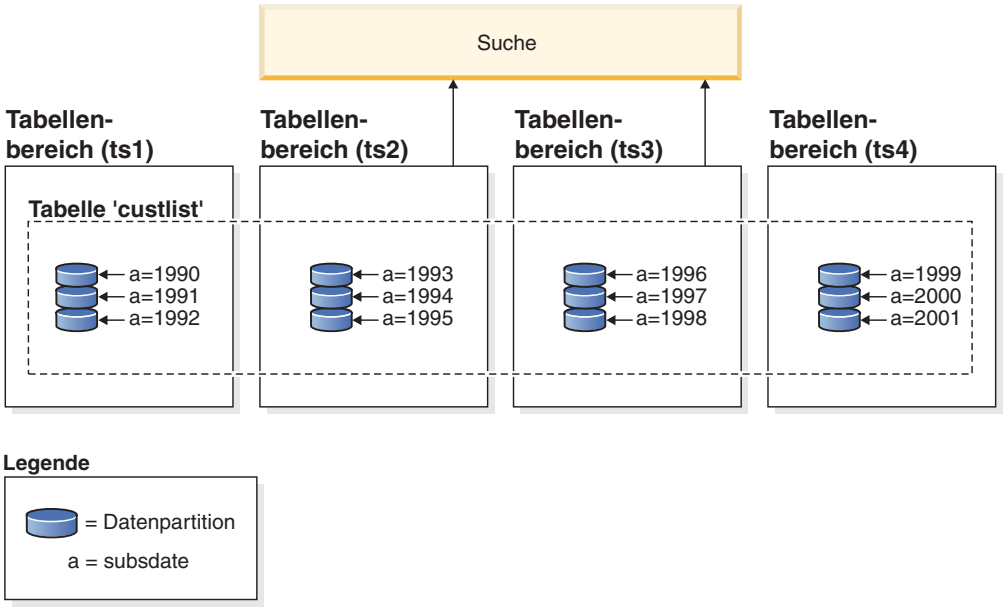


Abbildung 33. Die Leistungsvorteile des Datenpartitionsausschlusses

Ein weiteres Beispiel für den Ausschluss von Datenpartitionen basiert auf folgendem Schema:

```
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)
```

Nehmen Sie an, dass Sie die folgende Abfrage ausführen:

```
select * from multi
  where sale_date between '6/1/2005'
    and '7/31/2005' and region = 'NW'
```

Ohne Tabellenpartitionierung besteht ein wahrscheinlicher Plan in der logischen Verknüpfung der Indizes über AND. Beim logischen Verknüpfen von Indizes über AND (Index ANDing) werden die folgenden Aktionen ausgeführt:

- 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. 34 dargestellt, wird bei Tabellenpartitionierung der Index gelesen, um Übereinstimmungen für beide Spalten, d. h. REGION und SALE_DATE, zu ermitteln, sodass entsprechende Zeilen schnell abgerufen werden können.

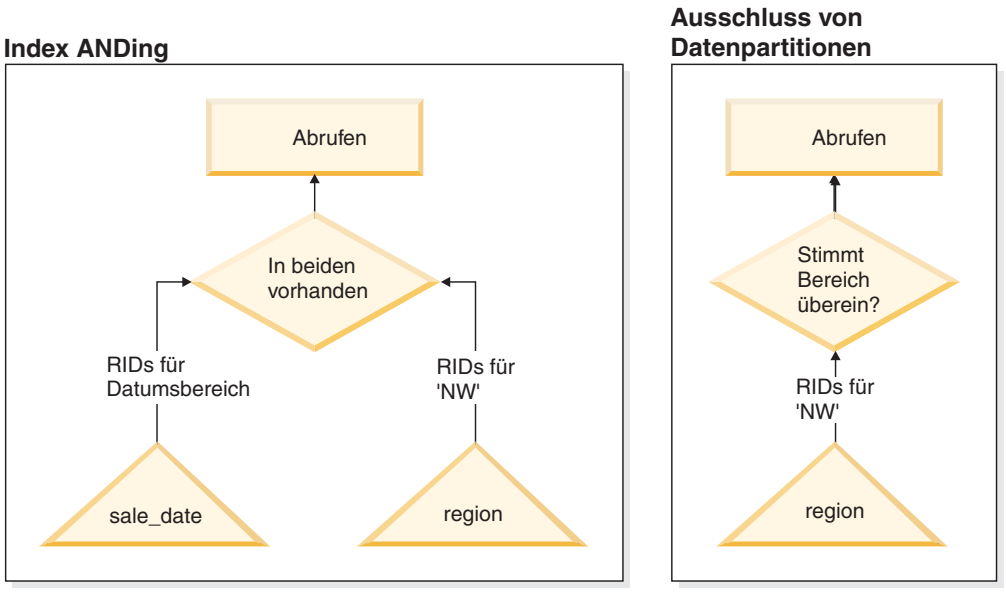


Abbildung 34. Der Entscheidungspfad des Optimierungsprogramms für die Tabellenpartitionierung und das logische Verknüpfen von Indizes über AND (Index ANDing)

DB2-EXPLAIN

Sie können auch die EXPLAIN-Funktion verwenden, um den Plan mit Datenpartitionsausschluss zu ermitteln, der vom Abfrageoptimierungsprogramm 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
DPNUMPRT: (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. Beispiel:

```
    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)

      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 bei Datenpartitionen, die mehrere Bereiche haben (d. h. Bereiche, die durch logisches OR verknüpft werden), zu erzielen. An der Tabelle SALES, die im vorigen Beispiel erstellt wurde, 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. Beispiel:

```
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, der zur Generierung der Spalten verwendet wird, außerdem monoton ist, übersetzt der Datenbankserver Vergleichselemente für die Quellenspalten in Vergleichselemente für die generierten Spalten, sodass der Ausschluss von Datenpartitionen über die generierten Spalten erfolgen kann. Beispiel:

```
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)
```

Die folgenden beiden 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.

Bei der Ausführung erfolgt, wenn T1 die innere Tabelle eines Joins mit Verschachtelungsschleife (NLJN) ist, der Ausschluss von Datenpartitionen auf der Basis der Vergleichselemente für jeden äußeren Wert von T2.A 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 im Tabellenbereich TS6 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 Tabellenbereich TS4.
4. Nach Anwendung von T1.A = 2 und T1.B > 15 durchsucht er die Datenpartitionen im Tabellenbereich TS4 von Tabelle T1, bis eine Zeile gefunden wird. Die erste qualifizierte Zeile, die gefunden wird, ist die dritte Zeile von T1.
5. Er verknüpft die übereinstimmenden Zeilen (Join).
6. Er durchsucht die Datenpartitionen in Tabellenbereich TS4 von Tabelle T1, 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 (wobei er den Wert 3 aus Spalte A nimmt). Dieses Verfahren wird fortgesetzt, bis alle Zeilen von T2 verarbeitet wurden.

Indizes zu XML-Daten

Ab DB2 Version 9.7 Fixpack 1 können Sie einen Index zu XML-Daten für eine partitionierte Tabelle entweder als partitionierten oder als nicht partitionierten Index erstellen. Standardmäßig wird ein partitionierter Index erstellt.

Partitionierte und nicht partitionierte XML-Indizes werden vom Datenbankmanager bei Einfüge-, Aktualisierungs- und Löschoperationen für Tabellen auf die gleiche Weise wie andere relationale Indizes für eine partitionierte Tabelle verwaltet. Nicht partitionierte Indizes zu XML-Daten für eine partitionierte Tabelle werden auf die gleiche Weise wie Indizes zu XML-Daten für eine nicht partitionierte Tabelle verwendet, um die Abfrageverarbeitung zu beschleunigen. Mithilfe des Abfragevergleichselements könnte zum Beispiel ermittelt werden, dass nur auf eine Untergruppe der Datenpartitionen in der partitionierten Tabelle zugegriffen werden muss, um die Abfrage zu erfüllen.

Der Ausschluss von Datenpartitionen und Indizes zu XML-Spalten können kombiniert werden, um die Abfrageleistung zu verbessern. Betrachten Sie die folgende partitionierte Tabelle:

```
create table employee (a int, b xml, c xml)
  index in tbspx
  partition by (a) (
    starting 0 ending 10,
    ending 20,
    ending 30,
    ending 40)
```

Betrachten Sie nun die folgende Abfrage:

```
select * from employee
  where a > 21
  and xmlexist('$doc/Person/Name/First[.="Eric"]'
    passing "EMPLOYEE"."B" as "doc")
```

Das Optimierungsprogramm kann die ersten beiden Partitionen aufgrund des Vergleichselements `a > 21` sofort ausschließen. Wenn der nicht partitionierte Index zu XML-Daten für Spalte B vom Optimierungsprogramm im Abfrageplan ausgewählt wird, kann eine Indexsuche im Index zu XML-Daten das Ergebnis des Datenpartitionsausschlusses des Optimierungsprogramms nutzen und nur Ergebnisse zurückgeben, die zu Partitionen gehören, die nicht durch die relationalen Vergleichselemente zum Datenpartitionsausschluss ausgeschlossen wurden.

Verbessern der Abfrageoptimierung mit MQTs (Materialized Query Tables)

MQTs (Materialized Query Tables, gespeicherte Abfragetabellen) bieten eine leistungsstarke Möglichkeit, die Antwortzeiten für komplexe Abfragen zu verbessern.

Dies gilt insbesondere für Abfragen, die eine oder mehrere der folgenden Operationen erfordern:

- Erstellen von Ergebnisdaten für eine oder mehrere Dimensionen
- Joins und Ergebnisberechnung von Daten einer Gruppe von Tabellen
- Bereitstellen von 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 Umgebung mit einer partitionierten Datenbank

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 in einer Abfrage, die auf die Basistabellen zugreift, eine MQT verwendet werden kann. Wenn eine MQT verwendet wird, kann die EXPLAIN-Funktion Informationen darüber bereitstellen, welche MQT ausgewählt wurde. In diesem Fall benötigen Benutzer Zugriffsberechtigungen für die Basistabellen und nicht für die weitergeleiteten MQTs.

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

Zur Veranschaulichung der Leistungsfähigkeit von MQTs zeigt das folgende Beispiel, wie eine mehrdimensionale Analyseabfrage von MQTs profitieren kann. Betrachten Sie ein Data-Warehouse, das eine Gruppe von Kunden und eine Gruppe 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 Schema) klassifiziert, weil es mit zwei großen Tabellen arbeitet, von denen eine die Transaktionsartikel und die andere die Kauftransaktionen enthält.

Eine Transaktion wird durch drei hierarchische Dimensionen beschrieben: 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 und Land, Region oder Gebiet, die in einer einzigen denormalisierten Tabelle gespeichert 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 für Verkäufe auf jeder Stufe der folgenden Hierarchien erstellt:

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

Viele Abfragen können aus diesen zusammengefassten Ergebnisdaten erfüllt werden. Das folgende Beispiel zeigt, wie eine MQT 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 außerdem 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.pg_salessum;
```

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

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

Obwohl die präzise Antwort für keine dieser Abfragen in der MQT enthalten ist, könnte der Aufwand zur Berechnung der Antwort mithilfe der MQT erheblich geringer ausfallen als der Aufwand zur 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 erzielen beträchtliche Leistungsverbesserungen, da sie die bereits berechneten Ergebnisse der Beispiel-MQT nutzen können.

Die erste Abfrage 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);
```

Die zweite Abfrage 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 bedeutender sind die potenziellen Leistungsverbesserungen für Antwortzeiten durch die Verwendung von MQTs. MQTs können sich überschneidende Arbeitsschritte von Abfragen effektiv eliminieren. Berechnungen werden nur einmal bei der Erstellung und einmal bei der Aktualisierung von MQTs durchgeführt. Der Inhalt von MQTs kann bei der Ausführung vieler Abfragen wiederverwendet werden.

EXPLAIN-Funktion

Die EXPLAIN-Funktion von DB2 stellt detaillierte Informationen zum Zugriffsplan bereit, den das Optimierungsprogramm für eine SQL- oder XQuery-Anweisung auswählt.

Die Informationen beschreiben die Entscheidungskriterien, anhand deren der Zugriffsplan ausgewählt wird, und können Ihnen helfen, die Anweisung oder die Konfiguration Ihrer Instanz zu optimieren, um die Leistung zu verbessern. Insbesondere können Ihnen die EXPLAIN-Informationen bei folgenden Aufgaben helfen:

- Untersuchen, wie der Datenbankmanager auf Tabellen und Indizes zur Erfüllung Ihrer Abfrage zugreift.
- Beurteilen Ihrer Maßnahmen zur Leistungsverbesserung. Untersuchen Sie nach dem Ändern einer Anweisung oder nach einer Konfigurationsänderung die neuen EXPLAIN-Informationen, um festzustellen, wie sich Ihre Maßnahme auf die Leistung ausgewirkt hat.

Zu den erfassten Informationen gehören folgende:

- Die Operationsfolge, die zur Verarbeitung der Abfrage ausgeführt wurde
- 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 wurde
- Werte für Hostvariablen, Parametermarken oder Sonderregister, die zur Reoptimierung der SQL- oder XQuery-Anweisung verwendet wurden

Die EXPLAIN-Funktion wird durch Absetzen der Anweisung EXPLAIN aufgerufen. Sie erfasst Informationen zu dem für eine bestimmte, mit EXPLAIN bearbeitbare Anweisung ausgewählten Zugriffsplan und schreibt diese Informationen in EXPLAIN-Tabellen. Sie müssen die EXPLAIN-Tabellen erstellen, bevor Sie die Anweisung EXPLAIN absetzen. Sie können auch die Sonderregister CURRENT EXPLAIN MODE oder CURRENT EXPLAIN SNAPSHOT definieren, die das Verhalten der EXPLAIN-Funktion steuern.

Informationen zu den Zugriffsrechten und Berechtigungen, die zur Verwendung des Dienstprogramms EXPLAIN erforderlich sind, finden Sie in der Beschreibung der Anweisung EXPLAIN. Die Berechtigung EXPLAIN kann einer Einzelperson erteilt werden, die Zugriff auf EXPLAIN-Informationen, jedoch nicht auf die Daten, die in der Datenbank gespeichert sind, benötigt. Diese Berechtigung umfasst einen Teil der Datenbankadministratorberechtigung und verfügt über kein inhärentes Zugriffsrecht auf Daten, die in Tabellen gespeichert sind.

Zum Anzeigen von EXPLAIN-Informationen können Sie entweder ein Befehlszeilentool oder Visual Explain verwenden. Das Tool, das Sie verwenden, bestimmt, wie Sie die Sonderregister definieren, die das Verhalten der EXPLAIN-Funktion steuern. 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.

Optimieren von SQL-Anweisungen mithilfe der EXPLAIN-Funktion

Die EXPLAIN-Funktion dient zum Anzeigen des Abfragezugriffsplans, der vom Abfrageoptimierungsprogramm zur Ausführung einer SQL-Anweisung ausgewählt wurde.

Sie gibt Auskunft über alle Details zu den relationalen Operationen, die zur Ausführung der SQL-Anweisung verwendet werden, wie zum Beispiel die Planoperatoren, ihre Argumente, die Reihenfolge der Ausführung und den Aufwand. Da der Abfragezugriffsplan einen der kritischsten Faktoren in der Abfrageleistung darstellt, ist es wichtig, die Ausgabe der EXPLAIN-Funktion bei der Diagnose von Problemen der Abfrageleistung verstehen zu können.

EXPLAIN-Informationen dienen in der Regel folgenden Zwecken:

- Analyse der Gründe für eine Änderung der Anwendungsleistung
- Beurteilung von Maßnahmen zur Leistungsoptimierung

Analyse von Leistungsänderungen

Zur Untersuchung der Ursachen für Änderungen in der Abfrageleistung führen Sie die folgenden Schritte aus, um EXPLAIN-Informationen „vor und nach" Änderungen zu erhalten:

1. Erfassen Sie EXPLAIN-Informationen für die Abfrage, bevor Sie Änderungen vornehmen, und speichern Sie die resultierenden EXPLAIN-Tabellen. Alternativ können Sie auch die Ausgabe aus dem Dienstprogramm **db2exfmt** speichern. Allerdings ermöglichen EXPLAIN-Informationen in den EXPLAIN-Tabellen eine einfache Abfrage der Tabellen durch SQL sowie eine differenziertere Analyse. Darüber hinaus bieten sie alle offenkundigen Verwaltungsvorteile von Daten, die in einem relationalen Datenbankmanagementsystem gespeichert sind. Das Tool **db2exfmt** kann jederzeit ausgeführt werden.
2. Speichern oder drucken Sie die aktuellen Katalogstatistiken, wenn Sie zum Anzeigen der entsprechenden Informationen nicht auf Visual Explain zugreifen können. Zu diesem Zweck können Sie auch den Befehl **db2look** verwenden. In DB2 Version 9.7 können Sie eine EXPLAIN-Momentaufnahme erfassen, wenn die EXPLAIN-Tabellen mit Daten gefüllt sind. Die EXPLAIN-Momentaufnahme enthält alle relevanten Statistikdaten für den Zeitpunkt, zu dem die Anweisung mit EXPLAIN bearbeitet wurde. Das Dienstprogramm **db2exfmt** führt eine automatische Formatierung der Statistikdaten aus, die in der Momentaufnahme enthalten sind. Dies ist insbesondere wichtig, wenn eine automatische oder echtzeitorientierte Statistikerfassung verwendet wird, weil die Statistikdaten, die für die Abfrageoptimierung verwendet werden, möglicherweise noch nicht in den Systemkatalogtabellen enthalten sind oder sich zwischen dem Zeitpunkt, zu dem die Anweisung mit EXPLAIN bearbeitet wurde, und dem Zeitpunkt, zu dem die Statistikdaten aus dem Systemkatalog abgerufen wurden, vielleicht geändert haben.

3. Sichern oder drucken Sie die Anweisungen der Datendefinitionssprache (DDL), einschließlich der Anweisungen für CREATE TABLE, CREATE VIEW, CREATE INDEX und CREATE TABLESPACE. Diese Task lässt sich auch mithilfe des Befehls **db2look** ausführen.

Die Informationen, die Sie auf diese Weise erfassen, stellen einen Referenzpunkt für zukünftige Analysen dar. Bei dynamischen SQL-Anweisungen können Sie diese Informationen bei der ersten Ausführung Ihrer Anwendung erfassen. Bei statischen SQL-Anweisungen können Sie diese Informationen auch beim Binden erfassen. Es ist besonders wichtig, diese Informationen zu erfassen, bevor eine größere Systemänderung, wie zum Beispiel die Installation eines neuen Service-Levels oder eines neuen DB2-Release durchgeführt wird oder bevor eine wesentliche Konfigurationsänderung, wie zum Beispiel das Hinzufügen oder Löschen von Datenbankpartitionen oder das Umverteilen von Daten, erfolgt. Dies hat den Grund, dass diese Typen von Systemänderungen zu negativen Änderungen an Zugriffsplänen führen können. Obwohl es nur selten zu einer Zugriffsplanregression kommen sollte, lassen sich Leistungsregressionen wahrscheinlich schneller lösen, wenn diese Informationen verfügbar sind. Zur Analyse einer Leistungsänderung vergleichen Sie die Informationen, die Sie zuvor 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 eines Zugriffsplans verwendet wird. Mithilfe der von Visual Explain oder **db2exfmt** angezeigten Informationen zu 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
- Erfassen von EXPLAIN-Informationen beim erneuten Binden (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 möglicherweise kein Problem mehr. Wenn der Index weiterhin nicht verwendet wird oder die Leistung problematisch bleibt, versuchen Sie eine zweite Maßnahme 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 oder das Erfassen aktueller Katalogstatistiken gehören.

Nach einer Änderung in einem dieser Bereiche können Sie mit der EXPLAIN-Funktion die Auswirkungen ermitteln, die die Änderung auf den ausgewählten Zugriffsplan hatte. Wenn Sie zum Beispiel einen Index oder eine MQT (Materialized Query Table, gespeicherte Abfragetabelle) gemäß den Richtlinien für Indizes hinzufügen, können Sie mithilfe 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 Vergleichstestverfahren (Benchmark Tests).

Richtlinien zur Erfassung von EXPLAIN-Informationen

EXPLAIN-Daten können auf Anforderung erfasst werden, wenn eine SQL- oder XQuery-Anweisung kompiliert wird.

Wenn SQL- oder XQuery-Anweisungen zum inkrementellen Binden während der Ausführung kompiliert werden, werden Daten während der Ausführung, nicht während des Bindens, in die EXPLAIN-Tabellen eingetragen. Für solche Anweisungen sind das eingefügte Tabellenqualifikationsmerkmal und die Berechtigungs-ID, die des Paketeigners und nicht die des Benutzers, der das Paket ausführt.

Die EXPLAIN-Informationen werden nur erfasst, wenn eine 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-Funktion aktiv ist. Wenn Sie dieselbe Anweisung PREPARE mehrmals für die gleiche Abfrageanweisung ausführen, erfolgt das Kompilieren der Abfrage und das Erfassen der EXPLAIN-Daten jedes Mal, wenn diese Anweisung vorbereitet oder ausgeführt wird.

Wenn ein Paket mit der Bindeoption REOPT ONCE oder ALWAYS gebunden wird, werden SQL- oder XQuery-Anweisungen, die Hostvariablen, Parametermarken, globale Variablen oder Sonderregister enthalten, kompiliert. Der Zugriffspfad wird mit realen Werten für diese Variablen, wenn sie bekannt sind, und mit Standardschätzwerten, wenn die Werte bei der Kompilierung nicht bekannt sind, erstellt.

Wenn die Option REOPT ONCE verwendet wird, wird ein Versuch unternommen, die angegebene SQL- oder XQuery-Anweisung mit der gleichen Anweisung im Paketcache abzugleichen. Werte für diese bereits reoptimierte Abfrageanweisung im Cache werden zur Reoptimierung der angegebenen Abfrageanweisung verwendet. Wenn der Benutzer über die erforderlichen Zugriffsrechte verfügt, enthalten die EXPLAIN-Tabellen in diesem Fall den neuen, reoptimierten Zugriffsplan sowie die Werte, die für die Reoptimierung verwendet wurden.

In einem Datenbanksystem 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.
  - Wenn das Sonderregister CURRENT EXPLAIN MODE auf einen der folgenden Werte gesetzt ist:
    - 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 Informationen zu empfohlenen Indizes werden in der Tabelle ADVISE_INDEX gespeichert. Die Abfrageanweisung wird jedoch nicht ausgeführt.
- EVALUATE INDEXES: Der SQL- und XQuery-Compiler verwendet Indizes zur Bewertung, die vom Benutzer in die Tabelle ADVISE_INDEX eingefügt wurden. In diesem Modus werden alle dynamischen Anweisungen so mit EXPLAIN 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 EXPLAIN-Daten für statische oder dynamische SQL- oder XQuery-Anweisungen bei der Reoptimierung der Anweisung während der Ausführung, wenn für Hostvariablen, Parametermarken, globale Variablen oder Sonderregister tatsächliche Werte zur Verfügung stehen.

• Wenn die Option EXPLAIN ALL im Befehl **BIND** oder **PREP** angegeben wurde, erfasst der Abfragecompiler während der Ausführung EXPLAIN-Daten für dynamische SQL- und XQuery-Anweisungen, selbst wenn das Sonderregister CURRENT EXPLAIN MODE auf NO gesetzt ist.

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. Zusätzliche Informationen zur EXPLAIN-Momentaufnahme, einschließlich Informationen zu Datenobjekten und Datenoperatoren, werden durch Visual Explain selbst zur Verfügung gestellt.

EXPLAIN-Momentaufnahmedaten werden erfasst, wenn eine SQL- oder XQuery-Anweisung kompiliert wird und EXPLAIN-Daten wie folgt angefordert wurden:

• Statische SQL- und XQuery-Anweisungen oder SQL- und XQuery-Anweisungen zum inkrementellen Binden

  Eine EXPLAIN-Momentaufnahme wird erfasst, wenn eine der beiden Klauseln EXPLSNAP ALL oder EXPLSNAP YES im Befehl **BIND** oder **PREP** angegeben wird oder wenn das Quellenprogramm eine statische EXPLAIN-Anweisung 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 ersten Klausel werden nur EXPLAIN-Momentaufnahmedaten, mit der zweiten Klausel sämtliche EXPLAIN-Informationen erfasst.
  – Wenn das Sonderregister CURRENT EXPLAIN SNAPSHOT auf einen der folgenden Werte gesetzt ist:
    - YES: Der SQL- und XQuery-Compiler erfasst EXPLAIN-Momentaufnahmedaten und führt die Abfrageanweisung aus.
    - EXPLAIN: Der SQL- und XQuery-Compiler erfasst EXPLAIN-Momentaufnahmedaten, führt die Abfrageanweisung jedoch nicht aus.

– Sie geben die Option EXPLSNAP ALL im Befehl **BIND** oder **PREP** an. Der Abfragecompiler erfasst EXPLAIN-Momentaufnahmedaten während der Ausführung, selbst wenn das Sonderregister CURRENT EXPLAIN SNAPSHOT auf NO gesetzt ist.

Richtlinien zur Erfassung von EXPLAIN-Informationen für Abschnitte

Die EXPLAIN-Funktionalität für Abschnitte erfasst (entweder direkt oder durch Tools) EXPLAIN-Informationen zu einer Anweisung nur unter Verwendung des Inhalts des Laufzeitabschnitts. Die EXPLAIN-Funktion für Abschnitte ist der Funktionalität ähnlich, die durch den Befehl **db2expln** bereitgestellt wird, jedoch bietet die EXPLAIN-Funktion für Abschnitte einen Detaillierungsgrad, der dem von der allgemeinen EXPLAIN-Funktion nahe kommt.

Durch Ausführen der EXPLAIN-Funktion für eine Anweisung unter Verwendung des Inhalts des Laufzeitabschnitts können Sie Informationen und Diagnosedaten dazu abrufen, was tatsächlich ausgeführt wird (bzw. ausgeführt wurde, wenn der Abschnitt nach der Ausführung erfasst wurde). Dies unterscheidet sich von der Ausführung einer Anweisung EXPLAIN, durch die ein anderer Zugriffsplan generiert werden kann. (Dies ist z. B. der Fall bei dynamischem SQL, wenn die Statistiken seit der letzten Ausführung der Anweisung aktualisiert wurden, sodass ein anderer Zugriffsplan gewählt wird, wenn die Anweisung EXPLAIN die mit EXPLAIN zu bearbeitende Anweisung kompiliert.)

Die EXPLAIN-Schnittstellen für Abschnitte füllen die EXPLAIN-Tabellen mit Informationen, die denen ähnlich sind, die durch eine EXPLAIN-Anweisung generiert werden. Es bestehen jedoch einige Unterschiede. Nachdem die Daten in die EXPLAIN-Tabellen geschrieben wurden, können sie nach Wunsch mit beliebigen der vorhandenen EXPLAIN-Tools (z. B. mit dem Befehl **db2exfmt**) verarbeitet werden.

EXPLAIN-Schnittstellen für Abschnitte

In der folgenden Liste sind vier Schnittstellenprozeduren aufgeführt, mit denen EXPLAIN-Operationen für Abschnitte ausgeführt werden können. Die Prozeduren unterscheiden sich nur durch Eingabe, die angegeben wird (d. h., durch die Methode, mit der der Abschnitt lokalisiert wird):

EXPLAIN_FROM_ACTIVITY
Empfängt die Anwendungs-ID, die Aktivitäts-ID, die UOW-ID und den Namen des Aktivitätsereignismonitors als Eingabe. Die Prozedur sucht nach dem Abschnitt, der der angegebenen Aktivität im Aktivitätsereignismonitor entspricht. (Eine SQL-Aktivität ist eine bestimmte Ausführung eines Abschnitts.) Die Daten einer EXPLAIN-Operation für Abschnitte, die mit dieser Schnittstelle arbeitet, enthalten Ist-Daten für einen Abschnitt, weil eine bestimmte Ausführung des Abschnitts ausgeführt wird.

EXPLAIN_FROM_CATALOG
Empfängt den Paketnamen, das Paketschema, die eindeutige ID und die Abschnittsnummer als Eingabe. Die Prozedur durchsucht die Katalogtabellen nach dem angegebenen Abschnitt.

EXPLAIN_FROM_DATA
Empfängt die Kennung des ausführbaren Abschnitts, den Abschnitt und den Anweisungstext als Eingabe.

EXPLAIN_FROM_SECTION

Empfängt die Kennung des ausführbaren Abschnitts und die Position als Eingabe, wobei die Position durch eine der folgenden Angaben bezeichnet wird:

- Paketcache im Speicher
- Name des Ereignismonitors für den Paketcache

Die Prozedur sucht nach dem Abschnitt an der angegebenen Position.

Eine Kennung (ID) für einen ausführbaren Abschnitt identifiziert einen Abschnitt eindeutig und konsistent. Die Kennung des ausführbaren Abschnitts ist ein nicht transparentes, binäres Token, das auf dem Datenserver für jeden Abschnitt generiert wird, der ausgeführt wurde. Die Kennung des ausführbaren Abschnitts wird als Eingabe zur Abfrage von Überwachungsdaten für den Abschnitt sowie zur Ausführung einer EXPLAIN-Operation für den Abschnitt verwendet.

In jedem Fall führt die Prozedur eine EXPLAIN-Operation unter Verwendung der Informationen aus, die in dem angegebenen Laufzeitabschnitt enthalten sind, und schreibt die EXPLAIN-Informationen in die EXPLAIN-Tabellen, die durch einen Eingabeparameter *explain_schema* angegeben werden. Es liegt in der Verantwortung des aufrufenden Programms, nach dem Aufruf der Prozedur eine Commitoperation auszuführen.

Untersuchen der Abfrageleistung mit EXPLAIN-Informationen aus einem Abschnitt:

Die EXPLAIN-Funktion kann zum Untersuchen des Zugriffsplans für eine bestimmte Anweisung in der Form, *in der sie tatsächlich ausgeführt werden wird* (oder in der sie bereits ausgeführt wurde), verwendet werden. Dies geschieht durch Generieren des Zugriffsplans aus dem Abschnitt für die Anweisung selbst. Im Unterschied dazu wird durch die Anweisung EXPLAIN der Zugriffsplan durch Neukompilieren der Anweisung erstellt. Die aus diesen beiden Methoden der Zugriffsplanerstellung resultierenden Zugriffspläne können sich voneinander unterscheiden. Wenn beispielsweise die Anweisung in einem Abschnitt vor zwei Stunden kompiliert wurde, kann sich der von ihr verwendete Zugriffsplan von dem unterscheiden, der durch Ausführen der Anweisung EXPLAIN für die Anweisung generiert wird.

Wenn Aktivitätsereignismonitordaten verfügbar sind, können Sie den Zugriffsplan für den Abschnitt nach dem Ausführen mit der Prozedur EXPLAIN_FROM_ACTIVITY erstellen. (Wenn die Ist-Daten des Abschnitts erfasst werden, können Sie diese Daten auch zusammen mit den geschätzten Daten anzeigen, die durch die EXPLAIN-Funktion im Zugriffsplan generiert wurden. Weitere Informationen hierzu finden Sie im Abschnitt „Erfassen und Zugreifen auf Ist-Daten für Abschnitte" auf Seite 301.)

Wenn für die Anweisung keine Aktivitätsereignismonitordaten verfügbar sind, können Sie mit der Prozedur EXPLAIN_FROM_SECTION den Zugriffsplan für die Anweisung in der Form generieren, in der sie ausgeführt werden wird. Die Basis dafür bildet der Abschnitt, der im Paketcache gespeichert wird. In diesem Abschnitt wird gezeigt, wie Zugriffsplandaten für eine Anweisung auf der Basis von Abschnittsdaten im Paketcache mit der Prozedur EXPLAIN_FROM_SECTION angezeigt werden.

Vorbereitende Schritte

Bei dieser Task wird davon ausgegangen, dass Sie bereits die EXPLAIN-Tabellen erstellt haben, die für die EXPLAIN-Funktion erforderlich sind.

Informationen zu diesem Vorgang

Gehen Sie in diesem Abschnitt davon aus, dass Sie mit der EXPLAIN-Funktion die Anweisung im Paketcache mit der größten CPU-Belastung untersuchen wollen.

Einschränkungen

Vorgehensweise

Im ersten Teil dieser Vorgehensweise wird gezeigt, wie die Anweisung mit der größten CPU-Belastung ermittelt wird. Danach wird demonstriert, wie mit der Prozedur EXPLAIN_FROM_SECTION die Zugriffsplandaten für die betreffende Anweisung in der Form, in der sie tatsächlich ausgeführt werden wird, angezeigt werden.

1. Ermitteln Sie die Anweisung, die am meisten Prozessorzeit benötigt:

```
SELECT SECTION_TYPE,
    CASE
      WHEN SUM(NUM_COORD_EXEC_WITH_METRICS) > 0 THEN
           SUM(TOTAL_CPU_TIME)/SUM(NUM_COORD_EXEC_WITH_METRICS)
      ELSE
           0
    END as  AVG_CPU_TIME,
    EXECUTABLE_ID,
    VARCHAR(STMT_TEXT, 200) AS TEXT
    FROM TABLE(MON_GET_PKG_CACHE_STMT ( 'D', NULL, NULL, -2)) as T
    WHERE T.NUM_EXEC_WITH_METRICS <> 0 AND STMT_TYPE_ID LIKE 'DML%'
    GROUP BY EXECUTABLE_ID, VARCHAR(STMT_TEXT, 200)
    ORDER BY AVG_CPU_TIME DESC
```

   Dieser SQL-Code ist so geschrieben, dass die Teilung durch 0 vermieden wird, wenn die durchschnittliche Prozessorzeit für die Teilkomponenten berechnet wird. Mit diesem Code werden auch ausschließlich DML-Anweisungen untersucht, da die EXPLAIN-Funktion nicht mit DDL-Anweisungen arbeitet. Diese Abfrage ergibt folgende Ergebnisse:

```
SECTION_TYPE AVG_CPU_TIME         EXECUTABLE_ID                                                                    TEXT
------------ -------------------- -------------------------------------------------------------------------------- --------------------------------------------------
D                          250000 x'01000000000000005F000000000000000000000000200201011081356293590000' select cust_last_name, cust_cc_number, cust_intere
SQL0445W  Value "SELECT POLICY FROM SYSTOOLS.POLICY WHERE MED='DB2TableMainte"
has been truncated.  SQLSTATE=01004

D                           15625 x'0100000000000000 1B000000000000000000000000200201011081350176250000' UPDATE SYSTOOLS.HMON_ATM_INFO SET STATS_LOCK = 'N'
D                           15625 x'0100000000000000 20000000000000000000000000200201011081350182960001' UPDATE SYSTOOLS.HMON_ATM_INFO AS ATM SET ATM.STATS
D                           15625 x'0100000000000000 21000000000000000000000000200201011081350183120001' UPDATE SYSTOOLS.HMON_ATM_INFO AS ATM SET STATS_FLA
D                            7812 x'0100000000000000 15000000000000000000000000200201011081350169840000' SELECT COLNAME, TYPENAME FROM  SYSCAT.COLUMNS WHER
D                               0 x'0100000000000000 16000000000000000000000000200201011081350171560000' SELECT TRIGNAME FROM  SYSCAT.TRIGGERS WHERE TABNAM
D                               0 x'0100000000000000 19000000000000000000000000200201011081350174840000' SELECT TABNAME FROM SYSCAT.TABLES  WHERE TABNAME='
D                               0 x'0100000000000000 1A000000000000000000000000200201011081350175000000' SELECT TABNAME FROM SYSCAT.TABLES  WHERE TABNAME='
D                               0 x'0100000000000000 1C000000000000000000000000200201011081350177500002' SELECT CREATOR, NAME, CTIME FROM SYSIBM.SYSTABLES
D                               0 x'0100000000000000 1D000000000000000000000000200201011081350178280001' SELECT CREATE_TIME FROM SYSTOOLS.HMON_ATM_INFO WHE
D                               0 x'0100000000000000 1E000000000000000000000000200201011081350180000001' DELETE FROM SYSTOOLS.HMON_ATM_INFO AS ATM WHERE NO
D                               0 x'0100000000000000 1F000000000000000000000000200201011081350180930000' SELECT POLICY FROM SYSTOOLS.POLICY WHERE MED='DB2T
D                               0 x'0100000000000000 22000000000000000000000000200201011081350183280001' SELECT IBM.TID, IBM.FID FROM SYSIBM.SYSTABLES AS I
D                               0 x'0100000000000000 23000000000000000000000000200201011081350183430001' SELECT POLICY FROM SYSTOOLS.POLICY WHERE MED='DB2C

  14 record(s) selected with 1 warning messages printed.
```

2. Generieren Sie auf der Basis der Ausgabe für diese Abfrage mit der Prozedur EXPLAIN_FROM_SECTION EXPLAIN-Informationen aus dem Abschnitt für die Anweisung mit der größten CPU-Belastung:

```
CALL EXPLAIN_FROM_SECTION (x'01000000000000005F000000000000000000000000200201011081356293590000' ,'M', NULL, 0, NULL, ?, ?, ?, ?, ? )
```

   Die Ausgabe der Prozedur EXPLAIN_FROM_SECTION sieht wie folgt aus:

```
Wert der Ausgabeparameter
-------------------------
Parametername: EXPLAIN_SCHEMA
Parameterwert: DB2DOCS

Parametername: EXPLAIN_REQUESTER
Parameterwert: DB2DOCS

Parametername: EXPLAIN_TIME
Parameterwert: 2010-11-08-13.57.52.984001

Parametername: SOURCE_NAME
Parameterwert: SQLC2H21

Parametername: SOURCE_SCHEMA
Parameterwert: NULLID

Parametername: SOURCE_VERSION
Parameterwert:
```

3. Jetzt können Sie die EXPLAIN-Informationen untersuchen, indem Sie die EXPLAIN-Tabellen mit SQL untersuchen oder mit dem Befehl **db2exfmt** die Informationen so formatieren, dass ihre Lesbarkeit verbessert wird. Wenn Sie beispielsweise den Befehl **db2exfmt -d gsdb -e db2docs -w 2010-11-08-13.57.52.984001 -n SQLC2H21 -s NULLID -t -#0** für die EXPLAIN-Informationen ausführen, die im vorangehenden Schritt erfasst wurden, wird dadurch folgende Ausgabe generiert:

```
Connecting to the Database.
DB2 Universal Database Version 9.7, 5622-044 (c) Copyright IBM Corp. 1991, 2008
Licensed Material - Program Property of IBM
IBM DATABASE 2 Explain Table Format Tool



******************** EXPLAIN INSTANCE ********************

DB2_VERSION:       09.07.2
SOURCE_NAME:       SQLC2H21
SOURCE_SCHEMA:     NULLID
SOURCE_VERSION:
EXPLAIN_TIME:      2010-11-08-13.57.52.984001
EXPLAIN_REQUESTER: DB2DOCS

Database Context:
----------------
   Parallelism:          None
   CPU Speed:            8.029852e-007
   Comm Speed:           100
   Buffer Pool size:     21418
   Sort Heap size:       6590
   Database Heap size:   1196
   Lock List size:       21386
   Maximum Lock List:    97
   Average Applications: 1
   Locks Available:      663821

Package Context:
---------------
   SQL Type:           Dynamic
   Optimization Level: 5
   Blocking:           Block All Cursors
   Isolation Level:    Cursor Stability



---------------- STATEMENT 1  SECTION 201 ----------------
   QUERYNO:       0
   QUERYTAG:      CLP
   Statement Type:   Select
   Updatable:        No
   Deletable:        No
   Query Degree:     1
```

```
Original Statement:
------------------
select cust_last_name, cust_cc_number, cust_interest_code
from gosalesct.cust_crdt_card C, gosalesct.cust_customer D,
        gosalesct.cust_interest E
where C.cust_code=d.cust_code AND c.cust_code=e.cust_code
group by d.cust_last_name,    c.cust_cc_number, e.cust_interest_code
order by d.cust_last_name ASC, c.cust_cc_number DESC, e.cust_interest_code
        ASC


Optimized Statement:
-------------------
SELECT Q5.CUST_LAST_NAME AS "CUST_LAST_NAME", Q5.CUST_CC_NUMBER AS
        "CUST_CC_NUMBER", Q5.CUST_INTEREST_CODE AS "CUST_INTEREST_CODE"
FROM
   (SELECT Q4.CUST_LAST_NAME, Q4.CUST_CC_NUMBER, Q4.CUST_INTEREST_CODE
   FROM
      (SELECT Q2.CUST_LAST_NAME, Q3.CUST_CC_NUMBER, Q1.CUST_INTEREST_CODE
      FROM GOSALESCT.CUST_INTEREST AS Q1, GOSALESCT.CUST_CUSTOMER AS Q2,
              GOSALESCT.CUST_CRDT_CARD AS Q3
      WHERE (Q3.CUST_CODE = Q1.CUST_CODE) AND (Q1.CUST_CODE = Q2.CUST_CODE))
           AS Q4
   GROUP BY Q4.CUST_INTEREST_CODE, Q4.CUST_CC_NUMBER, Q4.CUST_LAST_NAME) AS
        Q5
ORDER BY Q5.CUST_LAST_NAME, Q5.CUST_CC_NUMBER DESC, Q5.CUST_INTEREST_CODE

Explain level:    Explain from section

Access Plan:
-----------
   Total Cost:       1255.29
   Query Degree:      1

                            Rows
                           RETURN
                           (   1)
                            Cost
                             I/O
                             |
                           31255
                           GRPBY
                           (   2)
                           1255.29
                             NA
                             |
                           31255
                           TBSCAN
                           (   3)
                           1249.02
                             NA
                             |
                           31255
                           SORT
                           (   4)
                           1242.74
                             NA
                             |
                           31255
                           ^HSJOIN
                           (   5)
                           1134.96
                             NA
                        /---------+---------\
                     31255                   31255
                    HSJOIN                  TBSCAN
                    (   6)                  (   9)
                    406.871                 716.136
                      NA                NA
                  /------+-------\                |
               31255              31255         31255
              TBSCAN             IXSCAN   TABLE: GOSALESCT
              (   7)             (   8)     CUST_CUSTOMER
              235.505            159.488         Q2
                NA                NA
                |                   |
               31255               -1
 TABLE: GOSALESCT    INDEX: SYSIBM
  CUST_CRDT_CARD   SQL101108113609000
```

```
        Q3                Q1

⋮
 Objects Used in Access Plan:
---------------------------

   Schema: SYSIBM
   Name:    SQL101108113609000
   Type:    Index
         Last statistics update:    2010-11-08-13.29.58.531000
         Number of rows:       -1
         Number of buffer pool pages:    -1
         Distinct row values:       Yes
         Tablespace name:       GOSALES_TS
         Tablespace overhead:       7.500000
         Tablespace transfer rate:    0.060000
         Prefetch page count:       32
         Container extent page count:    32
         Index clustering statistic:    1.000000
         Index leaf pages:       37
         Index tree levels:       2
         Index full key cardinality:    31255
         Base Table Schema:       GOSALESCT
         Base Table Name:       CUST_INTEREST
         Columns in index:
            CUST_CODE(A)
            CUST_INTEREST_CODE(A)

   Schema: GOSALESCT
   Name:    CUST_CRDT_CARD
   Type:    Table
         Last statistics update:    2010-11-08-11.59.58.531000
         Number of rows:       31255
         Number of buffer pool pages:    192
         Distinct row values:       No
         Tablespace name:       GOSALES_TS
         Tablespace overhead:       7.500000
         Tablespace transfer rate:    0.060000
         Prefetch page count:       32
         Container extent page count:    32
         Table overflow record count:    0
         Table Active Blocks:       -1
         Average Row Compression Ratio:    0
         Percentage Rows Compressed:    0
         Average Compressed Row Size:    0

   Schema: GOSALESCT
   Name:    CUST_CUSTOMER
   Type:    Table
         Last statistics update:    2010-11-08-11.59.59.437000
         Number of rows:       31255
         Number of buffer pool pages:    672
         Distinct row values:       No
         Tablespace name:       GOSALES_TS
         Tablespace overhead:       7.500000
         Tablespace transfer rate:    0.060000
         Prefetch page count:       32
         Container extent page count:    32
         Table overflow record count:    0
         Table Active Blocks:       -1
         Average Row Compression Ratio:    0
         Percentage Rows Compressed:    0
         Average Compressed Row Size:    0

Base Table For Index Not Already Shown:
---------------------------------------

   Schema: GOSALESCT
   Name:    CUST_INTEREST
         Time of creation:       2010-11-08-11.30.28.203002
         Last statistics update:    2010-11-08-13.29.58.531000
         Number of rows:       31255
         Number of pages:       128
         Number of pages with rows:    124
         Table overflow record count:    0
         Indexspace name:       GOSALES_TS
         Tablespace name:       GOSALES_TS
         Tablespace overhead:       7.500000
```

```
Tablespace transfer rate:    0.060000
Prefetch page count:       -1
Container extent page count:    32

Long tablespace name:       GOSALES_TS
```

(Aus dieser Ausgabe wurden aus Gründen der Darstellung mehrere Zeilen entfernt.)

Nächste Schritte

Analysieren Sie die EXPLAIN-Ausgabe, um die Möglichkeiten zum Optimieren der Abfrage zu ermitteln.

Unterschiede zwischen der EXPLAIN-Ausgabe für Abschnitte und der Ausgabe der Anweisung EXPLAIN:

Die Ergebnisse, die nach der Ausführung einer EXPLAIN-Operation für Abschnitte abgerufen werden, sind denen ähnlich, die nach der Ausführung der Anweisung EXPLAIN erfasst werden. Es bestehen kleine Unterschiede, die für die jeweils betroffene EXPLAIN-Tabelle und im Hinblick auf die Auswirkungen auf die Ausgabe des Dienstprogramms **db2exfmt** (sofern zutreffend) beschrieben werden.

Die Ausgabeparameter EXPLAIN_REQUESTER, EXPLAIN_TIME, SOURCE_NAME, SOURCE_SCHEMA und SOURCE_VERSION der gespeicherten Prozedur stellen den Schlüssel dar, mit dem nach den Informationen für den Abschnitt in den EXPLAIN-Tabellen gesucht wird. Verwenden Sie diese Parameter mit beliebigen der vorhandenen EXPLAIN-Tools (z. B. **db2exfmt**), um die aus dem Abschnitt abgerufenen EXPLAIN-Informationen zu formatieren.

Tabelle EXPLAIN_INSTANCE

Die folgenden Spalten werden für die Zeile, die durch eine EXPLAIN-Operation für einen Abschnitt generiert wird, mit anderen Werten gefüllt:

- EXPLAIN_OPTION wird auf den Wert `S` gesetzt.
- SNAPSHOT_TAKEN wird immer auf den Wert `N` gesetzt.
- REMARKS ist immer NULL.

Tabelle EXPLAIN_STATEMENT

Wenn eine EXPLAIN-Operation für einen Abschnitt eine EXPLAIN-Ausgabe generiert hat, wird die Spalte EXPLAIN_LEVEL auf den Wert `S` gesetzt. Es ist wichtig zu beachten, dass die Spalte EXPLAIN_LEVEL Teil des Primärschlüssels der Tabelle und Teil des Fremdschlüssels der meisten anderen EXPLAIN-Tabellen ist. Daher ist dieser EXPLAIN_LEVEL-Wert auch in diesen anderen Tabellen enthalten.

In der Tabelle EXPLAIN_STATEMENT sind die übrigen Spaltenwerte, die normalerweise einer Zeile mit EXPLAIN_LEVEL = `P` zugeordnet sind, stattdessen vorhanden, wenn EXPLAIN_LEVEL = `S` ist, jedoch mit Ausnahme der Spalte SNAPSHOT. Die Spalte SNAPSHOT ist immer NULL, wenn EXPLAIN_LEVEL den Wert `S` hat.

Wenn die Originalanweisung zum Zeitpunkt der Generierung der EXPLAIN-Daten für den Abschnitt nicht verfügbar war (z. B. wenn der Anweisungstext für die Prozedur EXPLAIN_FROM_DATA nicht angegeben wurde, wird STATEMENT_TEXT auf die Zeichenfolge `UNKNOWN` gesetzt, wenn EXPLAIN_LEVEL auf den Wert `O` gesetzt wird.

In der Ausgabe von **db2exfmt** für einen mit EXPLAIN bearbeiteten Abschnitt wird die folgende zusätzliche Zeile nach der optimierten Anweisung angezeigt:

```
Explain level:    Explain from section
```

Tabelle EXPLAIN_OPERATOR

Von allen Spalten, in denen Aufwandswerte gespeichert werden, werden nur die Spalten TOTAL_COST und FIRST_ROW_COST nach einer EXPLAIN-Operation für einen Abschnitt mit einem Wert gefüllt. Alle anderen Spalten zum Speichern von Aufwandswerten haben den Wert -1.

In der Ausgabe von **db2exfmt** für einen mit EXPLAIN bearbeiteten Abschnitt ergeben sich die folgenden Unterschiede:

- Im Zugriffsplandiagramm wird der E/A-Aufwand (I/O Cost) als `NA` angegeben.
- In den Details für die einzelnen Operatoren werden nur die Aufwände `Cumulative Total Cost` und `Cumulative First Row Cost` angezeigt.

Tabelle EXPLAIN_PREDICATE

Keine Unterschiede.

Tabelle EXPLAIN_ARGUMENT

Eine kleine Anzahl von Argumenttypen wird nicht in die Tabelle EXPLAIN_ARGUMENT geschrieben, wenn eine EXPLAIN-Operation für einen Abschnitt ausgeführt wird.

Tabelle EXPLAIN_STREAM

Die folgenden Spalten enthalten nach einer EXPLAIN-Operation für einen Abschnitt keine Werte:

- SINGLE_NODE
- PARTITION_COLUMNS
- SEQUENCE_SIZES

Die folgende Spalte hat nach einer EXPLAIN-Operation für einen Abschnitt immer den Wert -1:

- PREDICATE_ID

Die folgenden Spalten haben nach einer EXPLAIN-Operation für einen Abschnitt nur Werte für Datenströme, die aus einem Basistabellenobjekt stammen, oder aber standardmäßig keinen Wert bzw. den Wert -1:

- COLUMN_NAMES
- COLUMN_COUNT

In der Ausgabe von **db2exfmt** für einen mit EXPLAIN bearbeiteten Abschnitt werden die Informationen aus diesen aufgeführten Spalten in den Abschnitten `Input Streams` und `Output Streams` für die einzelnen Operatoren nicht angezeigt, wenn diese keinen Wert oder den Wert -1 haben.

Tabelle EXPLAIN_OBJECT

Nach der Ausführung einer EXPLAIN-Operation für einen Abschnitt wird die Spalte STATS_SRC immer auf eine leere Zeichenfolge und die Spalte CREATE-_TIME auf den Wert NULL gesetzt.

Die folgenden Spalten haben nach einer EXPLAIN-Operation für einen Abschnitt immer den Wert -1:

- COLUMN_COUNT
- WIDTH
- FIRSTKEYCARD
- FIRST2KEYCARD
- FIRST3KEYCARD
- FIRST4KEYCARD
- SEQUENTIAL_PAGES
- DENSITY
- AVERAGE_SEQUENCE_GAP
- AVERAGE_SEQUENCE_FETCH_GAP
- AVERAGE_SEQUENCE_PAGES
- AVERAGE_SEQUENCE_FETCH_PAGES
- AVERAGE_RANDOM_PAGES
- AVERAGE_RANDOM_FETCH_PAGES
- NUMRIDS
- NUMRIDS_DELETED
- NUM_EMPTY_LEAFS
- ACTIVE_BLOCKS
- NUM_DATA_PART

In der Ausgabe von **db2exfmt** für einen mit EXPLAIN bearbeiteten Abschnitt werden die Informationen aus diesen aufgeführten Spalten unter den Statistikinformationen auf Tabellen- und Indexebene am unteren Ende der Ausgabe nicht angezeigt.

Die EXPLAIN-Daten für einen Abschnitt enthalten in der Ausgabe keine Objekte, auf die durch den Compiler verwiesen wurde (d. h. Zeilen, deren OBJECT_TYPE-Wert mit einem Zeichen '+' beginnt). Solche Objekte werden in der Ausgabe von **db2exfmt** nicht angezeigt.

Erfassen und Zugreifen auf Ist-Daten für Abschnitte:

Ist-Daten für Abschnitte sind Laufzeitstatistiken, die während der Ausführung eines Abschnitts für einen Zugriffsplan erfasst werden. Zur Erfassung eines Abschnitts mit Ist-Daten verwenden Sie den Aktivitätsereignismonitor. Für den Zugriff auf die Ist-Daten eines Abschnitts führen Sie die EXPLAIN-Funktion für den Abschnitt über die gespeicherte Prozedur EXPLAIN_FROM_ACTIVITY aus.

Um Ist-Daten für Abschnitte prüfen zu können, müssen Sie die EXPLAIN-Funktion für einen Abschnitt ausführen, für den Ist-Daten erfasst wurden (d. h., der Abschnitt und die Ist-Daten des Abschnitts sind Eingaben für die EXPLAIN-Funktion). Informationen zur Aktivierung und Erfassung von Ist-Daten für Abschnitte sowie zum Zugriff auf diese Daten werden in diesem Thema bereitgestellt.

Aktivieren von Ist-Daten für Abschnitte

Ist-Daten für Abschnitte werden während der Laufzeit nur aktualisiert, wenn sie aktiviert wurden. Um Ist-Daten für Abschnitte für die gesamte Datenbank zu aktivieren, verwenden Sie den Datenbankkonfigurationsparameter **section_actuals**. Um dies für eine bestimmte Anwendung durchzuführen, verwenden Sie die Prozedur WLM_SET_CONN_ENV.

Zur Aktivierung von Ist-Daten für Abschnitte setzen Sie den Parameter auf den Wert BASE (der Standardwert ist NONE). Beispiel:

```
db2 update database configuration using section_actuals base
```

Um Ist-Daten für Abschnitte für eine bestimmte Anwendung zu aktivieren, verwenden Sie die Prozedur WLM_SET_CONN_ENV und geben Sie für das Element **section_actuals** den Wert BASE an. Beispiel:

```
CALL WLM_SET_CONN_ENV(NULL, '<collectactdata>WITH DETAILS, SECTION</
collectactdata> <collectsectionactuals>BASE</collectsectionactuals>')
```

Anmerkung:

1. Die Einstellung des Datenbankkonfigurationsparameters **section_actuals**, die beim Start der UOW wirksam war, wird für alle Anweisungen in dieser UOW angewendet. Wenn der Datenbankkonfigurationsparameter **section_actuals** dynamisch geändert wird, wird der neue Wert erst bei der nächsten UOW von einer Anwendung erkannt.
2. Die Einstellung für **section_actuals**, die für eine Anwendung durch die Prozedur WLM_SET_CONN_ENV angegeben wird, wird sofort wirksam. Ist-Daten für Abschnitte werden für die nächste Anweisung erfasst, die von der Anwendung ausgegeben wird.
3. Ist-Daten für Abschnitte können nicht aktiviert werden, wenn die automatische Statistikprofilgenerierung aktiviert ist (SQLCODE -5153).

Erfassen von Ist-Daten für Abschnitte

Der Mechanismus zur Erfassung eines Abschnitts mit Ist-Daten für den Abschnitt, ist der Aktivitätsereignismonitor. Ein Aktivitätsereignismonitor zeichnet Details einer Aktivität auf, wenn die Ausführung der Aktivität abgeschlossen ist, sofern die Erfassung von Aktivitätsinformationen aktiviert ist. Die Erfassung von Aktivitätsinformationen wird mit der Klausel COLLECT ACTIVITY DATA in einer Auslastung (Workload), einer Serviceklasse, einem Schwellenwert oder einer Arbeitsaktion aktiviert. Zur Angabe der Erfassung eines Abschnitts und der zugehörigen Ist-Daten (falls letztere aktiviert sind) wird die Option SECTION der Klausel COLLECT ACTIVITY DATA verwendet. Die folgende Anweisung gibt zum Beispiel an, dass für eine beliebige SQL-Anweisung, die von einer Verbindung abgesetzt wird, die der Auslastung WL1 zugeordnet ist, Informationen (einschließlich Abschnitten und Ist-Daten) durch einen beliebigen aktiven Aktivitätsereignismonitor erfasst werden, wenn die Ausführung der Anweisung abgeschlossen wird:

```
ALTER WORKLOAD WL1 COLLECT ACTIVITY DATA WITH DETAILS,SECTION
```

In einer Umgebung mit partitionierten Datenbanken werden Ist-Daten für Abschnitte durch einen Aktivitätsereignismonitor in allen Partitionen erfasst, in denen die Aktivität ausgeführt wurde, sofern für die ausgeführte Anweisung die Klausel COLLECT ACTIVITY DATA verwendet wurde und in der Klausel COLLECT ACTIVITY DATA das Schlüsselwort SECTION und die Klausel ON ALL DATABASE PARTITIONS angegeben wurden. Wenn die Klausel ON ALL DATABASE PARTITIONS nicht angegeben wird, werden Ist-Daten nur in der Koordinatorpartition er-

fasst. Neben der Klausel COLLECT ACTIVITY DATA für eine Auslastung (Workload), eine Serviceklasse, einen Schwellenwert oder eine Arbeitsaktion kann die Aktivitätenerfassung zusätzlich (für eine einzelne Anwendung) dadurch aktiviert werden, dass die Prozedur WLM_SET_CONN_ENV mit einem zweiten Argument verwendet wird, das den Tag 'collectactdata' mit dem Wert 'WITH DETAILS, SECTION' enthält.

Einschränkungen

Die folgenden Einschränkungen in Bezug auf die Erfassung von Ist-Daten für Abschnitte sind zu beachten:

- Ist-Daten für Abschnitte werden nicht erfasst, wenn die gespeicherte Prozedur WLM_CAPTURE_ACTIVITY_IN_PROGRESS dazu verwendet wird, Informationen zu momentan ausgeführten Aktivitäten an einen Aktivitätsereignismonitor zu senden. Jeder Aktivitätsereignismonitorsatz, der durch die gespeicherte Prozedur WLM_CAPTURE_ACTIVITY_IN-_PROGRESS generiert wird, hat in der Spalte 'partial_record' (Partieller Datensatz) den Wert 1.
- Wenn gegen einen reaktiven Schwellenwert verstoßen wurde, werden Ist-Daten für Abschnitte nur in der Koordinatorpartition erfasst.
- EXPLAIN-Tabellen müssen auf DB2 Version 9.7 Fixpack 1 oder eine spätere Version migriert werden, bevor auf Ist-Daten für Abschnitte über eine EXPLAIN-Operation für Abschnitte zugegriffen werden kann. Wenn die EXPLAIN-Tabellen nicht migriert wurden, funktioniert die EXPLAIN-Operation für Abschnitte zwar, jedoch werden die Ist-Dateninformationen für Abschnitte nicht in die EXPLAIN-Tabellen eingefügt. In diesem Fall wird ein Eintrag in die Tabelle EXPLAIN_DIAGNOSTIC geschrieben.
- Vorhandene Tabellen für den Aktivitätsereignismonitor in DB2 Version 9.7 (insbesondere die Aktivitätstabelle) müssen erneut erstellt werden, bevor Ist-Daten für Abschnitte durch den Aktivitätsereignismonitor erfasst werden können. Wenn die logische Gruppe 'activity' (Aktivität) die Spalte SECTION_ACTUALS nicht enthält, wird die EXPLAIN-Operation für Abschnitte möglicherweise mit einem vom Aktivitätsereignismonitor erfassten Abschnitt zwar ausgeführt, jedoch enthält EXPLAIN keine Ist-Daten zu Abschnitten.

Zugreifen auf Ist-Daten für Abschnitte

Auf Ist-Daten für Abschnitte kann mithilfe der Prozedur EXPLAIN_FROM_ACTIVITY zugegriffen werden. Wenn Sie eine EXPLAIN-Operation für Abschnitte für eine Aktivität ausführen, für die Ist-Daten für Abschnitte erfasst wurden, wird die EXPLAIN-Tabelle EXPLAIN_ACTUALS mit den Ist-Dateninformationen gefüllt.

Anmerkung: Ist-Daten für Abschnitte sind nur verfügbar, wenn eine EXPLAIN-Operation für Abschnitte mit der Prozedur EXPLAIN_FROM_ACTIVITY ausgeführt wird.

Die Tabelle EXPLAIN_ACTUALS ist die untergeordnete Tabelle der vorhandenen EXPLAIN-Tabelle EXPLAIN_OPERATOR. Wenn die Prozedur EXPLAIN_FROM_ACTIVITY aufgerufen wird, wird die Tabelle EXPLAIN_ACTUALS mit den Ist-Daten gefüllt, wenn die Ist-Daten für Abschnitte verfügbar sind. Wenn die Ist-Daten in mehreren Datenbankpartitionen erfasst werden, wird eine Zeile pro Datenbankpartition für jeden Operator in die Tabelle EXPLAIN_ACTUALS eingefügt.

Abrufen von EXPLAIN-Informationen für Abschnitte mit Ist-Daten zur Untersuchung einer schwachen Abfrageleistung:

Zur Behebung einer Verlangsamung der SQL-Abfrageleistung können Sie beginnen, indem Sie EXPLAIN-Informationen zu einem Abschnitt abrufen, die Ist-Dateninformationen des Abschnitts enthalten. Die Ist-Datenwerte des Abschnitts können dann mit den geschätzten Zugriffsplanwerten verglichen werden, die vom Optimierungsprogramm zur Beurteilung der Gültigkeit des Zugriffsplans generiert werden. Diese Task führt Sie durch den Prozess des Abrufens von Ist-Werten eines Abschnitts, um eine schwache Abfrageleistung zu untersuchen.

Vorbereitende Schritte

Sie haben die Diagnosephase Ihrer Untersuchung abgeschlossen und festgestellt, dass es sich tatsächlich um eine Verlangsamung der SQL-Abfrageleistung handelt, und Sie haben einen Verdacht, welche Anweisung mit der Leistungsbeeinträchtigung zu tun haben könnte.

Informationen zu diesem Vorgang

Diese Task führt Sie durch den Prozess des Abrufens von Ist-Werten eines Abschnitts, um eine schwache Abfrageleistung zu untersuchen. Ein Vergleich der Informationen, die in den Ist-Werten für einen Abschnitt enthalten sind, mit den vom Optimierungsprogramm generierten Schätzwerten kann helfen, die Verlangsamung der Abfrageleistung zu beheben.

Einschränkungen

Informationen zu Einschränkungen finden Sie im Abschnitt „Erfassen und Zugreifen auf Ist-Daten für Abschnitte".

Vorgehensweise

Führen Sie die folgenden Schritte aus, um eine schwache Abfrageleistung für eine Abfrage zu untersuchen, die von der Anwendung `myApp.exe` ausgeführt wird:

1. Aktivieren Sie die Ist-Daten für Abschnitte:
   ```
   DB2 UPDATE DATABASE CONFIGURATION USING SECTION_ACTUALS BASE
   ```
2. Erstellen Sie die EXPLAIN-Tabellen im Schema MYSCHEMA mithilfe der Prozedur SYSINSTALLOBJECTS:
   ```
   CALL SYSINSTALLOBJECTS( 'EXPLAIN', 'C', NULL, 'MYSCHEMA' )
   ```

   Anmerkung: Dieser Schritt kann übersprungen werden, wenn die EXPLAIN-Tabellen bereits erstellt wurden.
3. Erstellen Sie eine Auslastung MYCOLLECTWL, um Aktivitäten zu erfassen, die von der Anwendung `myApp.exe` übergeben werden, und aktivieren Sie die Erfassung von Abschnittsdaten für diese Aktivitäten, indem Sie die beiden folgenden Befehle ausgeben:
   ```
   CREATE WORKLOAD MYCOLLECTWL APPLNAME( 'MYAPP.EXE')
   COLLECT ACTIVITY DATA WITH DETAILS,SECTION
   ```
   Geben Sie danach folgenden Befehl aus:
   ```
   GRANT USAGE ON WORKLOAD MYCOLLECTWL TO PUBLIC
   ```

Anmerkung: Die Verwendung einer separaten Auslastung (Workload) begrenzt den Umfang der Informationen, die durch den Aktivitätsereignismonitor erfasst werden.

4. Erstellen Sie einen Aktivitätsereignismonitor mit dem Namen ACTEVMON, indem Sie die folgende Anweisung ausführen:

   ```
   CREATE EVENT MONITOR ACTEVMON FOR ACTIVITIES WRITE TO TABLE
   ```

5. Aktivieren Sie den Aktivitätsereignismonitor ACTEVMON, indem Sie die folgende Anweisung ausführen:

   ```
   SET EVENT MONITOR ACTEVMON STATE 1
   ```

6. Führen Sie die Anwendung myApp.exe aus. Alle Anweisungen, die von der Anwendung abgesetzt werden, werden vom Aktivitätsereignismonitor erfasst.
7. Fragen Sie die Tabellen des Aktivitätsereignismonitors ab, um die Kennungsinformationen (IDs) für die fragliche Anweisung zu ermitteln, indem Sie die folgende Anweisung ausführen:

   ```
   SELECT APPL_ID,
          UOW_ID,
          ACTIVITY_ID,
          STMT_TEXT
   FROM ACTIVITYSTMT_ACTEVMON
   ```

   Das folgende Beispiel zeigt eine Ausgabe, die als Ergebnis der ausgeführten Anweisung SELECT generiert wurde:

   ```
   APPL_ID                    UOW_ID    ACTIVITY_ID     STMT_TEXT
   -------------------------  --------  --------------  ---------------
   *N2.DB2INST1.0B5A12222841         1               1  SELECT * FROM ...
   ```

8. Verwenden Sie die Informationen zur Aktivitäts-ID wie in der folgenden Anweisung CALL gezeigt als Eingabe für die Prozedur EXPLAIN_FROM_ACTIVITY, um EXPLAIN-Informationen zu Abschnitten mit Ist-Daten abzurufen:

   ```
   CALL EXPLAIN_FROM_ACTIVITY( '*N2.DB2INST1.0B5A12222841', 1, 1, 'ACTEVMON',
    'MYSCHEMA', ?, ?, ?, ?, ? )
   ```

   Das folgende Beispiel zeigt eine Ausgabe, die durch den Aufruf der Prozedur EXPLAIN_FROM_ACTIVITY generiert wurde:

   ```
   Wert der Ausgabeparameter
   -------------------------
   Parametername: EXPLAIN_SCHEMA
   Parameterwert: MYSCHEMA

   Parametername: EXPLAIN_REQUESTER
   Parameterwert: SWALKTY

   Parametername: EXPLAIN_TIME
   Parameterwert: 2009-08-24-12.33.57.525703

   Parametername: SOURCE_NAME
   Parameterwert: SQLC2H20

   Parametername: SOURCE_SCHEMA
   Parameterwert: NULLID

   Parametername: SOURCE_VERSION
   Parameterwert:

   Rückgabestatus = 0
   ```

9. Formatieren Sie die EXPLAIN-Daten mit dem Befehl **db2exfmt**, indem Sie als Eingabe den EXPLAIN-Instanzschlüssel angeben, der als Ausgabe von der Prozedur EXPLAIN_FROM_ACTIVITY zurückgegeben wurde. Geben Sie zum Beispiel Folgendes an:

   ```
   db2exfmt -d test -w 2009-08-24-12.33.57.525703 -n SQLC2H20 -s NULLID -# 0 -t
   ```

Die EXPLAIN-Instanzausgabe sieht beispielsweise wie folgt aus:

```
  ******************** EXPLAIN INSTANCE ********************

DB2_VERSION:       09.07.1
SOURCE_NAME:       SQLC2H20
SOURCE_SCHEMA:     NULLID
SOURCE_VERSION:
EXPLAIN_TIME:      2009-08-24-12.33.57.525703
EXPLAIN_REQUESTER: SWALKTY

Database Context:
----------------
        Parallelism:          None
        CPU Speed:            4.000000e-05
        Comm Speed:           0
        Buffer Pool size:     198224
        Sort Heap size:       1278
        Database Heap size:   2512
        Lock List size:       6200
        Maximum Lock List:    60
        Average Applications: 1
        Locks Available:      119040

Package Context:
---------------
        SQL Type:           Dynamic
        Optimization Level: 5
        Blocking:           Block All Cursors
        Isolation Level:    Cursor Stability



---------------- STATEMENT 1  SECTION 201 ----------------
        QUERYNO:       0
        QUERYTAG:      CLP
        Statement Type:   Select
        Updatable:        No
        Deletable:        No
        Query Degree:     1

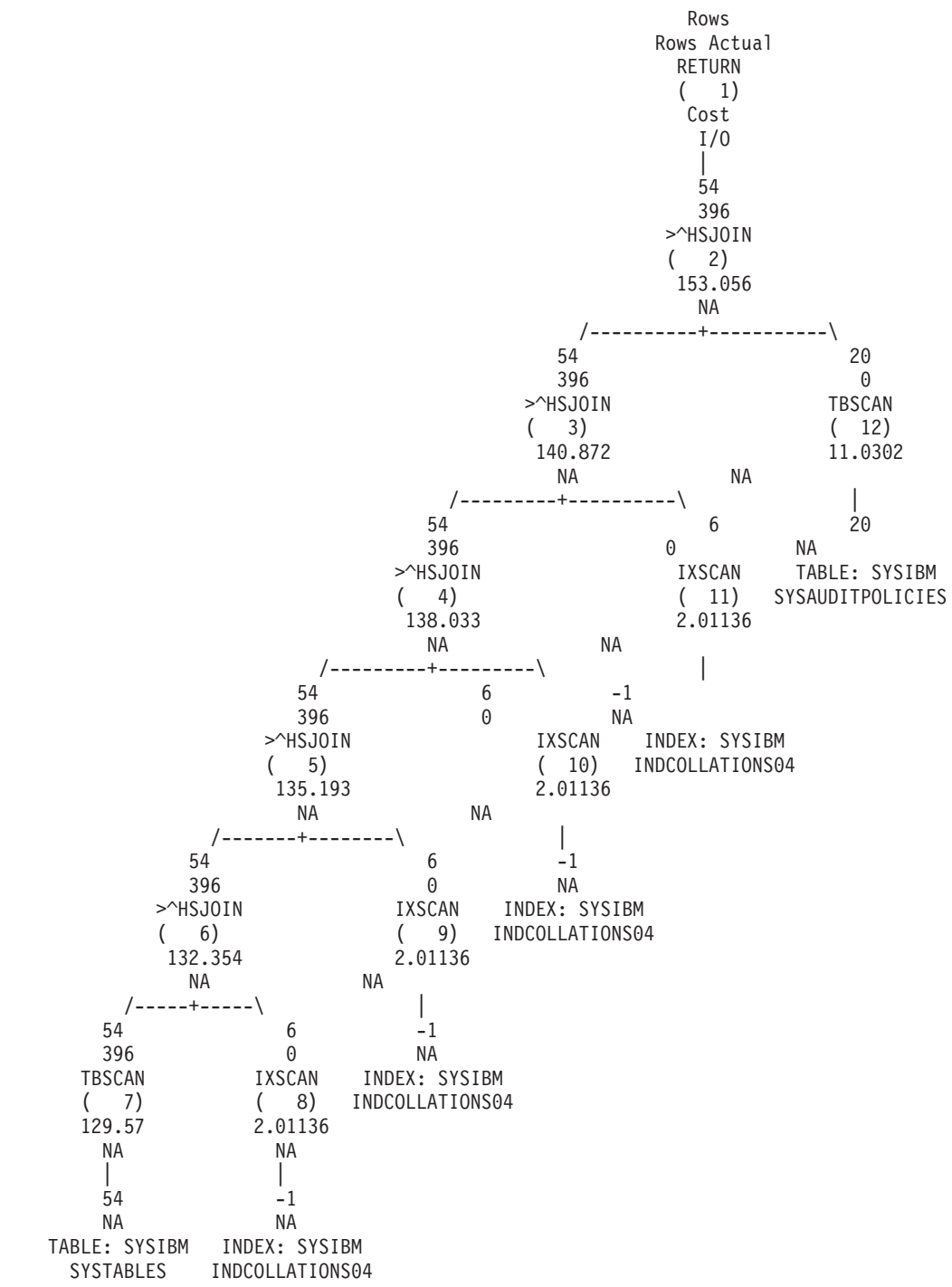
Original Statement:
------------------
select *
from syscat.tables

Optimized Statement:
-------------------
SELECT Q10.$C67 AS "TABSCHEMA", Q10.$C66 AS "TABNAME", Q10.$C65 AS "OWNER",
        Q10.$C64 AS "OWNERTYPE", Q10.$C63 AS "TYPE", Q10.$C62 AS "STATUS",
        Q10.$C61 AS "BASE_TABSCHEMA", Q10.$C60 AS "BASE_TABNAME", Q10.$C59 AS
        "ROWTYPESCHEMA", Q10.$C58 AS "ROWTYPENAME", Q10.$C57 AS "CREATE_TIME",
        Q10.$C56 AS "ALTER_TIME", Q10.$C55 AS "INVALIDATE_TIME", Q10.$C54 AS
        "STATS_TIME", Q10.$C53 AS "COLCOUNT", Q10.$C52 AS "TABLEID", Q10.$C51
        AS "TBSPACEID", Q10.$C50 AS "CARD", Q10.$C49 AS "NPAGES", Q10.$C48 AS
        "FPAGES", Q10.$C47 AS "OVERFLOW", Q10.$C46 AS "TBSPACE", Q10.$C45 AS
        "INDEX_TBSPACE", Q10.$C44 AS "LONG_TBSPACE", Q10.$C43 AS "PARENTS",
        Q10.$C42 AS "CHILDREN", Q10.$C41 AS "SELFREFS", Q10.$C40 AS
        "KEYCOLUMNS", Q10.$C39 AS "KEYINDEXID", Q10.$C38 AS "KEYUNIQUE",
        Q10.$C37 AS "CHECKCOUNT", Q10.$C36 AS "DATACAPTURE", Q10.$C35 AS
        "CONST_CHECKED", Q10.$C34 AS "PMAP_ID", Q10.$C33 AS "PARTITION_MODE",
        '0' AS "LOG_ATTRIBUTE", Q10.$C32 AS "PCTFREE", Q10.$C31 AS
        "APPEND_MODE", Q10.$C30 AS "REFRESH", Q10.$C29 AS "REFRESH_TIME",
...

Explain level:    Explain from section

Access Plan:
```

```
----------
        Total Cost:             154.035
        Query Degree:           1
```



...

10. Untersuchen Sie die Informationen zu Ist-Daten (Actual-Werten) in der EXPLAIN-Ausgabe. Vergleichen Sie die Werte der Ist-Daten des Abschnitts mit den geschätzten Werten des Zugriffsplans, der vom Optimierungsprogramm generiert wurde. Wenn eine Diskrepanz zwischen den Ist-Werten des Abschnitts und den geschätzten Werten für den Zugriffsplan zu erkennen ist, ermitteln Sie die Ursache für diese Diskrepanz und führen geeignete Behebungs-

maßnahmen aus. Sie könnten beispielsweise ermitteln, dass die Tabellenstatistiken für eine der abgefragten Tabellen nicht aktuell sind. Dies hat zur Folge, dass das Optimierungsprogramm einen falschen Zugriffsplan auswählt, der möglicherweise für die Verlangsamung der Abfrageleistung verantwortlich ist. Die geeignete Maßnahme wäre in diesem Fall, den Befehl **RUNSTATS** für die Tabelle auszuführen, um die Tabellenstatistiken zu aktualisieren.

11. Führen Sie die Anwendung erneut aus, um festzustellen, ob die Verlangsamung der Abfrageleistung weiter besteht.

Analysieren der Ist-Dateninformationen für Abschnitte in der EXPLAIN-Ausgabe:

Ist-Daten für Abschnitte werden, sofern sie verfügbar sind, in verschiedenen Teilen der EXPLAIN-Ausgabe angezeigt. Nachfolgend wird beschrieben, an welchen Stellen der EXPLAIN-Ausgabe Ist-Daten für Abschnitte und Operatordetails zu finden sind.

Ist-Daten für einen Abschnitt in der Diagrammausgabe von 'db2exfmt'

Wenn die EXPLAIN-Ist-Daten verfügbar sind, werden sie im Diagramm unter den geschätzten Zeilen angezeigt. EXPLAIN unterstützt Ist-Daten nur für Operatoren, nicht für Objekte. Für Objekte im Diagramm wird NA (nicht zutreffend) angezeigt. Das folgende Beispiel zeigt eine **db2exfmt**-Diagrammausgabe:

```
        Rows
     Rows Actual
       RETURN
       (   1)
        Cost
         I/O
         |
       3.21948  << Die vom Optimierungsprogramm verwendete Zeilenschätzung
         301    << Die tatsächlich zur Laufzeit erfassten Zeilen
        DTQ
       (   2)
       75.3961
         NA
         |
       3.21948
         130
       HSJOIN
       (   3)
       72.5927
         NA
       /--+---\
     674        260
     220        130
   TBSCAN     TBSCAN
   (   4)     (   5)
   40.7052    26.447
     NA         NA
     |          |
     337        130
     NA         NA  << EXPLAIN unterstützt keine Ist-Daten für Objekte.
  TABLE: FF  TABLE: FF
     T1         T2
```

In einer Umgebung mit partitionierten Datenbanken ist die im Diagramm angezeigte Kardinalität die durchschnittliche Kardinalität der Datenbankpartitionen, in denen die Ist-Daten erfasst wurden. Der Durchschnitt wird angezeigt, weil dies der vom Optimierungsprogramm geschätzte Wert ist. Der bereitgestellte tatsächliche

Durchschnitt liefert einen sinnvollen Wert, der mit dem geschätzten Durchschnitt verglichen werden kann. In einer Umgebung mit partitionierten Datenbanken wird in der Operatordetailausgabe eine Aufschlüsselung von Ist-Daten für Abschnitte pro Datenbankpartition bereitgestellt. Ein Benutzer kann diese Details untersuchen, um weitere Informationen zu ermitteln, wie zum Beispiel Gesamtsummen (über alle Partitionen hinweg), Minimalwerte, Maximalwerte usw.

Operatordetails in der Ausgabe von 'db2exfmt'

Die tatsächliche Kardinalität für einen Operator wird im Datenstromabschnitt nach der Zeile `Estimated number of rows` (geschätzte Anzahl Zeilen). In der EXPLAIN-Ausgabe entspricht dies der Zeile `Actual number of rows` (tatsächliche Anzahl Zeilen) angezeigt. Die angezeigte tatsächliche Kardinalität ist die durchschnittliche Kardinalität für eine Umgebung mit partitionierten Datenbanken, wenn der Operator in mehreren Datenbankteilkomponenten ausgeführt wird. Die nach Datenbankpartitionen aufgeschlüsselten Werte werden unter einem separaten Abschnitt `Explain Actuals` (Ist-Daten für Abschnitte) angezeigt. Der Abschnitt `Explain Actuals` wird nur in der Umgebung mit partitionierten Datenbanken, nicht im seriellen Modus angezeigt. Wenn die Ist-Daten für eine bestimmte Datenbankpartition nicht verfügbar sind, wird neben der Partitionsnummer der Wert `NA` in der Liste mit den nach Datenbankpartitionen aufgeschlüsselten Werten angezeigt. Der Wert für `Actual number of rows` ('Tatsächliche Anzahl Zeilen') im Abschnitt `Output Streams` ('Ausgabedatenströme') ist in diesem Fall ebenfalls `NA`. Das folgende Beispiel zeigt die Operatordetails in der Ausgabe von **db2exfmt**:

```
9) UNION : (Union)
   Cumulative Total Cost:     10.6858
   Cumulative First Row Cost:    9.6526

   Arguments:
   ---------
   UNIONALL: (UnionAll Parameterized Base Table)
      DISJOINT

   Input Streams:
   -------------
      5) From Operator #10

         Estimated number of rows:  30
         Actual number of rows:     63
         Partition Map ID:          3

      7) From Operator #11

         Estimated number of rows:  16
         Actual number of rows:     99
         Partition Map ID:          3

   Output Streams:
   --------------
      8) To Operator #8
         Estimated number of rows:  30
         Actual number of rows:     162
         Partition Map ID:          3

   Explain Actuals:      << Dieser Abschnitt wird nur in einer Umgebung mit partitionierten
                            Datenbanken angezeigt.
   ----------------
      DB Partition number   Cardinality
      -------------------   -----------
                1               193
                2               131
```

Richtlinien zur Verwendung von EXPLAIN-Informationen

Mithilfe von EXPLAIN-Informationen können Sie Ursachen für Änderungen in der Anwendungsleistung analysieren oder durchgeführte Maßnahmen zur Leistungsoptimierung bewerten.

Analysieren von Leistungsänderungen

Zur Untersuchung der Ursachen für Änderungen in der Abfrageleistung benötigen Sie EXPLAIN-Informationen zu den Situationen vor und nach den Änderungen. Diese Informationen können Sie durch die folgenden Schritte erhalten:

1. Erfassen Sie EXPLAIN-Informationen für die Abfrage, bevor Sie Änderungen vornehmen, und speichern Sie die resultierenden EXPLAIN-Tabellen. Alternativ können Sie auch die Ausgabe aus dem EXPLAIN-Tool **db2exfmt** speichern.
2. Speichern oder drucken Sie die aktuellen Katalogstatistiken, wenn Sie zum Anzeigen der entsprechenden Informationen nicht auf Visual Explain zugreifen können. Zur Ihrer Unterstützung bei dieser Aufgabe können Sie auch das Produktivitätstool **db2look** verwenden.
3. Sichern oder drucken Sie die Anweisungen der Datendefinitionssprache (DDL), einschließlich der Anweisungen CREATE TABLE, CREATE VIEW, CREATE INDEX oder 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- und XQuery-Anweisungen können Sie diese Informationen beim Binden erfassen. Zur Analyse einer Leistungsänderung vergleichen Sie diese Informationen, die Sie erfassen, mit den Referenzinformationen, die Sie zuvor erfasst haben.

Ihre Analyse könnte zum Beispiel ergeben, dass bei der Ermittlung eines Zugriffspfads ein Index nicht mehr verwendet wird. Mithilfe 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
- Erfassen von EXPLAIN-Informationen beim erneuten Binden (Rebind) der Abfrage

Nach der Durchführung einer dieser Maßnahmen untersuchen Sie den Zugriffsplan erneut. Wenn der Index verwendet wird, ist die Leistung der Abfrage möglicherweise kein Problem mehr. Wenn der Index weiterhin nicht verwendet wird oder die Leistung problematisch bleibt, führen Sie eine andere Maßnahme dieser Liste 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 Aktualisieren von Konfigurationsparametern, das Hinzufügen von Containern, das Erfassen aktueller Katalogstatistiken u. a. gehören.

Nach einer Änderung in einem dieser Bereiche ermitteln Sie mithilfe der EXPLAIN-Funktion die Auswirkungen, die die Änderung auf den ausgewählten Zugriffsplan hat (falls zutreffend). Wenn Sie zum Beispiel einen Index oder eine MQT (Materialized Query Table, gespeicherte Abfragetabelle) gemäß den Richtlinien für Indizes hinzufügen, können Sie mithilfe der EXPLAIN-Daten feststellen, ob der Index oder die MQT tatsächlich in der erwarteten Weise genutzt wird.

Obwohl Sie mithilfe der EXPLAIN-Ausgabe den ausgewählten Zugriffsplan und seinen relativen Aufwand ermitteln können, besteht die einzige Möglichkeit, die Verbesserung der Leistung für eine Abfrage präzise zu messen, in der Anwendung von Vergleichstestverfahren (Benchmark Tests).

Richtlinien zur Analyse von EXPLAIN-Informationen

Der primäre Zweck der EXPLAIN-Informationen ist die Analyse von Zugriffspfaden für Abfrageanweisungen. Die Analyse der EXPLAIN-Daten kann Ihnen auf verschiedene Weise helfen, Ihre Abfragen und Ihre Umgebung zu optimieren.

Ziehen Sie die folgenden Arten der Analyse in Betracht:

- Verwendung von Indizes

  Die geeigneten Indizes können die Leistung erheblich fördern. Anhand von EXPLAIN-Ausgaben können Sie ermitteln, ob die von Ihnen für eine bestimmte Gruppe von Abfragen erstellten Indizes tatsächlich verwendet werden. Prüfen Sie die folgende Bereiche auf die Verwendung von Indizes:

  – Joinvergleichselemente
  – Lokale Vergleichselemente
  – Klausel GROUP BY
  – Klausel ORDER BY
  – Klausel WHERE XMLEXISTS
  – SELECT-Liste

  Sie können die EXPLAIN-Funktion auch verwenden, um zu ermitteln, ob ein anderer Index oder gar kein Index besser wäre. Führen Sie nach der Erstellung eines neuen Index den Befehl **RUNSTATS** aus, um Statistikdaten für diesen Index zu erfassen, und kompilieren Sie anschließend die Abfrage erneut. Mit der Zeit stellen Sie (anhand von EXPLAIN-Daten) vielleicht fest, dass eine Tabellensuche anstelle einer Indexsuche 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 folgende Maßnahmen in Erwägung ziehen:

  – Reorganisieren der Tabelle, um die Daten diesem Index entsprechend in Clustern anzuordnen
  – Ausführen des Befehls **RUNSTATS** zur Erfassung von Statistikdaten für Index und Tabelle
  – Erneutes Kompilieren der Abfrage

  Stellen Sie anhand von EXPLAIN-Ausgaben fest, ob das Reorganisieren der Tabelle zu einer Verbesserung des Zugriffsplans geführt hat.

- Zugriffstyp

  Analysieren Sie die EXPLAIN-Ausgabe und suchen Sie nach Datenzugriffstypen, die für den Typ von Anwendung, die Sie ausführen, normalerweise nicht optimal sind. Beispiele:

  – OLTP-Abfragen (Onlinetransaktionsverarbeitung)

    OLTP-Anwendungen bieten sich in der Regel für Indexsuchen mit bereichsbegrenzenden Vergleichselementen an, weil sie meist nur wenige Zeilen zurück-

geben, die durch ein Gleichheitsvergleichselement für eine Schlüsselspalte ermittelt werden. Wenn Ihre OLTP-Abfragen eine Tabellensuche verwenden, können Sie die EXPLAIN-Daten analysieren, um herauszufinden, warum keine Indexsuche verwendet wird.

– Reine Suchabfragen

  Die Suchbedingungen für eine „reine Suchabfrage" können sehr vage sein, was bewirkt, dass eine große Menge von Zeilen den Bedingungen entspricht. Wenn sich Benutzer normalerweise nur einige Anzeigeseiten von Ausgabedaten ansehen, können Sie dafür sorgen, dass nicht die gesamte Antwortmenge berechnet werden muss, bevor einige Ergebnisse zurückgegeben werden. In diesem Fall unterscheiden sich die Ziele des Benutzers vom grundlegenden Arbeitsprinzip des Optimierungsprogramms, das versucht, den Ressourcenbedarf für die gesamte Abfrage und nicht nur für die ersten wenigen Datenanzeigen 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 vielen Zeilen, wie sie zum Beispiel bei Abfragen auf Entscheidungshilfedaten auftreten, werden in der Regel durch einen Hash-Join schneller als durch einen Mischjoin ausgeführt. Joins, die nur wenige Zeilen betreffen, wie zum Beispiel Joins in OLTP-Abfragen, sind zumeist als Joins mit Verschachtelungsschleifen schneller. In beiden Fällen kann es jedoch auch Umstände geben, wie beispielsweise die Verwendung lokaler Vergleichselemente oder Indizes, die die Arbeitsweise dieser typischen Joins möglicherweise ändern.

Verwenden von Zugriffsplänen zur Selbstdiagnose von Leistungsproblemen bei Anweisungen REFRESH TABLE und SET INTEGRITY

Durch Aufrufen des Dienstprogramms EXPLAIN für Anweisungen REFRESH TABLE und SET INTEGRITY können Sie Zugriffspläne generieren, die zur Selbstdiagnose von Leistungsproblemen mit diesen Anweisungen verwendet werden können. Diese Möglichkeit kann Ihnen helfen, Ihre MQTs (Materialized Query Tables) besser zu verwalten.

Zum Abrufen des Zugriffsplans für eine Anweisung REFRESH TABLE oder SET INTEGRITY können Sie eine der folgenden Methoden verwenden:

- Verwenden Sie die Option EXPLAIN PLAN FOR REFRESH TABLE oder EXPLAIN PLAN FOR SET INTEGRITY in der Anweisung EXPLAIN.
- Setzen Sie das Sonderregister CURRENT EXPLAIN MODE auf den Wert EXPLAIN, bevor Sie die Anweisung REFRESH TABLE bzw. SET INTEGRITY ausführen. Setzen Sie das Sonderregister CURRENT EXPLAIN MODE anschließend wieder auf NO.

Einschränkungen

- Die Anweisungen REFRESH TABLE und SET INTEGRITY kommen für die Reoptimierung nicht infrage. Daher ist der EXPLAIN-Modus REOPT (oder EXPLAIN SNAPSHOT) auf diese beiden Anweisungen nicht anwendbar.
- Die Klausel WITH REOPT ONCE der Anweisung EXPLAIN, die angibt, dass die angegebene mit EXPLAIN zu bearbeitende Anweisung reoptimiert werden soll, ist auf die Anweisungen REFRESH TABLE und SET INTEGRITY nicht anwendbar.

Szenario

Dieses Szenario veranschaulicht, wie Sie Zugriffspläne aus EXPLAIN- und REFRESH TABLE-Anweisungen generieren und zur Selbstdiagnose der Ursache Ihres Leistungsproblems verwenden können.

1. Erstellen Sie Ihre Tabellen und füllen Sie sie mit Daten. Beispiel:

```
create table t (
  i1 int not null,
  i2 int not null,
  primary key (i1)
);

insert into t values (1,1), (2,1), (3,2), (4,2);

create table mqt as (
  select i2, count(*) as cnt from t group by i2
)
data initially deferred
refresh deferred;
```

2. Setzen Sie die Anweisungen EXPLAIN und REFRESH TABLE wie folgt ab:

```
explain plan for refresh table mqt;
```

   Alternativ zu diesem Schritt kann der EXPLAIN-Modus im Sonderregister SET CURRENT EXPLAIN MODE wie folgt angegeben werden:

```
set current explain mode explain;
refresh table mqt;
set current explain mode no;
```

3. Formatieren Sie den Inhalt der EXPLAIN-Tabellen mit dem Befehl **db2exfmt** und rufen Sie den Zugriffsplan ab. Dieses Tool befindet sich im Unterverzeichnis `misc` des Verzeichnisses `sqllib` für Ihre Instanz.

```
db2exfmt –d datenbankname -o refresh.exp -1
```

4. Analysieren Sie den Zugriffsplan, um die Ursache des Leistungsproblems festzustellen. Wenn im vorherigen Beispiel T eine sehr große Tabelle ist, ist die Tabellensuche sehr aufwendig. In diesem Fall könnte die Leistung der Abfrage durch die Erstellung eines Index verbessert werden.

Tools zum Erfassen und Analysieren von EXPLAIN-Informationen

Der DB2-Datenbankserver verfügt über eine umfassende EXPLAIN-Funktion, 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 bieten Ihnen die erforderlichen flexiblen Möglichkeiten, die EXPLAIN-Informationen zu erfassen, anzuzeigen und zu analysieren.

Detaillierte Informationen des Abfrageoptimierungsprogramms, die eine eingehende Analyse eines Zugriffsplans ermöglichen, werden in EXPLAIN-Tabellen gespeichert, die vom eigentlichen Zugriffsplan getrennt sind. Es stehen mehrere Methoden zum Abrufen von Informationen aus den EXPLAIN-Tabellen zur Verfügung:

- Verwenden Sie das Tool **db2exfmt** zum Anzeigen von EXPLAIN-Informationen in einer formatierten Ausgabe.
- Schreiben Sie eigene Abfragen für die EXPLAIN-Tabellen. Das Schreiben eigener Abfragen bietet Ihnen die Möglichkeit, die Ausgabe auf einfache Weise zu bearbeiten und Vergleiche zwischen verschiedenen Abfragen oder Vergleiche zwischen Ausführungen derselben Abfrage über einen Zeitraum hinweg anzustellen.

Verwenden Sie das Tool **db2expln**, um die Zugriffsplaninformationen anzuzeigen, die für ein oder mehrere Pakete mit statischen SQL- oder XQuery-Anweisungen verfügbar sind. Dieses Dienstprogramm zeigt die tatsächliche Implementierung des gewählten Zugriffsplans, jedoch keine Informationen des Optimierungsprogramms an. Durch das Untersuchen des generierten Zugriffsplans bietet das Tool **db2expln** eine relativ kompakte verbale Übersicht über die Operationen, die bei der Ausführung stattfinden.

Das EXPLAIN-Befehlszeilentools befinden sich im Unterverzeichnis `misc` des Verzeichnisses `sqllib`.

Die folgende Tabelle enthält eine Übersicht über die verschiedenen Tools, die für die DB2-EXPLAIN-Funktion verfügbar sind. Verwenden Sie die Informationen dieser Tabelle, um das Tools auszuwählen, das für Ihre Anforderungen und Ihre Umgebung am besten geeignet ist.

Tabelle 54. Tools der EXPLAIN-Funktion

Gewünschte Merkmale	EXPLAIN-Tabellen	db2expln	db2exfmt
Textausgabe		Ja	Ja
Schnelle Grobanalyse für statisches SQL und XQuery		Ja	
Unterstützung für statisches SQL und XQuery	Ja	Ja	Ja
Unterstützung für dynamisches SQL und XQuery	Ja	Ja	Ja
Unterstützung für CLI-Anwendungen	Ja		Ja
Verfügbar für DRDA-Anwendungsrequester	Ja		
Detaillierte Informationen des Optimierungsprogramms	Ja		Ja
Geeignet zur Analyse mehrerer Anweisungen	Ja	Ja	Ja
Zugriff auf die Informationen aus einer Anwendung heraus	Ja		

Anzeigen der bei der EXPAIN-Bearbeitung gültigen Katalogstatistiken

Die EXPLAIN-Funktion erfasst die Statistikdaten, die gültig sind, wenn eine Anweisung mit EXPLAIN bearbeitet wird. Diese Statistiken können sich von denen unterscheiden, die im Systemkatalog gespeichert sind, insbesondere wenn die Echt-

zeitstatistikerfassung aktiviert ist. Wenn die EXPLAIN-Tabellen mit Daten gefüllt wurden, jedoch keine EXPLAIN-Momentaufnahme erstellt wird, sind nur einige Statistikdaten in der Tabelle EXPLAIN_OBJECT aufgezeichnet.

Zur Erfassung aller Katalogstatistiken, die für die Anweisung, die mit EXPLAIN bearbeitet wird, relevant sind, erstellen Sie eine EXPLAIN-Momentaufnahme zu dem Zeitpunkt, zu dem die EXPLAIN-Tabellen mit Daten gefüllt werden, und formatieren anschließend die Katalogstatistiken in der Momentaufnahme mit der Skalarfunktion SYSPROC.EXPLAIN_FORMAT_STATS.

Wenn das Tool **db2exfmt** zur Formatierung der EXPLAIN-Informationen verwendet wird und eine EXPLAIN-Momentaufnahme erfasst wurde, verwendet das Tool automatisch die Funktion SYSPROC.EXPLAIN_FORMAT_STATS, um die Katalogstatistiken anzuzeigen.

EXPLAIN-Tabellen und Organisation von EXPLAIN-Informationen

Sämtliche EXPLAIN-Informationen sind nach dem Konzept einer EXPLAIN-Instanz organisiert. Eine *EXPLAIN-Instanz* stellt einen Aufruf der EXPLAIN-Funktion für eine oder mehrere SQL- oder XQuery-Anweisungen dar. Die EXPLAIN-Informationen, die in einer EXPLAIN-Instanz erfasst werden, enthalten die Kompilierungsumgebung und den zur Ausführung der SQL- oder XQuery-Anweisung, die kompiliert wird, ausgewählten Zugriffsplan.

Zum Beispiel kann eine EXPLAIN-Instanz aus einer der folgenden Informationsgruppen bestehen:

- Alle infrage kommenden SQL- oder XQuery-Anweisungen in einem Paket bei statischen Abfrageanweisungen. Bei SQL-Anweisungen (einschließlich derer, die XML-Daten abfragen) können Sie EXPLAIN-Informationen für folgende Anweisungen erfassen: (dynamische) Compound-SQL-Anweisungen, CALL, DELETE, INSERT, MERGE, REFRESH TABLE, SELECT, SET INTEGRITY, SELECT INTO, UPDATE, VALUES und VALUES INTO. Bei XQuery-Anweisungen können Sie EXPLAIN-Informationen für XQUERY-Anweisungen mit 'db2-fn:xmlcolumn' und 'db2-fn:sqlquery' abrufen.

  Anmerkung: Anweisungen REFRESH TABLE und SET INTEGRITY werden nur dynamisch kompiliert.
- Eine bestimmte SQL-Anweisung bei SQL-Anweisungen zum inkrementellen Binden (Incremental Bind).
- Eine bestimmte SQL-Anweisung bei dynamischen SQL-Anweisungen.
- Jede EXPLAIN-Anweisung (dynamisch oder statisch).

Die EXPLAIN-Funktion, die durch Absetzen der Anweisung EXPLAIN oder durch Verwenden der EXPLAIN-Schnittstellen für Abschnitte aufgerufen wird, erfasst Informationen zu dem für eine bestimmte, mit EXPLAIN bearbeitbare Anweisung ausgewählten Zugriffsplan und schreibt diese Informationen in EXPLAIN-Tabellen. Sie müssen die EXPLAIN-Tabellen erstellen, bevor Sie die Anweisung EXPLAIN absetzen. Zur Erstellung der Tabellen führen Sie das Script `EXPLAIN.DDL` aus, das sich im Unterverzeichnis `misc` des Unterverzeichnisses `sqllib` befindet.

Mithilfe der Prozedur SYSPROC.SYSINSTALLOBJECTS können Sie EXPLAIN-Tabellen ebenfalls erstellen, löschen und prüfen. Die Tabellen können unter einem bestimmten Schema und in einem bestimmten Tabellenbereich erstellt werden. Ein Beispiel hierfür finden Sie in der Datei `EXPLAIN.DDL`.

EXPLAIN-Tabellen können für mehrere Benutzer gemeinsame Daten enthalten. Die Tabellen können für einen Benutzer definiert werden. Anschließend können für jeden weiteren Benutzer Aliasnamen erstellt werden, die auf die definierten Tabellen verweisen. Alternativ können die EXPLAIN-Tabellen 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 für diese Tabellen besitzen.

Tabelle 55. Zusammenfassung der EXPLAIN-Tabellen

Tabellenname	Beschreibung
ADVISE_INDEX	Speichert Informationen über empfohlene Indizes. Die Tabelle kann durch den Abfragecompiler, den Befehl **db2advis** oder einen Benutzer mit Daten gefüllt werden. Diese Tabelle wird zu folgenden Zwecken verwendet: • Zum Abrufen empfohlener Indizes • Zum Bewerten von Indizes auf der Grundlage von Eingaben zu vorgeschlagenen Indizes
ADVISE_INSTANCE	Enthält Informationen zur Ausführung von **db2advis**. Diese Informationen beinhalten auch die Startzeit. Für jede Ausführung von **db2advis** wird eine Zeile eingefügt.
ADVISE_MQT	Enthält die Abfrage, die die einzelnen empfohlenen MQTs (MQT, Materialized Query Table) definiert, die Statistikdaten für die MQTs, wie z. B. COLSTATS (im XML-Format), NUMROWS usw., sowie die Stichprobenabfrage zum Abrufen detaillierter Statistikdaten für die MQTs.
ADVISE_PARTITION	Speichert virtuelle Datenbankpartitionen, die von **db2advis** generiert und bewertet werden.
ADVISE_TABLE	Speichert die DDL-Anweisungen (Data Definition Language) für die Erstellung von Tabellen unter Verwendung der endgültigen Empfehlungen des Designadvisors für MQTs, MDC-Tabellen und Datenbankpartitionierung.
ADVISE_WORKLOAD	Jede Zeile in der Tabelle stellt eine SQL- oder XQuery-Anweisung in einer Auslastung dar. Der Befehl **db2advis** verwendet diese Tabelle, um Auslastungsdaten zu erfassen und zu speichern.
EXPLAIN_ACTUALS	Enthält die EXPLAIN-Informationen zu Ist-Daten für einen Abschnitt.
EXPLAIN_ARGUMENT	Enthält die spezifischen Merkmale der einzelnen Operatoren (sofern vorhanden).
EXPLAIN_DIAGNOSTIC	Enthält einen Eintrag für jede Diagnosenachricht, die für eine bestimmte Instanz einer EXPLAIN-Anweisung in der Tabelle EXPLAIN_STATEMENT erstellt wird.

Tabelle 55. Zusammenfassung der EXPLAIN-Tabellen (Forts.)

Tabellenname	Beschreibung
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 SQL-Anweisung beziehen, durch die die Nachricht generiert wurde.
EXPLAIN_INSTANCE	Die Hauptsteuertabelle für alle EXPLAIN-Informationen. Jede Zeile in den EXPLAIN-Tabellen ist explizit mit exakt einer Zeile in dieser Tabelle verbunden. In dieser Tabelle werden grundlegende Informationen über die Quelle der SQL- oder XQuery-Anweisungen, die mit EXPLAIN bearbeitet werden, sowie Umgebungsinformationen gespeichert.
EXPLAIN_OBJECT	Identifiziert die Datenobjekte, die für den Zugriffsplan erforderlich sind, der zur Erfüllung einer SQL- oder XQuery-Anweisung generiert wurde.
EXPLAIN_OPERATOR	Enthält alle Operatoren, die der Abfragecompiler zur Erfüllung einer SQL- oder XQuery-Anweisung benötigt.
EXPLAIN_PREDICATE	Enthält Informationen über die Vergleichselemente, die von einem bestimmten Operator angewendet 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-Informationen für Datenobjekte:

Ein einzelner Zugriffsplan kann zur Erfüllung einer SQL- oder XQuery-Anweisung ein oder mehrere Datenobjekte verwenden.

Objektstatistiken

Die EXPLAIN-Einrichtung zeichnet Informationen über jedes Objekt auf, wie zum Beispiel folgende Angaben:

- Die Erstellungszeit
- Den Zeitpunkt, zu dem zum letzten Mal Statistiken für das Objekt erfasst 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 das 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 im 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

EXPLAIN-Informationen für Datenoperatoren:

Ein einzelner Zugriffsplan kann mehrere Operationen an den Daten ausführen, um eine SQL- bzw. XQuery-Anweisung zu erfüllen und die Ergebnisse an Sie zurückzugeben. Der Abfragecompiler bestimmt die erforderlichen Operationen, wie zum Beispiel eine Tabellensuche, eine Indexsuche, einen Join mit Verschachtelungsschleife (Nested Loop Join) oder einen Gruppierungsoperator (GROUP BY).

Neben Informationen zu den einzelnen Operatoren, die in einem Zugriffsplan verwendet werden, zeigt die EXPLAIN-Ausgabe auch die kumulativen Auswirkungen des Zugriffsplans.

Informationen zum geschätzten Aufwand

Die folgenden Schätzungen zum kumulativen Aufwand für Operatoren werden aufgezeichnet. Diese Angaben beziehen sich auf den ausgewählten Zugriffsplan bis einschließlich zu dem Operator, für den die Informationen erfasst werden.

- Der Gesamtaufwand (in Timerons)
- Die Anzahl der Seiten-E/A-Operationen
- Die Anzahl von Verarbeitungsinstruktionen
- Der Aufwand (in Timerons) für den Abruf der ersten Zeile, einschließlich des erforderlichen Anfangsaufwands
- Der Übertragungsaufwand (in Rahmen (Frames))

Ein *Timeron* ist eine künstliche relative Maßeinheit. Timeronwerte werden durch das Optimierungsprogramm auf der Basis interner Werte ermittelt, wie zum Beispiel Statistiken, die sich im Lauf der Datenbankverwendung ändern. Infolgedes-

sen sind die Timeronwerte für eine SQL- bzw. XQuery-Anweisung nicht unbedingt jedes Mal gleich, wenn ein geschätzter Aufwand in Timerons ermittelt wird.

Operatormerkmale

Die folgenden Informationen, die die Merkmale der einzelnen Operatoren zu beschreiben, werden durch die EXPLAIN-Funktion erfasst:

- Die Gruppe von Tabellen, auf die zugegriffen wurde
- Die Gruppen von Spalten, auf die zugegriffen wurde
- Die Spalten, nach denen die Daten angeordnet werden, wenn das Optimierungsprogramm feststellt, dass diese Reihenfolge von nachfolgenden Operatoren genutzt werden kann
- Die Gruppe von Vergleichselementen, die angewendet wurden
- Die geschätzte Anzahl von Zeilen, die zurückgegeben werden (Kardinalität)

EXPLAIN-Informationen für Instanzen:

Informationen über EXPLAIN-Instanzen werden in der Tabelle EXPLAIN_INSTANCE gespeichert. Weitere spezifische Informationen zu den einzelnen Abfrageanweisungen in einer Instanz werden in der Tabelle EXPLAIN_STATEMENT gespeichert.

Kennzeichnung von EXPLAIN-Instanzen

Die folgenden Informationen helfen Ihnen bei der Identifizierung einer bestimmten EXPLAIN-Instanz sowie bei der Zuordnung der Informationen über bestimmte Anweisungen zu einem bestimmten Aufruf der EXPLAIN-Funktion:

- Der Benutzer, der die EXPLAIN-Informationen anforderte
- Der Zeitpunkt, zu dem die EXPLAIN-Anforderung begann
- Der Name des Pakets, das die mit EXPLAIN bearbeitete Anweisung enthält
- Das SQL-Schema des Pakets, das die mit EXPLAIN bearbeitete Anweisung enthält
- Die Version des Pakets, das die Anweisung enthält
- Eine Angabe, ob Momentaufnahmedaten (Snapshot) erfasst wurden

Umgebungseinstellungen

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 des DB2-Produkts
- Der Grad der Parallelität, unter dem die Abfrage kompiliert wurde

  Das Sonderregister CURRENT DEGREE, die Bindeoption DEGREE, der Befehl **SET RUNTIME DEGREE** und der Datenbankkonfigurationsparameter **dft_degree** bestimmen den Grad der Parallelität, unter dem eine bestimmte Abfrage kompiliert wird.
- Die Angabe, ob die Anweisung dynamisch oder statisch ist
- Die zum Kompilieren der Abfrage verwendete Optimierungsklasse
- Der Typ der Zeilenblockung für Cursor, der beim Kompilieren der Abfrage auftritt
- Die Isolationsstufe, unter der die Abfrage ausgeführt wird

- Die Werte verschiedener Konfigurationsparameter zum Zeitpunkt der Kompilierung der Abfrage. Werte für die folgenden Parameter werden bei der Erfassung einer EXPLAIN-Momentaufnahme (Snapshot) aufgezeichnet:
  - Sortierspeichergröße (**sortheap**)
  - Durchschnittliche Anzahl aktiver Anwendungen (**avg_appls**)
  - Datenbankzwischenspeicher (**dbheap**)
  - Maximaler Speicher für Sperrenliste (**locklist**)
  - Maximale Anzahl von Sperren vor Eskalation (**maxlocks**)
  - CPU-Geschwindigkeit (**cpuspeed**)
  - Kommunikationsbandbreite (**comm_bandwidth**)

Kennzeichnung von Anweisungen

Für jede EXPLAIN-Instanz können mehrere Anweisungen 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 oder SET INTEGRITY
- 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. Wenn EXPLAIN MODE oder EXPLAIN SNAPSHOT aktiv ist und dynamische EXPLAIN-Anweisungen in einer Sitzung des Befehlszeilenprozessors (CLP) oder von Call Level Interface (CLI) übergeben werden, wird der QUERYTAG-Wert auf „CLP" bzw. „CLI" gesetzt. 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-Anweisungen, die nicht über CLP bzw. CLI angefordert werden oder die die EXPLAIN-Anweisung nicht verwenden, wird der QUERYTAG-Wert auf Leerzeichen gesetzt, wobei der QUERYNO-Wert immer 1 ist.

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 Anweisungstexts aufgezeichnet. Eine Version ist der Code, den der Abfragecompiler von der Anwendung empfängt. Die andere Version ist die rückübersetzte Version aus der internen (im Compiler befindlichen) 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 Optimierungsprogramm den Zugriffsplan ausgewählt hat. Um zu verstehen, wie der Compiler Ihre Abfrage zur Optimierung umgeschrieben hat, vergleichen Sie den vom Benutzer geschriebenen Anweisungstext mit der internen Darstellung der Abfrageanweisung. Die umgeschriebene Anweisung enthält zudem weitere Faktoren, die sich auf Ihre Anweisung auswirken, wie zum Beispiel Trigger oder Integritätsbedingungen. Einige Kennungen, die in diesem „optimierten" Text verwendet werden, sind folgende:

$C*n* Der Name einer abgeleiteten Spalte, wobei *n* ein ganzzahliger Wert ist.

$CONSTRAINT$
Diese Kennung gibt eine Integritätsbedingung an, die der ursprünglichen Anweisung bei der Kompilierung hinzugefügt wurde und in Verbindung mit dem Präfix $WITH_CONTEXT$ zu sehen ist.

$DERIVED.T*n*
Der Name einer abgeleiteten Tabelle, wobei *n* ein ganzzahliger Wert ist.

$INTERNAL_FUNC$
Diese Kennung weist auf das Vorhandensein einer Funktion hin, die vom Compiler für die mit EXPLAIN bearbeitete Abfrage verwendet wird, jedoch nicht zur allgemeinen Verwendung verfügbar ist.

$INTERNAL_PRED$
Diese Kennung weist auf das Vorhandensein eines Vergleichselements hin, das 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 oder Integritätsbedingungen der ursprünglichen Anweisung hinzugefügten Zusatzkontext zu erfüllen.

$INTERNAL_XPATH$
Diese Kennung weist auf das Vorhandensein einer internen Tabellenfunktion hin, die als Parameter ein einziges mit Anmerkungen versehenes XPath-Muster akzeptiert und eine Tabelle mit einer oder mehreren Spalten zurückgibt, die dem Muster entsprechen.

RID Diese Kennung gibt die Satz-ID-Spalte (RID-Spalte) für eine bestimmte Zeile an.

$TRIGGER$
Diese Kennung gibt einen Trigger an, der der ursprünglichen Anweisung bei der Kompilierung hinzugefügt wurde und in Verbindung mit dem Präfix $WITH_CONTEXT$ zu sehen ist.

$WITH_CONTEXT$(...)
Dieses Präfix tritt am Anfang des Texts auf, wenn zusätzliche Trigger oder Integritätsbedingungen zur ursprünglichen Abfrageanweisung hinzugefügt

wurden. Diesem Präfix folgt eine Liste der Namen aller Trigger oder Integritätsbedingungen, die sich auf die Kompilierung und Auflösung der Anweisung auswirken.

SQL- und XQuery-EXPLAIN-Tool

Der Befehl **db2expln** beschreibt den Zugriffsplan, der für SQL- oder XQuery-Anweisungen ausgewählt wurde.

Mithilfe dieses Tools können Sie eine schnelle Erläuterung des ausgewählten Zugriffsplans abrufen, wenn keine EXPLAIN-Daten erfasst wurden. Für statische SQL- und XQuery-Anweisungen überprüft das Tool **db2expln** die Pakete, die im Systemkatalog gespeichert sind. Für dynamische SQL- und XQuery-Anweisungen überprüft das Tool **db2expln** die Abschnitte im Abfragecache.

Das EXPLAIN-Tool befindet sich im Unterverzeichnis `bin` des Verzeichnisses `sqllib` für Ihre Instanz. Wenn sich das Tool **db2expln** nicht in Ihrem aktuellen Verzeichnis befindet, muss es in einem Verzeichnis enthalten sein, das in Ihrer Umgebungsvariablen PATH definiert ist.

Der Befehl **db2expln** verwendet die Dateien `db2expln.bnd`, `db2exsrv.bnd` und `db2exdyn.bnd`, um sich an eine Datenbank zu binden, wenn auf die Datenbank zum ersten Mal zugegriffen wird.

Beschreibung der Ausgabe von 'db2expln':

Die EXPLAIN-Ausgabe des Befehls **db2expln** umfasst sowohl Paketinformationen als auch Abschnittsinformationen für jedes Paket.

- Die Paketinformationen enthalten das Datum der Bindeoperation und die relevanten Bindeoptionen.
- Die Abschnittsinformationen enthalten die Abschnittsnummer und die SQL- bzw. XQuery-Anweisung, die mit EXPLAIN bearbeitet wurde.

Die EXPLAIN-Ausgabe zu dem für die SQL- oder XQuery-Anweisung ausgewählten Zugriffsplan wird unter den Abschnittsinformationen angezeigt.

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 einen Zugriffsplan befinden sich Einrückungsbalken. Diese Balken markieren gleichzeitig den Geltungsbereich jeder einzelnen Operation. Operationen auf einer niedrigeren Einrückungsstufe weiter rechts werden vor denen verarbeitet, die auf der vorherigen Einrückungsstufe angezeigt werden.

Der ausgewählte Zugriffsplan basiert auf einer erweiterten Version der ursprünglichen SQL-Anweisung, auf der *effektiven SQL-Anweisung*, wenn der Anweisungskonzentrator aktiviert ist, oder auf der XQuery-Anweisung, die in der Ausgabe gezeigt wird. Da die Komponente des Abfragecompilers zum Umschreiben von Abfragen die SQL- oder XQuery-Anweisung in ein äquivalentes, jedoch effizienteres Format umschreiben kann, kann sich der in der EXPLAIN-Ausgabe gezeigte Zugriffsplan wesentlich vom erwarteten Zugriffsplan unterscheiden. Die EXPLAIN-Funktion, zu der die EXPLAIN-Tabellen, die Anweisung SET CURRENT EXPLAIN MODE 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.

Wenn Sie eine Ausgabe des Befehls **db2expln** mit einer Ausgabe der EXPLAIN-Funktion vergleichen, kann die Operator-ID-Option (-opids) sehr nützlich sein. Jedesmal, wenn **db2expln** mit der Verarbeitung eines neuen Operators aus der EXPLAIN-Funktion beginnt, wird die Operator-ID links neben dem von EXPLAIN gezeigten Plan ausgegeben. Die Operator-ID kann zum Vergleichen von Schritten in den verschiedenen Darstellungen des Zugriffsplans verwendet werden. Beachten Sie, dass es nicht immer eine Eins-zu-eins-Entsprechung zwischen den Operatoren in der Ausgabe der EXPLAIN-Funktion und den Operationen gibt, die von **db2expln** angezeigt werden.

Tabellenzugriffsinformationen:

Eine Angabe in der **db2expln**-Ausgabe enthält den Namen und den Typ der Tabelle, auf die zugegriffen wird.

Informationen zu regulären Tabellen enthalten eine der folgenden Tabellenzugriffsangaben:

```
Access Table Name = schema.name  ID = ts,n
Access Hierarchy Table Name = schema.name  ID = ts,n
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 Eintrag der Katalogsicht SYSCAT.TABLES für die Tabelle.

Informationen zu temporären Tabellen enthalten eine der folgenden Tabellenzugriffsangaben:

```
Access Temp Table  ID = tn
Access Global Temp Table  ID = ts,tn
```

Dabei ist ID die entsprechende TABLESPACEID aus dem Eintrag der Katalogsicht SYSCAT.TABLES für die Tabelle (*ts*) oder die entsprechende von **db2expln** zugeordnete Kennung (*tn*).

Nach der Tabellenzugriffsangabe werden die folgenden zusätzlichen Angaben aufgeführt, die den Zugriff näher beschreiben.

- Anzahl von Spalten
- Blockzugriff
- Parallelsuche
- Suchrichtung
- Zeilenzugriff
- Geplante Sperre
- Vergleichselement
- Verschiedene Variablen

Angabe der Anzahl von Spalten

Die folgende Angabe zeigt die Anzahl von Spalten an, die aus jeder Zeile der Tabelle verwendet werden:

```
#Columns = n
```

Angabe eines Blockzugriffs

Die folgende Angabe zeigt an, dass für die Tabelle ein oder mehrere Dimensionsblockindizes definiert sind:

```
Clustered by Dimension for Block Index Access
```

Wenn diese Angabe fehlt, wurde die Tabelle ohne die Klausel ORGANIZE BY DIMENSIONS erstellt.

Angabe einer Parallelsuche

Die folgende Angabe zeigt an, dass der Datenbankmanager mehrere Subagenten zum parallelen Lesen der Tabelle verwendet:

```
Parallel Scan
```

Wenn diese Angabe fehlt, wird die Tabelle nur von einem Agenten (bzw. Subagenten) gelesen.

Angabe der Suchrichtung

Die folgende Angabe zeigt an, dass der Datenbankmanager Zeilen in umgekehrter Reihenfolge liest:

```
Scan Direction = Reverse
```

Wenn diese Angabe fehlt, ist die Suchrichtung vorwärts (Standardwert).

Angaben für Zeilenzugriff

Eine der folgenden Angaben zeigt an, wie auf Zeilen in der Tabelle, die den angegebenen Bedingungen entsprechen, zugegriffen wird.

- Die Angabe `Relation Scan` bedeutet, dass die Tabelle sequenziell nach Zeilen durchsucht wird, die den angegebenen Bedingungen entsprechen.
  - 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, bestimmt 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 von Zeilen, die den angegebenen Kriterien entsprechen, und der Zugriff auf sie über einen Index erfolgen:

  ```
  Index Scan:  Name = schema.name  ID = xx
  |  Index type
  |  Index Columns:
  ```

  Dabei gilt:

  - *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 over XML data
```

Diesen Angaben folgen jeweils eine Ausgabezeile für jede Spalte im Index. Diese Informationen können folgende Formate haben:

```
n: spaltenname (Ascending)
n: spaltenname (Descending)
n: spaltenname (Include Column)
```

Die folgenden Angaben präzisieren den Typ der Indexsuche.

– Die bereichsbegrenzenden Vergleichselemente für den Index werden durch folgende Angaben dargestellt:

```
#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 des Schlüssels wird durch eines der folgenden Elemente 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: ?
  ```

  Nur die ersten 20 Zeichen einer Literalzeichenfolge werden angezeigt. Ist die Zeichenfolge länger als 20 Zeichen, wird dies durch ein Auslassungszeichen (...) am Ende der Zeichenfolge angezeigt. Einige Schlüssel können erst bestimmt werden, wenn der Abschnitt (section) ausgeführt wird. Dies wird durch ein Fragezeichen (?) als Wert angezeigt.

– `Index-Only Access`

  Wenn alle benötigten Spalten aus dem Indexschlüssel abgerufen werden können, wird diese Angabe für einen reinen Indexzugriff angezeigt. Es wird nicht auf Tabellendaten zugegriffen.

– Die folgende Angabe bedeutet, dass kein Vorablesezugriff 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 Vorablesezugriff auf Datenseiten erfolgt:

  ```
  Data Prefetch: None
  ```

– Die folgende Angabe weist darauf hin, dass Datenseiten wahrscheinlich vorabgelesen werden:

```
Data Prefetch: Eligible
```

– Wenn Vergleichselemente vorhanden sind, die an den Indexmanager übermittelt werden können, um bei der Ermittlung der entsprechenden Indexeinträge zu helfen, zeigt die folgende Angabe die Anzahl dieser Vergleichselemente an:

```
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 durch folgende Angabe gekennzeichnet:

```
Fetch Direct Using Block or Row IOs
```

Angaben von geplanten Sperren

Für jeden Tabellenzugriff wird der Typ der Sperre, die auf Tabellen- und Zeilenebene 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 Weak Exclusive
- None
- Share
- Update

Angaben von Vergleichselementen

Es gibt drei Typen von Angaben, die Informationen über die in einem Zugriffsplan verwendeten Vergleichselemente enthalten.

- 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
```

- Die folgende Angabe zeigt die Anzahl von Vergleichselementen, die ausgewertet werden, während auf die Daten zugegriffen wird. Diese Anzahl enthält keine Pushdown-Operationen, wie zum Beispiel Spaltenberechnungen oder Sortierungen:

  ```
  Sargable Predicate(s)
  |  #Predicates = n
  ```

- 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 diesen Angaben angezeigten Vergleichselemente spiegelt möglicherweise die Anzahl der Vergleichselemente, die in der Abfrageanweisung angegeben wurden, 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 den Tabellenzugriff verwendete Isolationsstufe nicht der Isolationsstufe der Anweisung entspricht:

  ```
  Isolation Level: xxxx
  ```

  Dafür kann es eine Reihe möglicher Gründe geben. Beispiele:

  – Ein Paket, das mit der Isolationsstufe 'Wiederholbares Lesen' (RR) gebunden wurde, hat Auswirkungen auf bestimmte referenzielle Integritätsbedingungen. Der Zugriff auf die übergeordnete Tabelle zum Prüfen dieser Integritätsbedingungen wird auf die Isolationsstufe 'Cursorstabilität' (CS) heruntergestuft, um unnötige Sperren für diese Tabelle zu vermeiden.

  – Ein Paket, das mit der Isolationsstufe 'Nicht festgeschriebenes Lesen' (UR) gebunden wurde, enthält eine DELETE-Anweisung. Der Zugriff auf die Tabelle für die Löschoperation wird auf 'CS' hochgestuft.

- Die folgende Angabe weist darauf hin, dass einige oder alle Zeilen, die aus einer temporären Tabelle gelesen wurden, außerhalb des Pufferpools zwischengespeichert werden, wenn genügend Sortierspeicher (**sortheap**) verfügbar ist:

  ```
  Keep Rows In Private Memory
  ```

- Die folgende Angabe zeigt an, dass für die Tabelle das Attribut für flüchtige Kardinalität definiert ist:

  ```
  Volatile Cardinality
  ```

Informationen zu temporären Tabellen:

Eine temporäre Tabelle wird als Arbeitstabelle während der Ausführung eines Zeitplans verwendet. 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 benötigt wird, enthält die Ausgabe des Befehls **db2expln** eine der folgenden Angaben.

```
Insert Into Temp Table  ID = tn     --> Normale temporäre Tabelle
Insert Into Shared Temp Table  ID = tn     --> Normale temporäre Tabelle wird von mehreren
                                               Subagenten parallel erstellt.
Insert Into Sorted Temp Table  ID = tn     --> Sortierte temporäre Tabelle
Insert Into Sorted Shared Temp Table  ID = tn     --> Sortierte temp. Tabelle wird von
                                                      mehreren Subagenten parallel erstellt.

Insert Into Global Temp Table  ID = ts,tn     --> Deklarierte globale temporäre Tabelle
Insert Into Shared Global Temp Table  ID = ts,tn     --> Deklarierte globale temporäre
                                                          Tabelle wird von mehreren Sub-
                                                          agenten parallel erstellt.
Insert Into Sorted Global Temp Table  ID = ts,tn     --> Sortierte dekl. glob. temp. Tabelle
Insert Into Sorted Shared Global Temp Table  ID = ts,tn     --> Sort. dekl. glob. temp.
                                                                Tabelle wird von mehreren
                                                                Subagenten parallel erstellt.
```

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.

Auf jede dieser Angaben folgt die folgende Angabe:

```
#Columns = n
```

Diese Angabe zeigt die Anzahl der Spalten in jeder Zeile an, die in die temporäre Tabelle eingefügt wird.

Sortierte temporäre Tabellen

Sortierte temporäre Tabellen treten zum Beispiel als Ergebnis der folgenden Operationen auf:

- ORDER BY
- DISTINCT
- GROUP BY
- Mischjoin (Merge Join)
- Unterabfrage '= ANY'
- Unterabfrage '<> ALL'
- INTERSECT oder EXCEPT
- UNION (ohne das Schlüsselwort ALL)

Eine Reihe von Angaben, die sich auf eine sortierte temporäre Tabelle beziehen, können in der Ausgabe des Befehls **db2expln** angezeigt werden.

- Die folgende Angabe zeigt die Anzahl der bei der Sortierung verwendeten Schlüsselspalten:

  ```
  #Sort Key Columns = n
  ```

  Dazu wird eine der folgenden Zeilen für jede Spalte im Sortierschlüssel 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, sodass die optimale Größe für den Sortierspeicher zur Laufzeit zugeordnet werden kann:

```
Sortheap Allocation Parameters:
|  #Rows     = n
|  Row Width = n
```

- Die folgende Angabe wird angezeigt, wenn nur die ersten Zeilen des sortierten Ergebnisses benötigt werden:

  ```
  Sort Limited To Estimated Row Count
  ```

- Für Sortiervorgänge, die in einer symmetrischen Mehrprozessorumgebung (SMP-Umgebung) ausgeführt werden, 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 sortierte Ergebnis im Sortierspeicher verbleibt oder nicht:

  ```
  Piped
  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 Sortieroperation entfernt werden:

  ```
  Duplicate Elimination
  ```

- Wenn Spaltenberechnungen während der Sortieroperation ausgeführt werden, wird eine der folgenden Angaben angezeigt:

  ```
  Partial Aggregation
  Intermediate Aggregation
  Buffered Partial Aggregation
  Buffered Intermediate Aggregation
  ```

Abschluss von temporären Tabellen

Eine Abschlussangabe ('Completion') wird angezeigt, wenn eine temporäre Tabelle im Geltungsbereich eines Tabellenzugriffs erstellt wird. Diese Angabe kann eine der folgenden sein:

```
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. Eine Tabellenfunktion wird durch die folgenden Angaben mit Details zu den Attributen der Funktion angezeigt. Der spezifische Name identifiziert die aufgerufene Tabellenfunktion eindeutig.

```
Access User Defined Table Function
|  Name = schema.funktionsname
|  Specific Name = spezifischer_name
|  SQL Access Level = zugriffsebene
|  Language = sprache
|  Parameter Style = parmstil
|  Fenced                      Not Deterministic
|  Called on NULL Input        Disallow Parallel
|  Not Federated               Not Threadsafe
```

Joininformationen:

Die Ausgabe des Befehls **db2expln** kann Informationen zu Joins in einer mit EXPLAIN bearbeiteten Anweisung enthalten.

Wenn ein Join ausgeführt wird, wird eine der folgenden Angaben angezeigt:

```
Hash Join
Merge Join
Nested Loop Join
```

Ein linker Outer Join wird durch eine der folgenden Angaben angezeigt:

```
Left Outer Hash Join
Left Outer Merge Join
Left Outer Nested Loop Join
```

Im Fall eines Mischjoins (Merge Join) oder eines Joins mit Verschachtelungsschleife (Nested Loop Join) ist die äußere Tabelle des Joins die Tabelle, auf die in der vorherigen Zugriffsangabe (in der Ausgabe) verwiesen wird. Die innere Tabelle des Joins ist die Tabelle, auf die in der Zugriffsangabe verwiesen wird, die sich im Bereich der Joinangabe befindet. Im Fall eines Hash-Joins sind die Zugriffsangaben umgekehrt: die äußere Tabelle ist im Joinbereich enthalten und die innere Tabelle wird vor dem Join angezeigt.

Im Fall eines Hash- oder Mischjoins können folgende weitere Angaben auftreten:

- `Early Out: Single Match Per Outer Row`

  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.
- ```
  Residual Predicate(s)
  |  #Predicates = n
  ```

  Es ist möglich, nach Abschluss eines Joins Vergleichselemente angewendet werden. Diese Angabe zeigt die Anzahl der Vergleichselemente an, die angewendet werden.

Im Fall eines Hash-Joins können folgende weitere Angaben auftreten:

- `Process Hash Table For Join`

  Die Hashtabelle wird aus der inneren Tabelle erstellt. Diese Angabe zeigt an, ob die Erstellung der Hashtabelle während des Zugriffs auf die innere Tabelle in ein Vergleichselement verschoben wurde.
- `Process Probe Table For Hash Join`

  Beim Zugriff der äußeren Tabelle kann eine Prüftabelle erstellt werden, um die Leistung des Joins zu verbessern. Diese Angabe zeigt an, ob eine Prüftabelle während des Zugriffs auf die äußere Tabelle erstellt wurde.
- `Estimated Build Size: n`

  Diese Angabe zeigt die geschätzte Anzahl Byte an, die zum Erstellen der Hashtabelle benötigt werden.
- `Estimated Probe Size: n`

  Diese Angabe zeigt die geschätzte Anzahl Byte an, die zum Erstellen der Prüftabelle benötigt werden.
  ```

Im Fall eines Joins mit Verschachtelungsschleife kann die folgende 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 *zusammengesetzte innere Tabelle* (Composite Inner Table) bezeichnet.

Wenn ein Join mehr als zwei Tabellen umfasst, müssen die EXPLAIN-Schritte von oben nach unten gelesen werden. Nehmen Sie zum Beispiel an, dass die EXPLAIN-Ausgabe die folgende Struktur hat:

```
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 Tabelle W.
2. Join einer Zeile aus W mit der nächsten Zeile aus Tabelle X und Bezeichnen des Ergebnisses als P1 (für partielles Joinergebnis Nr. 1).
3. Join von P1 mit der nächsten Zeile aus Tabelle Y zum Erstellen von P2.
4. Join von P2 mit der nächsten Zeile aus Tabelle 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

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 Datenstromkonzept ermöglicht es, eine Gruppe von Operationen innerhalb eines Zugriffsplans als Einheit zu steuern.

Der Beginn eines Datenstroms wird durch folgende Angabe in der Ausgabe des Befehls **db2expln** gekennzeichnet:

```
Data Stream n
```

Dabei ist *n* eine eindeutige Kennung, die zur leichteren Bezugnahme durch **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. Auf die einleitende Datenstromangabe können daher 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
  Not Piped
  ```

  Ein über eine Pipe geleiteter Datenstrom wird möglicherweise auf Platte geschrieben, wenn bei der Ausführung nicht genügend Hauptspeicher verfügbar ist. Der Zugriffsplan berücksichtigt beide Möglichkeiten.

- 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
```

Informationen zu INSERT-, UPDATE- und DELETE-Anweisungen:

Der EXPLAIN-Text für INSERT-, UPDATE- oder DELETE-Anweisungen ist selbsterklärend.

Der Text zur Angabe dieser SQL-Operationen in der Ausgabe des Befehls **db2expln** kann wie folgt aussehen:

```
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
```

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 (beim logischen Verknüpfen von Indizes über OR - Index ORing) oder wenn eine Technik verwendet wird, mit der festgestellt wird, welche IDs in allen Indizes, auf die zugegriffen wird, enthalten sind (beim logischen Verknüpfen von Indizes über AND - Index ANDing), bevor auf die Tabelle zugegriffen wird.

Die Informationen zur Vorbereitung von IDs (Kennungen), die in der EXPLAIN-Ausgabe angezeigt werden, haben drei Hauptverwendungszwecke:

- Die beiden folgenden Angaben bedeuten, dass ein logisches Verknüpfen von Indizes über OR (Index ORing) zur Vorbereitung der Liste der qualifizierten IDs verwendet wird:

  ```
  Index ORing Preparation
  Block Index ORing Preparation
  ```

Index ORing (logisches Verknüpfen von Indizes über OR) bezeichnet ein Verfahren, bei dem auf mehrere Indizes zugegriffen und die Ergebnisse kombiniert werden, um die unterschiedlichen (distinkten) IDs einzuschließen, die in beliebigen der Indizes enthalten sind. Das Optimierungsprogramm zieht das Index ORing-Verfahren in Betracht, wenn Vergleichselemente durch Schlüsselwörter OR verknüpft werden oder ein IN-Vergleichselement vorhanden ist.

- Eine der folgenden Angaben zeigt an, dass Eingabedaten zur Verwendung beim Vorablesezugriff über Listen vorbereitet wurden:

  ```
  List Prefetch Preparation
  Block List Prefetch RID Preparation
  ```

- *Index ANDing* (logisches Verknüpfen von Indizes über AND) bezeichnet ein Verfahren, bei dem auf mehrere Indizes zugegriffen und die Ergebnisse kombiniert werden, um die IDs einzuschließen, die in allen Indizes, auf die zugegriffen wird, enthalten sind. Das Index ANDing beginnt mit einer der folgenden Angaben:

  ```
  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 Index ANDing-Filteroperationen verarbeiten die IDs und verwenden Bitfilterverfahren, um die IDs zu bestimmen, die in jedem Index enthalten sind, auf den zugegriffen wird. Die folgenden Angaben weisen darauf hin, dass IDs für das logische Verknüpfen von Indizes über AND verarbeitet wurden:

  ```
  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 Bitzuordnung (Bitmap) abgeschätzt hat, wird der Schätzwert mit der folgenden Angabe angezeigt:

  ```
  Optimizer Estimate of Set Size: n
  ```

Wenn ein Vorablesezugriff über Listen für einen Typ von ID-Vorbereitung ausgeführt werden kann, wird dies durch die folgende Angabe angezeigt:

```
Prefetch: Enabled
```

Informationen zu Spaltenberechnungen (Aggregation):

Spaltenberechnungen werden an Zeilen ausgeführt, welche die Bedingungen erfüllen, die durch Vergleichselemente in einer SQL-Anweisung definiert werden.

Wenn eine Spaltenfunktion (Aggregatfunktion) ausgeführt wird, wird eine der folgenden Angaben in der Ausgabe des Befehls **db2expln** angezeigt:

```
Aggregation
Predicate Aggregation
Partial Aggregation
Partial Predicate Aggregation
```

```
Intermediate Aggregation
Intermediate Predicate Aggregation
Final Aggregation
Final Predicate Aggregation
```

Die Angabe 'Predicate aggregation' bedeutet, dass die Operation der Spaltenberechnung durch ein Vergleichselement beim Zugriff auf die Daten verarbeitet wurde.

Auf die Angabe der Spaltenberechnung folgt eine weitere Angabe, die den Typ der ausgeführten Spaltenfunktion bezeichnet:

```
Group By
Column Function(s)
Single Record
```

Die jeweilige Spaltenfunktion kann aus 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 durch ein Vergleichselement ('predicate aggregation') ausgeführt wurde, ist eine Operation zum Abschluss ('Completion') der Spaltenberechnung und eine entsprechende Ausgabe vorhanden:

```
Aggregation Completion
Partial Aggregation Completion
Intermediate Aggregation Completion
Final Aggregation Completion
```

Informationen zur Parallelverarbeitung:

Für die parallele Ausführung einer SQL-Anweisung (entweder mit partitionsinterner oder partitionsübergreifender Parallelität) sind einige besondere Zugriffsplanoperationen erforderlich.

- Bei der Ausführung eines Zugriffsplans mit partitionsinterner Parallelität werden Abschnitte des Plans gleichzeitig mithilfe mehrerer Subagenten ausgeführt. Die Erstellung dieser Subagenten wird durch folgende Angabe in der Ausgabe des Befehls **db2expln** 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 eine oder mehrere Datenbankpartitionen gesendet. Ein wichtiger Teilbereich ist der *Koordinatorteilbereich*. Der Koordinatorteilbereich ist der erste Teilbereich in jedem Plan. Er übernimmt zuerst die Steuerung und ist 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 folgende Angabe bedeutet, dass der Teilbereich abhängig vom Wert der Spalten zu einer Datenbankpartition 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 eine vorbestimmte Datenbankpartition gesendet wird. (Dies kommt häufig vor, wenn die Skalarfunktion DBPARTITIONNUM() in der Anweisung verwendet wird.)

    ```
    Directed by Node Number
    ```

– Die folgende Angabe weist darauf hin, dass der Teilbereich an die Datenbankpartition gesendet wird, die einer vorbestimmten Datenbankpartitionsnummer in der Datenbankpartitionsgruppe entspricht. (Dies kommt häufig vor, wenn die Skalarfunktion HASHEDVALUE() in der Anweisung verwendet wird.)

```
Directed by Partition Number
|  Partition Map ID = n, Nodegroup = ngname, #Nodes = n
```

– Die folgende Angabe bedeutet, dass der Teilbereich an die Datenbankpartition gesendet wird, die die aktuelle Zeile für den Cursor der Anwendung verfügbar gemacht hat.

```
Directed by Position
```

– Die folgende Angabe bedeutet, dass nur eine einzige Datenbankpartition, die bei der Kompilierung der Anweisung festgelegt wird, den Teilbereich empfängt.

```
Directed to Single Node
|  Node Number = n
```

– Die folgenden Angaben bedeuten, dass der Teilbereich in der Koordinatordatenbankpartition ausgeführt wird.

```
Directed to Application Coordinator Node
Directed to Local Coordinator Node
```

– Die folgende Angabe bedeutet, dass der Teilbereich an alle aufgelisteten Datenbankpartitionen gesendet wird.

```
Broadcast to Node List
|  Nodes = n1, n2, n3, ...
```

– Die folgende Angabe bedeutet, dass nur eine einzige Datenbankpartition, die 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.
  – 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 für Datenbankpartitionen wird das Ziel für Zeilen, die in die Tabellenwarteschlange eingefügt werden, durch eine der folgenden Angaben beschrieben:

    Jede Zeile wird an die Koordinatordatenbankpartition gesendet:

```
Broadcast to Coordinator Node
```

    Jede Zeile wird an jede Datenbankpartition gesendet, in 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 eine zufällig bestimmte Datenbankpartition gesendet:

```
Send to Random Node
```

– In einigen Fällen ist es möglich, dass eine Tabellenwarteschlange für Datenbankpartitionen einige Zeilen in eine temporäre Tabelle versetzen muss (Überlauf). Diese Möglichkeit wird durch folgende Angabe angezeigt:

```
Rows Can Overflow to Temporary Table
```

– Auf einen Tabellenzugriff, der eine Pushdown-Operation zum Einfügen von Zeilen in eine Tabellenwarteschlange einschließt, folgt eine Abschlussangabe ("Completion"), die Zeilen handhabt, die nicht unverzüglich gesendet werden konnten. In diesem Fall wird eine der folgenden Zeilen 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 Angaben folgt immer die Anzahl der Zeilen, die abgerufen werden.

```
#Columns = n
```

– Wenn die Tabellenwarteschlange die Zeilen auf der Empfängerseite sortiert, wird eine der folgenden Angaben angezeigt:

```
Output Sorted
Output Sorted and Unique
```

Diesen Angaben folgt die Anzahl der Schlüssel, die für die Sortierung verwendet werden.

```
#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 Angabe 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
```

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 Ausgabe des Befehls **db2expln** zeigt an, dass eine Datenquelle gelesen wird:

```
Ship Distributed Subquery #n
|   #Columns = n
```

Wenn Vergleichselemente auf Daten angewendet werden, die von einer verteilten Unterabfrage zurückgegeben werden, wird die Anzahl der angewendeten Vergleichselemente durch die folgenden Angaben angezeigt:

```
Residual Predicate(s)
|   #Predicates = n
```

Eine INSERT-, UPDATE- oder DELETE-Operation, die an einer Datenquelle stattfindet, wird durch eine der folgenden Angaben gezeigt:

```
Ship Distributed Insert #n
Ship Distributed Update #n
Ship Distributed Delete #n
```

Wenn eine Tabelle an einer Datenquelle explizit gesperrt wird, wird die folgende Angabe angezeigt:

```
Ship Distributed Lock Table #n
```

DDL-Anweisungen (Data Definition Language) für eine Datenquelle werden in zwei Teile aufgeteilt. Der Teil, der an der Datenquelle aufgerufen wird, wird durch folgende Angabe angezeigt:

```
Ship Distributed DDL Statement #n
```

Wenn der Server der föderierten Datenbanken eine partitionierte Datenbank ist, muss ein Teil der DDL-Anweisung in der Katalogdatenbankpartition ausgeführt werden. Dies wird durch folgende Angabe angezeigt:

```
Distributed DDL Statement #n Completion
```

Die Details für die einzelnen verteilten Unteranweisungen werden separat angezeigt.

- Die Datenquelle für die Unterabfrage wird durch eine der folgenden Angaben angezeigt:

  ```
  Server: servername (typ, version)
  Server: servername (typ)
  Server: servername
  ```

- Wenn es sich um eine relationale Datenquelle handelt, wird das SQL für die Unteranweisung wie folgt dargestellt:

  ```
  SQL Statement:
       anweisung
  ```

  Nicht relationale Datenquellen werden wie folgt angegeben:

  ```
  Non-Relational Data Source
  ```

- Kurznamen (Nicknames), auf die in der Unteranweisung verwiesen wird, werden folgendermaßen aufgelistet:

  ```
  Nicknames Referenced:
     schema.kurzname ID = n
  ```

  Bei einer relationalen Datenquelle wird die Basistabelle für den Kurznamen wie folgt angezeigt:

  ```
  Base = basisschema.basistabelle
  ```

  Bei einer nicht relationalen Datenquelle wird die Quellendatei für den Kurznamen wie folgt angezeigt:

  ```
  Source File = dateiname
  ```

- Wenn Werte vom Server mit föderierten Datenbanken an die Datenquelle übergeben werden, bevor die Unteranweisung ausgeführt wird, wird die Zahl der Werte durch folgende Angabe angezeigt:

  ```
  #Input Columns: n
  ```

- Wenn Werte von der Datenquelle an den Server mit föderierten Datenbanken übergeben werden, nachdem die Unteranweisung ausgeführt wurde, wird die Zahl der Werte durch folgende Angabe angezeigt:

  ```
  #Output Columns: n
  ```

Verschiedene EXPLAIN-Informationen:

Die Ausgabe des Befehls **db2expln** enthält zusätzliche nützliche Informationen, die sich einer einfachen Klassifizierung entziehen.

- Abschnitte für Anweisungen der Datendefinitionssprache (DDL, Data Definition Language) werden in der Ausgabe durch die folgende Angabe gekennzeichnet:

  ```
  DDL Statement
  ```

  Für DDL-Anweisungen wird keine weitere EXPLAIN-Ausgabe bereitgestellt.

- Abschnitte für SET-Anweisungen, die sich auf aktualisierbare Sonderregister wie CURRENT EXPLAIN SNAPSHOT beziehen, werden in der Ausgabe durch folgende Angabe angezeigt:

  ```
  SET Statement
  ```

  Für SET-Anweisungen wird keine weitere EXPLAIN-Ausgabe bereitgestellt.

- Wenn die SQL-Anweisung eine Klausel DISTINCT enthält, kann die folgende Angabe 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 distinkter Zeilenwerte müssen die Zeilen zunächst sortiert werden, um mehrfach auftretende gleiche Werte zu entfernen. Diese Angabe wird nicht angezeigt, wenn der Datenbankmanager gleiche Werte nicht explizit entfernen muss wie in den folgenden Fällen:

  – Es ist ein eindeutiger Index vorhanden und alle Spalten im Indexschlüssel sind Teil der DISTINCT-Operation.
  – Die mehrfach auftretenden gleichen 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 positionierte Operation 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 die SQL-Anweisung einen UNION-Verknüpfungsoperator enthält:

  ```
  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 vorhanden ist, die nur unter bestimmten Bedingungen verarbeitet wird:

  ```
  Conditional Evaluation
  |  Condition #n:
  |  #Predicates = n
  |  Action #n:
  ```

  Durch bedingte Auswertung werden Aktivitäten wie die 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 mit ALL, ANY oder EXISTS im Zugriffsplan verarbeitet wird:

  ```
  ANY/ALL Subquery
  EXISTS Subquery
  EXISTS SINGLE Subquery
  ```

- Vor bestimmten UPDATE- oder DELETE-Operationen muss die Position einer bestimmten Zeile in der Tabelle festgestellt werden. Dies wird durch folgende Angabe angezeigt:

  ```
  Establish Row Position
  ```

- Eine der folgenden Angaben wird für Löschoperationen an MDC-Tabellen angezeigt, für die eine Rollout-Optimierung in Betracht kommt:

  ```
  CELL DELETE with deferred cleanup
  CELL DELETE with immediate cleanup
  ```

- 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 Abschlussphasenangabe (Completion) in der Ausgabe angezeigt:

  ```
  Return Data Completion
  ```

- Die folgenden Angaben werden angezeigt, wenn eine gespeicherte Prozedur aufgerufen wird:

  ```
  Call Stored Procedure
  |    Name = schema.funktionsname
  |    Specific Name = spezifischer_name
  |    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 folgende Angabe wird angezeigt, wenn ein oder mehrere LOB-Querverweise freigegeben werden:

  ```
  Free LOB Locators
  ```

Optimieren von Abfragezugriffsplänen

Anweisungskonzentrator verringert Kompilierungsaufwand

Der Anweisungskonzentrator ändert dynamische SQL-Anweisungen auf dem Datenbankserver, sodass ähnliche, jedoch nicht identische, SQL-Anweisungen denselben Zugriffsplan gemeinsam nutzen können.

Bei der Onlinetransaktionsverarbeitung (OLTP, Online Transaction Processing) können einfache Anweisungen wiederholt, jedoch mit unterschiedlichen Literalwerten generiert werden. Bei solchen Auslastungen können die Kosten für die erneute Kompilierung der Anweisungen einen erheblichen Zusatzaufwand verursachen. Der Anweisungskonzentrator vermeidet diesen Aufwand, indem er eine Wiederverwendung kompilierter Anweisungen unabhängig von den Werten der Literale ermöglicht.

Der Anweisungskonzentrator ist standardmäßig inaktiviert. Er kann für alle dynamischen Anweisungen in einer Datenbank aktiviert werden, indem der Datenbankkonfigurationsparameter **stmt_conc** auf den Wert LITERALS gesetzt wird.

Der Anweisungskonzentrator verbessert die Leistung, indem er ankommende dynamische SQL-Anweisungen ändert. In einer Auslastung, die für die Nutzung des Anweisungskonzentrators geeignet ist, ist der Aufwand, der mit dem Ändern der ankommenden SQL-Anweisungen verbunden ist, im Vergleich zu den Einsparungen, die sich durch die Wiederverwendung von bereits im Paketcache befindlichen Anweisungen realisieren lassen, geringer.

Wenn eine dynamische Anweisung infolge der Anweisungskonzentration geändert wird, werden sowohl die ursprüngliche Anweisung als auch die geänderte Anweisung in der EXPLAIN-Ausgabe angezeigt. Die logischen Monitorelemente des Ereignismonitors und die Ausgabe der Tabellenfunktion MON_GET_ACTIVITY_DETAILS zeigen die ursprüngliche Anweisung an, wenn der Anweisungskonzentrator den ursprünglichen Anweisungstext geändert hat. Andere Überwachungsschnittstellen zeigen nur den geänderten Anweisungstext an.

Betrachten Sie das folgende Beispiel, in dem der Datenbankkonfigurationsparameter **stmt_conc** auf den Wert LITERALS gesetzt ist und die folgenden beiden Anweisungen ausgeführt werden:

```
select firstnme,lastname from employee where empno='000020'
select firstnme,lastname from employee where empno='000070'
```

Diese Anweisungen nutzen denselben Eintrag im Paketcache gemeinsam, der wiederum die folgende Anweisung verwendet:

```
select firstnme, lastname from employee where empno=:L0
```

Der Datenserver stellt einen Wert für `:L0` (entweder '000020' oder '000070') auf der Basis des Literals bereit, das in den ursprünglichen Anweisungen verwendet wurde.

Da die Anweisungskonzentration den Anweisungstext ändert, hat sie Auswirkungen auf die Zugriffsplanauswahl. Der Anweisungskonzentrator sollte verwendet werden, wenn ähnliche Anweisungen im Paketcache ähnliche Zugriffspläne haben. Wenn unterschiedliche Literalwerte in einer Anwendung zu sehr unterschiedlichen Zugriffsplänen führen, sollte der Anweisungskonzentrator für die betreffende Anweisung nicht aktiviert werden.

Der Anweisungskonzentrator kann dazu führen, dass die Längenattribute für die Zeichenfolgeliterale VARCHAR und VARGRAPHIC größer als die Länge des Zeichenfolgeliterals sind.

Wiederverwendung von Zugriffsplänen

Sie können anfordern, dass die für statische SQL-Anweisungen in einem Paket gewählten Zugriffspläne bei (erneuten) Bindeoperationen exakt oder zumindest weitestgehend den vorhandenen Zugriffsplänen entsprechen.

Die Wiederverwendung von Zugriffsplänen kann verhindern, dass wichtige Änderungen am Plan eintreten, ohne dass Sie ihnen explizit zugestimmt haben. Dies führt möglicherweise dazu, dass Ihre Abfragen nicht von potenziellen Planoptimierungen profitieren. Die Steuerungsmöglichkeiten, die die Wiederverwendung von Zugriffsplänen Ihnen bietet, lässt jedoch ein Testen und Implementieren dieser Verbesserungen zu, wenn Sie dazu bereit sind. Bis zu diesem Zeitpunkt können Sie auf eine stabile und berechenbare Leistung vertrauen, wenn Sie die vorhandenen Zugriffspläne weiter verwenden.

Aktivieren Sie die Wiederverwendung von Zugriffsplänen mit der Anweisung ALTER PACKAGE oder der Option APREUSE des Befehls **BIND**, **REBIND** oder **PRECOMPILE**. Pakete, für die die Wiederverwendung von Zugriffsplänen aktiviert ist, sind in der Spalte APREUSE der Katalogsicht SYSCAT.PACKAGES mit einem Y (für 'Ja') markiert.

Die Prozedur ALTER_ROUTINE_PACKAGE stellt eine einfache Möglichkeit zum Aktivieren der Wiederverwendung von Zugriffsplänen für kompilierte SQL-Objekte (z. B. SQL-Prozeduren) dar. Zugriffspläne können jedoch nicht während der kompilierten Objektreaktivierung wiederverwendet werden, da das Objekt vor dem erneuten Binden gelöscht wird. In diesem Fall wird APREUSE erst beim nächsten (erneuten) Binden des Pakets wirksam.

Die Wiederverwendung von Zugriffsplänen ist am effektivsten, wenn Änderungen an der Schema- und Kompilierungsumgebung gering gehalten werden. Wenn einschneidendere Änderungen vorgenommen werden, ist es unter Umständen nicht möglich, den vorherigen Zugriffsplan erneut zu erstellen. Beispiele für derartige Änderungen sind z. B. das Löschen eines Index, der in einem Zugriffsplan verwendet wird, oder das erneute Kompilieren einer SQL-Anweisung auf einer anderen Optimierungsebene. Einschneidende Änderungen an der Anweisungsanalyse des Abfragecompilers können darüber hinaus dazu führen, dass der vorherige Zugriffsplan nicht mehr verwertbar ist.

Sie können die Wiederverwendung von Zugriffsplänen mit Optimierungsrichtlinien kombinieren. Eine Richtlinie auf Anweisungsebene hat bei der statischen SQL-Anweisung, auf die sie sich bezieht, Vorrang vor der Wiederverwendung von Zugriffsplänen. Zugriffspläne für statische Anweisungen, für die es keine Richtlinien auf Anweisungsebene gibt, können wiederverwendet werden, wenn sie angegebenen allgemeinen Optimierungsrichtlinien nicht widersprechen. Mit einem Anweisungsprofil, das eine leere Richtlinie beinhaltet, kann die Wiederverwendung von

Zugriffspläne für eine bestimmte Anweisung inaktiviert werden, während die Wiederverwendung von Plänen gleichzeitig bei den übrigen statischen Anweisungen des Pakets aktiviert bleibt.

Anmerkung: Zugriffspläne aus Paketen, die mit Releases vor Version 9.7 erstellt wurden, können nicht wiederverwendet werden.

Kann ein Zugriffsplan nicht wiederverwendet werden, wird die Kompilierung fortgesetzt. Es wird jedoch eine Warnung (SQL20516W) mit einem Rückkehrcode ausgegeben, die darauf hinweist, dass der Zugriffsplan nicht wiederverwendet werden konnte. Nähere Informationen sind teilweise in den Diagnosenachrichten enthalten, die über die EXPLAIN-Funktion verfügbar sind.

Optimierungsklassen

Wenn Sie eine SQL- oder XQuery-Anweisung kompilieren, können Sie eine Optimierungsklasse angeben, die bestimmt, wie das Optimierungsprogramm den effizientesten Zugriffsplan für diese Anweisung auswählt.

Die Optimierungsklassen unterscheiden sich nach Anzahl und Typ der Optimierungsstrategien, die bei der Kompilierung einer Abfrage in Betracht gezogen werden. Obwohl Sie einerseits 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-Anweisung können Sie eine der folgenden Optimierungsklassen angeben:

0 Diese Klasse weist das Optimierungsprogramm an, nur eine Minimaloptimierung bei der Generierung eines Zugriffsplans durchzuführen, und ist durch folgende Merkmale gekennzeichnet:

- Statistiken zu Wertehäufigkeiten werden vom Optimierungsprogramm nicht berücksichtigt.
- Nur Grundregeln für das Umschreiben der Abfragen werden angewendet.
- Schnelle Joinaufzählung (Greedy Join Enumeration) wird verwendet.
- 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 angemessen indexierte Tabellen zugreifen.

1 Diese Optimierungsklasse ist durch folgende Merkmale gekennzeichnet:

- Statistiken zu Wertehäufigkeiten werden vom Optimierungsprogramm nicht berücksichtigt.
- Nur eine Teilmenge der Regeln für das Umschreiben von Abfragen wird angewendet.
- Schnelle Joinaufzählung (Greedy Join Enumeration) wird verwendet.

- Ein Vorablesezugriff wird in den generierten Zugriffsmethoden nicht verwendet.

Die Optimierungsklasse 1 ist der Klasse 0 ähnlich, abgesehen davon, dass zusätzlich Mischjoins und Tabellensuchen 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 zu Wertehäufigkeiten und 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.
- Schnelle Joinaufzählung (Greedy Join Enumeration) wird verwendet.
- Viele verschiedene Zugriffsmethoden werden berücksichtigt, einschließlich des Vorablesezugriffs 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 (Dynamic Programming Join Enumeration). 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 ist die Wahrscheinlichkeit gering, dass eine bestimmte Abfrage exakt wiederholt wird, sodass ein Zugriffsplan wahrscheinlich nicht bis zur nächsten Ausführung der Abfrage im Cache verbleibt.

3 Diese Klasse stellt einen gemäßigten Optimierungsgrad dar und entspricht am ehesten den Abfrageoptimierungsmerkmalen von DB2 für z/OS. Diese Optimierungsklasse ist durch folgende Merkmale gekennzeichnet:

- Statistiken zu Wertehäufigkeiten werden verwendet, wenn sie verfügbar sind.
- Die meisten Regeln zum Umschreiben von Abfragen, einschließlich der Umsetzungen von Unterabfragen in Joins, werden angewendet.
- Die dynamisch programmierte Joinaufzählung (Dynamic Programming Join Enumeration) wird wie folgt verwendet:
  - Eingeschränkte Verwendung zusammengesetzter innerer Tabellen
  - Eingeschränkte Verwendung kartesischer Produkte für Sternschemata, für die Suchtabellen erforderlich sind
- Viele verschiedene Zugriffsmethoden werden berücksichtigt, einschließlich des Vorablesezugriffs über Listen, logischer Verknüpfungen von Indizes über AND (Index ANDing) und Sternjoins.

Diese Klasse eignet sich für eine große Bandbreite von Anwendungen und verbessert Zugriffspläne für Abfragen mit vier und mehr Joins.

5 Diese Klasse weist das Optimierungsprogramm an, einen bedeutenden

Grad an Optimierung bei der Generierung eines Zugriffsplans durchzuführen, und ist durch folgende Merkmale gekennzeichnet:

- Alle verfügbaren Statistiken, einschließlich der Statistiken zu Wertehäufigkeiten und 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.
- Die dynamisch programmierte Joinaufzählung (Dynamic Programming Join Enumeration) wird wie folgt verwendet:
  - Eingeschränkte Verwendung zusammengesetzter innerer Tabellen
  - Eingeschränkte Verwendung kartesischer Produkte für Sternschemata, für die Suchtabellen erforderlich sind
- Viele verschiedene Zugriffsmethoden werden berücksichtigt, einschließlich des Vorablesezugriffs über Listen, logischer Verknüpfungen von Indizes über AND und der Weiterleitung an MQTs.

Die Abfrageoptimierungsklasse 5 (Standardeinstellung) ist hervorragend für eine gemischte Umgebung geeignet, in der sowohl eine Transaktionsaktionsverarbeitung als auch eine Ausführung komplexer Abfragen stattfindet. Diese Optimierungsklasse wurde zur Verwendung der wertvollsten Abfragetransformationen und anderer Optimierungstechniken für Abfragen in einer effizienten Weise entwickelt.

Wenn das Optimierungsprogramm feststellt, dass zusätzliche Ressourcen und Verarbeitungszeit für komplexe dynamische SQL- oder XQuery-Anweisungen nicht gerechtfertigt sind, wird die Optimierung reduziert. Der Umfang der Reduzierung hängt von der Maschinengröße und der Anzahl der Vergleichselemente ab. Wenn das Optimierungsprogramm den Grad der Abfrageoptimierung reduziert, wendet es weiterhin alle Regeln für das Umschreiben von Abfragen an, die normalerweise angewendet würden. Es verwendet jedoch die schnelle Joinaufzählung und verringert die Anzahl der Zugriffsplankombinationen, die in Betracht gezogen werden.

7 Diese Klasse weist das Optimierungsprogramm an, einen bedeutenden Grad an Optimierung bei der Generierung eines Zugriffsplans durchzuführen. Sie ist der Optimierungsklasse 5 ähnlich, reduziert jedoch den Umfang der Abfrageoptimierung für komplexe dynamische SQL- oder XQuery-Anweisungen nicht.

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 der Joinaufzählung, einschließlich kartesischer Produkte und uneingeschränkter zusammengesetzter innerer Tabellen
- Alle Zugriffsmethoden

Diese Klasse vergrößert die Anzahl der möglichen Zugriffspläne, die vom Optimierungsprogramm ausgewertet werden. Diese Klasse kann verwendet werden, um festzustellen, ob eine umfassendere Optimierung zur Generierung eines besseren Zugriffsplans für sehr komplexe oder sehr zeitintensive Abfragen auf große Tabellen führen würde. Überprüfen Sie anhand von EXPLAIN-Daten und Leistungswerten, ob ein besserer Plan gefunden wurde.

Auswählen von Optimierungsklassen:

Die Definition der Optimierungsklasse kann einige der Vorteile bringen, die durch die explizite Angabe von Optimierungstechniken erzielt werden.

Dies gilt insbesondere für folgende Zwecke:

- Verwalten sehr kleiner Datenbanken oder sehr einfacher dynamischer Abfragen
- Berücksichtigen von Speicherbeschränkungen auf dem Datenbankserver beim Kompilieren
- Verringern der Kompilierungsdauer für Abfragen, zum Beispiel bei der Anweisungsvorbereitung

Die meisten Anweisungen lassen sich in geeigneter Weise bei einem angemessenen Einsatz von Ressourcen unter Verwendung der Standardoptimierungsklasse 5 optimieren. Die Dauer der Abfragekompilierung und die Ressourcennutzung werden in erster Linie durch die Komplexität einer Abfrage, insbesondere durch die Anzahl der Joins und Unterabfragen, beeinflusst. Allerdings werden die Kompilierzeit und die Ressourcennutzung auch vom Grad der durchgeführten Optimierung beeinflusst.

Die Abfrageoptimierungsklassen 1, 2, 3, 5 und 7 sind alle für eine allgemeine Verwendung geeignet. Ziehen Sie die Klasse 0 nur in Betracht, wenn Sie die Kompilierzeit von Abfragen weiter verringern müssen die SQL- und XQuery-Anweisungen sehr einfach sind.

Tipp: Zur Analyse einer Abfrage mit langer Laufzeit führen Sie die Abfrage mit **db2batch** aus, um zu ermitteln, welcher Zeitaufwand für die Kompilierung und die Ausführung der Abfrage anfällt. Wenn die Kompilierzeit übermäßig lang ist, setzen Sie die Optimierungsklasse herab. Wenn die Ausführungszeit ein Problem darstellt, ziehen Sie eine höhere Optimierungsklasse in Betracht.

Beachten Sie bei der Auswahl einer Optimierungsklasse die folgenden allgemeinen Richtlinien:

- Beginnen Sie mit der Standardoptimierungsklasse 5 für Abfragen.
- Wenn Sie eine andere als die Standardklasse auswählen, 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:
  - Sie greifen nur auf eine oder einige wenige Tabellen zu.
  - Sie rufen nur eine oder einige wenige Zeilen ab.
  - Sie verwenden vollständig qualifizierte und eindeutige Indizes.
  - Sie sind Teil einer Onlinetransaktionsverarbeitung (OLTP).
- Verwenden Sie eine höhere Optimierungsklasse (3, 5 oder 7) für Abfragen mit längeren Laufzeiten von über 30 Sekunden.
- Ab Klasse 3 wird der Algorithmus der dynamisch programmierten Joinaufzählung (Dynamic Programming Join Enumeration) verwendet, bei dem zahlreiche

zusätzliche Alternativpläne geprüft werden und daher wesentlich mehr Kompilierzeit als bei den Klassen 0, 1 oder 2 anfallen kann, insbesondere wenn die Anzahl der Tabellen steigt.
- Verwenden Sie die Optimierungsklasse 9 nur, wenn Sie außergewöhnliche, 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 Sichten
- Eine große Anzahl von Vergleichselementen
- Zahlreiche Unterabfragen
- Zahlreiche Joins
- Zahlreiche Gruppenoperatoren wie UNION und INTERSECT
- Zahlreiche die Vergleichselemente erfüllende Zeilen
- Operationen GROUP BY und HAVING
- Verschachtelte Tabellenausdrücke

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 erfordern eine zusätzliche Optimierung zur Auswahl eines guten Zugriffsplans. Durch die Verwendung der Abfrageoptimierungsklasse 2 oder einer höheren Abfrageoptimierungsklasse können solche Abfragen verbessert werden.

Verwenden Sie für SAP-Anwendungen immer die Optimierungsklasse 5. Diese Optimierungsklasse aktiviert viele DB2-Funktionsmerkmale, die für SAP optimiert sind, wie zum Beispiel die Einstellung der Registrierdatenbankvariablen **DB2_REDUCED_OPTIMIZATION**.

In einer föderierten Datenbank gilt die Optimierungsklasse nicht für das ferne Optimierungsprogramm.

Einstellen der Optimierungsklasse:

Berücksichtigen Sie bei der Angabe einer Optimierungsklasse, ob eine Abfrage statische oder dynamische SQL- und XQuery-Anweisungen verwendet und ob die gleiche dynamische Abfrage wiederholt ausgeführt wird.

Informationen zu diesem Vorgang

Für statische SQL- und XQuery-Anweisungen werden die Abfragekompilierzeit und die Ressourcen nur einmal aufgewendet und der resultierende Plan kann mehrfach verwendet werden. Im Allgemeinen gilt, dass für SQL- und XQuery-Anweisungen stets die Standardoptimierungsklasse (5) 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 diesel-

be 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 sich nicht sicher sind, ob für eine Abfrage eine weitere Optimierung von Vorteil wäre oder wenn Sie Bedenken hinsichtlich der Kompilierzeit oder des Ressourcenbedarfs haben, ziehen Sie Vergleichstests (Benchmarktests) in Betracht.

Führen Sie die folgenden Schritte aus, um eine Abfrageoptimierungsklasse anzugeben:

Vorgehensweise

1. Analysieren Sie die Leistungsfaktoren.
   - Für eine dynamische Abfrageanweisung sollten Tests die durchschnittliche Laufzeit für die Anweisung vergleichen. Schätzen Sie die durchschnittliche Laufzeit mithilfe der folgenden Formel ab:

     ```
     Kompilierzeit + Summe der Ausführungszeiten aller Iterationen
     -------------------------------------------------------------
                          Anzahl der Iterationen
     ```

     Die Anzahl der Iterationen ist die von Ihnen erwartete Häufigkeit, mit der die Anweisung 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 Anweisung im Cache zwischengespeichert wurde, verwenden nachfolgende PREPARE-Anweisungen die Anweisung wieder, die sich im Cache befindet.
   - 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 der Gesamtzeitaufwand aus Kompilier- und Ausführungszeit für eine statische Anweisung in einem sinnvollen Kontext nur wenig aussagekräftig. Beim Vergleich der Gesamtzeiten wird die Tatsache außer Acht gelassen, dass eine statische Anweisung nach dem Binden viele Male ausgeführt werden kann und dass eine solche Anweisung in der Regel nicht während der Ausführungszeit gebunden wird.
2. Geben Sie die Optimierungsklasse an.
   - Dynamische SQL- und XQuery-Anweisungen verwenden die Optimierungsklasse, die im Sonderregister CURRENT QUERY OPTIMIZATION angegeben ist. 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, geben Sie eine SET-Anweisung im Anwendungsprogramm an.

     Wenn das Sonderregister CURRENT QUERY OPTIMIZATION nicht definiert ist, werden dynamische Anweisungen mit der Standardoptimierungsklasse für Abfragen gebunden. Der Standardwert für dynamische und statische Abfragen wird durch den Wert des Datenbankkonfigurationsparameters **dft_queryopt** festgelegt, der den Standardwert 5 hat. Die Standardwerte für die Bindeoption und das Sonderregister werden ebenfalls aus dem Datenbankkonfigurationsparameter **dft_queryopt** gelesen.

- Statische SQL- und XQuery-Anweisungen verwenden die Optimierungsklasse, die in den Befehlen **PREP** und **BIND** angegeben wird. In der Spalte QUERYOPT der Katalogsicht SYSCAT.PACKAGES wird die zum Binden eines Pakets verwendete Optimierungsklasse aufgezeichnet. Wenn das Paket erneut implizit oder mit dem Befehl **REBIND PACKAGE** gebunden wird, wird dieselbe Optimierungsklasse für statische Anweisungen verwendet. Verwenden Sie zum Ändern der Optimierungsklasse für solche statischen SQL- und XQuery-Anweisungen den Befehl **BIND**. Wenn Sie die Optimierungsklasse nicht angeben, verwendet der Datenserver die Standardoptimierungsklasse, die durch den Datenbankkonfigurationsparameter **dft_queryopt** definiert wird.

Verwenden von Optimierungsprofilen, wenn andere Optimierungsoptionen keine akzeptablen Ergebnisse liefern

Wenn Sie empfohlene Methoden befolgt haben und dennoch glauben, dass weiterhin eine nicht optimale Leistung erzielt wird, können Sie explizite Optimierungsrichtlinien für das DB2-Optimierungsprogramm angeben.

Diese Optimierungsrichtlinien sind in einem XML-Dokument enthalten, das als Optimierungsprofil bezeichnet wird. Das Profil definiert SQL-Anweisungen und die zugehörigen Optimierungsrichtlinien.

Wenn Sie Optimierungsprofile extensiv nutzen, erfordern Sie einen großen Verwaltungsaufwand. Wichtiger jedoch ist, dass Sie Optimierungsprofile nur zur Leistungsverbesserung für vorhandene SQL-Anweisungen verwenden können. Eine konsistente Befolgung der empfohlenen Methoden kann Ihnen helfen, eine stabile Abfrageleistung für alle Abfragen, einschließlich künftigen, sicherzustellen.

Optimierungsprofile und Optimierungsrichtlinien

Ein Optimierungsprofil ist ein XML-Dokument, das Optimierungsrichtlinien für eine oder auch mehrere SQL-Anweisungen enthalten kann. Die Beziehung zwischen einer SQL-Anweisung und den ihr zugeordneten Optimierungsrichtlinien wird durch den SQL-Text und andere Informationen hergestellt, die zur eindeutigen Identifizierung einer SQL-Anweisung erforderlich sind.

Das DB2-Optimierungsprogramm ist eines der höchstentwickelten aufwandsbasierten Optimierungsprogramme am Markt. In seltenen Fällen ist es jedoch möglich, dass das Optimierungsprogramm einen nicht optimalen Ausführungsplan auswählt. Als Datenbankadministrator (DBA), der mit der Datenbank vertraut ist, können Sie Dienstprogramme wie **db2advis**, RUNSTATS und **db2expln** sowie die Einstellung der Optimierungsklasse dazu verwenden, das Optimierungsprogramm im Hinblick auf bessere Datenbankleistung zu optimieren. Wenn sich nach Ausschöpfung aller Optimierungsoptionen nicht die erwarteten Ergebnisse einstellen, können Sie dem DB2-Optimierungsprogramm explizite Optimierungsrichtlinien zur Verfügung stellen.

Angenommen z. B:, dass das Optimierungsprogramm den Index I_SUPPKEY auch nach dem Aktualisieren der Datenbankstatistik und Ausführen aller sonstigen Optimierungsmaßnahmen nicht für den Zugriff auf die Tabelle SUPPLIERS in der folgenden Unterabfrage ausgewählt hat:

```
SELECT S.S_NAME, S.S_ADDRESS, S.S_PHONE, S.S_COMMENT
FROM PARTS P, SUPPLIERS S, PARTSUPP PS
WHERE P_PARTKEY = PS.PS_PARTKEY AND S.S_SUPPKEY =
PS.PS_SUPPKEY
  AND P.P_SIZE = 39
  AND P.P_TYPE = 'BRASS'
  AND S.S_NATION = 'MOROCCO'
  AND S.S_NATION IN ('MOROCCO', 'SPAIN')
        AND PS.PS_SUPPLYCOST = (SELECT MIN(PS1.PS_SUPPLYCOST)
                          FROM PARTSUPP PS1, SUPPLIERS S1
```

```
                                WHERE P.P_PARTKEY = PS1.PS_PARTKEY
                                  AND S1.S_SUPPKEY = PS1.PS_SUPPKEY
                                  AND S1.S_NATION = S.S_NATION))
```

In diesem Fall kann das Optimierungsprogramm mit einer expliziten Optimierungsrichtlinie beeinflusst werden. Beispiel:

```
<OPTGUIDELINES><IXSCAN TABLE="S" INDEX="I_SUPPKEY"/></OPTGUIDELINES>
```

Optimierungsrichtlinien werden mithilfe von einfachen XML-Spezifikationen angegeben. Jedes Element wird vom DB2-Optimierungsprogramm als Optimierungsrichtlinie interpretiert. Das folgende Beispiel enthält ein einziges Richtlinienelement für das Optimierungsprogramm. Über das Element IXSCAN wird angefordert, dass das Optimierungsprogramm über einen Index auf eine Tabelle zugreift. Das Attribut TABLE des Elements IXSCAN gibt den Zieltabellenverweis (mit dem exponierten Namen des Tabellenverweises) und das Attribut INDEX den Index an.

Das folgende Beispiel, das an der vorangehenden Abfrage ausgerichtet ist, zeigt, wie sich eine Optimierungsrichtlinie mithilfe eines Optimierungsprofils an das DB2-Optimierungsprogramm übergeben lässt.

```
<?xml version="1.0" encoding="UTF-8"?>

<OPTPROFILE VERSION="9.1.0.0">
<STMTPROFILE ID="Guidelines for TPCD Q9">
  <STMTKEY SCHEMA="TPCD">
    SELECT S.S_NAME, S.S_ADDRESS, S.S_PHONE, S.S_COMMENT
    FROM PARTS P, SUPPLIERS S, PARTSUPP PS
    WHERE P_PARTKEY = PS.PS_PARTKEY AND S.S_SUPPKEY = PS.PS_SUPPKEY
      AND P.P_SIZE = 39
      AND P.P_TYPE = 'BRASS'
      AND S.S_NATION = 'MOROCCO'
      AND S.S_NATION IN ('MOROCCO', 'SPAIN')
            AND PS.PS_SUPPLYCOST = (SELECT MIN(PS1.PS_SUPPLYCOST)
                                    FROM PARTSUPP PS1, SUPPLIERS S1
                                    WHERE P.P_PARTKEY = PS1.PS_PARTKEY
                                      AND S1.S_SUPPKEY = PS1.PS_SUPPKEY
                                                          AND S1.S_NATION = S.S_NATION))
  </STMTKEY>
  <OPTGUIDELINES><IXSCAN TABLE="S" INDEX="I_SUPPKEY"/></OPTGUIDELINES>
</STMTPROFILE>
</OPTPROFILE>
```

Jedes Element STMTPROFILE (Anweisungsprofil) stellt einen Satz von Optimierungsrichtlinien für eine Anwendungsanweisung bereit. Die betroffene Anweisung wird durch ein Unterelement STMTKEY (Anweisungsschlüssel) angegeben. Das Optimierungsprofil erhält einen durch ein Schema qualifizierten Namen und wird in die Datenbank eingefügt. Durch die Angabe dieses Namens im Befehl **BIND** oder **PRECOMPILE** wird das Optimierungsprofil für die Anweisung in Kraft gesetzt.

Durch Optimierungsprofile können dem Optimierungsprogramm Optimierungsrichtlinien zur Verfügung gestellt werden, ohne Änderungen an Anwendungen oder an der Datenbankkonfiguration vornehmen zu müssen. Sie erstellen lediglich das einfache XML-Dokument, fügen es in die Datenbank ein und geben den Namen des Optimierungsprofils im Befehl **BIND** oder **PRECOMPILE** an. Das Optimierungsprogramm ordnet die Optimierungsrichtlinien automatisch der richtigen Anweisung zu.

Optimierungsrichtlinien müssen nicht umfassend sein, sollten jedoch speziell auf einen gewünschten Ausführungsplan ausgerichtet werden. Das DB2-Optimierungsprogramm zieht auch weiterhin andere mögliche Zugriffspläne unter Verwendung der vorhandenen Methoden zur Aufwandsberechnung in Betracht. Optimierungsrichtlinien für bestimmte Tabellenverweise können allgemeine Optimierungseinstellungen nicht überschreiben. Zum Beispiel ist eine Optimierungsrichtlinie, die einen Mischjoin zwischen Tabelle A und Tabelle B angibt, bei Optimierungsklasse 0 nicht gültig.

Das Optimierungsprogramm ignoriert ungültige oder nicht anwendbare Optimierungsrichtlinien. Wenn Optimierungsrichtlinien ignoriert werden, wird ein Ausführungsplan generiert und eine Warnung SQL0437W mit dem Ursachencode 13 zurückgegeben. Anschließend können Sie mithilfe der Anweisung EXPLAIN detaillierte Diagnoseinformationen zur Verarbeitung von Optimierungsrichtlinien abrufen.

Optimierungsprofile:

Aufbau eines Optimierungsprofils:

Ein Optimierungsprofil kann globale Richtlinien enthalten, die für alle DML-Anweisungen (DML, Data Manipulation Language, Datenbearbeitungssprache) gelten, die ausgeführt werden, während das Profil wirksam ist. Und es kann spezielle Richtlinien enthalten, die sich auf einzelne DML-Anweisungen in einem Paket beziehen.

Beispiele:

- Sie könnten eine globale Optimierungsrichtlinie schreiben, die anfordert, dass das Optimierungsprogramm auf die MQTs (Materialized Query Tables) 'Test-.SumSales' und 'Test.AvgSales' zurückgreift, wenn eine Anweisung verarbeitet wird, während das aktuelle Optimierungsprofil aktiv ist.
- Sie könnten eine Optimierungsrichtlinie auf Anweisungsebene schreiben, die anfordert, dass der Index I_SUPPKEY für den Zugriff auf die Tabelle SUPPLIERS verwendet wird, wenn das Optimierungsprogramm auf die angegebene Anweisung trifft.

Ein Optimierungsprofil enthält zwei Hauptabschnitte, in denen Sie diese beiden Typen von Richtlinien angeben können: Ein Abschnitt für globale Optimierungsrichtlinien kann ein Element OPTGUIDELINES enthalten und ein Abschnitt für Anweisungsprofile kann eine beliebige Anzahl von Elementen STMTPROFILE enthalten. Ein Optimierungsprofil muss außerdem ein Element OPTPROFILE enthalten, das Metadaten und Verarbeitungsanweisungen enthält.

Der folgende Code ist ein Beispiel für ein gültiges Optimierungsprofil für DB2 Version 9.1, das einen Abschnitt für globale Optimierungsrichtlinien und einen Abschnitt für Anweisungsprofile mit einem Element STMTPROFILE enthält.

```
<?xml version="1.0" encoding="UTF-8"?>
<OPTPROFILE VERSION="9.1.0.0">

  <!--
       Abschnitt für globale Optimierungsrichtlinien.
       Optional, jedoch maximal ein Abschnitt dieser Art zulässig.
  -->
  <OPTGUIDELINES>
    <MQT NAME="Test.AvgSales"/>
    <MQT NAME="Test.SumSales"/>
  </OPTGUIDELINES>

  <!--
       Abschnitt für Anweisungsprofile.
       Null oder mehr Abschnitte möglich.
  -->
  <STMTPROFILE ID="Guidelines for TPCD Q9">
    <STMTKEY SCHEMA="TPCD">
      <![CDATA[SELECT S.S_NAME, S.S_ADDRESS, S.S_PHONE,
S.S_COMMENT FROM PARTS P, SUPPLIERS S, PARTSUPP PS
WHERE P_PARTKEY = PS.PS_PARTKEY AND S.S_SUPPKEY = PS.PS_SUPPKEY
AND P.P_SIZE = 39 AND P.P_TYPE = 'BRASS'
```

```
AND S.S_NATION = 'MOROCCO' AND S.S_NATION IN ('MOROCCO', 'SPAIN')
AND PS.PS_SUPPLYCOST = (SELECT MIN(PS1.PS_SUPPLYCOST)
FROM PARTSUPP PS1, SUPPLIERS S1
WHERE P.P_PARTKEY  = PS1.PS_PARTKEY AND S1.S_SUPPKEY = PS1.PS_SUPPKEY
AND S1.S_NATION  = S.S_NATION)]]>
    </STMTKEY>
    <OPTGUIDELINES>
      <IXSCAN TABID="Q1" INDEX="I_SUPPKEY"/>
    </OPTGUIDELINES>
  </STMTPROFILE>

</OPTPROFILE>
```

Das Element OPTPROFILE

Ein Optimierungsprofil beginnt mit dem Element OPTPROFILE. Im obigen Beispiel besteht dieses Element aus einem Attribut VERSION, das angibt, dass die Optimierungsprofilversion 9.1 ist.

Abschnitt für globale Optimierungsrichtlinien

Globale Optimierungsrichtlinien gelten für alle Anweisungen, für die das Optimierungsprofil wirksam ist. Der Abschnitt für globale Optimierungsrichtlinien wird durch das Element OPTGUIDELINES dargestellt. Im obigen Beispiel enthält dieser Abschnitt nur eine globale Optimierungsrichtlinie, die angibt, dass die MQTs (Materialized Query Tables) 'Test.AvgSales' und 'Test.SumSales' in Betracht gezogen werden sollen, wenn Anweisungen verarbeitet werden, für die das Optimierungsprofil wirksam ist.

Abschnitt für Anweisungsprofile

Ein Anweisungsprofil definiert die Optimierungsrichtlinien, die für eine bestimmte Anweisung gelten. In einem Optimierungsprofil können null oder mehr Anweisungsprofile enthalten sein. Der Abschnitt für Anweisungsprofile wird durch das Element STMTPROFILE dargestellt. Im obigen Beispiel enthält dieser Abschnitt Richtlinien für eine bestimmte Anweisung, für die das Optimierungsprofil wirksam ist.

Jedes Anweisungsprofil enthält einen Anweisungsschlüssel und Optimierungsrichtlinien auf Anweisungsebene, die durch die Elemente STMTKEY bzw. OPTGUIDELINES dargestellt werden.

- Der Anweisungsschlüssel gibt die Anweisung an, für die die Optimierungsrichtlinien auf Anweisungsebene gelten. In diesem Beispiel enthält das Element STMTKEY den ursprünglichen Anweisungstext und weitere Informationen, die zur eindeutigen Angabe der gewünschten Anweisung erforderlich sind. Anhand des Anweisungsschlüssels gleicht das Optimierungsprogramm ein Anweisungsprofil mit der entsprechenden Anweisung ab. Diese Beziehung bietet Ihnen die Möglichkeit, Optimierungsrichtlinien für eine Anweisung anzugeben, ohne die Anwendung modifizieren zu müssen.
- Der Abschnitt für Optimierungsrichtlinien auf Anweisungsebene des Anweisungsprofils wird durch das Element OPTGUIDELINES dargestellt. Dieser Abschnitt besteht aus einer oder mehreren Zugriffs- oder Joinanforderungen, durch die die Methoden für den Zugriff auf Tabellen bzw. für den Join von Tabellen in der Anweisung spezifiziert werden. Nach einem erfolgreichen Abgleich mit einem Anweisungsschlüssel in einem Anweisungsprofil greift das Optimierungsprogramm auf die zugeordneten Optimierungsrichtlinien auf der Anweisungsebene zurück, wenn es die Anweisung optimiert. Das Beispiel enthält eine

Zugriffsanforderung, die angibt, dass für die in der verschachtelten Subselect-Anweisung angegebene Tabelle SUPPLIERS ein Index mit dem Namen I_SUPP-KEY verwendet werden soll.

Erstellen eines Optimierungsprofils:

Ein Optimierungsprofil ist ein XML-Dokument, das Optimierungsrichtlinien für eine oder mehrere DML-Anweisungen (DML = Data Manipulation Language, Datenbearbeitungssprache) enthält.

Informationen zu diesem Vorgang

Da ein Optimierungsprofil zahlreiche Kombinationen von Richtlinien enthalten kann, werden im Folgenden nur die Schritte beschrieben, die bei jeder Erstellung eines Optimierungsprofils gleichermaßen auszuführen sind.

Führen Sie folgende Schritte aus, um ein Optimierungsprofil zu erstellen:

Vorgehensweise

1. Starten Sie einen XML-Editor. Verwenden Sie, falls möglich, einen Editor mit Schemaprüffunktionalität. Das Optimierungsprogramm führt keinerlei XML-Gültigkeitsprüfung durch. Ein Optimierungsprofil muss gemäß dem aktuellen Optimierungsprofilschema gültig sein.
2. Erstellen Sie ein neues XML-Dokument mit einem für Sie sinnvollen Namen. Sinnvoll wäre beispielsweise ein Name, der den Geltungsbereich der Anweisungen beschreibt, für die die Richtlinien gelten sollen. Beispiel: `inventory_db.xml`.
3. Fügen Sie dem Dokument die XML-Deklaration hinzu. Wenn Sie kein Codierformat ('encoding') angeben, wird UTF-8 angenommen. Speichern Sie das Dokument mit UTF-16-Codierung, sofern möglich. Der Datenserver kann diese Codierung effizienter verarbeiten.

   ```
   <?xml version="1.0" encoding="UTF-16"?>
   ```

4. Fügen Sie dem Dokument einen Abschnitt für Optimierungsprofile hinzu.

   ```
   <OPTPROFILE VERSION="9.1.0.0">
      </OPTPROFILE>
   ```

5. Erstellen Sie im Element OPTPROFILE je nach Bedarf globale Optimierungsrichtlinien oder Optimierungsrichtlinie auf Anweisungsebene und speichern Sie die Datei.

Konfigurieren des Datenservers für die Verwendung eines Optimierungsprofils:

Nach der Erstellung eines Optimierungsprofils und der Gültigkeitsprüfung seines Inhalts am aktuellen Optimierungsprofilschema (COPS) muss dem Inhalt ein eindeutiger, mit einem Schemanamen qualifizierter Name zugeordnet und der Inhalt selbst in der Tabelle SYSTOOLS.OPT_PROFILE gespeichert werden.

Vorgehensweise

Gehen Sie wie folgt vor, um den Datenserver für die Verwendung eines Optimierungsprofils zu konfigurieren:

1. Erstellen Sie die Optimierungsprofiltabelle.

   Weitere Informationen finden Sie unter „Tabelle SYSTOOLS.OPT_PROFILE" auf Seite 408. Jede Zeile der Tabelle kann genau ein Optimierungsprofil enthalten:

die Spalten SCHEMA und NAME geben den eindeutigen Namen des Optimierungsprofils an, und die Spalte PROFILE enthält den Text des Optimierungsprofils.

2. Optional: Sie können jede Berechtigung bzw. jedes Zugriffsrecht für die Tabelle erteilen, die bzw. das Ihre Anforderungen an die Datenbanksicherheit erfüllt. Dies hat keine Auswirkungen auf die Fähigkeit des Optimierungsprogramms, die Tabelle zu lesen.
3. Fügen Sie die gewünschten Optimierungsprofile in die Tabelle ein.

Feld STMTKEY in Optimierungsprofilen:

Innerhalb eines Elements STMTPROFILE wird die betroffene Anweisung durch ein Unterelement STMTKEY angegeben. Die im Feld STMTKEY definierte Anweisung muss genau mit der Anweisung übereinstimmen, die von der Anwendung ausgeführt wird, sodass DB2 die betroffene Anweisung eindeutig identifizieren kann. Leerzeichen innerhalb der Anweisungen sind zulässig.

Wenn DB2 einen Anweisungsschlüssel findet, der mit dem aktuellen Kompilierungsschlüssel übereinstimmt, wird die Suche gestoppt. Daher wird, wenn mehrere Anweisungsprofile in einem Optimierungsprofil vorhanden sind, deren Anwendungsschlüssel mit dem aktuellen Kompilierungsschlüssel übereinstimmt, nur das erste dieser Anweisungsprofile verwendet (entsprechend der Dokumentreihenfolge). In diesem Fall wird außerdem keine Fehlernachricht oder Warnung ausgegeben.

Der Anweisungsschlüssel entspricht der Anweisung ⌂`select * from orders where foo(orderkey)>20`", sofern der Kompilierungsschlüssel das Standardschema ⌂COLLEGE⌂ und den Funktionspfad `SYSIBM,SYSFUN,SYSPROC,DAVE` enthält.

```
<STMTKEY SCHEMA='COLLEGE' FUNCPATH='SYSIBM,SYSFUN,SYSPROC,DAVE'>
<![CDATA[select * from orders where foo(orderkey)>20[[>
</stmtkey>
```

Angeben des vom Optimierungsprogramm zu verwendenden Optimierungsprofils:

Mit der Bindeoption OPTPROFILE wird angegeben, dass ein Optimierungsprofil auf der Paketebene zu verwenden ist. Über das Sonderregister CURRENT OPTIMIZATION PROFILE wird angegeben, dass ein Optimierungsprofil auf der Anweisungsebene zu verwenden ist.

Dieses Sonderregister enthält den qualifizierten Namen des Optimierungsprofils, das für Anweisungen verwendet wird, die dynamisch zur Optimierung vorbereitet werden. Für CLI-Anwendungen können Sie dieses Sonderregister über die Clientkonfigurationsoption CURRENTOPTIMIZATIONPROFILE für jede Verbindung definieren.

Die Einstellung der Bindeoption OPTPROFILE gibt außerdem das Standardoptimierungsprofil für das Sonderregister CURRENT OPTIMIZATION PROFILE an. Für die Auswertung der Standardwerte gilt die folgende Reihenfolge:

- Die Bindeoption OPTPROFILE gilt für alle statischen Anweisungen ungeachtet aller anderen Einstellungen.
- Für dynamische Anweisungen wird der Wert des Sonderregisters CURRENTOPTIMIZATIONPROFILE durch folgende Werte in der Reihenfolge von der niedrigsten zur höchsten Priorität bestimmt:
  - Bindeoption OPTPROFILE
  - Clientkonfigurationsoption CURRENTOPTIMIZATIONPROFILE

– Letzte Anweisung SET CURRENT OPTIMIZATION PROFILE in der Anwendung

Festlegen eines Optimierungsprofils in einer Anwendung:

Sie können die Einstellung des aktuellen Optimierungsprofils für dynamische Anweisungen in einer Anwendung mithilfe der Anweisung SET CURRENT OPTIMIZATION PROFILE steuern.

Informationen zu diesem Vorgang

Der Optimierungsprofilname, den Sie in der Anweisung angeben, muss ein mit einem Schemanamen qualifizierter Name sein. Wenn Sie keinen Schemanamen angeben, wird der Wert des Sonderregisters CURRENT SCHEMA als implizites Schemaqualifikationsmerkmal verwendet.

Das Optimierungsprofil, das Sie angeben, gilt für alle nachfolgenden dynamischen Anweisungen, bis eine andere Anweisung SET CURRENT OPTIMIZATION PROFILE angetroffen wird. Statische Anweisungen sind nicht betroffen, da sie vorverarbeitet und in ein Paket eingefügt werden, bevor diese Einstellung ausgewertet wird.

Vorgehensweise

Gehen Sie wie folgt vor, um ein Optimierungsprofil in einer Anwendung festzulegen:

- Sie können die Anweisung SET CURRENT OPTIMIZATION PROFILE an einer beliebigen Stelle innerhalb Ihrer Anwendung verwenden. Zum Beispiel wird die letzte Anweisung in der folgenden Sequenz gemäß dem Optimierungsprofil JON.SALES optimiert.

```
EXEC SQL  SET CURRENT OPTIMIZATION PROFILE = 'NEWTON.INVENTDB';

/* Die folgenden beiden Anweisungen werden mit 'NEWTON.INVENTDB' optimiert. */
EXEC SQL PREPARE stmt FROM SELECT ... ;
EXEC SQL EXECUTE stmt;

EXEC SQL EXECUTE IMMEDIATE SELECT ... ;

EXEC SQL SET CURRENT OPTIMIZATION PROFILE = 'JON.SALES';

/* Die folgende Anweisung wird mit 'JON.SALES' optimiert. */
EXEC SQL EXECUTE IMMEDIATE SELECT ... ;
```

- Wenn das Optimierungsprogramm das Standardoptimierungsprofil verwenden soll, das wirksam war, als die Anwendung gestartet wurde, geben Sie den Wert NULL an. Beispiel:

```
EXEC SQL SET CURRENT OPTIMIZATION PROFILE = NULL;
```

- Wenn das Optimierungsprogramm keine Optimierungsprofile verwenden soll, geben Sie die leere Zeichenfolge an. Beispiel:

```
EXEC SQL SET CURRENT OPTIMIZATION PROFILE = '';
```

- Wenn Sie eine CLI-Anwendung verwenden, können Sie den Parameter CURRENTOPTIMIZATIONPROFILE zur Datei `db2cli.ini` hinzufügen; verwenden Sie hierzu den Konfigurationsassistenten oder den Befehl **UPDATE CLI CONFIGURATION**. Beispiel:

```
update cli cfg for section sanfran using currentoptimizationprofile jon.sales
```

  Durch diesen Befehl wird der folgende Eintrag in der Datei `'db2cli.ini'` erstellt:

```
[SANFRAN]
CURRENTOPTIMIZATIONPROFILE=JON.SALES
```

Anmerkung: Diese Einstellung wird von jeder Anweisung SET CURRENT OPTIMIZATION PROFILE in der Anwendung überschrieben.

Binden eines Optimierungsprofils an ein Paket:

Wenn Sie ein Paket mithilfe des Befehls **BIND** oder **PRECOMPILE** vorbereiten, können Sie die Option OPTPROFILE verwenden, um das Optimierungsprofil für das Paket anzugeben.

Informationen zu diesem Vorgang

Diese Methode ist die einzige Möglichkeit, ein Optimierungsprofil auf statische Anweisungen anzuwenden. Dabei gilt das angegebene Profil für alle statischen Anweisungen im Paket. Ein Optimierungsprofil, das auf diese Weise angegeben wird, ist gleichzeitig das Standardoptimierungsprofil, das für dynamische Anweisungen innerhalb des Pakets verwendet wird.

Vorgehensweise

Sie können ein Optimierungsprofil in SQLJ oder in eingebettetem SQL mithilfe von APIs (z. B. sqlaprep) oder über den Befehlszeilenprozessor (CLP) binden.
Der nachfolgend dargestellte Code zeigt beispielsweise, wie Sie ein Optimierungsprofil für eine Inventardatenbank über den Befehlszeilenprozessor an die Inventaranwendung binden:

```
db2 prep inventapp.sqc bindfile optprofile newton.inventdb
db2 bind inventapp.bnd
db2 connect reset
db2 terminate
xlc -I$HOME/sqllib/include -c inventapp.c -o inventapp.o
xlc -o inventapp inventapp.o -ldb2 -L$HOME/sqllib/lib
```

Wenn Sie keinen Schemanamen für das Optimierungsprofil angeben, wird der Wert der Option QUALIFIER als implizites Qualifikationsmerkmal verwendet.

Modifizieren eines Optimierungsprofils:

Sie können ein Optimierungsprofil modifizieren, indem Sie das Dokument bearbeiten, es anhand des aktuellen Schemas für Optimierungsprofile (COPS) prüfen und das ursprüngliche Dokument in der Tabelle SYSTOOLS.OPT_PROFILE durch die neue Version ersetzen.

Informationen zu diesem Vorgang

Wenn ein Optimierungsprofil referenziert wird, wird es kompiliert und im Cache gespeichert. Daher müssen diese Verweise ebenfalls entfernt werden. Verwenden Sie die Anweisung FLUSH OPTIMIZATION PROFILE CACHE, um das alten Profil aus dem Optimierungsprofilcache zu entfernen und alle Anweisungen im Cache für dynamische Pläne ungültig zu machen, die mit dem alten Profil vorbereitet wurden (*logische Ungültigmachung*). Führen Sie folgende Schritte aus, um ein Optimierungsprofil zu modifizieren:

Vorgehensweise

1. Bearbeiten Sie das Optimierungsprofil, nehmen Sie die erforderlichen Änderungen vor und prüfen Sie das XML.

2. Aktualisieren Sie die Tabelle SYSTOOLS.OPT_PROFILE mit dem neuen Profil.
3. Wenn Sie keine Trigger zum Löschen des Cache für das Optimierungsprofil erstellt haben, geben Sie die Anweisung FLUSH OPTIMIZATION PROFILE CACHE ein, um alle Versionen des Optimierungsprofils zu entfernen, die sich möglicherweise im Optimierungsprofilcache befinden.

   Anmerkung: Wenn Sie den Cache für das Optimierungsprofil löschen (FLUSH), werden auch alle dynamischen Anweisungen im Cache für dynamische Pläne ungültig gemacht, die mit dem alten Optimierungsprofil vorbereitet wurden.

Ergebnisse

Alle nachfolgenden Verweise auf das Optimierungsprofil veranlassen das Optimierungsprogramm, das neue Profil zu lesen und es erneut in den Optimierungsprofilcache zu laden. Aufgrund der logischen Ungültigmachung von Anweisungen, die unter dem alten Optimierungsprofil vorbereitet wurden, werden darüber hinaus alle Aufrufe an diese Anweisungen unter dem neuen Optimierungsprofil vorbereitet und erneut im Cache für dynamische Pläne zwischengespeichert.

Löschen eines Optimierungsprofils:

Sie können ein Optimierungsprofil entfernen, das nicht mehr benötigt wird, indem Sie es aus der Tabelle SYSTOOLS.OPT_PROFILE löschen. Wenn ein Optimierungsprofil angegeben wird, wird es kompiliert und im Cache gespeichert. Wenn also das ursprüngliche Profil bereits verwendet wurde, müssen Sie das gelöschte Optimierungsprofil auch aus dem Optimierungsprofilcache löschen (FLUSH).

Informationen zu diesem Vorgang

Führen Sie folgende Schritte aus, um ein Optimierungsprofil zu löschen:

Vorgehensweise

1. Löschen Sie das Optimierungsprofil aus der Tabelle SYSTOOLS.OPT_PROFILE. Beispiel:

   ```
   delete from systools.opt_profile
     where schema = 'NEWTON' and name = 'INVENTDB'
   ```

2. Wenn Sie keine Trigger zum Löschen des Cache für das Optimierungsprofil erstellt haben, geben Sie die Anweisung FLUSH OPTIMIZATION PROFILE CACHE ein, um alle Versionen des Optimierungsprofils zu entfernen, die sich möglicherweise im Optimierungsprofilcache befinden.

   Anmerkung: Wenn Sie den Cache für das Optimierungsprofil löschen (FLUSH), werden auch alle dynamischen Anweisungen im Cache für dynamische Pläne ungültig gemacht, die mit dem alten Optimierungsprofil vorbereitet wurden.

Ergebnisse

Alle nachfolgenden Verweise auf das Optimierungsprofil veranlassen das Optimierungsprogramm, eine Warnung SQL0437W mit dem Ursachencode 13 zurückzugeben.

Optimierungsrichtlinien:

Typen von Optimierungsrichtlinien:

Das DB2Optimierungsprogramm verarbeitet eine Anweisung in zwei Phasen: die Optimierungsphase der Abfrageumschreibung und die Planoptimierungsphase.

Die optimierte Anweisung wird durch die *Optimierungsphase der Abfrageumschreibung* bestimmt, in der die ursprüngliche Anweisung in eine semantisch äquivalente Anweisung umgeformt wird, die sich in der Planoptimierungsphase leichter optimieren lässt. In der *Planoptimierungsphase* werden die optimalen Zugriffsmethoden, Joinmethoden und Joinreihenfolgen für die *optimierte Anweisung* bestimmt, indem eine Anzahl von Alternativen aufgezählt und die Alternative ausgewählt wird, die den günstigsten Schätzwert für den Ausführungsaufwand liefert.

Die Abfrageumsetzungen, Zugriffsmethoden, Joinmethoden, Joinreihenfolgen und andere Optimierungsalternativen, die während der beiden Optimierungsphasen in Betracht gezogen werden, werden durch verschiedene DB2-Parameter gesteuert, wie zum Beispiel das Sonderregister CURRENT QUERY OPTIMIZATION, die Bindeoption REOPT und die Registrierdatenbankvariable **DB2_REDUCED_OPTIMIZATION**. Die Gruppe der Optimierungsalternativen wird als *Suchbereich* bezeichnet.

Die folgenden Typen von Optimierungsrichtlinien für Anweisungen werden unterstützt:

- *Allgemeine Optimierungsrichtlinien*, die zur Beeinflussung der Einstellung allgemeiner Optimierungsparameter verwendet werden können, werden zuerst angewendet, da sie sich auf den Suchbereich auswirken können.
- *Richtlinien für das Umschreiben von Abfragen*, die zur Beeinflussung der Umsetzungen, die in der Optimierungsphase der Abfrageumschreibung in Betracht gezogen werden, verwendet werden können, werden als Nächstes angewendet, weil sie sich auf die Anweisung auswirken können, die in der Planoptimierungsphase optimiert wird.
- *Richtlinien zur Planoptimierung*, die zur Beeinflussung der Zugriffsmethoden, Joinmethoden und Joinreihenfolgen, die in der Planoptimierungsphase in Betracht gezogen werden, verwendet werden können, werden zuletzt angewendet.

Allgemeine Optimierungsrichtlinien:

Allgemeine Optimierungsrichtlinien können zur Festlegung allgemeiner Optimierungsparameter verwendet werden.

Geltungsbereich jeder dieser Richtlinien ist die Anweisungsebene.

Optimierungsrichtlinien für das Umschreiben von Abfragen:

Richtlinien für das Umschreiben von Abfragen können verwendet werden, um die Umsetzungen zu beeinflussen, die in der Optimierungsphase der Abfrageumschreibung in Betracht gezogen werden, in der die ursprüngliche Anweisung in eine semantisch äquivalente, optimierte Anweisung umgeformt wird.

Der optimale Ausführungsplan für die optimierte Anweisung wird in der Phase der Planoptimierung bestimmt. Infolgedessen können sich Optimierungsrichtlinien für die Abfrageumschreibung auf die Anwendbarkeit der Planoptimierungsrichtlinien auswirken.

Jede Optimierungsrichtlinie für die Abfrageumschreibung entspricht einer der Abfrageumsetzungsregeln des Optimierungsprogramms. Die folgenden Abfrageumsetzungsregeln können durch Optimierungsrichtlinien für die Abfrageumschreibung beeinflusst werden:

- Regel zur Umsetzung von IN-Listen in Joins (INLIST2JOIN)
- Regel zur Umsetzung von Unterabfragen in Joins (SUBQ2JOIN)
- Regel zur Umsetzung von NOT EXISTS-Unterabfragen in Antijoins (NOTEX2AJ)
- Regel zur Umsetzung von NOT IN-Unterabfragen in Antijoins (NOTIN2AJ)

Optimierungsrichtlinien für das Umschreiben von Abfragen sind nicht immer anwendbar. Abfrageumschreibregeln werden jeweils einzeln umgesetzt. Infolgedessen können sich Abfrageumschreibregeln, die vor einer nachfolgenden Regel umgesetzt werden, auf die Optimierungsrichtlinie für das Umschreiben von Abfragen auswirken, die dieser Regel zugeordnet ist. Die Umgebungskonfiguration kann sich auf das Verhalten einiger Umschreibregeln auswirken, was sich wiederum auf die Anwendbarkeit der Optimierungsrichtlinie für das Umschreiben von Abfragen auf eine bestimmte Regel auswirken kann.

Um sicherzustellen, dass jedes Mal dieselben Ergebnisse erzielt werden, unterliegen Regeln für das Umschreiben von Abfragen bestimmten Bedingungen, die angewendet werden, bevor die Regeln umgesetzt werden. Wenn die Bedingungen, die einer Regel zugeordnet sind, zu dem Zeitpunkt, zu dem die Abfrageumschreibkomponente versucht, die Regel auf die Abfrage anzuwenden, nicht erfüllt sind, wird die Optimierungsrichtlinie für die Abfrageumschreibung für die Regel ignoriert. Wenn die Optimierungsrichtlinie für die Abfrageumschreibung nicht anwendbar ist und die Richtlinie eine aktivierende Richtlinie ist, wird die Nachricht SQL0437W mit Ursachencode 13 zurückgegeben. Wenn die Optimierungsrichtlinie für die Abfrageumschreibung nicht anwendbar ist und die Richtlinie eine inaktivierende Richtlinie ist, wird keine Nachricht zurückgegeben. Die Abfrageumschreibregel wird in diesem Fall nicht angewendet, weil die Regel so behandelt wird, als wäre sie inaktiviert worden.

Optimierungsrichtlinien für die Abfrageumschreibung können in zwei Kategorien untergliedert werden: Richtlinien auf Anweisungsebene und Richtlinien auf Vergleichselementebene. Alle Optimierungsrichtlinien für die Abfrageumschreibung unterstützen die Kategorie der Anweisungsebene. Nur INLIST2JOIN unterstützt die Kategorie der Vergleichselementebene. Die Optimierungsrichtlinien für die Abfrageumschreibung auf Anweisungsebene gelten für die gesamte Abfrage. Die Optimierungsrichtlinie für die Abfrageumschreibung auf Vergleichselementebene gilt nur für das bestimmte Vergleichselement. Wenn Optimierungsrichtlinien für die Abfrageumschreibung sowohl auf Anweisungs- als auch auf Vergleichselementebene angegeben werden, überschreibt die Richtlinie auf Vergleichselementebene die Richtlinie auf Anweisungsebene für das bestimmte Vergleichselement.

Jede Optimierungsrichtlinie für die Abfrageumschreibung wird durch ein entsprechendes Umschreibanforderungselement im Schema der Optimierungsrichtlinie dargestellt.

Das folgende Beispiel veranschaulicht eine Optimierungsrichtlinie für INLIST2JOIN-Abfrageumschreibung, dargestellt durch das Anforderungselement für Abfrageumschreibung INLIST2JOIN.

```
SELECT S.S_NAME, S.S_ADDRESS, S.S_PHONE, S.S_COMMENT
FROM "Tpcd".PARTS P, "Tpcd".SUPPLIERS S, "Tpcd".PARTSUPP PS
WHERE P_PARTKEY = PS.PS_PARTKEY AND S.S_SUPPKEY = PS.PS_SUPPKEY
  AND P_SIZE IN (35, 36, 39, 40)
```

```
  AND S.S_NATION IN ('INDIA', 'SPAIN')
  AND PS.PS_SUPPLYCOST = (SELECT MIN(PS1.PS_SUPPLYCOST)
                          FROM "Tpcd".PARTSUPP PS1, "Tpcd".SUPPLIERS S1
                          WHERE P.P_PARTKEY  = PS1.PS_PARTKEY AND
                            S1.S_SUPPKEY = PS1.PS_SUPPKEY
                            AND S1.S_NATION = S.S_NATION)
ORDER BY S.S_NAME
<OPTGUIDELINES><INLIST2JOIN TABLE='P'/></OPTGUIDELINES>;
```

Diese spezielle Optimierungsrichtlinie für die Abfrageumschreibung gibt an, dass die Liste der Konstanten im Vergleichselement `P_SIZE IN (35, 36, 39, 40)` in einen Tabellenausdruck umgewandelt werden soll. Dieser Tabellenausdruck kann anschließend zum Ausführen eines Joinzugriffs mit Verschachtelungsschleife und Index auf die Tabelle PARTS im Hauptsubselect ausgewählt werden. Das Attribut TABLE wird zur Kennzeichnung des Zielvergleichselements mit IN-Liste verwendet, indem es den Tabellenverweis angibt, auf den sich das Vergleichselement bezieht. Sind mehrere Vergleichselemente mit IN-Listen für den betreffenden Tabellenverweis vorhanden, wird das Anforderungselement für INLIST2JOIN-Abfrageumschreibung als nicht eindeutig betrachtet und ignoriert.

In diesen Fällen kann ein Attribut COLUMN hinzugefügt werden, um das Zielvergleichselement mit IN-Liste näher zu kennzeichnen. Beispiel:

```
SELECT S.S_NAME, S.S_ADDRESS, S.S_PHONE, S.S_COMMENT
FROM "Tpcd".PARTS P, "Tpcd".SUPPLIERS S, "Tpcd".PARTSUPP PS
WHERE P_PARTKEY = PS.PS_PARTKEY AND S.S_SUPPKEY = PS.PS_SUPPKEY
  AND P_SIZE IN (35, 36, 39, 40)
  AND P_TYPE IN ('BRASS', 'COPPER')
  AND S.S_NATION IN ('INDIA', 'SPAIN')
  AND PS.PS_SUPPLYCOST = (SELECT MIN(PS1.PS_SUPPLYCOST)
                          FROM "Tpcd".PARTSUPP PS1, "Tpcd".SUPPLIERS S1
                          WHERE P.P_PARTKEY  = PS1.PS_PARTKEY
                            AND S1.S_SUPPKEY = PS1.PS_SUPPKEY
                            AND S1.S_NATION = S.S_NATION)
ORDER BY S.S_NAME
<OPTGUIDELINES><INLIST2JOIN TABLE='P' COLUMN='P_SIZE'/></OPTGUIDELINES>;
```

Das Attribut TABLE des Elements INLIST2JOIN kennzeichnet den Verweis auf die Tabelle PARTS im Hauptsubselect. Das Attribut COLUMN dient zur Kennzeichnung des Vergleichselements mit IN-Liste für die Spalte P_SIZE als Zielelement. Im Allgemeinen kann der Wert des Attributs COLUMN den unqualifizierten Namen der Spalte enthalten, auf die im Zielvergleichselement mit IN-Liste verwiesen wird. Wird das Attribut COLUMN ohne das Attribut TABLE bereitgestellt, wird die Optimierungsrichtlinie für die Abfrageumschreibung als ungültig betrachtet und ignoriert.

Mit dem Attribut OPTION kann eine bestimmte Optimierungsrichtlinie für die Abfrageumschreibung aktiviert bzw. inaktiviert werden. Da das Attribut OPTION im folgenden Beispiel mit DISABLE definiert ist, wird die Liste der Konstanten im Vergleichselement `P_SIZE IN (35, 36, 39, 40)` nicht in einen Tabellenausdruck umgewandelt. Der Standardwert des Attributs OPTION ist ENABLE. ENABLE und DISABLE müssen vollständig in Großbuchstaben angegeben werden.

```
<OPTGUIDELINES>
  <INLIST2JOIN TABLE='P' COLUMN='P_SIZE' OPTION='DISABLE'/>
</OPTGUIDELINES>
```

Im folgenden Beispiel verfügt das Umschreibanforderungselement INLIST2JOIN nicht über das Attribut TABLE. Vom Optimierungsprogramm wird dies als eine Anforderung zum Inaktivieren der INLIST2JOIN-Abfrageumsetzung für alle Vergleichselemente mit IN-Listen in der Anweisung interpretiert.

```
<OPTGUIDELINES><INLIST2JOIN OPTION='DISABLE'/></OPTGUIDELINES>
```

Das folgende Beispiel veranschaulicht eine Optimierungsrichtlinie für eine Abfrageumschreibung mit Umsetzung von Unterabfragen in Joins, dargestellt durch das Umschreibanforderungselement SUBQ2JOIN. Bei einer Umsetzung von Unterabfragen in Joins wird eine Unterabfrage in einen äquivalenten Tabellenausdruck konvertiert. Umgesetzt werden Vergleichselemente für Unterabfragen, die durch EXISTS, IN, =SOME, =ANY, <>SOME oder <>ANY quantifiziert sind. Die Optimierungsrichtlinie für eine Abfrageumschreibung mit Umsetzung von Unterabfragen in Joins stellt nicht sicher, dass eine Unterabfrage zusammengeführt wird. Diese Optimierungsrichtlinie für die Abfrageumschreibung kann nicht auf eine bestimmte Unterabfrage ausgerichtet werden. Die Umsetzung kann nur auf Anweisungsebene aktiviert oder inaktiviert werden.

```
SELECT S.S_NAME, S.S_ADDRESS, S.S_PHONE, S.S_COMMENT
FROM "Tpcd".PARTS P, "Tpcd".SUPPLIERS S, "Tpcd".PARTSUPP PS
WHERE P_PARTKEY = PS.PS_PARTKEY AND S.S_SUPPKEY = PS.PS_SUPPKEY
  AND P_SIZE IN (35, 36, 39, 40)
  AND P_TYPE = 'BRASS'
  AND S.S_NATION IN ('INDIA', 'SPAIN')
  AND PS.PS_SUPPLYCOST = (SELECT MIN(PS1.PS_SUPPLYCOST)
                          FROM "Tpcd".PARTSUPP PS1, "Tpcd".SUPPLIERS S1
                          WHERE P.P_PARTKEY  = PS1.PS_PARTKEY
                          AND S1.S_SUPPKEY = PS1.PS_SUPPKEY
                            AND S1.S_NATION = S.S_NATION)
ORDER BY S.S_NAME
<OPTGUIDELINES><SUBQ2JOIN OPTION='DISABLE'/></OPTGUIDELINES>;
```

Das folgende Beispiel veranschaulicht eine Optimierungsrichtlinie für NOTEX2AJ-Abfrageumschreibung, dargestellt durch das Umschreibanforderungselement NOTEX2AJ. Bei einer NOTEX2AJ-Umsetzung wird eine Unterabfrage in einen Tabellenausdruck umgesetzt, der mit anderen Tabellen über Antijoin-Semantik (es werden nur Zeilen zurückgegeben, die keine Entsprechung aufweisen) verknüpft ist. Die Optimierungsrichtlinie für NOTEX2AJ-Abfrageumschreibung bezieht sich auf Vergleichselemente für Unterabfragen, die über NOT EXISTS quantifiziert werden. Die Optimierungsrichtlinie für NOTEX2AJ-Abfrageumschreibung stellt nicht sicher, dass eine Unterabfrage zusammengeführt wird. Diese Optimierungsrichtlinie für die Abfrageumschreibung kann nicht auf eine bestimmte Unterabfrage ausgerichtet werden. Die Umsetzung kann nur auf Anweisungsebene aktiviert oder inaktiviert werden.

```
SELECT S.S_NAME, S.S_ADDRESS, S.S_PHONE, S.S_COMMENT
FROM "Tpcd".PARTS P, "Tpcd".SUPPLIERS S, "Tpcd".PARTSUPP PS
WHERE P_PARTKEY = PS.PS_PARTKEY AND S.S_SUPPKEY = PS.PS_SUPPKEY
  AND P_SIZE IN (35, 36, 39, 40)
  AND P_TYPE = 'BRASS'
  AND S.S_NATION IN ('INDIA', 'SPAIN')
  AND NOT EXISTS (SELECT 1
                  FROM "Tpcd".SUPPLIERS S1
                  WHERE S1.S_SUPPKEY = PS.PS_SUPPKEY)
ORDER BY S.S_NAME
<OPTGUIDELINES><NOTEX2AJ OPTION='ENABLE'/></OPTGUIDELINES>;
```

Anmerkung: Wird die Regel für die Abfrageumsetzung auf Anweisungsebene aktiviert, garantiert dies nicht, dass die Regel auf einen bestimmten Teil der Anweisung angewendet wird. Ob eine Abfrageumsetzung stattfindet, richtet sich nach den üblichen Kriterien. Enthält der Abfrageblock z. B. mehrere Vergleichselemente NOT EXISTS, werden diese Vergleichselemente vom Optimierungsprogramm nicht für eine Konvertierung in Antijoins in Betracht gezogen. Auch eine explizite Aktivierung der Abfrageumsetzung auf Anweisungsebene ändert dieses Verhalten nicht.

Das folgende Beispiel veranschaulicht eine Optimierungsrichtlinie für NOTIN2AJ-Abfrageumschreibung, dargestellt durch das Umschreibanforderungselement NOTIN2AJ. Bei einer NOTIN2AJ-Umsetzung wird eine Unterabfrage in einen Tabellenausdruck umgesetzt, der mit anderen Tabellen über Antijoin-Semantik (es werden nur Zeilen zurückgegeben, die keine Entsprechung aufweisen) verknüpft ist. Die Optimierungsrichtlinie für NOTIN2AJ-Abfrageumschreibung bezieht sich auf Vergleichselemente für Unterabfragen, die über NOT IN quantifiziert werden. Die Optimierungsrichtlinie für NOTIN2AJ-Abfrageumschreibung stellt nicht sicher, dass eine Unterabfrage zusammengeführt wird. Diese Optimierungsrichtlinie für die Abfrageumschreibung kann nicht auf eine bestimmte Unterabfrage ausgerichtet werden. Die Umsetzung kann nur auf Anweisungsebene aktiviert oder inaktiviert werden.

```
SELECT S.S_NAME, S.S_ADDRESS, S.S_PHONE, S.S_COMMENT
FROM "Tpcd".PARTS P, "Tpcd".SUPPLIERS S, "Tpcd".PARTSUPP PS
WHERE P_PARTKEY = PS.PS_PARTKEY AND S.S_SUPPKEY = PS.PS_SUPPKEY
  AND P_SIZE IN (35, 36, 39, 40)
  AND P_TYPE = 'BRASS'
  AND S.S_NATION IN ('INDIA', 'SPAIN')
  AND PS.PS_SUPPKEY NOT IN (SELECT S1.S_SUPPKEY
                            FROM "Tpcd".SUPPLIERS S1
                            WHERE S1.S_NATION = 'CANADA')
ORDER BY S.S_NAME
<OPTGUIDELINES><NOTIN2AJ OPTION='ENABLE'/></OPTGUIDELINES>
```

Möglicherweise sind bestimmte Optimierungsrichtlinien für die Abfrageumschreibung nicht anwendbar, wenn andere Umsetzungen zur Abfrageumschreibung auf eine Anweisung angewendet werden. Kann eine Richtlinienanforderung zum Aktivieren einer Umsetzung nicht angewendet werden, wird eine Warnung zurückgegeben. Beispielsweise ist ein Anforderungselement für die Aktivierung der INLIST2JOIN-Umschreibung mit einem Vergleichselement als Ziel, das durch eine andere Abfrageumsetzung aus der Abfrage entfernt wurde, nicht anwendbar. Eine erfolgreiche Anwendung einer Optimierungsrichtlinie für die Abfrageumschreibung kann sogar die Anwendbarkeit anderer Umsetzungsregeln für die Abfrageumschreibung ändern. Eine Anforderung zum Umsetzen einer IN-Liste in einen Tabellenausdruck kann z. B. verhindern, dass eine andere IN-Liste in einen Tabellenausdruck umgesetzt wird, da das Optimierungsprogramm nur eine einzige INLIST2JOIN-Umsetzung pro Abfrageblock anwendet.

Richtlinien zur Planoptimierung:

Planoptimierungsrichtlinien werden in der auf Aufwandsberechnungen basierenden Phase der Optimierung angewendet, in der Zugriffsmethoden, Joinmethoden, Joinreihenfolgen und andere Details des Ausführungsplans für die Anweisung bestimmt werden.

Planoptimierungsrichtlinien brauchen nicht alle Aspekte eines Ausführungsplans anzugeben. Nicht angegebene Aspekte des Ausführungsplans werden durch das Optimierungsprogramm auf der Grundlage von Aufwandsberechnungen ermittelt.

Es gibt zwei Kategorien von Planoptimierungsrichtlinien:

- accessRequest – Eine Zugriffsanforderung gibt eine Zugriffsmethode zur Erfüllung eines Tabellenverweises in einer Anweisung an.
- joinRequest – Eine Joinanforderung gibt eine Methode und eine Reihenfolge für die Ausführung einer Joinoperation an. Joinanforderungen bestehen wiederum aus Zugriffs- oder anderen Joinanforderungen.

Optimierungsrichtlinien für Zugriffsanforderungen entsprechen den Datenzugriffsmethoden des Optimierungsprogramms, wie zum Beispiel Tabellensuche, Indexsuche und Vorablesezugriff über Listen. Die Richtlinien für Joinanforderungen entsprechen den Joinmethoden des Optimierungsprogramms, wie zum Beispiel Join mit Verschachtelungsschleife (Nested Loop Join), Hash-Join und Mischjoin (Merge Join). Jede Zugriffsanforderung und jede Joinanforderung werden durch ein entsprechendes Zugriffsanforderungs- bzw. Joinanforderungselement im Schema der Optimierungsrichtlinie für Anweisungen dargestellt.

Das folgende Beispiel veranschaulicht eine Zugriffsanforderung für Indexsuche, dargestellt durch das Zugriffsanforderungselement IXSCAN. Diese spezielle Anforderung gibt an, dass das Optimierungsprogramm den Index I_SUPPKEY für den Zugriff auf die Tabelle SUPPLIERS im Hauptsubselect der Anweisung verwenden soll. Das optionale Attribut INDEX gibt den gewünschten Index an. Das Tabellenattribut gibt den Tabellenverweis an, auf den die Zugriffsanforderung angewendet wird. Das Attribut TABLE muss den Zieltabellenverweis mit dem zugehörigen exponierten Namen angeben. In diesem Fall handelt es sich dabei um den Korrelationsnamen S.

SQL-Anweisung:

```
   select s.s_name, s.s_address, s.s_phone, s.s_comment
     from "Tpcd".parts, "Tpcd".suppliers s, "Tpcd".partsupp ps
     where p_partkey = ps.ps_partkey and
       s.s_suppkey = ps.ps_suppkey and
       p_size = 39 and
       p_type = 'BRASS' and
       s.s_nation in ('MOROCCO', 'SPAIN') and
       ps.ps_supplycost = (select min(ps1.ps_supplycost)
                             from "Tpcd".partsupp ps1, "Tpcd".suppliers s1
                             where "Tpcd".parts.p_partkey = ps1.ps_partkey and
                             s1.s_suppkey = ps1.ps_suppkey and
                             s1.s_nation = s.s_nation)
     order by s.s_name
```

Optimierungsrichtlinie:

```
       <OPTGUIDELINES>
     <IXSCAN TABLE='S' INDEX='I_SUPPKEY'/>
   </OPTGUIDELINES>
```

Das folgende Zugriffsanforderungselement für Indexsuche gibt an, dass das Optimierungsprogramm über einen Indexzugriff auf die Tabelle PARTS im Hauptsubselect der Anweisung zugreifen soll. Das Optimierungsprogramm wählt den Index nach geschätztem Ausführungsaufwand aus, da kein Attribut INDEX angegeben ist. Das Attribut TABLE gibt den qualifizierten Tabellennamen für den Zieltabellenverweis an, da kein zugeordneter Korrelationsname vorhanden ist.

```
       <OPTGUIDELINES>
     <IXSCAN TABLE='"Tpcd".PARTS'/>
   </OPTGUIDELINES>
```

Die folgende Zugriffsanforderung für Vorablesezugriff über Listen wird über das Zugriffsanforderungselement LPREFETCH dargestellt. Diese spezielle Anforderung gibt an, dass das Optimierungsprogramm den Index I_SNATION für den Zugriff auf die Tabelle SUPPLIERS im verschachtelten Subselect der Anweisung verwenden soll. Beim Attribut TABLE wird der Korrelationsname S1 verwendet, da es sich hierbei um den exponierten Namen handelt, der den Tabellenverweis für SUPPLIERS im verschachtelten Subselect kennzeichnet.

```
      <OPTGUIDELINES>
    <LPREFETCH TABLE='S1' INDEX='I_SNATION'/>
  </OPTGUIDELINES>
```

Das folgende Zugriffsanforderungselement für Indexsuche gibt an, dass das Optimierungsprogramm den Index I_SNAME für den Zugriff auf die Tabelle SUPPLIERS im Hauptsubselect verwenden soll. Das Attribut FIRST gibt an, dass diese Tabelle die erste Tabelle sein soll, auf die in der gewählten Joinsequenz für die entsprechende FROM-Klausel zugegriffen wird. Das Attribut FIRST kann jeder beliebigen Zugriffs- bzw. Joinanforderung hinzugefügt werden. Es darf jedoch in einer Klausel FROM nur eine einzige Zugriffs- oder Joinanforderung mit dem Attribut FIRST vorliegen, die auf Tabellen verweist.

SQL-Anweisung:

```
  select s.s_name, s.s_address, s.s_phone, s.s_comment
    from "Tpcd".parts, "Tpcd".suppliers s, "Tpcd".partsupp ps
    where p_partkey = ps.ps_partkey
      s.s_suppkey = ps.ps_suppkey and
      p_size = 39 and
      p_type = 'BRASS' and
      s.s_nation in ('MOROCCO', 'SPAIN') and
      ps.ps_supplycost = (select min(ps1.ps_supplycost)
                            from "Tpcd".partsupp ps1, "Tpcd".suppliers s1
                            where "Tpcd".parts.p_partkey = ps1.ps_partkey and
                              s1.s_suppkey = ps1.ps_suppkey and
                              s1.s_nation = s.s_nation)
    order by s.s_name
    optimize for 1 row
```

Optimierungsrichtlinien:

```
      <OPTGUIDELINES>
    <IXSCAN TABLE='S' INDEX='I_SNAME' FIRST='TRUE'/>
  </OPTGUIDELINES>
```

Das folgende Beispiel zeigt, wie mehrere Zugriffsanforderungen in einer einzigen Optimierungsrichtlinie für Anweisungen übergeben werden können. Das Zugriffsanforderungselement TBSCAN stellt eine Zugriffsanforderung für Tabellensuche dar. Diese spezielle Anforderung gibt an, dass der Zugriff auf die Tabelle im verschachtelten Subselect über eine vollständige Tabellensuche erfolgen soll. Das Zugriffsanforderungselement LPREFETCH gibt an, dass das Optimierungsprogramm den Index I_SUPPKEY beim Indexvorablesezugriff über Listen auf die Tabelle SUPPLIERS im Hauptsubselect verwenden soll.

```
      <OPTGUIDELINES>
    <TBSCAN TABLE='S1'/>
    <LPREFETCH TABLE='S' INDEX='I_SUPPKEY'/>
  </OPTGUIDELINES>
```

Das folgende Beispiel veranschaulicht eine Anforderung für eine Joinoperation mit Verschachtelungsschleife, dargestellt durch das Joinanforderungselement NLJOIN. Im Allgemeinen enthält eine Joinanforderung zwei untergeordnete Elemente. Das erste untergeordnete Element stellt die gewünschte äußere Eingabe für die Joinoperation dar, während das zweite untergeordnete Element die gewünschte innere Eingabe für die Joinoperation darstellt. Bei den untergeordneten Elementen kann es sich um Zugriffsanforderungen, andere Joinanforderungen oder eine Kombination aus beidem handeln. Im vorliegenden Beispiel gibt das erste Zugriffsanforderungselement IXSCAN an, dass die Tabelle PARTS im Hauptsubselect die äußere Tabelle der Joinoperation darstellen soll. Das Element gibt darüber hinaus an, dass der Zugriff auf die Tabelle PARTS über eine Indexsuche erfolgen soll. Das zweite Zugriffsanforderungselement IXSCAN gibt an, dass die Tabelle PARTSUPP im Hauptsubse-

lect die innere Tabelle der Joinoperation darstellen soll. Darüber hinaus gibt dieses Element an, dass der Zugriff auf die Tabelle über eine Indexsuche erfolgen soll.

```
    <OPTGUIDELINES>
  <NLJOIN>
    <IXSCAN TABLE='"Tpcd".Parts'/>
    <IXSCAN TABLE="PS"/>
  </NLJOIN>
</OPTGUIDELINES>
```

Das folgende Beispiel veranschaulicht eine Anforderung für Hash-Joins, dargestellt durch das Joinanforderungselement HSJOIN. Das Zugriffsanforderungselement ACCESS gibt an, dass die Tabelle SUPPLIERS im verschachtelten Subselect die äußere Tabelle der Joinoperation darstellen soll. Dieses Zugriffsanforderungselement ist hilfreich, wenn die Festlegung der Reihenfolge beim Join der Hauptzweck ist. Das Zugriffsanforderungselement IXSCAN gibt an, dass die Tabelle PARTSUPP im verschachtelten Subselect die innere Tabelle der Joinoperation darstellen und das Optimierungsprogramm über eine Indexsuche auf diese Tabelle zugreifen soll.

```
    <OPTGUIDELINES>
  <HSJOIN>
    <ACCESS TABLE='S1'/>
    <IXSCAN TABLE='PS1'/>
  </HSJOIN>
</OPTGUIDELINES>
```

Das folgende Beispiel veranschaulicht, wie umfangreichere Joinanforderungen mithilfe von verschachtelten Joinanforderungen erstellt werden können. Das Beispiel enthält eine Mischjoinanforderung, dargestellt durch das Joinanforderungselement MSJOIN. Die äußere Eingabe der Joinoperation ist das Ergebnis der Joinoperation für die beiden Tabellen PARTS und PARTSUPP des Hauptsubselects, dargestellt durch das Joinanforderungselement NLJOIN. Die innere Eingabe des Joinanforderungselements ist die Tabelle SUPPLIERS im Hauptsubselect, dargestellt durch das Zugriffsanforderungselement IXSCAN.

```
<OPTGUIDELINES>
  <MSJOIN>
    <NLJOIN>
      <IXSCAN TABLE='"Tpcd".Parts'/>
      <IXSCAN TABLE="PS"/>
    </NLJOIN>
    <IXSCAN TABLE='S'/>
  </MSJOIN>
</OPTGUIDELINES>
```

Bei gültigen Joinanforderungen müssen alle Zugriffsanforderungselemente, die direkt oder indirekt in der Anforderung verschachtelt sind, auf Tabellen in derselben Klausel FROM der optimierten Anweisung verweisen.

Optimierungsrichtlinien für den MQT-Abgleich

Benutzer können die Entscheidungen des Optimierungsprogramms überschreiben und mithilfe des Elements MQTENFORCE die Auswahl bestimmter MQTs (Materialized Query Tables) erzwingen. Das Element MQTENFORCE kann sowohl auf globaler Ebene als auch auf Anweisungsprofilebene angegeben werden und wird mit einem der folgenden Attribute verwendet:

NAME
Gibt den teilweise oder vollständig qualifizierten Namen der auszuwählenden MQT an.

TYPE Gibt eine Gruppe von MQTs anhand ihres Typs an. Mögliche Werte:

- NORMAL: Alle nicht replizierten MQTs.
- REPLICATED: Alle replizierten MQTs.
- ALL: Alle MQTs.

Im folgenden Beispiel ist eine Richtlinie dargestellt, mit der die Auswahl aller replizierten MQTs sowie von TPCD.PARTSMQT erzwungen wird.

```
<OPTGUIDELINES>
 <MQTENFORCE NAME='TPCD.PARTSMQT'/>
 <MQTENFORCE TYPE='REPLICATED'/>
</OPTGUIDELINES>
```

Anmerkung: Wenn Sie mehrere Attribute gleichzeitig angeben, wird nur das erste verwendet. Beispiel:

```
<MQTENFORCE NAME='TPCD.PARTSMQT' TYPE='REPLICATED'/>
```

In diesem Fall wird nur die Auswahl von PARTSMQT MQT erzwungen.

Erstellen von Optimierungsrichtlinien auf Anweisungsebene:

Der im Anweisungsprofil enthaltene Abschnitt für die Optimierungsrichtlinien auf Anweisungsebene besteht aus einer oder mehreren Zugriffs- oder Joinanforderungen, durch die die Methoden für den Zugriff auf bzw. den Join von Tabellen in der Anweisung spezifiziert werden.

Vorbereitende Schritte

Schöpfen Sie alle anderen Optimierungsoptionen aus. Zum Beispiel:

1. Stellen Sie sicher, dass die Statistikdaten zur Datenverteilung kürzlich durch das Dienstprogramm RUNSTATS aktualisiert wurden.
2. Stellen Sie sicher, dass der Datenserver mit der für die Auslastung geeigneten Optimierungsklasse arbeitet.
3. Stellen Sie sicher, dass dem Optimierungsprogramm die entsprechenden Indizes für den Zugriff auf die in der Abfrage referenzierten Tabellen zur Verfügung stehen.

Informationen zu diesem Vorgang

Gehen Sie wie folgt vor, um Optimierungsrichtlinien auf Anwendungsebene zu erstellen:

Vorgehensweise

1. Erstellen Sie das Optimierungsprofil, in das Sie die Richtlinien auf Anweisungsebene einfügen möchten.
2. Führen Sie die EXPLAIN-Funktion für die Anweisung aus, um festzustellen, ob Optimierungsrichtlinien von Nutzen sind. Ist dies der Fall, setzen Sie die Prozedur fort.
3. Rufen Sie die *ursprüngliche* Anweisung ab, indem Sie eine Abfrage ähnlich der folgenden ausführen:

```
select statement_text
  from explain_statement
  where explain_level = '0' and
    explain_requester = 'SIMMEN' and
```

```
explain_time      = '2003-09-08-16.01.04.108161' and
source_name       = 'SQLC2E03' and
source_version    = '' and
queryno           = 1
```

4. Bearbeiten Sie das Optimierungsprofil, und erstellen Sie ein Anweisungsprofil, indem Sie den Anweisungstext in den Anweisungsschlüssel einfügen. Zum Beispiel:

```
<STMTPROFILE ID="Guidelines for TPCD Q9">
   <STMTKEY SCHEMA="TPCD"><![CDATA[SELECT S.S_NAME, S.S_ADDRESS, S.S_PHONE,
     S.S_COMMENT
     FROM PARTS P, SUPPLIERS S, PARTSUPP PS
     WHERE P_PARTKEY = PS.PS_PARTKEY AND S.S_SUPPKEY = PS.PS_SUPPKEY
     AND P.P_SIZE  = 39 AND P.P_TYPE  = 'BRASS' AND S.S_NATION
     = 'MOROCCO' AND
     PS.PS_SUPPLYCOST = (SELECT MIN(PS1.PS_SUPPLYCOST)
     FROM PARTSUPP PS1, SUPPLIERS S1
     WHERE P.P_PARTKEY  = PS1.PS_PARTKEY AND S1.S_SUPPKEY = PS1.PS_SUPPKEY
     AND S1.S_NATION  = S.S_NATION)]]>
   </STMTKEY>
</STMTPROFILE>
```

5. Fügen Sie Optimierungsrichtlinien auf Anweisungsebene nach dem Anweisungsschlüssel ein. Verwenden Sie exponierte Namen zur Identifikation der Objekte, die in den Zugriffs- und Joinanforderungen referenziert werden. Das folgende Beispiel zeigt eine Joinanforderung:

```
<OPTGUIDELINES>
  <HSJOIN>
    <TBSCAN TABLE='PS1'/>
    <IXSCAN TABLE='S1'
      INDEX='I1'/>
  </HSJOIN>
</OPTGUIDELINES>
```

6. Führen Sie eine Gültigkeitsprüfung für die Datei aus, und speichern Sie die Datei.

Ergebnisse

Wenn nicht das erwartete Ergebnis erzielt wird, ändern Sie die Richtlinien oder erstellen Sie zusätzliche Richtlinien und aktualisieren Sie das Optimierungsprofil entsprechend.

Bildung von Tabellenverweisen in Optimierungsrichtlinien:

Der Begriff *Tabellenverweis* wird zur Bezeichnung einer beliebigen Angabe einer Tabelle oder Sicht, eines Tabellenausdrucks oder einer Tabelle, auf die ein Aliasname verweist, in einer SQL-Anweisung bzw. Sichtdefinition verwendet. Eine Optimierungsrichtlinie kann einen Tabellenverweis durch den in der ursprünglichen Anweisung angegebenen exponierten Namen (engl. exposed name) oder durch den eindeutigen Korrelationsnamen angeben, der dem Tabellenverweis in der optimierten Anweisung zugeordnet wird.

Erweiterte Namen, die aus Folgen von exponierten Namen bestehen, helfen bei der eindeutigen Angabe von Tabellenverweisen, die in Sichten eingebettet sind. Ein Aliasname kann nicht als Tabellenverweis in einer Optimierungsrichtlinie verwendet werden. In einem solchen Fall wird jede Richtlinie, die auf den Tabellenverweis Bezug nimmt, ignoriert. Optimierungsrichtlinien, die exponierte Namen oder erweiterte Namen angeben, die im Kontext der gesamten Anweisung nicht eindeutig sind, werden als mehrdeutig angesehen und nicht angewendet. Wenn darüber hinaus derselbe Tabellenverweis von mehr als einer Optimierungsrichtlinie angegeben wird, werden alle Optimierungsrichtlinien, die diesen Tabellenverweis ange-

ben, als gegenseitig unverträglich betrachtet und nicht angewendet. Aufgrund möglicher Abfrageumsetzungen ist nicht zu garantieren, dass ein exponierter oder erweiterter Name während der Optimierung erhalten bleibt. Eine Richtlinie, die solche Tabellenverweise angibt, wird dementsprechend ignoriert.

Verwenden von exponierten Namen in der ursprünglichen Anweisung zur Angabe von Tabellenverweisen

Ein Tabellenverweis wird unter Verwendung des exponierten Namens der Tabelle angegeben. Der exponierte Name wird auf dieselbe Weise angegeben, in der auch eine Tabelle in einer SQL-Anweisung qualifiziert wird.

Die Regeln zur Angabe von SQL-Kennungen gelten ebenso für den Wert des Attributs TABLE einer Optimierungsrichtlinie. Der Wert des Attributs TABLE wird mit jedem exponierten Namen in der Anweisung verglichen. In diesem DB2-Release ist nur eine einzige Übereinstimmung zulässig. Wenn der Wert des Attributs TABLE durch den Schemanamen qualifiziert ist, entspricht er jedem äquivalenten, qualifizierten exponierten Tabellennamen. Wenn der Wert des Attributs TABLE nicht qualifiziert ist, entspricht er jedem äquivalenten Korrelationsnamen oder exponierten Tabellennamen. Der Wert des Attributs TABLE wird daher implizit als mit dem für die Anweisung gültigen Standardschema qualifiziert betrachtet. Diese Konzepte werden im folgenden Beispiel veranschaulicht. Nehmen Sie an, dass die Anweisung unter Verwendung des Standardschemas 'Tpcd' optimiert wird.

```
select s_name, s_address, s_phone, s_comment
  from parts, suppliers, partsupp ps
  where p_partkey = ps.ps_partkey and
    s.s_suppkey = ps.ps_suppkey and
    p_size = 39 and
    p_type = 'BRASS'
```

Werte des Attributs TABLE, die einen Tabellenverweis in der Anweisung angeben, sind '"Tpcd".PARTS', 'PARTS', 'Parts' (da die Kennung keine Begrenzungszeichen hat, wird sie in Großschreibung umgesetzt). Werte des Attributs TABLE, die einen Tabellenverweis in der Anweisung nicht korrekt angeben, sind zum Beispiel '"Tpcd2".SUPPLIERS', 'PARTSUPP' (kein exponierter Name) und 'Tpcd.PARTS' (die Kennung Tpcd muss mit Begrenzern angegeben werden, da sie ansonsten in Großbuchstaben umgesetzt wird).

Der exponierte Name kann zur Angabe eines beliebigen Tabellenverweises in der ursprünglichen Anweisung, Sicht, SQL-Funktion oder im ursprünglichen Trigger verwendet werden.

Verwenden exponierter Namen in der ursprünglichen Anweisung zur Angabe von Tabellenverweisen in Sichten

Optimierungsrichtlinien können eine erweiterte Syntax zur Angabe von Tabellenverweisen verwenden, die in Sichten eingebettet sind. Siehe dazu das folgende Beispiel:

```
create view "Rick".v1 as
  (select * from employee a where salary > 50000)

create view "Gustavo".v2 as
  (select * from "Rick".v1
     where deptno in ('52', '53', '54')

select * from "Gustavo".v2 a
  where v2.hire_date > '01/01/2004'
```

```
    <OPTGUIDELINES>
  <IXSCAN TABLE='A/"Rick".V1/A'/>
</OPTGUIDELINES>
```

Das Zugriffsanforderungselement IXSCAN gibt an, dass eine Indexsuche für den Verweis auf die Tabelle EMPLOYEE verwendet werden soll, der in die Sichten "Gustavo".V2 und "Rick".V1 eingebettet ist. Die erweiterte Syntax für die Angabe von Tabellenverweisen in Sichten sieht eine Folge exponierter Namen vor, die durch einen Schrägstrich getrennt sind. Der Wert für das Attribut TABLE A/"Rick".V1/A verdeutlicht diese erweiterte Syntax. Der letzte exponierte Name in der Folge (A) kennzeichnet den Tabellenverweis, der ein Ziel der Optimierungsrichtlinie darstellt. Der erste exponierte Name in der Folge (A) kennzeichnet die Sicht, auf die in der ursprünglichen Anweisung direkt verwiesen wird. Die exponierten Namen in der Mitte der Folge ("Rick".V1) gehören zu den Sichtverweisen im Pfad von der direkten Sichtreferenz zum Zieltabellenverweis. Die im vorherigen Abschnitt beschriebenen Regeln für Verweise auf exponierte Namen in Optimierungsrichtlinien gelten für jeden einzelnen Schritt in der erweiterten Syntax.

Wäre der exponierte Name im Verweis für die Tabelle EMPLOYEE in der Sicht für alle Tabellen eindeutig, auf die direkt oder indirekt in der Anweisung verwiesen wird, wäre die erweiterte Namenssyntax nicht erforderlich.

Die erweiterte Syntax kann für einen beliebigen Zieltabellenverweis in der ursprünglichen Anweisung oder SQL-Funktion bzw. im ursprünglichen Trigger verwendet werden.

Angeben von Tabellenverweisen mit Korrelationsnamen in der optimierten Anweisung

Eine Optimierungsrichtlinie kann einen Tabellenverweis auch durch die eindeutigen Korrelationsnamen angeben, die dem Tabellenverweis in der optimierten Anweisung zugeordnet werden. Die optimierte Anweisung ist eine semantisch äquivalente Version der ursprünglichen Anweisung, die in der Optimierungsphase der Abfrageumschreibung bestimmt wird. Die optimierte Anweisung kann aus den EXPLAIN-Tabellen abgerufen werden. Das Attribut TABID einer Optimierungsrichtlinie dient zur Angabe der Tabellenverweise in der optimierten Anweisung. Beispiel:

Ursprüngliche Anweisung:

```
select s.s_name, s.s_address, s.s_phone, s.s_comment
  from sm_tpcd.parts p, sm_tpcd.suppliers s, sm_tpcd.partsupp ps
  where p_partkey = ps.ps_partkey and
    s.s_suppkey = ps.ps_suppkey and
    p.p_size = 39 and
    p.p_type = 'BRASS' and
    s.s_nation in ('MOROCCO', 'SPAIN') and
    ps.ps_supplycost = (select min(ps1.ps_supplycost)
                          from sm_tpcd.partsupp ps1, sm_tpcd.suppliers s1
                          where p.p_partkey = ps1.ps_partkey and
                            s1.s_suppkey = ps1.ps_suppkey and
                            s1.s_nation = s.s_nation)

<OPTGUIDELINES>
  <HSJOIN>
    <TBSCAN TABLE='S1'/>
    <IXSCAN TABID='Q2'/>
  </HSJOIN>
</OPTGUIDELINES>
```

Optimierte Anweisung:

```
select q6.s_name as "S_NAME", q6.s_address as "S_ADDRESS",
    q6.s_phone as "S_PHONE", q6.s_comment as "S_COMMENT"
  from (select min(q4.$c0)
          from (select q2.ps_supplycost
                  from sm_tpcd.suppliers as q1, sm_tpcd.partsupp as q2
                  where q1.s_nation = 'MOROCCO' and
                    q1.s_suppkey = q2.ps_suppkey and
                    q7.p_partkey = q2.ps_partkey
                  ) as q3
          ) as q4, sm_tpcd.partsupp as q5, sm_tpcd.suppliers as q6,
              sm_tpcd.parts as q7
  where p_size = 39 and
    q5.ps_supplycost = q4.$c0 and
    q6.s_nation in ('MOROCCO', 'SPAIN') and
    q7.p_type = 'BRASS' and
    q6.s_suppkey = q5.ps_suppkey and
    q7.p_partkey = q5.ps_partkey
```

Diese Optimierungsrichtlinie beinhaltet eine Anforderung für Hash-Joins, bei der die Tabelle SUPPLIERS im verschachtelten Subselect die äußere Tabelle darstellt, wie im Zugriffsanforderungselement TBSCAN angegeben, und die Tabelle PARTSUPP im verschachtelten Subselect die innere Tabelle, wie im Zugriffsanforderungselement IXSCAN angegeben. Das Zugriffsanforderungselement TBSCAN verwendet das Attribut TABLE, um den Verweis auf die Tabelle SUPPLIERS mit dem entsprechenden exponierten Namen in der ursprünglichen Anweisung zu kennzeichnen. Das Zugriffsanforderungselement IXSCAN dagegen verwendet das Attribut TABID, um den Verweis auf die Tabelle PARTSUPP mit dem eindeutigen Korrelationsnamen zu kennzeichnen, der dem Tabellenverweis in der optimierten Anweisung zugeordnet ist.

Wenn in einer Optimierungsrichtlinie sowohl das Attribut TABLE als auch das Attribut TABID angegeben sind, müssen sie sich auf denselben Tabellenverweis beziehen. Ansonsten wird die Optimierungsrichtlinie ignoriert.

Anmerkung: Zurzeit gibt es keine Garantie, dass Korrelationsnamen in der optimierten Anweisung bei einem Upgrade auf ein neues Release des DB2-Produkts stabil bleiben.

Mehrdeutige Tabellenverweise

Eine Optimierungsrichtlinie gilt als ungültig und wird nicht angewendet, wenn sie mehreren exponierten oder erweiterten Namen entspricht. Beispiel:

```
create view v1 as
  (select * from employee
     where salary > (select avg(salary) from employee)

select * from v1
  where deptno in ('M62', 'M63')

<OPTGUIDE>
  <IXSCAN TABLE='V1/EMPLOYEE'/>
</OPTGUIDE>
```

Für das Optimierungsprogramm ist das Zugriffsanforderungselement IXSCAN nicht eindeutig, da der exponierte Name EMPLOYEE innerhalb der Definition der Sicht V1 nicht eindeutig definiert ist.

Zur Sicherstellung eindeutiger Namen kann die Sicht so umgeschrieben werden, dass sie eindeutige Korrelationsnamen verwendet, oder es kann das Attribut TABID verwendet werden. Tabellenverweise, die im Attribut TABID angegeben werden, sind nie mehrdeutig, da alle Korrelationsnamen in der optimierten Anweisung eindeutig sind.

Unverträgliche Optimierungsrichtlinien

Der gleiche Tabellenverweis kann nicht von mehreren Optimierungsrichtlinien angegeben werden. Beispiel:

```
<OPTGUIDELINES>
  <IXSCAN TABLE='"Tpcd".PARTS' INDEX='I_PTYPE'/>
  <IXSCAN TABLE='"Tpcd".PARTS' INDEX='I_SIZE'/>
</OPTGUIDELINES>
```

Jedes einzelne Element IXSCAN verweist auf die Tabelle "Tpcd".PARTS im Hauptsubselect.

Wenn sich mehrere Richtlinien auf dieselbe Tabelle beziehen, wird nur die erste angewendet. Alle anderen Richtlinien werden unter Rückgabe eines Fehlers ignoriert.

Es kann nur ein Element INLIST2JOIN zur Anforderung einer Abfrageumschreibung auf Vergleichselementebene pro Abfrage aktiviert werden. Das folgende Beispiel veranschaulicht eine nicht unterstützte Optimierungsrichtlinie für das Umschreiben von Abfragen, bei der zwei Vergleichselemente mit IN-Listen auf Vergleichselementebene aktiviert sind. Beide Richtlinien werden ignoriert und es wird eine Warnung zurückgegeben.

```
<OPTGUIDELINES>
  <INLIST2JOIN TABLE='P' COLUMN='P_SIZE'/>
 <INLIST2JOIN TABLE='P' COLUMN='P_TYPE'/>
</OPTGUIDELINES>
```

Überprüfen, ob die Optimierungsrichtlinien verwendet wurden:

Das Optimierungsprogramm unternimmt jeden Versuch, die in einem Optimierungsprofil angegebenen Optimierungsrichtlinien einzuhalten. Das Optimierungsprogramm kann jedoch ungültige oder nicht anwendbare Richtlinien zurückweisen.

Vorbereitende Schritte

Bevor die EXPLAIN-Funktion verwendet werden kann, müssen EXPLAIN-Tabellen vorhanden sein. Die DDL-Anweisungen (DDL = Data Definition Language, Datendefinitionssprache) zur Erstellung der EXPLAIN-Tabellen sind in der Datei `EXPLAIN.DDL` enthalten, die sich im Unterverzeichnis `misc` des Verzeichnisses `sqllib` befindet.

Informationen zu diesem Vorgang

Führen Sie die folgenden Schritte aus, um zu prüfen, ob eine gültige Optimierungsrichtlinie verwendet wurde:

Vorgehensweise

1. Geben Sie die EXPLAIN-Anweisung für die Anweisung ein, für die die Richtlinien gelten. Wenn eine Optimierungsrichtlinie für die Anweisung durch ein Optimierungsprofil wirksam war, wird der Name des Optimierungsprofils als

Argument des Operators RETURN in der Tabelle EXPLAIN_ARGUMENT ausgewiesen. Und wenn die Optimierungsrichtlinie eine eingebettete SQL-Optimierungsrichtlinie bzw. ein Anweisungsprofil enthielt, die der aktuellen Anweisung entsprach, wird der Name des Anweisungsprofils als Argument eines Operators RETURN ausgegeben. Die Typen dieser beiden neuen Argumentwerte sind OPT_PROF und STMTPROF.

2. Untersuchen Sie die Ergebnisse der mit EXPLAIN bearbeiteten Anweisung. Die folgende Abfrage auf die EXPLAIN-Tabellen kann modifiziert werden, um den Namen des Optimierungsprofils und den Namen des Anweisungsprofils für Ihre bestimmte Kombination aus EXPLAIN_REQUESTER, EXPLAIN_TIME, SOURCE_NAME, SOURCE_VERSION und QUERYNO zurückzugeben:

```
SELECT VARCHAR(B.ARGUMENT_TYPE,   9) as TYPE,
            VARCHAR(B.ARGUMENT_VALUE, 24) as VALUE

   FROM   EXPLAIN_STATEMENT A, EXPLAIN_ARGUMENT B

   WHERE  A.EXPLAIN_REQUESTER = 'SIMMEN'
       AND  A.EXPLAIN_TIME      = '2003-09-08-16.01.04.108161'
       AND  A.SOURCE_NAME       = 'SQLC2E03'
       AND  A.SOURCE_VERSION    = ''
       AND  A.QUERYNO           = 1

       AND  A.EXPLAIN_REQUESTER = B.EXPLAIN_REQUESTER
       AND  A.EXPLAIN_TIME      = B.EXPLAIN_TIME
       AND  A.SOURCE_NAME       = B.SOURCE_NAME
       AND  A.SOURCE_SCHEMA     = B.SOURCE_SCHEMA
       AND  A.SOURCE_VERSION    = B.SOURCE_VERSION
       AND  A.EXPLAIN_LEVEL     = B.EXPLAIN_LEVEL
       AND  A.STMTNO            = B.STMTNO
       AND  A.SECTNO            = B.SECTNO

       AND  A.EXPLAIN_LEVEL     = 'P'

       AND  (B.ARGUMENT_TYPE  = 'OPT_PROF' OR ARGUMENT_TYPE = 'STMTPROF')
       AND  B.OPERATOR_ID = 1
```

Wenn die Optimierungsrichtlinie aktiv ist und die mit EXPLAIN-bearbeitete Anweisung der Anweisung entspricht, die im Element STMTKEY (Anweisungsschlüssel) der Optimierungsrichtlinie enthalten ist, liefert eine Abfrage ähnlich dem vorhergehenden Beispiel eine Ausgabe ähnlich der nachfolgend dargestellten Ausgabe. Der Wert des STMTPROF-Arguments ist derselbe wie der des ID-Attributs im Element STMTPROFILE.

```
TYPE      VALUE
--------- -------------------------
OPT_PROF  NEWTON.PROFILE1
STMTPROF  Guidelines for TPCD Q9
```

XML-Schema für Optimierungsprofile und -richtlinien:

Aktuelles Schema für Optimierungsprofile:

Der gültige Inhalt eines Optimierungsprofils für ein bestimmtes DB2-Release wird durch ein XML-Schema beschrieben, das als aktuelles Optimierungsprofilschema (COPS - Current Optimization Profile Schema) bezeichnet wird. Ein Optimierungsprofil gilt nur für DB2 Database für Linux-, UNIX- und Windows-Server.

Nachfolgend wird das aktuelle Optimierungsprofil (COPS) für das aktuelle Release des DB2-Produkts dargestellt. Das aktuelle Optimierungsprofil ist außerdem in der Datei `DB2OptProfile.xsd` enthalten, die sich im Unterverzeichnis `misc` des Verzeichnisses `sqllib` befindet.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified" version="1.0">
<!--*********************************************************************************************-->
<!-- Lizensiertes Material - Eigentum von IBM                                                    -->
<!-- (C) Copyright International Business Machines Corporation 2009. Alle Rechte vorbehalten.-->
<!-- U.S. Government Users Restricted Rights;  Use, duplication or disclosure restricted by  -->
<!-- GSA ADP Schedule Contract with IBM Corp.                                                    -->
<!--*********************************************************************************************-->
<!--*********************************************************************************************-->
<!-- Definition des aktuellen Optimierungsprofilschemas für Version Version 9.7.0.0          -->
<!--                                                                                             -->
<!--   Ein Optimierungsprofil setzt sich aus folgenden Abschnitten zusammen:                 -->
<!--                                                                                             -->
<!--    + Ein Abschnitt für globale Optimierungsrichtlinien (maximal einer), in dem          -->
<!--      Optimierungsrichtlinien für alle Anweisungen definiert werden, für die das         -->
<!--      Optimierungsprofil wirksam ist.                                                    -->
<!--                                                                                             -->
<!--    +  Null oder mehr Anweisungsprofilabschnitte, die jeweils Optimierungsrichtlinien    -->
<!--       für eine bestimmte Anweisung definieren, für die das Optimierungsprofil           -->
<!--       wirksam ist.                                                                      -->
<!--                                                                                             -->
<!-- Das Attribut VERSION gibt die Version dieses Optimierungsprofilschemas an.              -->
<!--                                                                                             -->
<!--*********************************************************************************************-->
<xs:element name="OPTPROFILE">
   <xs:complexType>
      <xs:sequence>
         <!-- Abschnitt für globale Optimierungsrichtlinien. Nur ein solcher Abschnitt ist zulässig. -->
         <xs:element name="OPTGUIDELINES" type="globalOptimizationGuidelinesType" minOccurs="0"/>
         <!-- Anweisungsprofilabschnitt. Null oder mehr Abschnitte zulässig -->
                  <xs:element name="STMTPROFILE" type="statementProfileType" minOccurs="0" maxOccurs="unbounded"/>
      </xs:sequence>
      <!-- Attribut VERSION ist gegenwärtig optional. -->
      <xs:attribute name="VERSION" use="optional"/>
   </xs:complexType>
</xs:element>

<!--*********************************************************************************************-->
<!--  In dieser Version unterstützte globale Optimierungsrichtlinien:                        -->
<!--  + MQTOptimizationChoices-Elemente beeinflussen die vom Optimierungsprogramm in         -->
<!--    Betracht gezogenen MQTs (Materialized Query Tables).                                 -->
<!--  + computationalPartitionGroupOptimizationsChoices-Elemente können die Optimierungen    -->
<!--    im Hinblick auf die Neupartitionierung mit Kurznamen beeinflussen.                   -->
<!--  + Allgemeine Anforderungen beeinflussen den Suchbereich, der die alternativen         -->
<!--    Abfrageumsetzungen, Zugriffsmethoden, Joinmethoden, Joinreihenfolgen und andere      -->
<!--    Optimierungen definiert, die von Optimierungsprogramm und Compiler erwägt werden.    -->
<!--  + MQT-Durchsetzungsanforderungen geben semantisch vergleichbare MQTs an, deren         -->
<!--    Verwendung in Zugriffsplänen unabhängig vom geschätzten Ausführungsaufwand           -->
<!--    durchgesetzt werden soll.                                                            -->
<!-- *********************************************************************************************-->
<xs:complexType name="globalOptimizationGuidelinesType">
   <xs:sequence>
      <xs:group ref="MQTOptimizationChoices" />
      <xs:group ref="computationalPartitionGroupOptimizationChoices" />
      <xs:group ref="generalRequest"/>
      <xs:group ref="mqtEnforcementRequest" />
   </xs:sequence>
</xs:complexType>
<!-- *********************************************************************************************-->
<!--  Elemente zur Beeinflussung der Optimierung mit MQTs (Materialized Query Tables):       -->
<!--                                                                                             -->
<!--  + MQTOPT - Kann zur Inaktivierung der Optimierung mit MQTs verwendet werden.           -->
<!--           Bei Inaktivierung zieht das Optimierungsprogramm keine MQTs zur Optimierung   -->
<!--           der Anweisung in Betracht.                                                    -->
<!--                                                                                             -->
<!--  + MQT - Mehrere dieser Elemente können angegeben werden. Jedes gibt eine MQT an,       -->
<!--     die bei der Optimierung der Anweisung berücksichtigt werden sollte. Nur            -->
<!--     angegebene MQTs werden berücksichtigt.                                              -->
<!--                                                                                             -->
<!--*********************************************************************************************-->
<xs:group name="MQTOptimizationChoices">
   <xs:choice>
      <xs:element name="MQTOPT" minOccurs="0" maxOccurs="1">
         <xs:complexType>
            <xs:attribute name="OPTION" type="optionType" use="optional"/>
         </xs:complexType>
      </xs:element>
      <xs:element name="MQT" minOccurs="0" maxOccurs="unbounded">
         <xs:complexType>
```

```
            <xs:attribute name="NAME" type="xs:string" use="required"/>
         </xs:complexType>
      </xs:element>
   </xs:choice>
</xs:group>
<!-- ********************************************************************************************-->
<!--  Elemente zur Beeinflussung der Rechenpartitionsgruppenoptimierung (CPG-Optimierung):  -->
<!--                                                                                            -->
<!--   + PARTOPT - Kann zur Inaktivierung der Rechenpartitionsgruppenoptimierung verwendet  -->
<!--               werden, die dazu dient, Eingaben an Join-, Aggregations-, und UNION-     -->
<!--               Operationen dynamisch neu zu verteilen, wenn diese Eingaben Ergebnisse   -->
<!--               von Fernabfragen sind.                                                   -->
<!--                                                                                        -->
<!--    + PART - Definiert die in CPG-Optimierungen zu verwendenden Partitionsgruppen.      -->
<!--                                                                                        -->
<!-- ********************************************************************************************-->
<xs:group name="computationalPartitionGroupOptimizationChoices">
   <xs:choice>
      <xs:element name="PARTOPT" minOccurs="0" maxOccurs="1">
         <xs:complexType>
            <xs:attribute name="OPTION" type="optionType" use="optional"/>
      </xs:complexType>
      </xs:element>
      <xs:element name="PART" minOccurs="0" maxOccurs="1">
         <xs:complexType>
            <xs:attribute name="NAME" type="xs:string" use="required"/>
         </xs:complexType>
      </xs:element>
   </xs:choice>
</xs:group>
<!-- ********************************************************************************************-->
<!--  Definition eines Anweisungsprofils.                                                        -->
<!--  Besteht aus einem Anweisungsschlüssel und Optimierungsrichtlinien.                         -->
<!--  Der Anweisungsschlüssel enthält semantische Informationen, die zur Angabe der Anweisung-->
<!--  dienen, für die die Optimierungsrichtlinien gelten. Das optionale Attribut ID stellt   -->
<!--  einen Namen für das Anweisungsprofil in EXPLAIN-Ausgaben bereit.                           -->
<!-- ********************************************************************************************-->
<xs:complexType name="statementProfileType">
   <xs:sequence>
      <!-- Anweisungsschlüsselelement -->
      <xs:element name="STMTKEY" type="statementKeyType"/>
      <xs:element name="OPTGUIDELINES" type="optGuidelinesType"/>
   </xs:sequence>
   <!-- Attribut ID. Dient in EXPLAIN-Ausgaben zur Angabe, dass das Anweisungsprofil verwendet wurde. -->
   <xs:attribute name="ID" type="xs:string" use="optional"/>
</xs:complexType>
<!--********************************************************************************************-->
<!--  Definition des Anweisungsschlüssels. Der Anweisungsschlüssel stellt semantische       -->
<!--  Informationen zur Angabe der Anweisung bereit, für die die Optimierungsrichtlinien    -->
<!--  gelten.                                                                               -->
<!-- Der Anweisungsschlüssel besteht aus:                                                   -->
<!--      + Anweisungstext (wie in der Anwendung geschrieben)                               -->
<!--      + Standardschema (zur Auflösung unqualifizierter Tabellennamen in der Anweisung)  -->
<!--      + Funktionspfad (zur Auflösung unqualifizierter Typen und Funktionen in der       -->
<!--        Anweisung)                                                                      -->
<!-- Der Anweisungstext wird in Form von Elementdaten bereitgestellt, während Standardschema-->
<!--  und Funktionspfad über die Elemente SCHEMA und FUNCPATH angegeben werden.             -->
<!--********************************************************************************************-->
<xs:complexType name="statementKeyType" mixed="true">
   <xs:attribute name="SCHEMA" type="xs:string" use="optional"/>
   <xs:attribute name="FUNCPATH" type="xs:string" use="optional"/>
</xs:complexType>

<!--********************************************************************************************-->
<!--                                                                                            -->
<!-- Optimierungsrichtlinienelemente können aus allgemeinen Anforderungen, Umschreibe-      -->
<!-- anforderungen, Zugriffsanforderungen oder Joinanforderungen ausgewählt werden.         -->
<!--                                                                                            -->
<!-- Allgemeine Anforderungen beeinflussen den Suchbereich, der die alternativen            -->
<!-- Abfrageumsetzungen, Zugriffsmethoden, Joinmethoden, Joinreihenfolgen und andere        -->
<!-- Optimierungen definiert, die vom Optimierungsprogramm in Betracht gezogen werden.      -->
<!--                                                                                            -->
<!-- Umschreibanforderungen beeinflussen die Abfrageumsetzungen, die bei der Bestimmung     -->
<!-- der optimierten Anweisung verwendet werden.                                            -->
<!--                                                                                            -->
<!-- Zugriffsanforderungen beeinflussen die Zugriffsmethoden, die vom aufwandsbasierten     -->
<!-- Optimierungsprogramm in Betracht gezogen werden. Joinanforderungen beeinflussen        -->
<!-- die Joinmethoden und die Joinreihenfolge im Ausführungsplan.                           -->
<!--                                                                                            -->
```

```
<!-- MQT-Durchsetzungsanforderungen geben semantisch vergleichbare MQTs an, deren          -->
<!--   Verwendung in Zugriffsplänen unabhängig vom geschätzten Ausführungsaufwand          -->
<!--   durchgesetzt werden soll.                                                           -->
<!--                                                                                       -->
<!--*****************************************************************************************-->
<xs:element name="OPTGUIDELINES" type="optGuidelinesType"/>
<xs:complexType name="optGuidelinesType">
   <xs:sequence>
      <xs:group ref="generalRequest" minOccurs="0" maxOccurs="1"/>
      <xs:choice maxOccurs="unbounded">
         <xs:group ref="rewriteRequest" />
         <xs:group ref="accessRequest"/>
         <xs:group ref="joinRequest"/>
         <xs:group ref="mqtEnforcementRequest"/>
      </xs:choice>
   </xs:sequence>
</xs:complexType>
<!--***************************************************************************************** -->
<!-- Auswahlmöglichkeiten für allgemeine Anforderungselemente.                              -->
<!-- REOPT kann zum Überschreiben der Einstellung der Bindeoption REOPT verwendet werden. -->
<!-- DPFXMLMOVEMENT kann beim Versetzen von XML-Dokumenten auf andere                       -->
<!-- Datenbankpartitionen zum Beeinflussen des Optimierungsprogrammplans verwendet werden.-->
<!-- Mögliche Werte: NONE, REFERENCE und COMBINATION. Standardwert: NONE.                   -->
<!--***************************************************************************************** -->
<xs:group name="generalRequest">
   <xs:sequence>
      <xs:element name="REOPT" type="reoptType" minOccurs="0" maxOccurs="1"/>
      <xs:element name="DEGREE" type="degreeType" minOccurs="0" maxOccurs="1"/>
      <xs:element name="QRYOPT" type="qryoptType" minOccurs="0" maxOccurs="1"/>
      <xs:element name="RTS" type="rtsType" minOccurs="0" maxOccurs="1"/>
      <xs:element name="DPFXMLMOVEMENT" type="dpfXMLMovementType" minOccurs="0" maxOccurs="1"/>
   </xs:sequence>
</xs:group>
<!--*****************************************************************************************-->
<!--  Auswahlmöglichkeiten für Umschreibanforderungselemente.                           -->
<!--*****************************************************************************************-->
<xs:group name="rewriteRequest">
   <xs:sequence>
      <xs:element name="INLIST2JOIN" type="inListToJoinType" minOccurs="0"/>
      <xs:element name="SUBQ2JOIN" type="subqueryToJoinType" minOccurs="0"/>
      <xs:element name="NOTEX2AJ" type="notExistsToAntiJoinType" minOccurs="0"/>
      <xs:element name="NOTIN2AJ" type="notInToAntiJoinType" minOccurs="0"/>
   </xs:sequence>
</xs:group>
<!--***************************************************************************************** -->
<!--  Auswahlmöglichkeiten für Zugriffsanforderungselemente.                             -->
<!--  TBSCAN - Zugriffsanforderungselement für Tabellensuche                             -->
<!--  IXSCAN - Zugriffsanforderungselement für Indexsuche                                -->
<!--  LPREFETCH - Zugriffsanforderungselement für Vorablesezugriff über Listen           -->
<!--  IXAND - Zugriffsanforderungselement mit logischem Verknüpfen von Indizes über AND  -->
<!--  IXOR - Zugriffsanforderungselement mit logischem Verknüpfen von Indizes über OR    -->
<!--  XISCAN - Zugriffsanforderungselement für Zugriffe über XML-Indizes                 -->
<!--  XANDOR - Zugriffsanforderungselement XANDOR                                        -->
<!--  ACCESS - Gibt an, dass das Optimierungsprogramm die Zugriffsmethode für die Tabelle -->
<!--           auswählen soll.                                                           -->
<!--***************************************************************************************** -->
<xs:group name="accessRequest">
   <xs:choice>
      <xs:element name="TBSCAN" type="tableScanType"/>
      <xs:element name="IXSCAN" type="indexScanType"/>
      <xs:element name="LPREFETCH" type="listPrefetchType"/>
      <xs:element name="IXAND" type="indexAndingType"/>
      <xs:element name="IXOR" type="indexOringType"/>
      <xs:element name="XISCAN" type="indexScanType"/>
      <xs:element name="XANDOR" type="XANDORType"/>
      <xs:element name="ACCESS" type="anyAccessType"/>
   </xs:choice>
</xs:group>
<!--***************************************************************************************** -->
<!-- Auswahlmöglichkeiten für Joinanforderungslemente.                                      -->
<!-- NLJOIN - Anforderungselement für Joins mit Verschachtelungsschleife (Nested Loop)    -->
<!-- MSJOIN - Anforderungselement für Mischjoins (Sort-Merge)                           -->
<!-- HSJOIN - Anforderungselement für Hash-Joins                                        -->
<!-- JOIN - Gibt an, dass das Optimierungsprogramm die Joinmethode auswählen soll.        -->
<!--***************************************************************************************** -->
<xs:group name="joinRequest">
   <xs:choice>
      <xs:element name="NLJOIN" type="nestedLoopJoinType"/>
      <xs:element name="HSJOIN" type="hashJoinType"/>
```

```
      <xs:element name="MSJOIN" type="mergeJoinType"/>
      <xs:element name="JOIN" type="anyJoinType"/>
   </xs:choice>
</xs:group>
<!--*****************************************************************************************-->
<!-- Anforderungselement für MQT-Durchsetzung                                              -->
<!-- MQTENFORCE - Mit diesem Element können semantisch vergleichbare MQTs angegeben      -->
<!-- werden, deren Verwendung in Zugriffsplänen unabhängig vom geschätzten               -->
<!-- Ausführungsaufwand des Optimierungsprogramms durchgesetzt werden soll.              -->
<!-- MQTs können direkt über das Attribut NAME oder allgemein über das Attribut          -->
<!-- TYPE angegeben werden.                                                              -->
<!-- Es wird nur das erste gültige Attribut verwendet, alle folgenden werden ignoriert.  -->
<!-- Da das Element mehrmals angegeben werden kann, können gleichzeitig mehrere MQTs     -->
<!-- durchgesetzt werden.                                                                -->
<!-- Dabei ist zu beachten, dass bei einem Konflikt beim Abgleich von zwei               -->
<!-- durchgesetzten MQTs für dieselbe Datenquelle (Basistabelle oder abgeleitete Tabelle)-->
<!-- die Entscheidung für eine MQT nach vorhandenen Zuteilungsrichtlinien (heuristisch   -->
<!-- oder nach Ausführungsaufwand) gefällt wird.                                         -->
<!-- Außerdem gilt, dass diese Anforderung alle in einem Profil angegebenen              -->
<!-- MQT-Optimierungsoptionen überschreibt, d. h. die MQT wird auch durchgesetzt, wenn   -->
<!-- MQTOPT mit DISABLE definiert ist und wenn die angegebene(n) MQT(s) in der über      -->
<!-- MQT-Elemente angegebenen Auswahlliste nicht enthalten ist/sind.                     -->
<!--*****************************************************************************************-->
<xs:group name="mqtEnforcementRequest">
   <xs:sequence>
      <xs:element name="MQTENFORCE" type="mqtEnforcementType" minOccurs="0" maxOccurs="unbounded"/>
   </xs:sequence>
</xs:group>
<!--***************************************************************************************** -->
<!--  Allgemeines REOPT-Anforderungselement. Kann die REOPT-Einstellung auf Paket-           -->
<!--  Datenbank- und Datenbankmanagerebene überschreiben.                                    -->
<!--***************************************************************************************** -->
<xs:complexType name="reoptType">
   <xs:attribute name="VALUE" use="required">
      <xs:simpleType>
         <xs:restriction base="xs:string">
            <xs:enumeration value="ONCE"/>
            <xs:enumeration value="ALWAYS"/>
         </xs:restriction>
      </xs:simpleType>
   </xs:attribute>
</xs:complexType>
<!--*****************************************************************************************-->
<!-- Allgemeines RTS-Anforderungselement zum Aktivieren, Inaktivieren der Echtzeitstatstik-    -->
<!-- erfassung bzw. zur Festlegung eines Zeitbudgets für diese Erfassung.                     -->
<!-- Attribut OPTION: Aktivieren oder Inaktivieren der Echtzeitstatistiken.                   -->
<!-- Attribut TIME: Zeitbudget in Millisekunden für Echtzeitstatistikerfassung.               -->
<!--*****************************************************************************************-->
<xs:complexType name="rtsType">
   <xs:attribute name="OPTION" type="optionType" use="optional" default="ENABLE"/>
   <xs:attribute name="TIME" type="xs:nonNegativeInteger" use="optional"/>
</xs:complexType>
<!--*****************************************************************************************-->
<!--   Definition einer Anforderung zum Umschreiben von "IN-Listen in Joins"                 -->
<!--   Attribut OPTION ermöglicht Aktivierung oder Inaktivierung der Alternative.            -->
<!--   Attribut TABLE ermöglicht Anforderung gewünschter IN-Listenvergleichselemente, die auf -->
<!--   einen bestimmten Tabellenverweis angewendet werden. Attribut COLUMN ermöglicht        -->
<!--   Anforderung für ein bestimmtes IN-Listenvergleichselement.                            -->
<!--*****************************************************************************************-->
<xs:complexType name="inListToJoinType">
   <xs:attribute name="OPTION" type="optionType" use="optional" default="ENABLE"/>
   <xs:attribute name="TABLE" type="xs:string" use="optional"/>
   <xs:attribute name="COLUMN" type="xs:string" use="optional"/>
</xs:complexType>
<!--*****************************************************************************************-->
<!--  Definition einer Anforderung zur Umschreibung von Unterabfragen in Joins               -->
<!--  Das Attribut OPTION ermöglicht die Aktivierung oder Inaktivierung der Alternative.     -->
<!--*****************************************************************************************-->
<xs:complexType name="subqueryToJoinType">
   <xs:attribute name="OPTION" type="optionType" use="optional" default="ENABLE"/>
</xs:complexType>
<!--*****************************************************************************************-->
<!--  Definition einer Anforderung zur Umschreibung von NOT EXISTS-Unterabfragen in Antijoins -->
<!--  Das Attribut OPTION ermöglicht die Aktivierung oder Inaktivierung der Alternative.     -->
<!--*****************************************************************************************-->
<xs:complexType name="notExistsToAntiJoinType">
   <xs:attribute name="OPTION" type="optionType" use="optional" default="ENABLE"/>
</xs:complexType>
<!--*****************************************************************************************-->
```

```
<!--  Definition einer Anforderung zur Umschreibung von NOT IN-Unterabfragen in Antijoins     -->
<!--  Das Attribut OPTION ermöglicht die Aktivierung oder Inaktivierung der Alternative.      -->
<!--*****************************************************************************************-->
<xs:complexType name="notInToAntiJoinType">
   <xs:attribute name="OPTION" type="optionType" use="optional" default="ENABLE"/>
</xs:complexType>
<!--*****************************************************************************************-->
<!-- Effektiv die Superklasse, aus der alle Zugriffsanforderungselemente übernehmen.          -->
<!-- Dieser Typ definiert zurzeit die Attribute TABLE und TABID, die zur Bindung einer        -->
<!-- Zugriffsanforderung an einen Tabellenverweis in der Abfrage verwendet werden können.     -->
<!-- Der Wert des Attributs TABLE dient zur Angabe eines Tabellenverweises durch Kennungen    -->
<!-- in der ursprünglichen SQL-Anweisung. Der Wert des Attributs TABID dient zur Angabe eines -->
<!-- Tabellenverweises durch den eindeutigen Korrelationsnamen, der durch die optimierte      -->
<!-- Anweisung bereitgestellt wird. Wenn TABLE und TABID zusammen angegeben werden, wird das  -->
<!-- Feld TABID ignoriert. Das Attribut FIRST gibt an, dass der Zugriff der erste Zugriff     -->
<!-- in der Joinreihenfolge für die FROM-Klausel sein soll.                                   -->
<!-- Das Attribut SHARING gibt an, dass der Zugriff für andere ähnliche gleichzeitige Zugriffe-->
<!-- sichtbar sein muss, die dadurch Pufferpoolseiten gemeinsam nutzen können. Das Attribut   -->
<!-- WRAPPING gibt an, dass beim Zugriff Umlauf möglich sein soll, sodass der Zugriff in der  -->
<!-- Mitte beginnen kann, um die gemeinsame Nutzung mit anderen gleichzeitigen Zugriffen zu   -->
<!-- verbessern. Das Attribut THROTTLE gibt an, dass beim Zugriff eine Drosselung möglich     -->
<!-- sein soll, wenn dies für andere gleichzeitige Zugriffe vorteilhaft ist. Das Attribut     -->
<!-- SHARESPEED gibt an, ob der Zugriff als schneller oder langsamer Zugriff gewertet         -->
<!-- werden soll und erleichtert so die Gruppierung mit gleichzeitigen Zugriffen.             -->
<!--***************************************************************************************** -->
<xs:complexType name="accessType" abstract="true">
   <xs:attribute name="TABLE" type="xs:string" use="optional"/>
   <xs:attribute name="TABID" type="xs:string" use="optional"/>
   <xs:attribute name="FIRST" type="xs:string" use="optional" fixed="TRUE"/>
   <xs:attribute name="SHARING" type="optionType" use="optional" default="ENABLE"/>
   <xs:attribute name="WRAPPING" type="optionType" use="optional" default="ENABLE"/>
   <xs:attribute name="THROTTLE" type="optionType" use="optional" default="ENABLE"/>
   <xs:attribute name="SHARESPEED" type="shareSpeed" use="optional"/>
</xs:complexType>
<!--*****************************************************************************************-->
<!-- Definition eines Zugriffsanforderungselements für Tabellensuche.                        -->
<!--*****************************************************************************************-->
<xs:complexType name="tableScanType">
   <xs:complexContent>
      <xs:extension base="accessType"/>
   </xs:complexContent>
</xs:complexType>
<!-- *****************************************************************************************-->
<!-- Definition eines Zugriffsanforderungselements für Indexsuche.                           -->
<!-- Der Indexname ist optional.                                                             -->
<!--***************************************************************************************** -->
<xs:complexType name="indexScanType">
   <xs:complexContent>
      <xs:extension base="accessType">
         <xs:attribute name="INDEX" type="xs:string" use="optional"/>
      </xs:extension>
   </xs:complexContent>
</xs:complexType>
<!--*****************************************************************************************-->
<!-- Definition eines Zugriffsanforderungselements für Vorablesezugriff über Listen.         -->
<!-- Der Indexname ist optional.                                                             -->
<!--***************************************************************************************** -->
<xs:complexType name="listPrefetchType">
   <xs:complexContent>
      <xs:extension base="accessType">
         <xs:attribute name="INDEX" type="xs:string" use="optional"/>
      </xs:extension>
   </xs:complexContent>
</xs:complexType>
<!--*****************************************************************************************-->
<!-- Definition eines erweiterten Zugriffselements, das von IXAND- und ACCESS-               -->
<!-- Anforderungen genutzt wird.                                                             -->
<!-- Ein Index kann durch das Attribut INDEX angegeben werden. Mehrere Indizes können        -->
<!-- durch INDEX-Elemente angegeben werden. Die Angabe von INDEX-Elementen überschreibt      -->
<!-- die Attributangabe. Wenn nur ein Index angegeben wird, verwendet das Optimierungs-      -->
<!-- programm den Index als ersten Index bei der Zugriffsmethode mit AND-Verknüpfung         -->
<!-- von Indizes. Weitere Indizes werden nach Aufwandsberechnung gewählt. Wenn mehrere       -->
<!-- Indizes angegeben werden, verwendet das Optimierungsprogramm diese exakt in der         -->
<!-- angegebenen Reihenfolge. Wenn keine Indizes durch das Attribut INDEX bzw. durch         -->
<!-- INDEX-Elemente angegeben werden, wählt das Optimierungsprogramm alle Indizes nach       -->
<!-- Aufwandsberechnung aus.                                                                 -->
<!-- Erweiterung für XML-Unterstützung:                                                      -->
<!-- TYPE: Optionales Attribut. Zulässiger Wert: XMLINDEX. Wenn der Typ nicht angegeben      -->
<!-- ist, entscheidet das Optimierungsprogramm je nach Ausführungsaufwand.                   -->
```

```
<!-- ALLINDEXES: Optionales Attribut. Der zulässige Wert ist TRUE.                      -->
<!-- Standardwert: FALSE.                                                               -->
<!--*********************************************************************************** -->
<xs:complexType name="extendedAccessType">
   <xs:complexContent>
      <xs:extension base="accessType">
         <xs:sequence minOccurs="0">
            <xs:element name="INDEX" type="indexType" minOccurs="2" maxOccurs="unbounded"/>
         </xs:sequence>
         <xs:attribute name="INDEX" type="xs:string" use="optional"/>
         <xs:attribute name="TYPE" type="xs:string" use="optional" fixed="XMLINDEX"/>
         <xs:attribute name="ALLINDEXES" type="boolType" use="optional" fixed="TRUE"/>
      </xs:extension>
   </xs:complexContent>
</xs:complexType>
<!--*********************************************************************************--->
<!-- Definition eines Zugriffsanforderungselements für AND-Verknüpfung von Indizes.    -->
<!-- Erweiterung für XML-Unterstützung:                                                 -->
<!-- Alle Attribute und Elemente in 'extendedAccessType' sind eingeschlossen.           -->
<!-- Achtung: Die Option ALLINDEXES ist nur gültig, wenn für TYPE XMLINDEX definiert ist. -->
<!-- STARJOIN-IndexANDing: Die Angabe STARJOIN='TRUE' oder eines oder mehrerer         -->
<!-- NLJOIN-Elemente gibt die Anforderung zum logischen Verknüpfen von Indizes über AND   -->
<!-- als Anforderung für einen Sternjoin an. In diesem Fall sind folgende Punkte zu     -->
<!-- beachten:                                                                          -->
<!--   TYPE kann nicht XMLINDEX sein (d. h. ALLINDEXES kann nicht angegeben werden).    -->
<!--   Weder das Attribut INDEX noch INDEX-Elemente können angegeben werden.            -->
<!--   Das Attribut TABLE oder TABID gibt die Fakttabelle an.                           -->
<!--   Null oder mehr Semi-Joins können mit NLJOIN-Elementen angegeben werden.          -->
<!--   Wenn keine Semi-Joins angegeben werden, werden sie vom Optimierungsprogramm      -->
<!--   ausgewählt.                                                                      -->
<!--   Wenn nur ein Semi-Join angegeben wird, verwendet das Optimierungsprogramm dieses -->
<!--   als erstes Semi-Join und wählt die übrigen selbst aus.                           -->
<!--   Wenn mehrere Semi-Joins angegeben werden, verwendet das Optimierungsprogramm genau -->
<!--   diese Semi-Joins in der angegebenen Reihenfolge.                                 -->
<!--*********************************************************************************** -->
<xs:complexType name="indexAndingType">
   <xs:complexContent>
      <xs:extension base="extendedAccessType">
         <xs:sequence minOccurs="0">
            <xs:element name="NLJOIN" type="nestedLoopJoinType" minOccurs="1" maxOccurs="unbounded"/>
         </xs:sequence>
         <xs:attribute name="STARJOIN" type="boolType" use="optional"/>
      </xs:extension>
   </xs:complexContent>
</xs:complexType>
<!--*********************************************************************************--->
<!-- Definition einer INDEX-Elementmethode. Indexgruppe ist optional. Bei Angabe       -->
<!-- müssen mindestens 2 angegeben werden.                                             -->
<!--*********************************************************************************** -->
<xs:complexType name="indexType">
   <xs:attribute name="IXNAME" type="xs:string" use="optional"/>
</xs:complexType>
<!--*********************************************************************************--->
<!-- Definition einer XANDOR-Zugriffsanforderungsmethode                               -->
<!--*********************************************************************************--->
<xs:complexType name="XANDORType">
   <xs:complexContent>
      <xs:extension base="accessType"/>
   </xs:complexContent>
</xs:complexType>
<!--*********************************************************************************--->
<!-- Zur Verwendung für den Zugriff auf abgeleitete Tabellen oder in anderen Fällen,   -->
<!-- in denen es nicht auf die Zugriffsmethode ankommt.                                -->
<!-- Erweiterung für XML-Unterstützung:                                                 -->
<!-- Alle Attribute und Elemente in 'extendedAccessType' sind eingeschlossen.           -->
<!-- Achtung: INDEX-Attribute/-Elemente und ALLINDEXES sind nur als Optionen gültig,   -->
<!-- wenn TYPE mit XMLINDEX definiert ist.                                             -->
<!--*********************************************************************************--->
<xs:complexType name="anyAccessType">
   <xs:complexContent>
      <xs:extension base="extendedAccessType"/>
   </xs:complexContent>
</xs:complexType>
<!--*********************************************************************************--->
<!-- Definition eines Zugriffs mit logischer OR-Verknüpfung von Indizes                -->
<!-- Angabe weiterer Details (z. B. Indizes) nicht möglich. Optimierungsprogramm wählt -->
<!-- die Details auf der Grundlage einer Aufwandsberechnung aus.                       -->
<!--*********************************************************************************** -->
<xs:complexType name="indexOringType">
```

```
   <xs:complexContent>
      <xs:extension base="accessType"/>
   </xs:complexContent>
</xs:complexType>
<!--*****************************************************************************************-->
<!-- Effektiv die Superklasse, aus der Joinanforderungselemente übernehmen.                  -->
<!-- Dieser Typ definiert zurzeit Joinelementeingaben und das Attribut FIRST.                -->
<!-- Eine Joinanforderung muss genau 2 verschachtelte Unterelemente enthalten. Die Unter- -->
<!-- elemente können entweder eine Zugriffsanforderung oder eine weitere Joinanforderung  -->
<!-- sein. Das erste Unterelement stellt die äußere Tabelle der Joinoperation dar, das    -->
<!-- zweite die innere Tabelle. Das Attribut FIRST gibt an, dass das Joinergebnis der     -->
<!-- erste Join in Beziehung zu anderen Tabellen in derselben FROM-Klausel sein soll.     -->
<!--***************************************************************************************** -->
<xs:complexType name="joinType" abstract="true">
   <xs:choice minOccurs="2" maxOccurs="2">
      <xs:group ref="accessRequest"/>
      <xs:group ref="joinRequest"/>
   </xs:choice>
   <xs:attribute name="FIRST" type="xs:string" use="optional" fixed="TRUE"/>
</xs:complexType>
<!--***************************************************************************************** -->
<!-- Definition einer Zugriffsanforderung für Join mit Verschachtelungsschleife.          -->
<!-- Unterklasse von 'joinType'. Fügt keine Elemente oder Attribute hinzu.                -->
<!--***************************************************************************************** -->
<xs:complexType name="nestedLoopJoinType">
   <xs:complexContent>
      <xs:extension base="joinType"/>
   </xs:complexContent>
</xs:complexType>
<!--*****************************************************************************************-->
<!-- Definition einer Zugriffsanforderung mit Mischjoin.                                  -->
<!-- Unterklasse von 'joinType'. Fügt keine Elemente oder Attribute hinzu.                -->
<!--***************************************************************************************** -->
<xs:complexType name="mergeJoinType">
   <xs:complexContent>
      <xs:extension base="joinType"/>
   </xs:complexContent>
</xs:complexType>
<!--***************************************************************************************** -->
<!-- Definition einer Zugriffsanforderung mit Hash-Join.                                  -->
<!-- Unterklasse von 'joinType'. Fügt keine Elemente oder Attribute hinzu.                -->
<!--***************************************************************************************** -->
<xs:complexType name="hashJoinType">
   <xs:complexContent>
      <xs:extension base="joinType"/>
   </xs:complexContent>
</xs:complexType>
<!--***************************************************************************************** -->
<!-- Der Typ 'anyJoinType' ist eine Unterklasse des binären Joins. Erweitert diesen       -->
<!-- in keiner Weise. Fügt keine Elemente oder Attribute hinzu.                           -->
<!--***************************************************************************************** -->
<xs:complexType name="anyJoinType">
   <xs:complexContent>
      <xs:extension base="joinType"/>
   </xs:complexContent>
</xs:complexType>
<!--*****************************************************************************************-->
<!-- Das Element MQTENFORCE kann mit einem der folgenden Attribute angegeben werden:      -->
<!-- NAME: Geben Sie den MQT-Namen direkt als Wert für dieses Attribut an.                -->
<!-- TYPE: Geben Sie hiermit den Typ der MQTs an, die durchgesetzt werden sollen.         -->
<!-- Beachten Sie, dass nur der Wert für das erste gefundene gültige Attribut verwendet   -->
<!-- wird. Alle nachfolgenden Attribute werden ignoriert.                                 -->
<!--*****************************************************************************************-->
<xs:complexType name="mqtEnforcementType">
   <xs:attribute name="NAME" type="xs:string"/>
   <xs:attribute name="TYPE" type="mqtEnforcementTypeType"/>
</xs:complexType>
<!--*********************************************************************************************-->
<!-- Zulässige Werte für das Attribut TYPE eines Elements MQTENFORCE:                        -->
<!-- NORMAL: Alle semantisch vergleichbaren MQTs, replizierte ausgenommen, durchsetzen       -->
<!-- REPLICATED: Nur die Verwendung semantisch vergleichbarer replizierter MQTS durchsetzen  -->
<!-- ALL: Verwendung der semantisch vergleichbaren MQTs durchsetzen                          -->
<!--***************************************************************************************** -->
<xs:simpleType name="mqtEnforcementTypeType">
   <xs:restriction base="xs:string">
      <xs:enumeration value="NORMAL"/>
      <xs:enumeration value="REPLICATED"/>
      <xs:enumeration value="ALL"/>
   </xs:restriction>
```

```
</xs:simpleType>
<!--*****************************************************************************************-->
<!-- Zulässige Werte für boolesches Attribut                                                 -->
<!--***************************************************************************************** -->
<xs:simpleType name="boolType">
   <xs:restriction base="xs:string">
      <xs:enumeration value="TRUE"/>
      <xs:enumeration value="FALSE"/>
   </xs:restriction>
</xs:simpleType>
<!--*****************************************************************************************-->
<!-- Zulässige Werte für Attribut OPTION                                                     -->
<!--***************************************************************************************** -->
<xs:simpleType name="optionType">
   <xs:restriction base="xs:string">
      <xs:enumeration value="ENABLE"/>
      <xs:enumeration value="DISABLE"/>
   </xs:restriction>
</xs:simpleType>
<!--*****************************************************************************************-->
<!-- Zulässige Werte für Attribut SHARESPEED                                                 -->
<!--***************************************************************************************** -->
<xs:simpleType name="shareSpeed">
   <xs:restriction base="xs:string">
      <xs:enumeration value="FAST"/>
      <xs:enumeration value="SLOW"/>
   </xs:restriction>
</xs:simpleType>
<!--*****************************************************************************************-->
<!-- Definition von 'qryoptType': Zulässig sind nur die Werte 0, 1, 2, 3, 5, 7 und 9.        -->
<!--*****************************************************************************************-->
<xs:complexType name="qryoptType">
   <xs:attribute name="VALUE" use="required">
      <xs:simpleType>
         <xs:restriction base="xs:string">
            <xs:enumeration value="0"/>
            <xs:enumeration value="1"/>
            <xs:enumeration value="2"/>
            <xs:enumeration value="3"/>
            <xs:enumeration value="5"/>
            <xs:enumeration value="7"/>
            <xs:enumeration value="9"/>
         </xs:restriction>
      </xs:simpleType>
   </xs:attribute>
</xs:complexType>
<!--*****************************************************************************************-->
<!-- Definition von 'degreeType': Beliebiger Wert zwischen 1 und 32767 bzw. die              -->
<!-- Zeichenfolgen ANY oder '-1'                                                             -->
<!--***************************************************************************************** -->
<xs:simpleType name="intStringType">
   <xs:union>
      <xs:simpleType>
         <xs:restriction base="xs:integer">
            <xs:minInclusive value="1"></xs:minInclusive>
            <xs:maxInclusive value="32767"></xs:maxInclusive>
         </xs:restriction>
      </xs:simpleType>
      <xs:simpleType>
         <xs:restriction base="xs:string">
            <xs:enumeration value="ANY"/>
            <xs:enumeration value="-1"/>
         </xs:restriction>
      </xs:simpleType>
   </xs:union>
</xs:simpleType>

<xs:complexType name="degreeType">
         <xs:attribute name="VALUE" type="intStringType"></xs:attribute>
</xs:complexType>
<!--*****************************************************************************************-->
<!-- Definition der Typen für DPF-XML-Versetzung                                             -->
<!--***************************************************************************************** -->
<xs:complexType name="dpfXMLMovementType">
    <xs:attribute name="VALUE" use="required">
     <xs:simpleType>
      <xs:restriction base="xs:string">
        <xs:enumeration value="REFERENCE"/>
        <xs:enumeration value="COMBINATION"/>
```

```
      </xs:restriction>
    </xs:simpleType>
  </xs:attribute>
 </xs:complexType>

</xs:schema>
```

XML-Schema für das Element OPTPROFILE:

Das Element OPTPROFILE ist das Stammelement (Root) eines Optimierungsprofils.

Dieses Element wird wie folgt definiert:

```
XML-Schema
   <xs:element name="OPTPROFILE">
      <xs:complexType>
         <xs:sequence>
            <!-- Abschnitt für globale Optimierungsrichtlinien.
            Nur ein solcher Abschnitt ist zulässig. -->
                        <xs:element name="OPTGUIDELINES" type=
                        "globalOptimizationGuidelinesType"
                                minOccurs="0"/>
            <!-- Anweisungsprofilabschnitt. Null oder mehr Abschnitte
                        zulässig -->
                        <xs:element name="STMTPROFILE"
                        type="statementProfileType" minOccurs="0"
                                maxOccurs="unbounded"/>
         </xs:sequence>
         <!-- Attribut VERSION ist gegenwärtig optional. -->
         <xs:attribute name="VERSION" use="optional"/>
      </xs:complexType>
   </xs:element>
```

Beschreibung

Das optionale Unterelement OPTGUIDELINES definiert die globalen Optimierungsrichtlinien für das Optimierungsprofil. Jedes Unterelement STMTPROFILE definiert ein Anweisungsprofil. Das Attribut VERSION gibt das aktuelle Optimierungsprofilschema an, unter dem ein bestimmtes Optimierungsprofil erstellt und auf Gültigkeit geprüft wurde.

XML-Schema für das globale OPTGUIDELINES-Element:

Das Element OPTGUIDELINES definiert die globalen Optimierungsrichtlinien für das Optimierungsprofil.

Es wird durch den komplexen Typ 'globalOptimizationGuidelinesType' definiert.

```
XML-Schema

   <xs:complexType name="globalOptimizationGuidelinesType">
      <xs:sequence>
         <xs:group ref="MQTOptimizationChoices"/>
         <xs:group ref="computationalPartitionGroupOptimizationChoices"/>
         <xs:group ref="generalRequest"/>
         <xs:group ref="mqtEnforcementRequest"/>
      </xs:sequence>
   </xs:complexType>
```

Beschreibung

Globale Optimierungsrichtlinien können mit Elementen aus den Gruppen 'MQTOptimizationChoices', 'computationalPartitionGroupChoices' oder 'generalRequest' definiert werden.

- Elemente der Gruppe 'MQTOptimizationChoices' können zur Beeinflussung der MQT-Substitution verwendet werden.
- Elemente der Gruppe 'computationalPartitionGroupOptimizationChoices' können zur Beeinflussung der Rechenpartitionsgruppenoptimierung verwendet werden, die eine dynamische Neuverteilung von Daten beinhaltet, die aus fernen Datenquellen gelesen werden. Sie ist nur auf Konfigurationen mit partitionierten und föderierten Datenbanken anwendbar.
- Die Elemente der Gruppe 'generalRequest' beziehen sich nicht speziell auf eine bestimmte Phase des Optimierungsprozesses und können zur Änderung des Suchbereichs des Optimierungsprogramms verwendet werden. Sie können global oder auf Anweisungsebene angegeben werden.
- MQT-Durchsetzungsanforderungen geben semantisch vergleichbare MQTs an, deren Verwendung in Zugriffsplänen unabhängig vom geschätzten Ausführungsaufwand durchgesetzt werden soll.

Auswahlmöglichkeiten für die MQT-Optimierung:

Die Gruppe 'MQTOptimizationChoices' definiert einen Satz von Elementen, die zur Beeinflussung der Optimierung mit MQTs (Materialized Query Tables) verwendet werden können. Insbesondere können diese Elemente dazu verwendet werden, die Prüfung, ob eine MQT-Substitution in Betracht kommt, zu aktivieren bzw. zu inaktivieren, oder den kompletten Satz von MQTs anzugeben, der vom Optimierungsprogramm in Betracht gezogen werden soll.

XML-Schema

```
<xs:group name="MQTOptimizationChoices">
   <xs:choice>
      <xs:element name="MQTOPT" minOccurs="0" maxOccurs="1">
         <xs:complexType>
            <xs:attribute name="OPTION" type="optionType" use="optional"/>
         </xs:complexType>
      </xs:element>
      <xs:element name="MQT" minOccurs="0" maxOccurs="unbounded">
         <xs:complexType>
            <xs:attribute name="NAME" type="xs:string" use="required"/>
         </xs:complexType>
      </xs:element>
   </xs:choice>
</xs:group>
```

Beschreibung

Das MQTOPT-Element dient zur Aktivierung bzw. Inaktivierung der Berücksichtigung der MQT-Optimierung. Das Attribut OPTION kann die Werte ENABLE (Standardwert) oder DISABLE haben.

Das Attribut NAME eines MQT-Elements gibt eine MQT an, die vom Optimierungsprogramm in Betracht gezogen werden soll. Die Regeln für die Bildung eines Verweises auf eine MQT im Attribut NAME sind die gleichen wie die für die Bildung von Verweisen auf exponierte Tabellennamen. Wenn ein oder mehrere MQT-Elemente angebeben werden, werden nur diese MQTs vom Optimierungsprogramm berücksichtigt. Die Entscheidung, eine MQT-Substitution mit einer oder mehreren der angegebenen MQTs durchzuführen, erfolgt weiterhin auf der Basis einer Aufwandsberechnung.

Beispiele

Das folgende Beispiel zeigt, wie die MQT-Optimierung inaktiviert wird:

```
<OPTGUIDELINES>
   <MQTOPT OPTION='DISABLE'/>
</OPTGUIDELINES>
```

Das folgende Beispiel zeigt, wie die MQT-Optimierung auf die Tabelle 'Tpcd-.PARTSMQT' und die Tabelle 'COLLEGE.STUDENTS' eingegrenzt wird:

```
<OPTGUIDELINES>
   <MQT NAME='Tpcd.PARTSMQT'/>
   <MQT NAME='COLLEGE.STUDENTS'/>
<OPTGUIDELINES>
```

Auswahlmöglichkeiten für die Rechenpartitionsgruppenoptimierung:

Die Gruppe 'computationalPartitionGroupOptimizationChoices' definiert einen Satz von Elementen, die zur Beeinflussung der Rechenpartitionsgruppenoptimierung verwendet werden können. Insbesondere können diese Elemente dazu verwendet werden, die Rechengruppenoptimierung zu aktivieren bzw. zu inaktivieren oder die Partitionsgruppe anzugeben, die zur Rechenpartitionsgruppenoptimierung verwendet werden soll.

XML-Schema

```
<xs:group name="computationalPartitionGroupOptimizationChoices">
   <xs:choice>
      <xs:element name="PARTOPT" minOccurs="0" maxOccurs="1">
         <xs:complexType>
            <xs:attribute name="OPTION" type="optionType" use="optional"/>
         </xs:complexType>
      </xs:element>
      <xs:element name="PART" minOccurs="0" maxOccurs="1">
         <xs:complexType>
            <xs:attribute name="NAME" type="xs:string" use="required"/>
         </xs:complexType>
      </xs:element>
   </xs:choice>
</xs:group>
```

Beschreibung

Das PARTOPT-Element dient zur Aktivierung bzw. Inaktivierung der Berücksichtigung der Rechenpartitionsgruppenoptimierung. Das Attribut OPTION kann die Werte ENABLE (Standardwert) oder DISABLE haben.

Das Element PART kann verwendet werden, um die Partitionsgruppe anzugeben, die für die Rechenpartitionsgruppenoptimierung verwendet werden soll. Das Attribut NAME muss eine vorhandene Partitionsgruppe angeben. Die Entscheidung, eine dynamische Neuverteilung unter Verwendung der angegebenen Partitionsgruppe auszuführen, erfolgt weiterhin auf der Basis einer Aufwandsberechnung.

Beispiele

Das folgende Beispiel zeigt, wie die Rechenpartitionsgruppenoptimierung inaktiviert wird:

```
<OPTGUIDELINES>
   <PARTOPT OPTION='DISABLE'/>
</OPTGUIDELINES>
```

Das folgende Beispiel zeigt, wie angegeben wird, dass die Partitionsgruppe WORKPART für die Rechenpartitionsgruppenoptimierung zu verwenden ist:

```
<OPTGUIDELINES>
   <MQT NAME='Tpcd.PARTSMQT'/>
   <PART NAME='WORKPART'/>
<OPTGUIDELINES>
```

Allgemeine Optimierungsrichtlinien als globale Anforderungen:

Die Gruppe 'generalRequest' definiert Richtlinien, die sich nicht speziell auf eine bestimmte Phase des Optimierungsprozesses beziehen und zur Änderung des Suchbereichs des Optimierungsprogramms verwendet werden können.

Allgemeine Optimierungsrichtlinien können global oder auf Anweisungsebene angegeben werden. Die Beschreibung und Syntax von Elementen allgemeiner Optimierungsrichtlinien ist für globale Optimierungsrichtlinien und Optimierungsrichtlinien auf Anweisungsebene identisch. Weitere Informationen finden Sie in der Beschreibung zum „XML-Schema für allgemeine Optimierungsrichtlinien".

XML-Schema für das Element STMTPROFILE:

Das Element STMTPROFILE definiert ein Anweisungsprofil innerhalb eines Optimierungsprofils.

Es wird durch den komplexen Typ 'statementProfileType' definiert.

XML-Schema

```
<xs:complexType name="statementProfileType">
   <xs:sequence>
      <xs:element name="STMTKEY" type="statementKeyType"/>
      <xs:element name="OPTGUIDELINES" type="optGuidelinesType"/>
   </xs:sequence>
   <xs:attribute name="ID" type="xs:string" use="optional"/>
</xs:complexType>
```

Beschreibung

Ein Anweisungsprofil gibt Optimierungsrichtlinien für eine bestimmte Anweisung an und enthält die folgenden Teile.

- Anweisungsschlüssel (STMTKEY)

  Ein Optimierungsprofil kann für mehr als eine Anweisung in einer Anwendung wirksam sein. Anhand des Anweisungsschlüssels gleicht das Optimierungsprogramm automatisch jedes Anweisungsprofil mit einer entsprechenden Anweisung in der Anwendung ab. Dies bietet Ihnen die Möglichkeit, Optimierungsrichtlinien für eine Anweisung anzugeben, ohne die Anwendung zu bearbeiten. Der Anweisungsschlüssel enthält den Text der Anweisung (wie in der Anwendung geschrieben) und weitere Informationen, die zur eindeutigen Kennzeichnung der richtigen Anweisung erforderlich sind. Der Anweisungsschlüssel wird durch das Unterelement STMTKEY dargestellt.
- Optimierungsrichtlinien auf Anweisungsebene (OPTGUIDELINES)

  Dieser Teil des Anweisungsprofils gibt die Optimierungsrichtlinien an, die für die Anweisung, die durch den Anweisungsschlüssel angegeben ist, wirksam sind. Weitere Informationen finden Sie in der Beschreibung zum „XML-Schema für das Element OPTGUIDELINES auf Anweisungsebene".
- Name des Anweisungsprofils (ID)

  Ein benutzerdefinierter Name, der in der Diagnosenachricht angezeigt wird, um ein bestimmtes Anweisungsprofil zu identifizieren.

XML-Schema für das STMTKEY-Element:

Das Element STMTKEY ermöglicht dem Optimierungsprogramm, ein Anweisungsprofil mit einer entsprechenden Anweisung in einer Anwendung abzugleichen.

Es wird durch den komplexen Typ 'statementKeyType' definiert.

XML-Schema

```
<xs:complexType name="statementKeyType" mixed="true">
   <xs:attribute name="SCHEMA" type="xs:string" use="optional"/>
   <xs:attribute name="FUNCPATH" type="xs:string" use="optional"/>
</xs:complexType>
</xs:schema>
```

Beschreibung

Das optionale Attribut SCHEMA kann verwendet werden, um die Komponente des Standardschemas des Anweisungsschlüssels anzugeben.

Das optionale Attribut FUNCPATH kann verwendet werden, um die Komponente des Funktionspfads des Anweisungsschlüssels anzugeben. Mehrere Pfade müssen durch Kommas getrennt werden und die angegebenen Funktionspfade müssen exakt mit den im Kompilierungsschlüssel angegebenen Funktionspfaden übereinstimmen.

Beispiel

Das folgende Beispiel zeigt eine Anweisungsschlüsseldefinition, die eine bestimmte Anweisung einem Standardschema 'COLLEGE' und einem Funktionspfad 'SYSIBM,SYSFUN,SYSPROC,DAVE' zuordnet:

```
<STMTKEY SCHEMA='COLLEGE' FUNCPATH='SYSIBM,SYSFUN,SYSPROC,DAVE'>
   <![CDATA[select * from orders" where foo(orderkey) > 20]]>
</STMTKEY>
```

Die CDATA-Codierung (beginnend mit <![CDATA[und endend mit]]>) ist erforderlich, weil der Anweisungstext das XML-Sonderzeichen '>' enthält.

Abgleich von Anweisungs- und Kompilierungsschlüsseln:

Der Anweisungsschlüssel dient zur Angabe der Anwendungsanweisung, für die Optimierungsrichtlinien der Anweisungsebene gelten.

Wenn eine SQL-Anweisung kompiliert wird, beeinflussen verschiedene Faktoren, wie die Anweisung semantisch vom Compiler interpretiert wird. Die SQL-Anweisung und die Einstellungen von Parametern des SQL-Compilers bilden zusammen den Kompilierungsschlüssel. Jede Komponente eines Anweisungsschlüssels entspricht einer bestimmten Komponente eines Kompilierungsschlüssels.

Ein Anweisungsschlüssel setzt sich aus folgenden Komponenten zusammen:

- Anweisungstext: Der Text der Anweisung, wie er in der Anwendung geschrieben wurde.
- Standardschema: Der Name des Schemas, der als implizites Qualifikationsmerkmal für nicht qualifizierte Tabellennamen verwendet wird. Diese Komponente ist optional. Sie sollte jedoch angegeben werden, wenn die Anweisung nicht qualifizierte Tabellennamen enthält.

- Funktionspfad: Dies ist der Funktionspfad, der zur Auflösung nicht qualifizierter Funktions- und Datentypverweise verwendet wird. Diese Komponente ist optional. Sie sollte jedoch angegeben werden, wenn die Anweisung nicht qualifizierte benutzerdefinierte Funktionen oder benutzerdefinierte Datentypen enthält.

Wenn der Datenserver eine SQL-Anweisung kompiliert und ein aktives Optimierungsprofil findet, versucht er, jeden Anweisungsschlüssel im Optimierungsprofil mit dem aktuellen Kompilierungsschlüssel abzugleichen. Ein Anweisungsschlüssel und ein Kompilierungsschlüssel werden als übereinstimmend betrachtet, wenn jede angegebene Komponente des Anweisungsschlüssels mit der entsprechenden Komponente des Kompilierungsschlüssels übereinstimmt. Wenn eine Komponente des Anweisungsschlüssels nicht angegeben ist, wird die ausgelassene Komponente standardmäßig als übereinstimmend betrachtet. Jede nicht angegebene Komponente des Anweisungsschlüssels wird als Platzhalter in der Weise interpretiert, dass sie mit der entsprechenden Komponente eines beliebigen Kompilierungsschlüssels übereinstimmt.

Wenn der Datenserver einen Anweisungsschlüssel findet, der mit dem aktuellen Kompilierungsschlüssel übereinstimmt, stoppt er die Suche. Wenn mehrere Anweisungsprofile vorhanden sind, deren Anweisungsschlüssel mit dem aktuellen Kompilierungsschlüssel übereinstimmt, wird nur das erste dieser Anweisungsprofile (entsprechend der Dokumentreihenfolge) verwendet.

XML-Schema für das Element OPTGUIDELINES auf Anweisungsebene:

Das Element OPTGUIDELINES eines Anweisungsoptimierungsprofils definiert die wirksamen Optimierungsrichtlinien für die Anweisung, die durch den zugeordneten Anweisungsschlüssel (STMTKEY) angegeben wird. Es wird durch den komplexen Typ 'optGuidelinesType' definiert.

XML-Schema

```
<xs:element name="OPTGUIDELINES" type="optGuidelinesType"/>
<xs:complexType name="optGuidelinesType">
   <xs:sequence>
      <xs:group ref="general request" minOccurs="0" maxOccurs="1"/>
      <xs:choice maxOccurs="unbounded">
         <xs:group ref="rewriteRequest"/>
         <xs:group ref="accessRequest"/>
         <xs:group ref="joinRequest"/>
         <xs:group ref="mqtEnforcementRequest"/>
      </xs:choice>
  </xs:sequence>
</xs:complexType>
```

Beschreibung

Die Gruppe 'optGuidelinesType' definiert den Satz gültiger Unterelemente des Elements OPTGUIDELINES. Jedes Unterelement wird vom DB2-Optimierungsprogramm als Optimierungsrichtlinie interpretiert. Unterelemente können in allgemeine Anforderungselemente, Umschreibanforderungselemente, Zugriffsanforderungselemente oder Joinanforderungselemente kategorisiert werden.

- *Allgemeine Anforderungselemente* dienen zur Angabe allgemeiner Optimierungsrichtlinien, die zum Ändern des Suchbereichs des Optimierungsprogramms verwendet werden können.

- *Umschreibanforderungselemente* dienen zur Angabe von Optimierungsrichtlinien für das Umschreiben von Abfragen, mit denen die Abfrageumsetzungen beeinflusst werden können, die angewendet werden, wenn die optimierte Anweisung ermittelt wird.
- *Zugriffsanforderungselemente* und *Joinanforderungselemente* sind Planoptimierungsrichtlinien, die zur Beeinflussung von Zugriffsmethoden, Joinmethoden und Joinreihenfolgen verwendet werden können, die im Ausführungsplan für die optimierte Anweisung verwendet werden.
- *Anforderungselemente für MQT-Durchsetzung* geben semantisch vergleichbare MQTs an, deren Verwendung in Zugriffsplänen unabhängig vom geschätzten Ausführungsaufwand durchgesetzt werden soll.

Anmerkung: Optimierungsrichtlinien, die in einem Anweisungsprofil angegeben sind, haben Vorrang vor den Richtlinien, die im globalen Abschnitt eines Optimierungsprofils angegeben sind.

XML-Schema für allgemeine Optimierungsrichtlinien:

Die Gruppe 'generalRequest' definiert Richtlinien, die sich nicht speziell auf eine bestimmte Phase des Optimierungsprozesses beziehen und zur Änderung des Suchbereichs des Optimierungsprogramms verwendet werden können.

```
<!--****************************************************************************** --> \
<!-- Auswahlmöglichkeiten für allgemeine Anforderungselemente.                    --> \
<!-- REOPT kann zum Überschreiben der Einstellung der Bindeoption REOPT verwendet werden. --> \
<!-- DPFXMLMOVEMENT kann beim Versetzen von XML-Dokumenten auf andere             --> \
<!-- Datenbankpartitionen zum Beeinflussen des Optimierungsprogrammplans verwendet werden.--> \
<!-- Zulässige Werte: REFERENCE und COMBINATION. Standardwert: NONE.              --> \
<!--****************************************************************************** --> \
<xs:group name="generalRequest">
   <xs:sequence>
      <xs:element name="REOPT" type="reoptType" minOccurs="0" maxOccurs="1"/>
      <xs:element name="DEGREE" type="degreeType" minOccurs="0" maxOccurs="1"/>
      <xs:element name="QRYOPT" type="qryoptType" minOccurs="0" maxOccurs="1"/>
      <xs:element name="RTS" type="rtsType" minOccurs="0" maxOccurs="1"/>
      <xs:element name="DPFXMLMOVEMENT" type="dpfXMLMovementType" minOccurs="0" maxOccurs="1"/>
   </xs:sequence>
</xs:group>
```

Anmerkung: Allgemeine Optimierungsrichtlinien können global oder auf Anweisungsebene angegeben werden. Die Beschreibung und Syntax von Elementen allgemeiner Optimierungsrichtlinien ist für globale Optimierungsrichtlinien und Optimierungsrichtlinien auf Anweisungsebene identisch.

Beschreibung

Allgemeine Anforderungselemente können zur Definition allgemeiner Optimierungsrichtlinien verwendet werden, die den Optimierungssuchbereich beeinflussen und daher die Anwendbarkeit von Optimierungsrichtlinien für das Umschreiben von Abfragen sowie von aufwandsbasierten Optimierungsrichtlinien beeinflussen können.

DEGREE-Anforderungen:

Das allgemeine Anforderungselement DEGREE kann zum Überschreiben der Einstellung der Bindeoption DEGREE, des Datenbankkonfigurationsparameters **dft_degree** oder des Ergebnisses einer vorherigen Anweisung SET CURRENT DEGREE verwendet werden.

Das allgemeine Anforderungselement DEGREE wird nur berücksichtigt, wenn die Instanz für partitionsinterne Parallelität konfiguriert ist. Ansonsten wird eine Warnung zurückgegeben. Es wird durch den komplexen Typ 'degreeType' definiert.

XML-Schema

```
<xs:simpleType name="intStringType">
   <xs:union>
      <xs:simpleType>
         <xs:restriction base="xs:integer">
            <xs:minInclusive value="1"></xs:minInclusive>
            <xs:maxInclusive value="32767"></xs:maxInclusive>
         </xs:restriction>
      </xs:simpleType>
      <xs:simpleType>
         <xs:restriction base="xs:string">
            <xs:enumeration value="ANY"/>
            <xs:enumeration value="-1"/>
         </xs:restriction>
      </xs:simpleType>
   </xs:union>
</xs:simpleType>

<xs:complexType name="degreeType">
   <xs:attribute name="VALUE"
      type="intStringType"></xs:attribute>
</xs:complexType>
```

Beschreibung

Das allgemeine Anforderungselement DEGREE hat ein erforderliches Attribut VALUE, das die Einstellung der Option DEGREE angibt. Das Attribut kann einen ganzzahligen Wert von 1 bis 32.767 oder die Zeichenfolgewerte -1 oder ANY erhalten. Der Wert -1 (bzw. ANY) gibt an, dass der Grad der Parallelität vom Datenserver bestimmt werden soll. Der Wert 1 gibt an, dass die Abfrage keine partitionsinterne Parallelität verwenden soll.

DPFXMLMOVEMENT-Anforderungen:

Mit dem allgemeinen Anforderungselement DPFXMLMOVEMENT können in Umgebungen mit partitionierten Datenbanken Entscheidungen des Optimierungsprogramms überschrieben werden, um einen Plan zu verwenden, bei dem eine Spalte vom Typ XML versetzt oder ein Verweis auf die betreffende Spalte auf eine andere Datenbankpartition versetzt wird. Das Element wird durch den komplexen Typ 'dpfXMLMovementType' definiert.

```
<xs:complexType name="dpfXMLMovementType">
   <xs:attribute name="VALUE" use="required">
      <xs:simpleType>
         <xs:restriction base="xs:string"
            <xs:enumeration value="REFERENCE"/>
            <xs:enumeration value="COMBINATION"/>
         </xs:restriction>
      </xs:simpleType>
   </xs:attribute>
</xs:complexType>
```

Beschreibung

In Umgebungen mit partitionierten Datenbanken müssen Daten teilweise mithilfe von Anweisungen zwischen Datenbankpartitionen versetzt werden. Bei XML-Spalten kann das Optimierungsprogramm auswählen, ob die in diesen Spalten enthaltenen Dokumente selbst oder lediglich als Verweis auf die Quellendokumente in den ursprünglichen Datenbankpartitionen versetzt werden.

Für das allgemeine Anforderungselement DPFXMLMOVEMENT ist das Attribut VALUE erforderlich. Für dieses Attribut sind folgende Werte möglich: REFERENCE oder COMBINATION. Wenn eine Zeile mit einer XML-Spalte auf eine andere Partition versetzt werden muss, gilt Folgendes:

- REFERENCE gibt an, dass Verweise auf die XML-Dokumente über den Tabellenwarteschlangenoperator (TQ) versetzt werden sollen. Die Dokumente selbst bleiben auf der Quellendatenbankpartition.
- COMBINATION gibt an, dass einige XML-Dokumente versetzt werden und nur Verweise auf die verbleibenden XML-Dokumente über den Tabellenwarteschlangenoperator versetzt werden.

Die Entscheidung, ob die Dokumente oder lediglich Verweise auf diese Dokumente versetzt werden, richtet sich nach den Bedingungen bei der Abfrageausführung. Wird das allgemeine Anforderungselement DPFXMLMOVEMENT nicht angegeben, fällt das Optimierungsprogramm am geschätzten Ausführungsaufwand ausgerichtete Entscheidungen, die in erster Linie dazu dienen, die Leistung zu maximieren.

QRYOPT-Anforderungen:

Das allgemeine Anforderungselement QRYOPT kann zum Überschreiben der Einstellung der Bindeoption QUERYOPT, des Datenbankkonfigurationsparameters **dft_queryopt** oder des Ergebnisses einer vorherigen Anweisung SET CURRENT QUERY OPTIMIZATION verwendet werden. Es wird durch den komplexen Typ 'qryoptType' definiert.

XML-Schema

```
<xs:complexType name="qryoptType">
   <xs:attribute name="VALUE" use="required">
      <xs:simpleType>
         <xs:restriction base="xs:string">
            <xs:enumeration value="0"/>
            <xs:enumeration value="1"/>
            <xs:enumeration value="2"/>
            <xs:enumeration value="3"/>
            <xs:enumeration value="5"/>
            <xs:enumeration value="7"/>
            <xs:enumeration value="9"/>
         </xs:restriction>
      </xs:simpleType>
   </xs:attribute>
</xs:complexType>
```

Beschreibung

Das allgemeine Anforderungselement QRYOPT hat ein erforderliches Attribut VALUE, das die Einstellung der Option QUERYOPT angibt. Das Attribut kann einen der folgenden Werte erhalten: 0, 1, 2, 3, 5, 7 oder 9. Detaillierte Informationen zur Bedeutung dieser Werte finden Sie im Abschnitt „Optimierungsklassen".

REOPT-Anforderungen:

Das allgemeine Anforderungselement REOPT kann zum Überschreiben der Einstellung der Bindeoption REOPT verwendet werden, die sich auf die Optimierung von Anweisungen auswirkt, die Parametermarken oder Hostvariablen enthalten. Es wird durch den komplexen Typ 'reoptType' definiert.

XML-Schema

```
<xs:complexType name="reoptType">
```

```
      <xs:attribute name="VALUE" use="required">
         <xs:simpleType>
            <xs:restriction base="xs:string">
               <xs:enumeration value="ONCE"/>
               <xs:enumeration value="ALWAYS"/>
            </xs:restriction>
         </xs:simpleType>
      </xs:attribute>
   </xs:complexType>
```

Beschreibung

Das allgemeine Anforderungselement REOPT hat ein erforderliches Attribut VALUE, das die Einstellung der Option REOPT angibt. Das Attribut kann den Wert ONCE oder den Wert ALWAYS annehmen. Der Wert ONCE gibt an, dass die Anweisung für den ersten Satz von Werten für Hostvariablen bzw. Parametermarken optimiert werden soll. Der Wert ALWAYS gibt an, dass die Anweisung für jeden Satz von Werten für Hostvariablen bzw. Parametermarken optimiert werden soll.

RTS-Anforderungen:

Das allgemeine RTS-Anforderungselement kann zur Aktivierung bzw. Inaktivierung der Echtzeitstatistikerfassung (RTS - Real-Time Statistics) verwendet werden. Es kann außerdem dazu verwendet werden, den Zeitaufwand für die Echtzeitstatistikerfassung zu begrenzen.

Für bestimmte Abfragen oder Auslastungen kann es empfehlenswert sein, Erfassung von Echtzeitstatistiken zu begrenzen, sodass zusätzlicher Aufwand bei der Kompilierung von Anweisungen vermieden werden kann. Das allgemeine Anforderungselement RTS wird durch den komplexen Typ 'rtsType' definiert.

```
<!--*****************************************************************************************--> \
<!-- Allgemeine RTS-Anforderung zum Aktivieren oder Inaktivieren der Echtzeitstatistik-       --> \
<!-- erfassung bzw. zur Festlegung eines Zeitbudgets für diese Erfassung.                     --> \
<!-- Attribut OPTION: Aktivieren oder Inaktivieren der Echtzeitstatistiken.                   --> \
<!-- Attribut TIME: Zeitbudget in Millisekunden für Echtzeitstatistikerfassung.               --> \
<!--***************************************************************************************** --> \
<xs:complexType name="rtsType">
   <xs:attribute name="OPTION" type="optionType" use="optional" default="ENABLE"/>
   <xs:attribute name="TIME" type="xs:nonNegativeInteger" use="optional"/>
</xs:complexType>
```

Beschreibung

Das allgemeine Anforderungselement für die Echtzeitstatistikerfassung (RTS) hat zwei optionale Attribute.

- Das Attribut OPTION dient zum Aktivieren oder Inaktivieren der Echtzeitstatistikerfassung. Es kann den Wert ENABLE (Standardwert) oder DISABLE erhalten.
- Das Attribut TIME gibt den maximalen Zeitaufwand (in Millisekunden) an, der für die Erfassung von Echtzeitstatistiken bei der Anwendungskompilierung (pro Anweisung) aufgewendet werden kann.

Wenn für das Attribut OPTION der Wert ENABLE angegeben wird, muss die automatische Statistikerfassung und die Echtzeitstatistikerfassung über die entsprechenden Konfigurationsparameter aktiviert sein. Ansonsten wird die Optimierungsrichtlinie nicht angewendet und eine Nachricht SQL0437W mit Ursachencode 13 zurückgegeben.

XML-Schema für Optimierungsrichtlinien zum Umschreiben von Abfragen:

Die Gruppe 'rewriteRequest' definiert Richtlinien, die die Phase der Abfrageumschreibung des Optimierungsprozesses beeinflussen.

XML-Schema

```
<xs:group name="rewriteRequest">
   <xs:sequence>
      <xs:element name="INLIST2JOIN" type="inListToJoinType" minOccurs="0"/>
      <xs:element name="SUBQ2JOIN"   type="subqueryToJoinType" minOccurs="0"/>
      <xs:element name="NOTEX2AJ"    type="notExistsToAntiJoinType" minOccurs="0"/>
      <xs:element name="NOTIN2AJ"    type="notInToAntiJoinType" minOccurs="0"/>
   </xs:sequence>
</xs:group>
```

Beschreibung

Wenn das Element INLIST2JOIN dazu verwendet wird, Optimierungsrichtlinien sowohl auf Anweisungsebene als auch auf Vergleichselementebene anzugeben, überschreiben die Richtlinien der Vergleichselementebene die Richtlinien der Anweisungsebene.

Umschreibanforderungen INLIST2JOIN:

Ein Anforderungselement INLIST2JOIN zur Abfrageumschreibung kann zur Aktivierung bzw. Inaktivierung der Umsetzung von IN-Listenvergleichselementen in Joins verwendet werden. Dieses Element kann als Optimierungsrichtlinie auf Anweisungsebene oder auf Vergleichselementebene angegeben werden. Im letzteren Fall kann nur eine einzige Richtlinie pro Abfrage aktiviert sein. Das Anforderungselement INLIST2JOIN wird durch den komplexen Typ 'inListToJoinType' definiert.

XML-Schema

```
<xs:complexType name="inListToJoinType">
   <xs:attribute name="OPTION" type="optionType" use="optional" default="ENABLE"/>
   <xs:attribute name="TABLE"  type="xs:string"  use="optional"/>
   <xs:attribute name="COLUMN" type="xs:string"  use="optional"/>
</xs:complexType>
```

Beschreibung

Das Anforderungselement INLIST2JOIN zur Abfrageumschreibung besitzt drei optionale Attribute und keine Unterelemente. Das Attribut OPTION kann die Werte ENABLE (Standardwert) oder DISABLE haben. Die Attribute TABLE und COLUMN dienen zur Angabe eines IN-Listenvergleichselements. Wenn diese Attribute nicht angegeben sind oder mit einer leeren Zeichenfolge („“) angegeben ist, wird die Richtlinie wie eine Richtlinie auf Anweisungsebene behandelt. Wenn eines oder beide dieser Attribute angegeben sind, wird sie als Richtlinie auf Vergleichselementebene behandelt. Wenn das Attribute TABLE nicht angegeben ist oder mit einem leeren Zeichenfolgewert, jedoch das Attribut COLUMN angegeben ist, wird die Optimierungsrichtlinie ignoriert und eine Nachricht SQL0437W mit Ursachencode 13 zurückgegeben.

Umschreibanforderungen NOTEX2AJ:

Das Anforderungselement NOTEX2AJ zur Abfrageumschreibung kann zur Aktivierung bzw. Inaktivierung der Umsetzung von NOT EXISTS-Vergleichselementen in Antijoins verwendet werden. Es kann nur als Optimierungsrichtlinie auf Anweisungsebene angegeben werden. Das Anforderungselement NOTEX2AJ wird durch den komplexen Typ 'notExistsToAntiJoinType' definiert.

XML-Schema

```
<xs:complexType name="notExistsToAntiJoinType">
   <xs:attribute name="OPTION" type="optionType" use="optional" default="ENABLE"/>
</xs:complexType>
```

Beschreibung

Das Anforderungselement NOTEX2AJ zur Abfrageumschreibung besitzt ein optionales Attribut und keine Unterelemente. Das Attribut OPTION kann die Werte ENABLE (Standardwert) oder DISABLE haben.

Umschreibanforderungen NOTIN2AJ:

Das Anforderungselement NOTIN2AJ zur Abfrageumschreibung kann zur Aktivierung bzw. Inaktivierung der Umsetzung von NOT IN-Vergleichselementen in Antijoins verwendet werden. Es kann nur als Optimierungsrichtlinie auf Anweisungsebene angegeben werden. Das Anforderungselement NOTIN2AJ wird durch den komplexen Typ 'notInToAntiJoinType' definiert.

XML-Schema

```
<xs:complexType name="notInToAntiJoinType">
   <xs:attribute name="OPTION" type="optionType" use="optional" default="ENABLE"/>
</xs:complexType>
```

Beschreibung

Das Anforderungselement NOTIN2AJ zur Abfrageumschreibung besitzt ein optionales Attribut und keine Unterelemente. Das Attribut OPTION kann die Werte ENABLE (Standardwert) oder DISABLE haben.

Umschreibanforderungen SUBQ2JOIN:

Das Anforderungselement SUBQ2JOIN zur Abfrageumschreibung kann zur Aktivierung bzw. Inaktivierung der Umsetzung von Unterabfragen in Joins verwendet werden. Es kann nur als Optimierungsrichtlinie auf Anweisungsebene angegeben werden. Das Anforderungselement SUBQ2JOIN wird durch den komplexen Typ 'subqueryToJoinType' definiert.

XML-Schema

```
<xs:complexType name="subqueryToJoinType">
   <xs:attribute name="OPTION" type="optionType" use="optional" default="ENABLE"/>
</xs:complexType>
```

Beschreibung

Das Anforderungselement SUBQ2JOIN zur Abfrageumschreibung besitzt ein optionales Attribut und keine Unterelemente. Das Attribut OPTION kann die Werte ENABLE (Standardwert) oder DISABLE haben.

XML-Schema für Planoptimierungsrichtlinien:

Planoptimierungsrichtlinien können aus Zugriffsanforderungen oder Joinanforderungen bestehen.

- Eine *Zugriffsanforderung* gibt eine Zugriffsmethode für einen Tabellenverweis an.
- Eine *Joinanforderung* gibt eine Methode und eine Reihenfolge für die Ausführung einer Joinoperation an. Joinanforderungen bestehen wiederum aus weiteren Zugriffs- oder Joinanforderungen.

Die meisten der verfügbaren Zugriffsanforderungen entsprechen den Datenzugriffsmethoden des Optimierungsprogramms, wie zum Beispiel Tabellensuche, Indexsuche und Vorablesezugriff über Listen. Die meisten verfügbaren Joinanforderungen entsprechen den Joinmethoden des Optimierungsprogramms, wie zum

Beispiel Join mit Verschachtelungsschleife (Nested Loop Join), Hash-Join und Mischjoin (Merge Join). Jedes Zugriffs- bzw. Joinanforderungselement kann zur Beeinflussung der Planoptimierung verwendet werden.

Zugriffsanforderungen:

Die Gruppe 'accessRequest' definiert den Satz der gültigen Zugriffsanforderungselemente. Eine Zugriffsanforderung gibt eine Zugriffsmethode für einen Tabellenverweis an.

XML-Schema

```
<xs:group name="accessRequest">
   <xs:choice>
      <xs:element name="TBSCAN" type="tableScanType"/>
      <xs:element name="IXSCAN" type="indexScanType"/>
      <xs:element name="LPREFETCH" type="listPrefetchType"/>
      <xs:element name="IXAND" type="indexAndingType"/>
      <xs:element name="IXOR" type="indexOringType"/>
      <xs:element name="XISCAN" type="indexScanType"/>
      <xs:element name="XANDOR" type="XANDORType"/>
      <xs:element name="ACCESS" type="anyAccessType"/>
   </xs:choice>
</xs:group>
```

Beschreibung

- TBSCAN, IXSCAN, LPREFETCH, IXAND, IXOR, XISCAN und XANDOR

  Diese Elemente entsprechen den DB2-Datenzugriffsmethoden und können nur auf lokale Tabellen angewendet werden, die in einer Anweisung angegeben werden. Sie können sich nicht auf Kurznamen (ferne Tabellen) oder abgeleitete Tabellen (Ergebnis eines Subselects) beziehen.
- ACCESS

  Dieses Element, mit dem das Optimierungsprogramm veranlasst wird, die Zugriffsmethode auszuwählen, kann verwendet werden, wenn die Joinreihenfolge (nicht die Zugriffsmethode) von primärem Interesse ist. Das Element ACCESS muss verwendet werden, wenn der Zieltabellenverweis eine abgeleitete Tabelle ist. Bei XML-Abfragen kann dieses Element auch mit dem Attribut TYPE = XMLINDEX verwendet werden, um anzugeben, dass das Optimierungsprogramm Zugriffspläne mit XML-Indizes auswählen soll.

Zugriffstypen:

Allgemeine Aspekte der Elemente TBSCAN, IXSCAN, LPREFETCH, IXAND, IXOR, XISCAN, XANDOR und ACCESS werden durch den abstrakten Typ 'accessType' definiert.

XML-Schema

```
<xs:complexType name="accessType" abstract="true">
   <xs:attribute name="TABLE" type="xs:string" use="optional"/>
   <xs:attribute name="TABID" type="xs:string" use="optional"/>
   <xs:attribute name="FIRST" type="xs:string" use="optional" fixed="TRUE"/>
   <xs:attribute name="SHARING" type="optionType" use="optional" default="ENABLE"/>
   <xs:attribute name="WRAPPING" type="optionType" use="optional" default="ENABLE"/>
   <xs:attribute name="THROTTLE" type="optionType" use="optional"/>
   <xs:attribute name="SHARESPEED" type="shareSpeed" use="optional"/>
</xs:complexType>

<xs:complexType name="extendedAccessType">
   <xs:complexContent>
      <xs:extension base="accessType">
```

```
          <xs:sequence minOccurs="0">
             <xs:element name="INDEX" type="indexType" minOccurs="2"
               maxOccurs="unbounded"/>
          </xs:sequence>
          <xs:attribute name="INDEX" type="xs:string" use="optional"/>
          <xs:attribute name="TYPE" type="xs:string" use="optional" fixed="XMLINDEX"/>
          <xs:attribute name="ALLINDEXES" type="boolType" use="optional" fixed="TRUE"/>
        </xs:extension>
    </xs:complexContent>
  </xs:complexType>
```

Beschreibung

Alle Zugriffsanforderungselemente erweitern den komplexen Typ 'accessType'. Jedes Element dieser Art muss den Zieltabellenverweis entweder durch das Attribut TABLE oder durch das Attribut TABID angeben. Informationen dazu, wie ordnungsgemäße Tabellenverweise aus einem Zugriffselement gebildet werden, finden Sie im Abschnitt „Bildung von Tabellenverweisen in Optimierungsrichtlinien".

Zugriffsanforderungen können außerdem das optionale Attribut FIRST angeben. Wenn das Attribut FIRST angegeben wird, muss es den Wert TRUE haben. Durch Hinzufügen des Attributs FIRST in einem Zugriffsanforderungselement wird angeben, dass der Ausführungsplan die angegebene Tabelle als erste Tabelle in der Joinreihenfolge der entsprechenden FROM-Klausel enthält. Nur in einer Zugriffs- oder Joinanforderung pro FROM-Klausel kann das Attribut FIRST angegeben werden. Wenn mehrere Zugriffs- oder Joinanforderungen, die Tabellen derselben FROM-Klausel angeben, das Attribut FIRST enthalten, werden alle außer der ersten Anforderung ignoriert und eine Warnung (SQL0437W mit Ursachencode 13) zurückgegeben.

Neue Optimierungsprogrammrichtlinien bieten Ihnen die Möglichkeit, die Entscheidungen des Compilers im Hinblick auf das Scan-Sharing (gemeinsame Nutzung von Suchläufen) zu beeinflussen. In Fällen, in denen der Compiler die gemeinsame Nutzung von Suchläufen, Suchläufe mit Umlauf oder eine Drosselung zugelassen hätte, kann eine entsprechende Richtlinie dies verhindern. Ein gemeinsam genutzter Suchlauf ist ein Suchlauf, der für andere Suchläufe, die am Scan-Sharing beteiligt sind, sichtbar ist, sodass diese anderen Suchläufe bestimmte Entscheidungen auf der Basis dieser Informationen treffen können. Suchläufe mit Umlauf sind Suchläufe, die an einem beliebigen Punkt in der Tabelle beginnen können, um Seiten, die bereits in den Pufferpool eingelesen wurden, vorteilhaft zu nutzen. Bei einem gedrosselten Suchlauf handelt es sich um einen Suchlauf, der verzögert wurde, um den Grad der gemeinsamen Nutzung insgesamt zu erhöhen.

Gültige Werte für 'optionType' (für die Attribute SHARING, WRAPPING und THROTTLE) sind DISABLE und ENABLE (Standardwert). Die Attribute SHARING und WRAPPING können nicht aktiviert werden, wenn vom Compiler eine Inaktivierung festgelegt wurde. In diesen Fällen bleibt der Wert ENABLE ohne Auswirkung. Das Attribut THROTTLE kann aktiviert oder inaktiviert werden. Gültige Werte für SHARESPEED (zum Überschreiben der Suchgeschwindigkeitsschätzung des Compilers) sind FAST und SLOW. Die Standardeinstellung ist, dass der Compiler Werte auf der Basis seiner Schätzung bestimmen kann.

Beim Attribut TYPE wird als einziger Wert XMLINDEX unterstützt. Dieser Wert gibt dem Optimierungsprogramm an, dass auf eine Tabelle mit einer der XML-Indexzugriffsmethoden IXAND, IXOR, XANDOR oder XISCAN zugegriffen werden muss. Wird dieses Attribut nicht angegeben, wählt das Optimierungsprogramm den Zugriffsplan für die angegebene Tabelle anhand des geschätzten Ausführungsaufwands aus.

Mit dem optionalen Attribut INDEX kann nur ein Indexname angegeben werden.

Mit dem optionalen Element INDEX können mehrere Indexnamen als Indexelemente angegeben werden. Werden sowohl das Attribut INDEX als auch das Element INDEX angegeben, wird das Attribut INDEX ignoriert.

Das optionale Attribut ALLINDEXES, bei dem als einziger Wert TRUE unterstützt wird, kann nur angegeben werden, wenn das Attribut TYPE den Wert XMLINDEX aufweist. Wird das Attribut ALLINDEXES angegeben, muss das Optimierungsprogramm unabhängig vom Ausführungsaufwand alle entsprechenden relationalen Indizes und Indizes für XML-Daten für den Zugriff auf die angegebene Tabelle verwenden.

Anforderungen für beliebigen Zugriffstyp:

Mit dem Zugriffsanforderungselement ACCESS kann festgelegt werden, dass das Optimierungsprogramm eine geeignete Methode für den Zugriff auf eine Tabelle auf der Basis einer Aufwandsberechnung auswählen soll. Dieses Element muss verwendet werden, wenn auf eine abgeleitete Tabelle verwiesen wird. Eine abgeleitete Tabelle ist das Ergebnis einer anderen Subselect-Anweisung. Dieses Zugriffsanforderungselement wird durch den komplexen Typ 'anyAccessType' definiert.

```
XML-Schema

<xs:complexType name="anyAccessType">
   <xs:complexContent>
      <xs:extension base="extendedAccessType"/>
   </xs:complexContent>
</xs:complexType>
```

Beschreibung

Der komplexe Typ 'anyAccessType' ist eine einfache Erweiterung des abstrakten Typs 'extendedAccessType'. Es werden keine neuen Elemente oder Attribute hinzugefügt.

Das Attribut TYPE, bei dem als einziger Wert XMLINDEX unterstützt wird, gibt dem Optimierungsprogramm an, dass auf eine Tabelle mit einer der XML-Indexzugriffsmethoden IXAND, IXOR, XANDOR oder XISCAN zugegriffen werden soll. Wird dieses Attribut nicht angegeben, wählt das Optimierungsprogramm den Zugriffsplan für die angegebene Tabelle anhand des geschätzten Ausführungsaufwands aus.

Mit dem optionalen Attribut INDEX kann nur ein Indexname angegeben werden, wenn das Attribut TYPE den Wert XMLINDEX aufweist. Wird dieses Attribut angegeben, kann das Optimierungsprogramm einen der folgenden Pläne auswählen:

- XISCAN-Plan unter Verwendung des angegebenen Indexes für XML-Daten
- XANDOR-Plan, bei dem der angegebene Index für XML-Daten einen der Indizes unter XANDOR darstellt. Das Optimierungsprogramm verwendet alle entsprechenden Indizes für XML-Daten im XANDOR-Plan.
- IXAND-Plan, bei dem der angegebene Index den führenden Index von IXAND darstellt. Das Optimierungsprogramm wird am geschätzten Ausführungsaufwand ausgerichtet weitere Indizes zum IXAND-Plan hinzufügen.
- Am geschätzten Ausführungsaufwand ausgerichteter IXOR-Plan

Mit dem optionalen Element INDEX können nur mehrere Indexnamen als Indexelemente angegeben werden, wenn das Attribut TYPE den Wert XMLINDEX aufweist. Wird dieses Element angegeben, kann das Optimierungsprogramm einen der folgenden Pläne auswählen:

- XANDOR-Plan, bei dem die angegebenen Indizes für XML-Daten unter XANDOR erscheinen. Das Optimierungsprogramm wird alle entsprechenden Indizes für XML-Daten im XANDOR-Plan verwenden.
- IXAND-Plan, bei dem die angegebenen Indizes IXAND-Indizes darstellen (in der angegebenen Reihenfolge).
- Am geschätzten Ausführungsaufwand ausgerichteter IXOR-Plan

Werden sowohl das Attribut INDEX als auch das Element INDEX angegeben, wird das Attribut INDEX ignoriert.

Das optionale Attribut ALLINDEXES, bei dem als einziger Wert TRUE unterstützt wird, kann nur angegeben werden, wenn das Attribut TYPE den Wert XMLINDEX aufweist. Wird dieses Attribut angegeben, muss das Optimierungsprogramm unabhängig vom Ausführungsaufwand alle entsprechenden relationalen Indizes und Indizes für XML-Daten für den Zugriff auf die angegebene Tabelle verwenden. Das Optimierungsprogramm wählt einen der folgenden Pläne aus:

- XANDOR-Plan mit allen entsprechenden Indizes für XML-Daten, die unter dem Operator XANDOR erscheinen
- IXAND-Plan mit allen entsprechenden relationalen Indizes und Indizes für XML-Daten, die unter dem Operator IXAND erscheinen
- IXOR-Plan
- XISCAN-Plan, wenn nur ein einziger Index für die Tabelle definiert ist und dieser Index den Typ XML aufweist

Beispiele

Die folgende Richtlinie stellt ein Beispiel für eine beliebige Zugriffsanforderung dar:

```
<OPTGUIDELINES>
  <HSJOIN>
    <ACCESS TABLE='S1'/>
    <IXSCAN TABLE='PS1'/>
  </HSJOIN>
</OPTGUIDELINES>
```

Das folgende Beispiel beinhaltet eine Zugriffsrichtlinie, die angibt, dass ein XML-Indexzugriff auf die Tabelle SECURITY verwendet werden soll. Das Optimierungsprogramm kann einen beliebigen XML-Indexplan, z. B. einen XISCAN-, IXAND-, XANDOR- oder IXOR-Plan, auswählen.

```
SELECT * FROM security
  WHERE XMLEXISTS('$SDOC/Security/SecurityInformation/
    StockInformation[Industry= "OfficeSupplies"]')

<OPTGUIDELINES>
  <ACCESS TABLE='SECURITY' TYPE='XMLINDEX'/>
</OPTGUIDELINES>
```

Das folgende Beispiel beinhaltet eine Zugriffsrichtlinie, die angibt, dass alle möglichen Indexzugriffe auf die Tabelle SECURITY verwendet werden sollen. Über die Methode entscheidet das Optimierungsprogramm. Angenommen, zwei XML-Indizes mit der Bezeichnung SEC_INDUSTRY und SEC_SYMBOL entsprechen den bei-

den XML-Vergleichselementen. Das Optimierungsprogramm kann je nach Ausführungsaufwand die XANDOR- oder IXAND-Zugriffsmethode auswählen.

```
SELECT * FROM security
  WHERE XMLEXISTS('$SDOC/Security/SecurityInformation/
      StockInformation[Industry= "Software"]') AND
    XMLEXISTS('$SDOC/Security/Symbol[.="IBM"]')

<OPTGUIDELINES>
  <ACCESS TABLE='SECURITY' TYPE='XMLINDEX' ALLINDEXES='TRUE'/>
</OPTGUIDELINES>
```

Das folgende Beispiel beinhaltet eine Zugriffsrichtlinie, die angibt, dass beim Zugriff auf die Tabelle SECURITY zumindest der Index SEC_INDUSTRY XML verwendet werden soll. Das Optimierungsprogramm wählt entsprechend dem geschätzten Ausführungsaufwand einen der folgenden Zugriffspläne aus:

- XISCAN-Plan unter Verwendung des Index SEC_INDUSTRY XML
- IXAND-Plan mit dem Index SEC_INDUSTRY als erster Index für IXAND. Das Optimierungsprogramm kann nun je nach Aufwandsanalyse weitere relationale Indizes oder XML-Indizes im XAND-Plan verwenden. Ist ein relationaler Index für die Spalte TRANS_DATE verfügbar, kann dieser Index z. B. als ein zusätzlicher Index für IXAND erscheinen, wenn dies vom Optimierungsprogramm als vorteilhaft bewertet wird.
- XANDOR-Plan mit dem Index SEC_INDUSTRY und anderen entsprechenden XML-Indizes

```
SELECT * FROM security
  WHERE trans_date = CURRENT DATE AND
    XMLEXISTS('$SDOC/Security/SecurityInformation/
      StockInformation[Industry= "Software"]') AND
    XMLEXISTS('$SDOC/Security/Symbol[.="IBM"]')

<OPTGUIDELINES>
  <ACCESS TABLE='SECURITY' TYPE='XMLINDEX' INDEX='SEC_INDUSTRY'/>
</OPTGUIDELINES>
```

Zugriffsanforderungen mit logischem Verknüpfen von Indizes über AND (Index ANDing):

Das Zugriffsanforderungselement IXAND kann verwendet werden, um anzugeben, dass das Optimierungsprogramm die Datenzugriffsmethode mit logischer Verknüpfung von Indizes über AND (Index ANDing) für den Zugriff auf eine lokale Tabelle verwenden soll. Es wird durch den komplexen Typ 'indexAndingType' definiert.

```
XML-Schema

   <xs:complexType name="indexAndingType">
      <xs:complexContent>
         <xs:extension base="extendedAccessType">
            <xs:sequence minOccurs="0">
               <xs:element name="NLJOIN" type="nestedLoopJoinType" minOccurs="1"
                  maxOccurs="unbounded"/>
            </xs:sequence>
            <xs:attribute name="STARJOIN" type="boolType" use="optional"/>
         </xs:extension>
      </xs:complexContent>
   </xs:complexType>
```

Beschreibung

Der komplexe Typ 'indexAndingType' ist eine Erweiterung des Typs 'extendedAccessType'. Wenn kein Attribut STARJOIN und keine NLJOIN-Elemente angegeben werden, wird 'indexAndingType' zu einer einfachen Erweiterung des Typs 'exten-

dedAccessType'. Der Typ 'extendedAccessType' erweitert den abstrakten Typ 'accessType', indem das optionale Attribut INDEX, optionale INDEX-Unterelemente, das optionale Attribut TYPE und das optionale Attribut ALLINDEXES hinzugefügt werden. Mit dem Attribut INDEX kann der Index angegeben werden, der als erster Index in einer Operation zum logischen Verknüpfen von Indizes über AND verwendet werden soll. Wenn das Attribut INDEX verwendet wird, wählt das Optimierungsprogramm zusätzliche Indizes und die Zugriffsreihenfolge auf der Basis einer Aufwandsberechnung aus. Mit den INDEX-Unterelementen kann der exakte Satz von Indizes und die Zugriffsreihenfolge angegeben werden. Die Reihenfolge, in der die INDEX-Unterelemente angegeben sind, bestimmt die Reihenfolge, in der die einzelnen Indexsuchen ausgeführt werden sollen. Die Angabe von INDEX-Unterelementen überschreibt die Angabe des Attributs INDEX.

- Wenn keine Indizes angegeben werden, wählt das Optimierungsprogramm sowohl die Indizes als auch die Zugriffsreihenfolge auf der Basis einer Aufwandsberechnung aus.
- Wenn Indizes entweder durch das Attribut oder durch Unterelemente angegeben werden, müssen diese Indizes für die durch das Attribut TABLE oder TABID angegebene Tabelle definiert sein.
- Wenn für die Tabelle keine Indizes definiert sind, wird die Zugriffsanforderung ignoriert und eine Fehlernachricht zurückgegeben.

Das Attribut TYPE, bei dem als einziger Wert XMLINDEX unterstützt wird, gibt dem Optimierungsprogramm an, dass auf eine Tabelle über mindestens einen Index zu XML-Daten zugegriffen werden soll.

Mit dem optionalen Attribut INDEX kann nur ein XML-Indexname angegeben werden, wenn das Attribut TYPE den Wert XMLINDEX hat. Für das optionale Attribut INDEX kann unabhängig von der Angabe für das Attribut TYPE ein relationaler Index angegeben werden. Der angegebene Index wird vom Optimierungsprogramm als führender Index eines IXAND-Plans verwendet. Vom Optimierungsprogramm werden je nach geschätztem Ausführungsaufwand weitere Indizes zum IXAND-Plan hinzugefügt.

Mit dem optionalen Element INDEX können nur mehrere Namen von Indizes zu XML-Daten als Indexelemente angegeben werden, wenn das Attribut TYPE den Wert XMLINDEX hat. Für die optionalen INDEX-Elemente können unabhängig von der Angabe für das Attribut TYPE relationale Indizes angegeben werden. Die angegebenen Indizes werden vom Optimierungsprogramm in der angegebenen Reihenfolge als Indizes eines IXAND-Plans verwendet.

Ist kein Attribut TYPE angegeben, sind INDEX-Attribute und -Elemente weiterhin für relationale Indizes gültig.

Wenn sowohl das Attribut INDEX als auch das Element INDEX angegeben werden, wird das Attribut INDEX ignoriert.

Das optionale Attribut ALLINDEXES, bei dem als einziger Wert TRUE unterstützt wird, kann nur angegeben werden, wenn das Attribut TYPE den Wert XMLINDEX hat. Wird dieses Attribut angegeben, muss das Optimierungsprogramm unabhängig vom Ausführungsaufwand alle entsprechenden relationalen Indizes und Indizes für XML-Daten in einem IXAND-Plan für den Zugriff auf die angegebene Tabelle verwenden.

Wenn das Attribut TYPE angegeben wird, das Attribut INDEX, das Element INDEX und das Attribut ALLINDEXES jedoch nicht, wählt das Optimierungspro-

gramm einen IXAND-Plan mit mindestens einem Index zu XML-Daten aus. Bei den übrigen Indizes im Plan kann es sich um relationale Indizes oder Indizes zu XML-Daten handeln. Reihenfolge und Auswahl der Indizes werden vom Optimierungsprogramm auf der Grundlage von Aufwandsberechnungen festgelegt.

Blockindizes müssen in einer Zugriffsanforderung mit logischer AND-Verknüpfung von Indizes vor Satzindizes angegeben werden. Wenn diese Voraussetzung nicht erfüllt wird, wird eine Fehlernachricht zurückgegeben. Die Zugriffsmethode mit logischer AND-Verknüpfung von Indizes erfordert, dass mindestens ein Vergleichselement indexiert werden kann. Wenn der Zugriff mit logischer AND-Verknüpfung von Indizes nicht auswählbar ist, weil das erforderliche Vergleichselement nicht vorhanden ist, wird die Zugriffsanforderung ignoriert und eine Fehlernachricht zurückgegeben. Wenn die Datenzugriffsmethode mit logischer AND-Verknüpfung von Indizes nicht im Suchbereich enthalten ist, der für die Anweisung wirksam ist, wird die Zugriffsanforderung ignoriert und eine Fehlernachricht zurückgegeben.

Mit dem Zugriffsanforderungselement IXAND können Sie einen Plan mit Sternjoin und logischer AND-Verknüpfung von Indizes anfordern. Das optionale Attribut STARJOIN im IXAND-Element gibt an, dass sich das IXAND-Element auf einen Plan mit Sternjoin und logischer AND-Verknüpfung von Indizes bezieht. NLJOIN-Elemente können Unterelemente von IXAND sein und müssen ordnungsgemäß konstruierte Sternjoin-Semi-Join-Verknüpfungen sein. `STARJOIN="FALSE"` gibt eine Anforderung für einen regulären Basiszugriffsplan mit logischer AND-Verknüpfung von Indizes an. `STARJOIN="TRUE"` gibt eine Anforderung für einen Sternjoinplan mit logischer AND-Verknüpfung von Indizes an. Der Standardwert wird durch den Kontext bestimmt: Wenn das IXAND-Element mindestens ein untergeordnetes Semi-Join-Element hat, ist der Standardwert TRUE. Andernfalls ist der Standardwert FALSE. Wenn `STARJOIN="TRUE"` angegeben wird, sind folgende Punkte zu beachten:

- Die Attribute INDEX, TYPE und ALLINDEXES können nicht angegeben werden.
- Es können keine INDEX-Elemente angegeben werden.

Wenn NLJOIN-Elemente angegeben werden, sind folgende Punkte zu beachten:

- Die Attribute INDEX, TYPE und ALLINDEXES können nicht angegeben werden.
- Es können keine INDEX-Elemente angegeben werden.
- Der einzige unterstützte Wert für das Attribut STARJOIN ist TRUE.

Das folgende Beispiel veranschaulicht eine Zugriffsanforderung mit logischer AND-Verknüpfung von Indizes:

```
SQL-Anweisung:

   select s.s_name, s.s_address, s.s_phone, s.s_comment
     from "Tpcd".parts, "Tpcd".suppliers s, "Tpcd".partsupp ps
     where p_partkey = ps.ps_partkey and
       s.s_suppkey = ps.ps_suppkey and
       p_size = 39 and
       p_type = 'BRASS' and
       s.s_nation in ('MOROCCO', 'SPAIN') and
       ps.ps_supplycost = (select min(ps1.ps_supplycost)
                             from "Tpcd".partsupp ps1, "Tpcd".suppliers s1
                             where "Tpcd".parts.p_partkey = ps1.ps_partkey and
                               s1.s_suppkey = ps1.ps_suppkey and
                               s1.s_nation = s.s_nation)
     order by s.s_name
     optimize for 1 row

Optimierungsrichtlinie:
```

```
<OPTGUIDELINES>
  <IXAND TABLE='"Tpcd".PARTS' FIRST='TRUE'>
    <INDEX IXNAME='ISIZE'/>
    <INDEX IXNAME='ITYPE'/>
  </IXAND>
</OPTGUIDELINES>
```

Die Zugriffsanforderung mit logischer AND-Verknüpfung von Indizes gibt an, dass auf die Tabelle PARTS im Hauptsubselect über eine logische AND-Verknüpfung von Indizes zugegriffen werden soll. Bei der ersten Indexsuche wird der Index ISIZE, bei der zweiten der Index ITYPE verwendet. Die Indizes werden über das Attribut IXNAME des Elements INDEX angegeben. Die Einstellung für das Attribut FIRST gibt an, dass die Tabelle PARTS die erste Tabelle in der Joinsequenz mit den Tabellen SUPPLIERS und PARTSUPP sowie abgeleiteten Tabellen in derselben Klausel FROM darstellen soll.

Das folgende Beispiel zeigt eine Sternjoinrichtlinie mit logischer AND-Verknüpfung von Indizes, die das erste Semi-Join-Element angibt, jedoch dem Optimierungsprogramm die Auswahl der übrigen Elemente überlässt. Außerdem überlässt sie dem Optimierungsprogramm die Auswahl der speziellen Zugriffsmethode für die äußere Tabelle und des Index für die innere Tabelle in der angegebenen Semi-Join-Verknüpfung.

```
<IXAND TABLE="F">
  <NLJOIN>
    <ACCESS TABLE="D1"/>
    <IXSCAN TABLE="F"/>
  </NLJOIN>
</IXAND>
```

Die folgende Richtlinie gibt alle Semi-Join-Verknüpfungen, einschließlich Details, an und lässt dem Optimierungsprogramm keine Auswahlmöglichkeiten für den Plan im IXAND-Element und unterhalb dieses Elements.

```
<IXAND TABLE="F" STARJOIN="TRUE">
  <NLJOIN>
    <TBSCAN TABLE="D1"/>
    <IXSCAN TABLE="F" INDEX="FX1"/>
  </NLJOIN>
  <NLJOIN>
    <TBSCAN TABLE="D4"/>
    <IXSCAN TABLE="F" INDEX="FX4"/>
  </NLJOIN>
  <NLJOIN>
    <TBSCAN TABLE="D3"/>
    <IXSCAN TABLE="F" INDEX="FX3"/>
  </NLJOIN>
</IXAND>
```

Zugriffsanforderungen mit logischem Verknüpfen von Indizes über OR (Index ORing):

Das Zugriffsanforderungselement IXOR kann verwendet werden, um anzugeben, dass das Optimierungsprogramm die Datenzugriffsmethode mit logischer Verknüpfung von Indizes über OR (Index ORing) für den Zugriff auf eine lokale Tabelle verwenden soll. Es wird durch den komplexen Typ 'indexOringType' definiert.

XML-Schema

```
<xs:complexType name="indexOringType">
```

```
      <xs:complexContent>
         <xs:extension base="accessType"/>
      </xs:complexContent>
   </xs:complexType>
```

Beschreibung

Der komplexe Typ 'indexOringType' ist eine einfache Erweiterung des abstrakten Typs 'accessType'. Es werden keine neuen Elemente oder Attribute hinzugefügt. Wenn die Zugriffsmethode mit logischer OR-Verknüpfung von Indizes nicht im Suchbereich enthalten ist, der für die Anweisung wirksam ist, wird die Zugriffsanforderung ignoriert und eine Nachricht SQL0437W mit Ursachencode 13 zurückgegeben. Das Optimierungsprogramm wählt die Vergleichselemente und Indizes, die in der OR-Verknüpfungsoperation von Indizes verwendet werden, auf der Basis einer Aufwandsberechnung aus. Die Zugriffsmethode mit logischer OR-Verknüpfung von Indizes erfordert, dass mindestens ein IN-Vergleichselement indexiert werden kann oder dass ein Vergleichselement mit Termen indexiert und durch eine logische OR-Operation verbunden werden kann. Wenn der Zugriff mit logischer OR-Verknüpfung von Indizes nicht auswählbar ist, weil das erforderliche Vergleichselement bzw. die erforderlichen Indizes nicht vorhanden sind, wird die Anforderung ignoriert und eine Nachricht SQL0437W mit Ursachencode 13 zurückgegeben.

Das folgende Beispiel veranschaulicht eine Zugriffsanforderung mit logischer Verknüpfung über OR:

```
SQL-Anweisung:

   select s.s_name, s.s_address, s.s_phone, s.s_comment
     from "Tpcd".parts, "Tpcd".suppliers s, "Tpcd".partsupp ps
     where p_partkey = ps.ps_partkey and
       s.s_suppkey = ps.ps_suppkey and
       p_size = 39 and
       p_type = 'BRASS' and
       s.s_nation in ('MOROCCO', 'SPAIN') and
       ps.ps_supplycost = (select min(ps1.ps_supplycost)
                             from "Tpcd".partsupp ps1, "Tpcd".suppliers s1
                             where "Tpcd".parts.p_partkey = ps1.ps_partkey and
                               s1.s_suppkey = ps1.ps_suppkey and
                               s1.s_nation = s.s_nation)
     order by s.s_name
     optimize for 1 row

Optimierungsrichtlinie:

   <OPTGUIDELINES>
     <IXOR TABLE='S'/>
   </OPTGUIDELINES>
```

Diese Zugriffsanforderung mit logischer Verknüpfung von Indizes über OR gibt an, dass auf die Tabelle SUPPLIERS, auf die im Hauptsubselect verwiesen wird, über die Datenzugriffsmethode mit logischer Verknüpfung von Indizes über OR zugegriffen werden soll. Das Optimierungsprogramm wählt die geeigneten Vergleichselemente und Indizes für die Operation zur OR-Verknüpfung von Indizes anhand des geschätzten Ausführungsaufwands aus.

Zugriffsanforderungen für Indexsuchen:

Das Zugriffsanforderungselement IXSCAN kann verwendet werden, um anzugeben, dass das Optimierungsprogramm eine Indexsuche für den Zugriff auf eine lokale Tabelle verwenden soll. Es wird durch den komplexen Typ 'indexScanType' definiert.

XML-Schema

```
<xs:complexType name="indexScanType">
   <xs:complexContent>
      <xs:extension base="accessType">
         <xs:attribute name="INDEX" type="xs:string" use="optional"/>
      </xs:extension>
   </xs:complexContent>
</xs:complexType>
```

Beschreibung

Der komplexe Typ 'indexScanType' erweitert den abstrakten Typ 'accessType', indem er das optionale Attribut INDEX hinzufügt. Das Attribut INDEX gibt den Namen des Index an, der für den Zugriff auf die Tabelle zu verwenden ist.

- Wenn die Zugriffsmethode der Indexsuche nicht im Suchbereich enthalten ist, der für die Anweisung wirksam ist, wird die Zugriffsanforderung ignoriert und eine Nachricht SQL0437W mit Ursachencode 13 zurückgegeben.
- Wenn das Attribut INDEX angegeben wird, muss es einen Index angeben, der für die durch das Attribut TABLE oder TABID angegebene Tabelle definiert ist. Wenn der Index nicht vorhanden ist, wird die Zugriffsanforderung ignoriert und eine Nachricht SQL0437W mit Ursachencode 13 zurückgegeben.
- Wenn das Attribut INDEX nicht angegeben wird, wählt das Optimierungsprogramm einen Index auf der Basis einer Aufwandsberechnung aus. Wenn für die Zieltabelle keine Indizes definiert sind, wird die Zugriffsanforderung ignoriert und eine Nachricht SQL0437W mit Ursachencode 13 zurückgegeben.

Die folgende Richtlinie stellt ein Beispiel für eine Zugriffsanforderung für Indexsuche dar:

```
<OPTGUIDELINES>
  <IXSCAN TABLE='S' INDEX='I_SUPPKEY'/>
</OPTGUIDELINES>
```

Anforderungen für Vorablesezugriff über Listen:

Das Zugriffsanforderungselement LPREFETCH kann verwendet werden, um anzugeben, dass das Optimierungsprogramm eine Indexsuche mit Vorablesezugriff über Listen für den Zugriff auf eine lokale Tabelle verwenden soll. Es wird durch den komplexen Typ 'listPrefetchType' definiert.

XML-Schema

```
<xs:complexType name="listPrefetchType">
   <xs:complexContent>
      <xs:extension base="accessType">
         <xs:attribute name="INDEX" type="xs:string" use="optional"/>
      </xs:extension>
   </xs:complexContent>
</xs:complexType>
```

Beschreibung

Der komplexe Typ 'listPrefetchType' erweitert den abstrakten Typ 'accessType', indem er das optionale Attribut INDEX hinzufügt. Das Attribut INDEX gibt den Namen des Index an, der für den Zugriff auf die Tabelle zu verwenden ist.

- Wenn die Zugriffsmethode mit Vorablesezugriff über Listen nicht im Suchbereich enthalten ist, der für die Anweisung wirksam ist, wird die Zugriffsanforderung ignoriert und eine Nachricht SQL0437W mit Ursachencode 13 zurückgegeben.

- Die Zugriffsmethode mit Vorablesezugriff über Listen erfordert, dass mindestens ein Vergleichselement indexiert werden kann. Wenn der Vorablesezugriff über Listen nicht auswählbar ist, weil das erforderliche Vergleichselement nicht vorhanden ist, wird die Zugriffsanforderung ignoriert und eine Nachricht SQL0437W mit Ursachencode 13 zurückgegeben.
- Wenn das Attribut INDEX angegeben wird, muss es einen Index angeben, der für die durch das Attribut TABLE oder TABID angegebene Tabelle definiert ist. Wenn der Index nicht vorhanden ist, wird die Zugriffsanforderung ignoriert und eine Nachricht SQL0437W mit Ursachencode 13 zurückgegeben.
- Wenn das Attribut INDEX nicht angegeben wird, wählt das Optimierungsprogramm einen Index auf der Basis einer Aufwandsberechnung aus. Wenn für die Zieltabelle keine Indizes definiert sind, wird die Zugriffsanforderung ignoriert und eine Nachricht SQL0437W mit Ursachencode 13 zurückgegeben.

Die folgende Richtlinie ist ein Beispiel für eine Zugriffsanforderung für Vorablesezugriff über Listen:

```
<OPTGUIDELINES>
  <LPREFETCH TABLE='S1' INDEX='I_SNATION'/>
</OPTGUIDELINES>
```

Zugriffsanforderungen für Tabellensuchen:

Das Zugriffsanforderungselement TBSCAN kann verwendet werden, um anzugeben, dass das Optimierungsprogramm eine sequenzielle Tabellensuche für den Zugriff auf eine lokale Tabelle verwenden soll. Es wird durch den komplexen Typ 'tableScanType' definiert.

XML-Schema

```
<xs:complexType name="tableScanType">
   <xs:complexContent>
      <xs:extension base="accessType"/>
   </xs:complexContent>
</xs:complexType>
```

Beschreibung

Der komplexe Typ 'tableScanType' ist eine einfache Erweiterung des abstrakten Typs 'accessType'. Es werden keine neuen Elemente oder Attribute hinzugefügt. Wenn die Zugriffsmethode der Tabellensuche nicht im Suchbereich enthalten ist, der für die Anweisung wirksam ist, wird die Zugriffsanforderung ignoriert und eine Nachricht SQL0437W mit Ursachencode 13 zurückgegeben.

Die folgende Richtlinie ist ein Beispiel für eine Zugriffsanforderung für Tabellensuche:

```
<OPTGUIDELINES>
  <TBSCAN TABLE='S1'/>
</OPTGUIDELINES>
```

Zugriffsanforderungen mit logischer Verknüpfung von XML-Indizes über AND und OR:

Mit dem Zugriffsanforderungselement XANDOR kann angegeben werden, dass das Optimierungsprogramm XML-Datensuchen über mehrere logisch verknüpfte (AND/OR) Indizes für den Zugriff auf eine lokale Tabelle verwenden soll. Das Element wird durch den komplexen Typ 'XANDORType' definiert.

XML-Schema

```
<xs:complexType name="XANDORType">
```

```
      <xs:complexContent>
         <xs:extension base="accessType"/>
      </xs:complexContent>
   </xs:complexType>
```

Beschreibung

Der komplexe Typ 'XANDORType' ist eine einfache Erweiterung des abstrakten Typs 'accessType'. Es werden keine neuen Elemente oder Attribute hinzugefügt.

Beispiel

Angenommen, es soll die folgende Abfrage ausgeführt werden:

```
SELECT * FROM security
  WHERE trans_date = CURRENT DATE AND
    XMLEXISTS('$SDOC/Security/SecurityInformation/
      StockInformation[Industry = "Software"]') AND
    XMLEXISTS('$SDOC/Security/Symbol[.="IBM"]')
```

Die folgende Richtlinie für XANDOR gibt an, dass der Zugriff auf die Tabelle SECURITY über eine XANDOR-Operation erfolgen soll, die für alle entsprechenden XML-Indizes ausgeführt wird. Relationale Indizes für die Tabelle SECURITY werden dabei nicht berücksichtigt, da relationale Indizes nicht mit einem Operator XANDOR verwendet werden können.

```
<OPTGUIDELINES>
  <XANDOR TABLE='SECURITY'/>
</OPTGUIDELINES>
```

Zugriffsanforderungen für XML-Indexsuchen:

Mit dem Zugriffsanforderungselement XISCAN kann angegeben werden, dass das Optimierungsprogramm Datensuchen über Indizes für XML-Daten für den Zugriff auf eine lokale Tabelle verwenden soll. Es wird durch den komplexen Typ 'indexScanType' definiert.

```
XML-Schema

   <xs:complexType name="indexScanType">
      <xs:complexContent>
         <xs:extension base="accessType"/>
            <xs:attribute name="INDEX" type="xs:string" use="optional"/>
         </xs:extension>
      </xs:complexContent>
   </xs:complexType>
```

Beschreibung

Der komplexe Typ 'indexScanType' erweitert den abstrakten Typ 'accessType', indem er das optionale Attribut INDEX hinzufügt. Das Attribut INDEX gibt den Namen des Index für XML-Daten an, der für den Zugriff auf die Tabelle zu verwenden ist.

- Wenn die Zugriffsmethode mit einer Datensuche über Indizes für XML-Daten nicht in dem Suchbereich enthalten ist, der für die Anweisung wirksam ist, wird die Zugriffsanforderung ignoriert und eine Nachricht SQL0437W mit Ursachencode 13 zurückgegeben.
- Wenn das Attribut INDEX angegeben wird, muss es einen Index für XML-Daten angeben, der für die durch das Attribut TABLE oder TABID angegebene Tabelle definiert ist. Wenn der Index nicht vorhanden ist, wird die Zugriffsanforderung ignoriert und eine Nachricht SQL0437W mit Ursachencode 13 zurückgegeben.

- Wenn das Attribut INDEX nicht angegeben wird, wählt das Optimierungsprogramm einen Index für XML-Daten auf der Basis einer Aufwandsberechnung aus. Wenn für die Zieltabelle keine Indizes für XML-Daten definiert sind, wird die Zugriffsanforderung ignoriert und eine Nachricht SQL0437W mit Ursachencode 13 zurückgegeben.

Beispiel

Angenommen, es soll die folgende Abfrage ausgeführt werden:

```
SELECT * FROM security
  WHERE XMLEXISTS('$SDOC/Security/SecurityInformation/
    StockInformation[Industry = "OfficeSupplies"]')
```

Die folgende XISCAN-Richtlinie gibt an, dass auf die Tabelle SECURITY über einen XML-Index mit der Bezeichnung SEC_INDUSTRY zugegriffen werden soll.

```
<OPTGUIDELINES>
  <XISCAN TABLE='SECURITY' INDEX='SEC_INDUSTRY'/>
</OPTGUIDELINES>
```

Joinanforderungen:

Die Gruppe 'joinRequest' definiert den Satz der gültigen Joinanforderungselemente. Eine Joinanforderung gibt eine Methode zum Verknüpfen (Join) zweier Tabellen an.

XML-Schema

```
<xs:group name="joinRequest">
   <xs:choice>
      <xs:element name="NLJOIN" type="nestedLoopJoinType"/>
      <xs:element name="HSJOIN" type="hashJoinType"/>
      <xs:element name="MSJOIN" type="mergeJoinType"/>
      <xs:element name="JOIN" type="anyJoinType"/>
   </xs:choice>
</xs:group>
```

Beschreibung

- NLJOIN, MSJOIN und HSJOIN

  Diese Elemente entsprechen den Methoden Join mit Verschachtelungsschleife (Nested Loop Join), Mischjoin (Merge Join) bzw. Hash-Join.
- JOIN

  Dieses Element, mit dem das Optimierungsprogramm veranlasst wird, die Joinmethode auszuwählen, kann verwendet werden, wenn die Joinreihenfolge nicht von primärem Interesse ist.

Alle Joinanforderungselemente enthalten zwei Unterelemente, die die Eingabetabellen der Joinoperation darstellen. Joinanforderungen können außerdem das optionale Attribut FIRST angeben.

Die folgende Richtlinie ist ein Beispiel für eine Joinanforderung:

```
<OPTGUIDELINES>
  <HSJOIN>
    <ACCESS TABLE='S1'/>
    <IXSCAN TABLE='PS1'/>
  </HSJOIN>
</OPTGUIDELINES>
```

Die Joinreihenfolge wird letzten Endes durch die Verschachtelungsreihenfolge bestimmt. Das folgende Beispiel veranschaulicht, wie umfangreichere Joinanforderungen aus kleineren Joinanforderungen erstellt werden können:

```
<OPTGUIDELINES>
  <MSJOIN>
    <NLJOIN>
      <IXSCAN TABLE='"Tpcd".Parts'/>
      <IXSCAN TABLE="PS"/>
    </NLJOIN>
    <IXSCAN TABLE='S'/>
  </MSJOIN>
</OPTGUIDELINES>
```

Jointypen:

Allgemeine Aspekte aller Joinanforderungselemente werden durch den abstrakten Typ 'joinType' definiert.

XML-Schema

```
<xs:complexType name="joinType" abstract="true">
   <xs:choice minOccurs="2" maxOccurs="2">
      <xs:group ref="accessRequest"/>
      <xs:group ref="joinRequest"/>
   </xs:choice>
   <xs:attribute name="FIRST" type="xs:string" use="optional" fixed="TRUE"/>
</xs:complexType>
```

Beschreibung

Joinanforderungselemente, die den komplexen Typ 'joinType' erweitern, müssen genau zwei Unterelemente besitzen. Beide Unterelemente können ein aus der Gruppe 'accessRequest' ausgewähltes Zugriffsanforderungselement oder ein aus der Gruppe 'joinRequest' ausgewähltes Joinanforderungselement sein. Das erste Unterelement, dass in der Joinanforderung auftritt, gibt die äußere Tabelle der Joinoperation an. Das zweite Element gibt die innere Tabelle an.

Wenn das Attribut FIRST angegeben wird, muss es den Wert TRUE haben. Durch Hinzufügen des Attributs FIRST in einem Joinanforderungselement wird angeben, dass ein Ausführungsplan gewünscht wird, bei dem die Tabellen, die in der Joinanforderung angegeben sind, die äußersten Tabellen in der Joinreihenfolge für die entsprechende FROM-Klausel sind. Nur in einer Zugriffs- oder Joinanforderung pro FROM-Klausel kann das Attribut FIRST angegeben werden. Wenn mehrere Zugriffs- oder Joinanforderungen, die Tabellen derselben FROM-Klausel angeben, das Attribut FIRST enthalten, werden alle außer der ersten Anforderung ignoriert und eine Nachricht SQL0437W mit Ursachencode 13 zurückgegeben.

Beliebige Joinanforderungen:

Das Joinanforderungselement JOIN kann verwendet werden, um anzugeben, dass das Optimierungsprogramm eine geeignete Methode für die Verknüpfung (Join) von zwei Tabellen in einer bestimmten Reihenfolge auswählen soll.

Jede der beiden Tabellen kann eine lokale oder abgeleitete Tabelle sein, wie dies durch ein Unterelement einer Zugriffsanforderung angegeben wird, oder sie kann das Ergebnis einer Joinoperation sein, wie dies durch ein Unterelement einer Joinanforderung angegeben wird. Eine abgeleitete Tabelle ist das Ergebnis einer anderen Subselect-Anweisung. Dieses Joinanforderungselement wird durch den komplexen Typ 'anyJoinType' definiert.

XML-Schema

```
<xs:complexType name="anyJoinType">
   <xs:complexContent>
      <xs:extension base="joinType"/>
   </xs:complexContent>
</xs:complexType>
```

Beschreibung

Der komplexe Typ 'anyJoinType' ist eine einfache Erweiterung des abstrakten Typs 'joinType'. Es werden keine neuen Elemente oder Attribute hinzugefügt.

Das folgende Beispiel zeigt, wie das Joinanforderungselement JOIN verwendet wird, um eine bestimmte Joinreihenfolge für eine Reihe von Tabellen zu erzwingen:

SQL-Anweisung:

```
  select s.s_name, s.s_address, s.s_phone, s.s_comment
     from "Tpcd".parts, "Tpcd".suppliers s, "Tpcd".partsupp ps
     where p_partkey = ps.ps_partkey and
       s.s_suppkey = ps.ps_suppkey and
       p_size = 39 and
       p_type = 'BRASS' and
       s.s_nation in ('MOROCCO', 'SPAIN') and
       ps.ps_supplycost = (select min(ps1.ps_supplycost)
                             from "Tpcd".partsupp ps1, "Tpcd".suppliers s1
                             where "Tpcd".parts.p_partkey = ps1.ps_partkey and
                               s1.s_suppkey = ps1.ps_suppkey and
                               s1.s_nation = s.s_nation)
     order by s.s_name
```

Optimierungsrichtlinie:

```
   <OPTGUIDELINES>
     <JOIN>
       <JOIN>
         <ACCESS TABLE='Tpcd".PARTS'/>
         <ACCESS TABLE='S'/>
       </JOIN>
       <ACCESS TABLE='PS'>
     </JOIN>
   </OPTGUIDELINES>
```

Die Joinanforderungselemente JOIN geben an, dass die Tabelle PARTS im Hauptsubselect mit der Tabelle SUPPLIERS und das Ergebnis aus dieser Joinoperation mit der Tabelle PARTSUPP verknüpft werden soll. Die für diese spezielle Joinsequenz verwendete Methode wird vom Optimierungsprogramm je nach Ausführungsaufwand ausgewählt.

HSJOIN-Anforderungen:

Das Joinanforderungselement HSJOIN kann verwendet werden, um anzugeben, dass das Optimierungsprogramm zwei Tabellen durch die Hash-Joinmethode verknüpfen soll.

Jede der beiden Tabellen kann eine lokale oder abgeleitete Tabelle sein, wie dies durch ein Unterelement einer Zugriffsanforderung angegeben wird, oder sie kann das Ergebnis einer Joinoperation sein, wie dies durch ein Unterelement einer Join-

anforderung angegeben wird. Eine abgeleitete Tabelle ist das Ergebnis einer anderen Subselect-Anweisung. Dieses Joinanforderungselement wird durch den komplexen Typ 'hashJoinType' definiert.

XML-Schema

```
<xs:complexType name="hashJoinType">
   <xs:complexContent>
      <xs:extension base="joinType"/>
   </xs:complexContent>
</xs:complexType>
```

Beschreibung

Der komplexe Typ 'hashJoinType' ist eine einfache Erweiterung des abstrakten Typs 'joinType'. Es werden keine neuen Elemente oder Attribute hinzugefügt. Wenn die Hash-Joinmethode nicht im Suchbereich enthalten ist, der für die Anweisung wirksam ist, wird die Joinanforderung ignoriert und eine Nachricht SQL0437W mit Ursachencode 13 zurückgegeben.

Die folgende Richtlinie ist ein Beispiel für eine HSJOIN-Anforderung:

```
<OPTGUIDELINES>
  <HSJOIN>
    <ACCESS TABLE='S1'/>
    <IXSCAN TABLE='PS1'/>
  </HSJOIN>
</OPTGUIDELINES>
```

MSJOIN-Anforderungen:

Das Joinanforderungselement MSJOIN kann verwendet werden, um anzugeben, dass das Optimierungsprogramm zwei Tabellen durch die Mischjoinmethode (Merge Join) verknüpfen soll.

Jede der beiden Tabellen kann eine lokale oder abgeleitete Tabelle sein, wie dies durch ein Unterelement einer Zugriffsanforderung angegeben wird, oder sie kann das Ergebnis einer Joinoperation sein, wie dies durch ein Unterelement einer Joinanforderung angegeben wird. Eine abgeleitete Tabelle ist das Ergebnis einer anderen Subselect-Anweisung. Dieses Joinanforderungselement wird durch den komplexen Typ 'mergeJoinType' definiert.

XML-Schema

```
<xs:complexType name="mergeJoinType">
   <xs:complexContent>
      <xs:extension base="joinType"/>
   </xs:complexContent>
</xs:complexType>
```

Beschreibung

Der komplexe Typ 'mergeJoinType' ist eine einfache Erweiterung des abstrakten Typs 'joinType'. Es werden keine neuen Elemente oder Attribute hinzugefügt. Wenn die Mischjoinmethode nicht im Suchbereich enthalten ist, der für die Anweisung wirksam ist, wird die Joinanforderung ignoriert und eine Nachricht SQL0437W mit Ursachencode 13 zurückgegeben.

Die folgende Richtlinie ist ein Beispiel für eine Mischjoinanforderung:

```
<OPTGUIDELINES>
  <MSJOIN>
    <NLJOIN>
```

```
      <IXSCAN TABLE='"Tpcd".Parts'/>
      <IXSCAN TABLE="PS"/>
    </NLJOIN>
    <IXSCAN TABLE='S'/>
  </MSJOIN>
</OPTGUIDELINES>
```

NLJOIN-Anforderungen:

Das Joinanforderungselement NLJOIN kann verwendet werden, um anzugeben, dass das Optimierungsprogramm zwei Tabellen durch die Joinmethode mit Verschachtelungsschleife (Nested Loop Join) verknüpfen soll.

Jede der beiden Tabellen kann eine lokale oder abgeleitete Tabelle sein, wie dies durch ein Unterelement einer Zugriffsanforderung angegeben wird, oder sie kann das Ergebnis einer Joinoperation sein, wie dies durch ein Unterelement einer Joinanforderung angegeben wird. Eine abgeleitete Tabelle ist das Ergebnis einer anderen Subselect-Anweisung. Dieses Joinanforderungselement wird durch den komplexen Typ 'nestedLoopJoinType' definiert.

```
XML-Schema

<xs:complexType name="nestedLoopJoinType">
   <xs:complexContent>
      <xs:extension base="joinType"/>
   </xs:complexContent>
</xs:complexType>
```

Beschreibung

Der komplexe Typ 'nestedLoopJoinType' ist eine einfache Erweiterung des abstrakten Typs 'joinType'. Es werden keine neuen Elemente oder Attribute hinzugefügt. Wenn die Joinmethode mit Verschachtelungsschleife nicht im Suchbereich enthalten ist, der für die Anweisung wirksam ist, wird die Joinanforderung ignoriert und eine Nachricht SQL0437W mit Ursachencode 13 zurückgegeben.

Die folgende Richtlinie ist ein Beispiel für eine NLJOIN-Anforderung:

```
<OPTGUIDELINES>
  <NLJOIN>
    <IXSCAN TABLE='"Tpcd".Parts'/>
    <IXSCAN TABLE="PS"/>
  </NLJOIN>
</OPTGUIDELINES>
```

Tabelle SYSTOOLS.OPT_PROFILE:

Die Tabelle SYSTOOLS.OPT_PROFILE enthält alle Optimierungsprofile.

Diese Tabelle kann mit zwei Methoden erstellt werden:

- Rufen Sie die Prozedur SYSINSTALLOBJECTS auf:

  ```
  db2 "call sysinstallobjects('opt_profiles', 'c', '', '')"
  ```

- Führen Sie die Anweisung CREATE TABLE aus:

  ```
  create table systools.opt_profile (
    schema  varchar(128) not null,
    name    varchar(128) not null,
    profile blob (2m)    not null,
    primary key (schema, name)
  )
  ```

Die Spalten in der Tabelle SYSTOOLS.OPT_PROFILE sind wie folgt definiert:

SCHEMA
Gibt den Schemanamen für ein Optimierungsprofil an. Der Name kann bis zu 30 alphanumerische Zeichen und Unterstreichungszeichen enthalten, muss jedoch wie gezeigt als Typ VARCHAR(128) definiert werden.

NAME
Gibt den Basisnamen für ein Optimierungsprofil an. Der Name kann bis zu 128 alphanumerische Zeichen oder Unterstreichungszeichen enthalten.

PROFILE
Gibt ein XML-Dokument an, in dem das Optimierungsprofil definiert ist.

Trigger zum Löschen des Cache für das Optimierungsprofil:

Der Cache für das Optimierungsprofil wird automatisch gelöscht (FLUSH), wenn ein Eintrag in der Tabelle SYSTOOLS.OPT_PROFILE aktualisiert oder gelöscht wird.

Die folgende SQL-Prozedur und die folgenden Trigger müssen erstellt werden, bevor das automatische Löschen des Profilcache erfolgen kann.

```
CREATE PROCEDURE SYSTOOLS.OPT_FLUSH_CACHE( IN SCHEMA VARCHAR(128),
                                           IN NAME VARCHAR(128) )
LANGUAGE SQL
MODIFIES SQL DATA
BEGIN ATOMIC
 -- FLUSH stmt (33) + angegebenes Schema (130) + Punkt (1) + angegebener Name (130) = 294
 DECLARE FSTMT VARCHAR(294) DEFAULT 'FLUSH OPTIMIZATION PROFILE CACHE '; --

 IF NAME IS NOT NULL THEN
   IF SCHEMA IS NOT NULL THEN
     SET FSTMT = FSTMT || '"' || SCHEMA || '".'; --
   END IF; --

   SET FSTMT = FSTMT || '"' || NAME || '"'; --

   EXECUTE IMMEDIATE FSTMT; --
 END IF; --
END;

CREATE TRIGGER SYSTOOLS.OPT_PROFILE_UTRIG AFTER UPDATE ON SYSTOOLS.OPT_PROFILE
REFERENCING OLD AS O
FOR EACH ROW
  CALL SYSTOOLS.OPT_FLUSH_CACHE( O.SCHEMA, O.NAME );

CREATE TRIGGER SYSTOOLS.OPT_PROFILE_DTRIG AFTER DELETE ON SYSTOOLS.OPT_PROFILE
REFERENCING OLD AS O
FOR EACH ROW
  CALL SYSTOOLS.OPT_FLUSH_CACHE( O.SCHEMA, O.NAME );
```

Verwalten der Tabelle SYSTOOLS.OPT_PROFILE:

Optimierungsprofile müssen einen eindeutigen, durch ein Schema qualifizierten Namen haben und in der Tabelle SYSTOOLS.OPT_PROFILE gespeichert werden. Zur Verwaltung der Dateien in dieser Tabelle können die Befehle **LOAD**, **IMPORT** und **EXPORT** verwendet werden.

Zum Beispiel können mithilfe des Befehls **IMPORT** von einem beliebigen DB2-Client aus Daten in die Tabelle SYSTOOLS.OPT_PROFILE eingefügt oder dort aktualisiert werden. Der Befehl **EXPORT** kann zum Kopieren eines Profils aus der Tabelle SYSTOOLS.OPT_PROFILE in eine Datei verwendet werden.

Das folgende Beispiel zeigt, wie drei neue Profile in die Tabelle SYSTOOLS.OPT-_PROFILE eingefügt werden. Nehmen Sie dazu an, dass sich die Dateien im aktuellen Verzeichnis befinden.

1. Erstellen Sie eine Eingabedatei (z. B. `profiledata`) mit dem Schema, dem Namen und dem Dateinamen für jedes Profil in einer separaten Zeile:

```
"ROBERT","PROF1","ROBERT.PROF1.xml"
"ROBERT","PROF2","ROBERT.PROF2.xml"
"DAVID", "PROF1","DAVID.PROF1.xml"
```

2. Führen Sie den Befehl **IMPORT** aus:

```
import from profiledata of del
  modified by lobsinfile
  insert into systools.opt_profile
```

Zum Aktualisieren vorhandener Zeilen verwenden Sie die Option INSERT_UPDATE im Befehl **IMPORT**:

```
import from profiledata of del
  modified by lobsinfile
  insert_update into systools.opt_profile
```

Zum Kopieren des Profils ROBERT.PROF1 in die Datei `ROBERT.PROF1.xml` unter der Annahme, dass das Profil weniger als 32.700 Byte lang ist, verwenden Sie den Befehl **EXPORT**:

```
export to robert.prof1.xml of del
  select profile from systools.opt_profile
    where schema='ROBERT' and name='PROF1'
```

Weitere Informationen, zum Beispiel auch dazu, wie mehr als 32.700 Byte an Daten exportiert werden, finden Sie in der Beschreibung zum „Befehl EXPORT".

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 so aufgeteilt werden, dass 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 durch Kollokation zusammengefasst werden.

Abhängig von der Größe der Tabelle reduziert das Verteilen der Daten über mehrere Datenbankpartitionen die geschätzte Dauer der Abfrageausführung. 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.

Erfassen präziser Katalogstatistiken, einschließlich Verwendung erweiterter Statistikfunktionen

Präzise Datenbankstatistiken sind für die Abfrageoptimierung von entscheidender Bedeutung. Führen Sie regelmäßig das Dienstprogramm RUNSTATS für alle Tabellen aus, die für die Abfrageleistung kritisch sind.

Es kann auch sinnvoll sein, Statistiken zu Systemkatalogtabellen zu erfassen, wenn eine Anwendung diese Tabellen direkt abfragt und erhebliche Katalogaktualisierungsaktivitäten zu verzeichnen sind, wie zum Beispiel durch die Ausführung von DDL-Anweisungen (DDL, Data Definition Language, Datendefinitionssprache). Die automatische Statistikerfassung kann aktiviert werden, um den DB2-Datenserver zu veranlassen, eine RUNSTATS-Operation automatisch auszuführen. Die Echtzeitstatistikerfassung kann aktiviert werden, um den DB2-Datenserver zu veranlassen, noch mehr zeitgerechte Statistiken, die unmittelbar vor der Optimierung von Abfragen erfasst werden, bereitzustellen.

Wenn Sie Statistikdaten manuell mit dem Befehl **RUNSTATS** erfassen, sollten Sie mindestens die folgenden Optionen verwenden:

```
RUNSTATS ON TABLE DB2USER.DAILY_SALES
  WITH DISTRIBUTION AND SAMPLED DETAILED INDEXES ALL
```

Verteilungsstatistiken geben dem Optimierungsprogramm Anhaltspunkte über eine ungleiche Datenverteilung. Detaillierte Indexstatistiken stellen mehr Details über die erforderlichen E/A-Operationen zum Abrufen von Datenseiten bereit, wenn auf die Tabelle über einen bestimmten Index zugegriffen wird. Die Erfassung detaillierter Indexstatistiken benötigt beträchtlichen Verarbeitungszeitaufwand und Speicher bei großen Tabellen. Die Option SAMPLED stellt detaillierte Indexstatistiken mit annähernd identischer Präzision bereit, erfordert jedoch nur einen Bruchteil der CPU-Zeit und Speicherressourcen. Diese Standardoptionen werden auch bei der automatischen Statistikerfassung verwendet, wenn für eine Tabelle kein Statistikprofil bereitgestellt wurde.

Ziehen Sie zur Verbesserung der Abfrageleistung eine Erfassung etwas erweiterter Statistiken, wie Spaltengruppenstatistiken oder LIKE-Statistiken oder die Erstellung von Statistiksichten in Betracht.

Spaltengruppenstatistiken

Wenn Ihre Abfrage mehr als ein Joinvergleichselement enthält, mit dem zwei Tabellen verknüpft werden, berechnet das DB2-Optimierungsprogramm die Selektivität der einzelnen Vergleichselemente, bevor es einen Plan zur Ausführung der Abfrage auswählt.

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
```

Das Optimierungsprogramm geht davon aus, dass die beiden Vergleichselemente unabhängig sind, das heißt, dass alle Variationen von Elastizität für jede Farbe vorkommen. 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.

Möglicherweise sind diese beiden Vergleichselemente jedoch nicht unabhängig. Zum Beispiel könnten hochelastische Materialien in nur wenigen Farben und die sehr wenig elastischen Materialien nur in einigen anderen Farben verfügbar sein, die sich von den Farben der elastischen Materialien unterscheiden. In diesem Fall schließt die kombinierte Selektivität der Vergleichselemente weniger Zeilen aus, sodass die Abfrage mehr Zeilen zurückgibt. Ohne diese Informationen würde das Optimierungsprogramm vielleicht nicht mehr den besten Plan auswählen.

Zur Erfassung der Spaltengruppenstatistiken für die Spalten PRODUCT.COLOR und PRODUCT.ELASTICITY, führen Sie den folgenden Befehl **RUNSTATS** aus:

```
RUNSTATS ON TABLE PRODUCT ON COLUMNS ((COLOR, ELASTICITY))
```

Das Optimierungsprogramm verwendet diese Statistiken, um Fälle von Korrelation zu erkennen und die errechneten kombinierten Selektivitäten von korrelierten Vergleichselementen dynamisch anzupassen, um so eine präzisere Schätzung der Größe und des Aufwands eines Joins zu erhalten.

Wenn eine Abfrage Daten nach Schlüsselwörtern wie GROUP BY oder DISTINCT gruppiert, bieten Spaltengruppenstatistiken dem Optimierungsprogramm auch die Möglichkeit, die Anzahl der unterschiedlichen Gruppierungen zu berechnen.

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 (und in diesem Fall auch die Anzahl der zurückgegebenen Zeilen) dem Produkt aus der Anzahl der unterschiedlichen Werte in den Spalten DEPTNO, MGR und YEAR_HIRED. Dieser Schätzwert basiert auf der Annahme, dass die Gruppierungsschlüsselspalten unabhängig sind. Diese Annahme kann jedoch falsch sein, wenn jeder Manager genau eine Abteilung leitet. Darüber hinaus ist es unwahrscheinlich, dass jede Abteilung jedes Jahr Mitarbeiter einstellt. 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.

Spaltengruppenstatistikdaten, die für die Spalten DEPTNO, MGR und YEAR_HIRED erfasst werden, liefern dem Optimierungsprogramm die genaue Anzahl der unterschiedlichen Gruppierungen für die vorige Abfrage:

```
RUNSTATS ON TABLE EMPLOYEE ON COLUMNS ((DEPTNO, MGR, YEAR_HIRED))
```

Neben der Korrelation von Joinvergleichselementen verwaltet das Optimierungsprogramm die Korrelation mit einfachen Gleichheitsvergleichselementen wie zum Beispiel:

```
DEPTNO = 'Sales' AND  MGR = 'John'
```

In diesem Beispiel sind Vergleichselemente für die Spalte DEPTNO in der Tabelle EMPLOYEE wahrscheinlich von Vergleichselementen für die Spalte YEAR unabhängig. Vergleichselemente für die Spalten DEPTNO und MGR sind jedoch bestimmt nicht unabhängig, da jede Abteilung (DEPTNO) normalerweise zu einem Zeitpunkt jeweils von einem Manager (MGR) geleitet wird. Das Optimierungspro-

gramm verwendet Statistikdaten über Spalten, um die kombinierte Zahl unterschiedlicher Werte festzustellen, und passt anschließend die Kardinalitätsschätzung an, um die Korrelation zwischen Spalten mit einzukalkulieren.

Korrelation einfacher Gleichheitsvergleichselemente

Neben der Korrelation von Joinvergleichselementen verwaltet das Optimierungsprogramm die Korrelation mit einfachen Gleichheitsvergleichselementen des Typs `SPALTE =`.

Betrachten Sie zum Beispiel eine Tabelle mit verschiedenen Typen von Fahrzeugen, die jeweils mit MARKE (d. h. Hersteller), MODELL, JAHR, FARBE und AUSFÜHRUNG (z. B. Limousine, Kombi oder Geländewagen) eingetragen sind. Da die weitaus meisten Hersteller die gleichen Standardfarben 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. Auf MARKE und MODELL basierende Vergleichselemente sind jedoch nicht 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.

Wenn für die beiden Spalten MARKE und MODELL ein Index vorhanden ist oder Spaltengruppenstatistiken erfasst sind, verwendet das Optimierungsprogramm Statistikdaten über den Index oder die Spalten, um die kombinierte Zahl an unterschiedlichen Werten festzustellen und die Selektivitäts- oder Kardinalitätsschätzungen für die Korrelation zwischen diesen beiden Spalten anzupassen. Wenn die Vergleichselemente lokale Gleichheitsvergleichselemente sind, benötigt das Optimierungsprogramm keinen eindeutigen Index, um eine Anpassung vorzunehmen.

Statistiksichten

Das aufwandsbasierte DB2-Optimierungsprogramm verwendet einen Schätzwert für die Anzahl von Zeilen (Kardinalität), die von einem Zugriffsplanoperator verarbeitet werden, um den Aufwand für diesen Operator präzise zu ermitteln. Diese Kardinalitätsschätzung ist die wichtigste einzelne Eingabe für das Aufwandsmodell des Optimierungsprogramms. Ihre Genauigkeit hängt größtenteils von den Statistiken ab, die das Dienstprogramm RUNSTATS aus der Datenbank erfasst.

Differenziertere Statistiken sind erforderlich erforderlich, um komplexere Beziehungen darzustellen, wie zum Beispiel Vergleiche mit Ausdrücken (z. B. `price > MSRP + Dealer_markup`), Beziehungen über mehrere Tabellen hinweg (z. B. `product.name = 'Alloy wheels' and product.key = sales.product_key`) sowie jeden anderen Typ von Element, der komplexer ist als Vergleichselemente mit unabhängigen Attributen und einfache Vergleichsoperationen. Statistiksichten können diese Typen komplexer Beziehungen darstellen, weil die Statistikdaten für die Ergebnismenge, die von der Sicht zurückgegeben werden, und nicht für die Basistabelle, auf die von der Sicht verwiesen wird, erfasst werden.

Wenn eine Abfrage kompiliert wird, gleicht das Optimierungsprogramm die Abfrage mit den verfügbaren Statistiksichten ab. Wenn das Optimierungsprogramm die Kardinalitätsschätzungen für Zwischenergebnismengen berechnet, verwendet es die Statistiken aus der Sicht, um eine bessere Schätzung zu berechnen.

Abfragen brauchen auf eine Statistiksicht nicht direkt zu verweisen, damit das Optimierungsprogramm die Statistiksicht verwendet. Das Optimierungsprogramm verwendet denselben Mechanismus, der auch für MQTs (Materialized Query Tables) verwendet wird, um Abfragen mit Statistiksichten abzugleichen. In dieser Hin-

sicht sind Statistiksichten MQTs sehr ähnlich, abgesehen davon, dass sie nicht permanent gespeichert werden, keinen Festplattenspeicher belegen und nicht gepflegt werden müssen.

Eine Statistiksicht wird erstellt, indem zuerst eine Sicht erstellt und anschließend diese Sicht für die Optimierung mit der Anweisung ALTER VIEW aktiviert wird. Anschließend wird der Befehl **RUNSTATS** für die Statistiksicht ausgeführt, um die Systemkatalogtabellen mit Statistiken zu dieser Sicht zu füllen. Wenn Sie zum Beispiel eine Statistiksicht erstellen wollen, die den Join zwischen der Dimensionstabelle TIME und der Fakttabelle in einem Sternschema darstellt, geben Sie folgende Anweisungen und Befehle ein:

```
CREATE VIEW SV_TIME_FACT AS (
  SELECT T.* FROM TIME T, SALES S
    WHERE T.TIME_KEY = S.TIME_KEY)

ALTER VIEW SV_TIME_FACT ENABLE QUERY OPTIMIZATION

RUNSTATS ON TABLE DB2DBA.SV_TIME_FACT WITH DISTRIBUTION
```

Diese Statistiksicht kann zur Verbesserung der Kardinalitätsschätzung und daher zur Verbesserung des Zugriffsplans und der Abfrageleistung für Abfragen wie die folgende verwendet werden:

```
SELECT SUM(S.PRICE)
  FROM SALES S, TIME T, PRODUCT P
  WHERE
    T.TIME_KEY = S.TIME_KEY AND
    T.YEAR_MON = 200712 AND
    P.PROD_KEY = S.PROD_KEY AND
    P.PROD_DESC = 'Power drill'
```

Ohne Statistiksicht nimmt das Optimierungsprogramm an, dass alle Werte der Spalte TIME_KEY der Fakttabelle, die einem bestimmten Wert der Spalte YEAR_MON der Dimension TIME entsprechen, gleichmäßig in der Fakttabelle verteilt sind. Der Verkauf könnte jedoch im Dezember besonders stark gewesen sein, sodass wesentlich mehr Verkaufstransaktionen als in den anderen Monaten ausgeführt wurden.

Eine automatische Statistikerfassung ist für Statistiksichten gegenwärtig nicht verfügbar. Erfassen Sie Statistiken für solche Sichten, wenn die Basistabellen, auf die Statistiksichten verweisen, beträchtlich aktualisiert wurden.

Eine Statistiksicht kann nicht direkt oder indirekt auf eine Katalogtabelle verweisen.

Verwenden von Statistiksichten

Die Optimierung für eine Sicht muss aktiviert werden, bevor die Statistikdaten der Sicht zur Optimierung einer Abfrage herangezogen werden können. Eine Sicht, deren Optimierung aktiviert ist, wird als *Statistiksicht* bezeichnet.

Informationen zu diesem Vorgang

Eine Sicht, deren Optimierung nicht aktiviert ist, wird nicht als Statistiksicht, sondern als *reguläre Sicht* bezeichnet. Nach der Erstellung ist die Optimierung für eine Sicht zunächst inaktiviert. Mit der Anweisung ALTER VIEW können Sie die Optimierung für eine Sicht aktivieren. Informationen zu den Zugriffsrechten und Berechtigungen, die zur Ausführung dieser Task erforderlich sind, finden Sie in der Beschreibung der Anweisung ALTER VIEW. Informationen zu den Zugriffsrechten

und Berechtigungen, die zur Ausführung des Dienstprogramms RUNSTATS für eine Sicht erforderlich sind, finden Sie in der Beschreibung des Befehls **RUNSTATS**.

Die Optimierung für eine Sicht kann nicht 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 Statistiksicht kann auf eine Statistiksicht verweisen.)
- Die Sicht verweist direkt oder indirekt auf eine Katalogtabelle.
- Die Sicht ist unbrauchbar.
- Die Sicht ist eine typisierte Sicht.
- In derselben Anweisung ALTER VIEW ist eine weitere Sichtänderungsanforderung enthalten.

Wenn die Definition einer Sicht, die zur Aktivierung der Optimierung geändert wird, eines der folgenden Elemente enthält, wird eine Warnung zurückgegeben und die Statistikdaten der Sicht werden vom Optimierungsprogramm nicht genutzt:

- Spaltenberechnungsoperationen oder DISTINCT-Operationen
- UNION-, EXCEPT- oder INTERSECT-Operationen
- OLAP-Spezifikation

Vorgehensweise

1. Aktivieren Sie die Optimierung für die Sicht.

   Die Optimierung für eine Sicht kann mithilfe der Klausel ENABLE OPTIMIZATION in der Anweisung ALTER VIEW aktiviert werden. Für eine Sicht, für die die Optimierung aktiviert wurde, kann die Optimierung nachfolgend mit der Klausel DISABLE OPTIMIZATION inaktiviert werden. Geben Sie zum Beispiel die folgende Anweisung ein, um die Optimierung für eine Sicht mit dem Namen MYVIEW zu aktivieren:

   ```
   alter view myview enable query optimization
   ```

2. Rufen Sie den Befehl **RUNSTATS** auf. Geben Sie zum Beispiel den folgenden Befehl ein, um Statistikdaten für MYVIEW zu erfassen:

   ```
   runstats on table db2dba.myview
   ```

   Geben Sie zum Beispiel den folgenden Befehl ein, wenn bei der Erfassung der Sichtstatistiken eine Stichprobenentnahme auf Zeilenebene von 10 % der Zeilen, einschließlich Verteilungsstatistiken, erfolgen soll:

   ```
   runstats on table db2dba.myview with distribution tablesample bernoulli (10)
   ```

   Geben Sie zum Beispiel den folgenden Befehl ein, wenn bei der Erfassung der Sichtstatistiken eine Stichprobenentnahme auf Seitenebene von 10 % der Seiten, einschließlich Verteilungsstatistiken, erfolgen soll:

   ```
   runstats on table db2dba.myview with distribution tablesample system (10)
   ```

3. Optional: Wenn Abfragen, die von der Sichtdefinition beeinflusst werden, Teil statischer SQL-Pakete sind, binden Sie diese Pakete erneut (REBIND), um die Vorteile von Änderungen an Zugriffsplänen zu nutzen, die sich aus den neuen Statistiken ergeben.

Sichtstatistiken mit Relevanz für die Optimierung

Nur Statistikdaten, welche die Datenverteilung der Abfrage kennzeichnen, die eine Statistiksicht definiert, zum Beispiel CARD und COLCARD, werden bei der Optimierung in Betracht gezogen.

Die folgenden Statistikdaten, die sich auf Sichtdatensätze beziehen, können zur Verwendung durch das Optimierungsprogramm erfasst werden.

- Tabellenstatistiken (SYSCAT.TABLES, SYSSTAT.TABLES)
  - CARD - Anzahl von Zeilen im Ergebnis für eine Sicht
- Spaltenstatistiken (SYSCAT.COLUMNS, SYSSTAT.COLUMNS)
  - COLCARD - Anzahl unterschiedlicher Werte einer Spalte im Ergebnis für eine Sicht
  - AVGCOLLEN - Durchschnittliche Länge einer Spalte 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 von Nullwerten in einer Spalte im Ergebnis für eine Sicht
  - SUB_COUNT - Durchschnittliche Anzahl von Unterelementen in einer Spalte im Ergebnis für eine Sicht
  - SUB_DELIM_LENGTH - Durchschnittliche Länge der einzelnen Begrenzer, die Unterelemente voneinander trennen
- Spaltenverteilungsstatistiken (SYSCAT.COLDIST, SYSSTAT.COLDIST)
  - DISTCOUNT - Anzahl unterschiedlicher Quantilwerte, die kleiner oder gleich dem COLVALUE-Statistikwert sind
  - SEQNO - Stelle in der Häufigkeitsrangfolge einer Folgenummer, 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 ein 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.

Szenario: Verbessern der Kardinalitätsschätzung durch Statistiksichten

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 Statistiksichten (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 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 von Statistiksichten zur Lösung dieses Typs von Problem schrittweise veranschaulicht.

Betrachten Sie eine Datenbank aus einem typischen Data-Warehouse, in dem STORE, CUSTOMER, PRODUCT, PROMOTION und PERIOD die Dimensionstabellen sind und DAILY_SALES die Fakttabelle ist. In den folgenden Tabellen sind die Definitionen für diese Tabellen aufgeführt.

Tabelle 56. STORE (63 Zeilen)

Spalte	storekey	store_number	city	state	district	...
Attribut	integer not null Primär-schlüssel	char(2)	char(20)	char(5)	char(14)	...

Tabelle 57. CUSTOMER (1.000.000 Zeilen)

Spalte	custkey	name	address	age	gender	...
Attribut	integer Primärschlüssel	char(30)	char(40)	smallint	char(1)	...

Tabelle 58. PRODUCT (19.450 Zeilen)

Spalte	prodkey	category	item_desc	price	cost	...
Attribut	integer not null Primärschlüssel	integer	char(30)	decimal(11)	decimal(11)	...

Tabelle 59. PROMOTION (35 Zeilen)

Spalte	promokey	promotype	promodesc	promovalue	...
Attribut	integer not null Primärschlüssel	integer	char(30)	decimal(5)	...

Tabelle 60. PERIOD (2922 Zeilen)

Spalte	perkey	calendar_date	month	period	year	...
Attribut	integer not null Primärschlüssel	date	char(3)	smallint	smallint	...

Tabelle 61. 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 feststellen, ob Kunden ein Produkt noch einmal kaufen, wenn ihnen ein Rabatt bei einem weiteren Besuch angeboten wird. Nehmen Sie außerdem an, dass diese Studie nur für das Geschäft (STORE) '01' durchgeführt wird, das 18 Niederlassungen im Land hat. In Tabelle 62 sind Informationen zu den verschiedenen Kategorien von Maßnahmen zur Verkaufsförderung (PROMOTION) aufgeführt, die verfügbar sind.

Tabelle 62. PROMOTION (35 Zeilen)

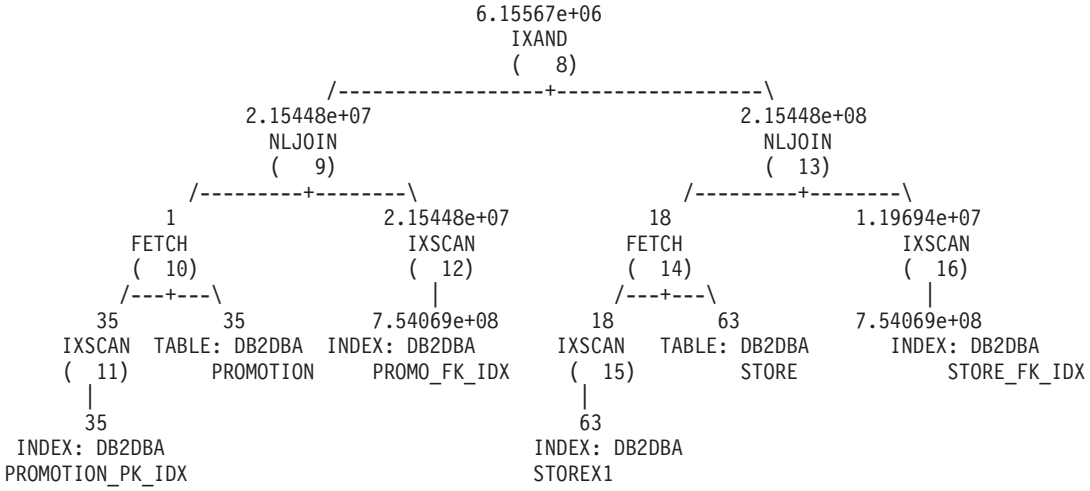
promotype	promodesc	COUNT (promotype)	Prozentsatz vom Gesamtwert
1	Return customers	1	2,86%
2	Coupon	15	42,86%
3	Advertisement	5	14,29%
4	Manager's special	3	8,57%
5	Overstocked items	4	11,43%
6	End aisle display	7	20,00%

Die Tabelle zeigt, dass Rabatte für wiederkehrende Kunden (Return customers) nur 2,86 % der 35 angebotenen Arten von Verkaufsförderungsmaßnahmen ausmachen.

Die folgende Abfrage gibt eine ermittelte Anzahl von 12.889.514 zurück:

```
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 die erste Zeile die Kardinalitätsschätzung, die zweite Zeile den Operatortyp und die dritte Zeile (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 ÷7,54069e+08 ≈ 0,0286). Beachten Sie, dass dies vor und nach dem Join der Tabelle PROMOTION mit der Tabelle DAILY_SALES der gleiche Wert ist. In Tabelle 63 sind die Kardinalitätsschätzwerte und ihre Prozentsätze (Filtereffekt) vor und nach dem Join zusammengefasst.

Tabelle 63. Geschätzte Kardinalitäten vor und nach dem Join mit der Tabelle DAILY_SALES

	Vor dem Join		Nach dem Join	
Vergleichselement	**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 (12.889.514 ÷ 6.155.670 ≈ 2,09) unter der tatsächlichen Anzahl.

Was ist die Ursache dafür, dass das Optimierungsprogramm die Anzahl der Datensätze, die die beiden Vergleichselemente erfüllen, 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 63 auf Seite 419 irreführend sein. Der tatsächliche Prozentsatz in der Tabelle DAILY_SALES könnte sehr wohl ein anderer Wert sein als der projektierte.

Statistiksichten können dem Optimierungsprogramm helfen, solche Schätzwerte zu korrigieren. Zunächst müssen zwei Statistiksichten erstellt werden, welche die beiden Semi-Join-Verknüpfungen in der vorherigen Abfrage darstellen. Die erste Statistiksicht stellt die Verteilungsdaten von Geschäften für alle täglichen Umsätze bereit. Die zweite Statistiksicht stellt die Verteilung von Verkaufsförderungstypen für alle täglichen Umsätze dar. Beachten Sie, dass diese Statistiksichten 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 Tabellen abgefragt, um die erforderlichen Statistikdaten zur Aktualisierung der beiden Statistiksichten 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)
   ```

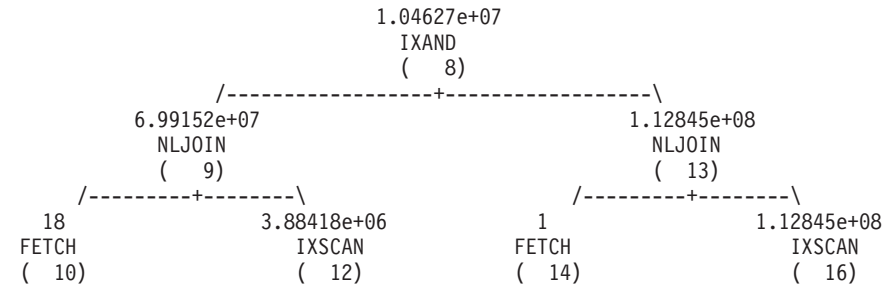
3. Machen Sie die Sichten zu Statistiksichten, indem Sie sie für die Abfrageoptimierung aktivieren:

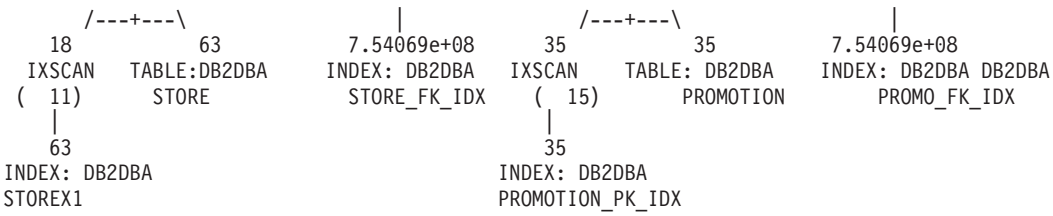
   ```
   alter view sv_store_dailysales enable query optimization
   alter view sv_promotion_dailysales enable query optimization
   ```

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 Fakttabelle und den Dimensionstabellen anzupassen. Dies führt zu einer Umkehrung der Reihenfolge der Semi-Join-Verknüpfungen, die ohne diese Statistiken ausgewählt wurden. Der neue Plan sieht folgendermaßen aus:





In Tabelle 64 sind die Kardinalitätsschätzwerte und ihre Prozentsätze (der Filtereffekt) vor und nach dem Join für jeden Semi-Join zusammengefasst.

Tabelle 64. Geschätzte Kardinalitäten vor und nach dem Join mit der Tabelle DAILY_SALES

	Vor dem Join		**Nach dem Join (*ohne Statistiksichten*)**		**Nach dem Join (*mit Statistiksichten*)**	
Vergleichs-element	**Anzahl**	**Prozent-satz quali-fizierter 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%

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 Statistiksichten dem Optimierungsprogramm im Wesentlichen mitteilen, dass das Vergleichselement `store_number = '01'` mehr Zeilen herausfiltert als das Vergleichselement `promotype = 1`. Dieses Mal schätzt das Optimierungsprogramm, dass ca. 10.462.700 Produkte verkauft wurden. Dieser Schätzwert liegt um den Faktor 1,23 (12.889.514 ÷ 10.462.700 ≈ 1,23) unter der tatsächlichen Anzahl, was eine erhebliche Verbesserung gegenüber dem Schätzwert ohne Statistiksichten (in Tabelle 63 auf Seite 419) darstellt.

Katalogstatistiken

Wenn der Abfragecompiler Abfragepläne optimiert, werden die Entscheidungen in hohem Maße von statistischen Informationen über die Größe der Tabellen, Indizes und Statistiksichten der Datenbank beeinflusst. Diese Informationen werden in Systemkatalogtabellen gespeichert.

Das Optimierungsprogramm verwendet außerdem Informationen über die Verteilung von Daten in bestimmten Spalten von Tabellen, Indizes und Statistiksichten, 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.

Statistische Informationen über das Clusterverhältnis (CLUSTERRATIO) 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 können ebenfalls erfasst werden. Diese Informationen können Ihnen bei der Entscheidung helfen, wann eine Reorganisation von Tabellen oder Indizes durchzuführen ist.

Tabellenstatistikdaten in einer Umgebung mit partitionierten Datenbanken werden nur für den Teil der Tabelle, der sich in der Datenbankpartition befindet, in der das Dienstprogramm ausgeführt wird, oder für die erste Datenbankpartition in der Da-

tenbankpartitionsgruppe, die die Tabelle enthält, erfasst. Informationen zu Statistiksichten werden für alle Datenbankpartitionen erfasst.

Vom Dienstprogramm RUNSTATS aktualisierte Statistiken

Katalogstatistiken werden vom Dienstprogramm RUNSTATS aktualisiert, das durch den Befehl **RUNSTATS**, durch Aufrufen der Prozedur ADMIN_CMD oder durch Aufrufen der API db2Runstats gestartet werden kann. Aktualisierungen können entweder manuell oder automatisch initialisiert werden.

Statistiken über deklarierte temporäre Tabellen werden nicht im Systemkatalog, sondern in Speicherstrukturen gespeichert, die die Kataloginformationen für deklarierte temporäre Tabellen darstellen. Es ist möglich (und es kann in einigen Fällen nützlich sein), das Dienstprogramm RUNTSTATS für eine deklarierte temporäre Tabelle auszuführen.

Das Dienstprogramm RUNSTATS erfasst die folgenden Informationen zu Tabellen und Indizes:

- Die Anzahl der Seiten, die Zeilen enthalten
- Die Anzahl der Seiten, die belegt sind
- Die Anzahl von Zeilen in der Tabelle (*Kardinalität*)
- Die Anzahl der Zeilen, die überlaufen
- Für Tabellen mit mehrdimensionalem Clustering (MDC-Tabellen): die Anzahl von Blöcken, die Daten enthalten
- Für partitionierte Tabellen: der Grad der Datenclusterbildung in einer einzigen Datenpartition
- Datenverteilungsstatistiken, die vom Optimierungsprogramm verwendet werden, um effiziente Zugriffspläne für Tabellen und Statistiksichten abzuschätzen, deren Daten nicht gleichmäßig verteilt sind und deren Spalten eine beträchtliche Anzahl mehrfach auftretender Werte aufweisen
- Detaillierte Indexstatistiken, anhand deren das Optimierungsprogramm die Effizienz eines Zugriffs auf Tabellendaten über einen Index bestimmen kann
- Statistiken zu Unterelementen für LIKE-Vergleichselemente, insbesondere für solche, die in Zeichenfolgen nach Mustern suchen (z. B. `LIKE %disk%`), werden ebenfalls vom Optimierungsprogramm verwendet

Das Dienstprogramm RUNSTATS erfasst die folgenden statistischen Daten für die einzelnen Datenpartitionen einer Tabelle. Diese statistischen Daten dienen nur dazu zu ermitteln, ob eine Partition reorganisiert werden sollte:

- Die Anzahl der Seiten, die Zeilen enthalten
- Die Anzahl der Seiten, die belegt sind
- Die Anzahl von Zeilen in der Tabelle (Kardinalität)
- Die Anzahl der Zeilen, die überlaufen
- Für MDC-Tabellen: die Anzahl von Blöcken, die Daten enthalten

Verteilungsstatistiken werden in folgenden Fällen nicht erfasst:

- Wenn die Datenbankkonfigurationsparameter **num_freqvalues** und **num_quantiles** auf den Wert 0 gesetzt sind.
- Wenn die Verteilung von Daten bekannt ist, zum Beispiel wenn jeder Datenwert nur einmal vorkommt.
- Wenn die Spalte Daten der Typen LONG, LOB oder eines strukturierten Datentyps enthält.

- Wenn es sich um Zeilentypen in untergeordneten Tabellen handelt (die Statistiken für NPAGES, FPAGES und OVERFLOW auf Tabellenebene werden 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.

Das Dienstprogramm RUNSTATS erfasst die folgenden Informationen zu jeder Spalte in einer Tabelle oder Statistiksicht sowie zur ersten Spalte in einem Indexschlüssel:

- Die Kardinalität der Spalte
- Die durchschnittliche Länge der Spalte (der durchschnittliche Speicherplatz in Byte, der benötigt wird, wenn die Spalte im Datenbankspeicher oder in einer temporären Tabelle gespeichert wird)
- Der zweithöchste Wert in der Spalte
- Der zweitniedrigste Wert in der Spalte
- Die Anzahl von Nullwerten in der Spalte

Für Spalten mit LOB-Datentypen (große Objekte) oder LONG-Datentypen erfasst das Dienstprogramm RUNSTATS nur die durchschnittliche Länge der Spalte und die Anzahl der Nullwerte in der Spalte. Die durchschnittliche Länge der Spalte stellt die Länge des Datendeskriptors dar, außer wenn LOB-Daten integriert ('inline') auf der Datenseite gespeichert werden. Der durchschnittliche Speicherplatzbedarf zum Speichern der Spalte auf der Platte kann vom Wert dieser Statistik abweichen.

Das Dienstprogramm erfasst die folgenden Informationen zu jeder XML-Spalte:

- Die Anzahl von XML-Dokumenten mit dem Wert NULL
- Die Anzahl von XML-Dokumenten mit dem Wert ungleich NULL
- Die Anzahl distinkter Pfade
- Die Summe der Knotenzahlen für die einzelnen distinkten Pfade
- Die Summe der Dokumentzahlen für die einzelnen distinkten Pfade
- Die *k* Paare aus Pfad und Knotenzahl für die größte Knotenzahl
- Die *k* Paare aus Pfad und Dokumentzahl für die größte Dokumentzahl
- Die *k* Triplets aus Pfad, Wert und Knotenzahl für die größte Knotenzahl
- Die *k* Triplets aus Pfad, Wert und Dokumentzahl für die größte Dokumentzahl
- Für jeden distinkten Pfad, der zu einem Text- oder Attributwert führt:
  - Die Anzahl der unterschiedlichen Werte, die dieser Pfad annehmen kann
  - Den höchsten Wert
  - Den niedrigsten Wert
  - Die Anzahl der Text- oder Attributknoten
  - Die Anzahl der Dokumente mit den Text- oder Attributknoten

In jeder Zeile einer XML-Spalte wird ein XML-Dokument gespeichert. Die Knotenzahl eines angegebenen Pfads oder eines Pfad-Wert-Paars bezieht sich auf die Anzahl der Knoten, die durch den Pfad oder das Pfad-Wert-Paar erreichbar sind. Die Dokumentzahl eines angegebenen Pfads oder eines Pfad-Wert-Paars bezieht sich auf die Anzahl der Dokumente, die den angegebenen Pfad oder das angegebene Pfad-Wert-Paar enthalten.

In DB2 Version 9.7 Fixpack 1 und späteren Releases gelten die folgenden Punkte für die Erfassung von Verteilungsstatistiken für eine XML-Spalte:

- Verteilungsstatistiken werden für jeden Index zu XML-Daten erfasst, der für eine XML-Spalte angegeben ist.
- Das Dienstprogramm RUNSTATS muss Verteilungsstatistiken und Tabellenstatistiken sammeln, um Verteilungsstatistiken für einen Index zu XML-Daten zu erfassen. Die Tabellenstatistiken müssen gesammelt werden, damit Verteilungsstatistiken erfasst werden, da XML-Verteilungsstatistiken zusammen mit Tabellenstatistiken gespeichert werden.

  Wenn nur Indexstatistiken erfasst werden oder Indexstatistiken bei der Indexerstellung erfasst werden, werden dabei keine Verteilungsstatistiken für einen Index zu XML-Daten erfasst.

  Standardmäßig erfasst das Dienstprogramm RUNSTATS maximal 250 Quantile für die Verteilungsstatistik für jeden Index zu XML-Daten. Die maximale Anzahl von Quantilen für eine Spalte kann bei der Ausführung des Dienstprogramms RUNSTATS angegeben werden.
- Verteilungsstatistikdaten werden für Indizes zu XML-Daten vom Typ VARCHAR, DOUBLE, TIMESTAMP und DATE erfasst. XML-Verteilungsstatistikdaten werden nicht für Indizes zu XML-Daten vom Typ VARCHAR HASHED erfasst.
- XML-Verteilungsstatistikdaten werden erfasst, wenn automatische RUNSTATS-Operationen für Tabellen ausgeführt werden.
- XML-Verteilungsstatistiken werden nicht erstellt, wenn Daten mit der Option STATISTICS geladen werden.
- XML-Verteilungsstatistikdaten werden nicht für partitionierte Indizes zu XML-Daten erfasst, die für eine partitionierte Tabelle definiert sind.

Das Dienstprogramm RUNSTATS erfasst die folgenden Informationen zu Spaltengruppen:

- Ein auf einer Zeitmarke basierender Name für die Spaltengruppe
- Die Kardinalität der Spaltengruppe

Das Dienstprogramm RUNSTATS erfasst die folgenden Informationen zu 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 der Blattseiten, die sich auf dem Datenträger in der Reihenfolge des Indexschlüssels befinden, 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 durchschnittliche Schlüsselgröße für Blattseiten ohne INCLUDE-Spalten
- Die durchschnittliche Schlüsselgröße für Blattseiten mit INCLUDE-Spalten
- 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 Indexstatistiken anfordern, werden zusätzliche Informationen zum Grad der Clusterbildung der Tabellendaten im Verhältnis zum Index sowie zu den Seitenabrufschätzwerten für verschiedene Puffergrößen erfasst.

Bei einem partitionierten Index beziehen sich diese statistischen Daten auf eine einzige Indexpartition, die Angaben zu unterschiedlichen Werten in der ersten Spalte des Index, in den ersten zwei, drei, vier Spalten des Index und in allen Spalten ausgenommen. Die statistischen Daten für die einzelnen Indexpartitionen werden auch erfasst, um festzustellen, welche Indexpartition reorganisiert werden muss.

Eine Statistikerfassung macht im Cache gespeicherte dynamische Anweisungen, die auf Tabellen verweisen, für die Statistiken erfasst wurden, ungültig. Dies geschieht, damit im Cache gespeicherte dynamische Anweisungen mit den aktuellsten Statistiken reoptimiert werden können.

Katalogstatistiktabellen

Statistische Informationen über die Größe von Tabellen, Indizes und Statistiksichten werden in Systemkatalogtabellen gespeichert.

Die folgenden Tabellen enthalten eine Kurzbeschreibung dieser statistischen Informationen und geben an, wo die Informationen gespeichert werden.

- Die Spalte „Tabelle" gibt an, ob die jeweilige Statistik erfasst wird, wenn die Option FOR INDEXES oder AND INDEXES im Befehl RUNSTATS nicht angegeben wurde.
- Die Spalte „Indizes" gibt an, ob die jeweilige Statistik erfasst wird, wenn die Option FOR INDEXES oder AND INDEXES angegeben wurde.

Einige Statistiken können nur von der Tabelle, einige nur von den Indizes und einige von beiden bereitgestellt werden.

- Tabelle 1. Tabellenstatistiken (SYSCAT.TABLES und SYSSTAT.TABLES)
- Tabelle 2. Spaltenstatistiken (SYSCAT.COLUMNS und SYSSTAT.COLUMNS)
- Tabelle 3. Statistiken für Spaltengruppen (SYSCAT.COLGROUPS und SYSSTAT-.COLGROUPS)
- Tabelle 4. Verteilungsstatistiken für Spaltengruppen (SYSCAT.COLGROUPDIST und SYSSTAT.COLGROUPDIST)
- Tabelle 5. Verteilungsstatistiken für Spaltengruppen (SYSCAT.COLGROUPDIST-COUNTS und SYSSTAT.COLGROUPDISTCOUNTS)
- Tabelle 6. Indexstatistiken (SYSCAT.INDEXES und SYSSTAT.INDEXES)
- Tabelle 7. Spaltenverteilungsstatistiken (SYSCAT.COLDIST und SYSSTAT.COL-DIST)

Die Verteilungsstatistiken für Spaltengruppen in Tabelle 4. Verteilungsstatistiken für Spaltengruppen (SYSCAT.COLGROUPDIST und SYSSTAT.COLGROUPDIST) und in Tabelle 5. Verteilungsstatistiken für Spaltengruppen (SYSCAT.COLGROUP-DISTCOUNTS und SYSSTAT.COLGROUPDISTCOUNTS) werden nicht vom Dienstprogramm RUNSTATS erfasst. Sie können nicht manuell aktualisiert werden.

Tabelle 65. 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 einer 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 definiert wurden und Sie die Indexstatistik anfordern, wird keine CARD-Statistik aktualisiert. Die vorige CARD-Statistik wird beibehalten.

Tabelle 66. 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 einer Spalte	Ja	Ja (Anm. 1)
HIGH2KEY	Zweithöchster Wert einer Spalte	Ja	Ja (Anm. 1)
LOW2KEY	Zweitniedrigster Wert einer Spalte	Ja	Ja (Anm. 1)
NUMNULLS	Die Anzahl von Nullwerten in einer Spalte	Ja	Ja (Anm. 1)
SUB_COUNT	Die durchschnittliche Anzahl von Unterelementen	Ja	Nein (Anm. 2)
SUB_DELIM_LENGTH	Durchschnittslänge jedes Begrenzers, der Unterelemente trennt	Ja	Nein (Anm. 2)

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 Statistikwerte SUB_COUNT und SUB_DELIM_LENGTH werden nur für Spalten der Typen CHAR und VARCHAR mit einem Codepageattribut eines Einzelbytezeichensatzes (SBCS), mit FOR BIT DATA oder mit UTF-8 erfasst.

Tabelle 67. Statistiken für Spaltengruppen (SYSCAT.COLGROUPS und SYSSTAT.COLGROUPS)

Statistik	Beschreibung	RUNSTATS-Option	
		Tabelle	**Indizes**
COLGROUPCARD	Kardinalität der Spaltengruppe	Ja	Nein

Tabelle 68. Verteilungsstatistiken für Spaltengruppen (SYSCAT.COLGROUPDIST und SYSSTAT.COLGROUPDIST)

Statistik	Beschreibung	RUNSTATS-Option	
		Tabelle	**Indizes**
TYPE	F = Häufigkeitswert Q = Quantilwert	Ja	Nein
ORDINAL	Ordinalzahl der Spalte in der Gruppe	Ja	Nein
SEQNO	Folgenummer *n*, die den *n*-ten TYPE-Wert darstellt	Ja	Nein
COLVALUE	Der Datenwert als Zeichenliteral oder Nullwert	Ja	Nein

Tabelle 69. Verteilungsstatistiken für Spaltengruppen (SYSCAT.COLGROUPDISTCOUNTS und SYSSTAT.COLGROUPDISTCOUNTS)

Statistik	Beschreibung	RUNSTATS-Option	
		Tabelle	**Indizes**
TYPE	F = Häufigkeitswert Q = Quantilwert	Ja	Nein
SEQNO	Folgenummer *n*, die den *n*-ten TYPE-Wert darstellt	Ja	Nein
VALCOUNT	Bei TYPE = F ist VALCOUNT die Anzahl Vorkommen von COLVALUE für die Spaltengruppe mit dieser Folgenummer (SEQNO). Bei TYPE = Q ist VALCOUNT die Anzahl Zeilen, deren Wert kleiner oder gleich COLVALUE für die Spaltengruppe mit dieser Folgenummer ist.	Ja	Nein

Tabelle 69. Verteilungsstatistiken für Spaltengruppen (SYSCAT.COLGROUPDISTCOUNTS und SYSSTAT.COLGROUPDISTCOUNTS) (Forts.)

Statistik	Beschreibung	RUNSTATS-Option	
		Tabelle	Indizes
DISTCOUNT	Bei TYPE = Q enthält diese Spalte die Anzahl unterschiedlicher Werte, die kleiner oder gleich COLVALUE für die Spaltengruppe mit dieser Folgenummer (SEQNO) sind. Null, falls nicht verfügbar.	Ja	Nein

Tabelle 70. 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
FIRST4KEYCARD	Anzahl der unterschiedlichen Werte in den ersten vier Spalten des Index	Nein	Ja

Tabelle 70. Indexstatistiken (SYSCAT.INDEXES und SYSSTAT.INDEXES) (Forts.)

Statistik	Beschreibung	RUNSTATS-Option	
		Tabelle	**Indizes**
FULLKEYCARD	Anzahl der unterschiedlichen Werte in allen Spalten des Index, mit Ausnahme der Schlüsselwerte in einem Index, 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, die auf der Platte in der durch den Indexschlüssel definierten Reihenfolge mit wenigen oder keinen großen dazwischen liegenden Lücken gespeichert sind	Nein	Ja

Tabelle 70. Indexstatistiken (SYSCAT.INDEXES und SYSSTAT.INDEXES) (Forts.)

Statistik	Beschreibung	RUNSTATS-Option	
		Tabelle	**Indizes**
AVERAGE_SEQUENCE_PAGES	Anzahl von Indexseiten, auf die in der 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	Anzahl von Tabellenseiten, auf die in der Reihenfolge zugegriffen werden kann; dies ist die Anzahl von Tabellenseiten, die von Vorablesefunktionen als in der richtigen Reihenfolge vorliegend erkannt werden können, wenn sie Tabellenzeilen über den Index abrufen	Nein	Ja (Anm. 4)
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)

Tabelle 70. Indexstatistiken (SYSCAT.INDEXES und SYSSTAT.INDEXES) (Forts.)

Statistik	Beschreibung	RUNSTATS-Option	
		Tabelle	Indizes
NUMRIDS	Anzahl von Satz-IDs (RIDs) im Index, einschließlich gelöschter Satz-IDs	Nein	Ja
NUMRIDS_DELETED	Gesamtanzahl von Satz-IDs (RIDs) im Index, die als gelöscht markiert sind, außer Satz-IDs auf Blattseiten, auf denen alle Satz-IDs als gelöscht markiert sind	Nein	Ja
NUM_EMPTY_LEAFS	Gesamtanzahl von Blattseiten, auf 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 ausreichende Größe (mehr als ca. 25 Seiten) aufweist. In diesem Fall ist CLUSTERRATIO (Clusterverhältnis) -1 (d. h. wird nicht erfasst). Wenn die Tabelle relativ klein ist, werden nur Daten für die Spalte CLUSTERRATIO vom Dienstprogramm RUNSTATS erfasst, während für die Spalten CLUSTERFACTOR und PAGE_FETCH_PAIRS keine Daten erfasst werden. Wenn die Klausel DETAILED nicht angegeben wird, werden nur die Daten 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 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 Statistik kann nicht berechnet werden, wenn sich die Tabelle in einem DMS-Tabellenbereich befindet.
5. Statistiken über Vorablesezugriffe werden beim Laden oder Erstellen von Indizes nicht erfasst, und zwar auch dann nicht, wenn die Funktion zum Erfassen von Statistiken beim Aufrufen des Befehls angegeben wird. Statistiken über Vorablesezugriffe werden ebenfalls nicht erfasst, wenn der Datenbankkonfigurationsparameter **seqdetect** auf den Wert NO gesetzt ist.
6. Wenn unter 'RUNSTATS-Option' für 'Tabelle' der Wert „Nein" angegeben ist, bedeutet dies, dass die entsprechenden Statistikdaten bei der Erfassung von Tabellenstatistiken nicht erfasst werden. Wenn unter 'RUNSTATS-Option' für 'Indizes' der Wert „Ja" angegeben ist, heißt dies, dass die entsprechenden Statistikdaten erfasst werden, wenn der Befehl **RUNSTATS** mit den INDEXES-Optionen ausgeführt wird.

Tabelle 71. Spaltenverteilungsstatistiken (SYSCAT.COLDIST und SYSSTAT.COLDIST)

Statistik	Beschreibung	RUNSTATS-Option	
		Tabelle	**Indizes**
DISTCOUNT	Bei TYPE = Q enthält DISTCOUNT 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 Quantile 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 Quantilen erfasst werden	Verteilung	Nein
VALCOUNT	Häufigkeit, mit der der Datenwert in einer Spalte auftritt; bei Quantilen die Anzahl der Werte, die kleiner oder gleich dem Datenwert (COLVALUE) sind	Verteilung	Nein

Anmerkung:

1. Verteilungsstatistikdaten über Spalten werden erfasst, indem die Klausel WITH DISTRIBUTION im Befehl **RUNSTATS** angegeben wird. Verteilungsstatistikdaten können nicht erfasst 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.

Automatische Statistikerfassung

Das DB2-Optimierungsprogramm verwendet Katalogstatistiken, um den effizientesten Zugriffsplan für eine Abfrage zu ermitteln. Tabellen- oder Indexstatistiken, die nicht auf dem neuesten Stand oder unvollständig sind, könnten dazu führen, dass das Optimierungsprogramm einen suboptimalen Plan auswählt, sodass sich die Abfrageausführung verlangsamt. Allerdings ist die Entscheidung, welche Statistiken für eine gegebene Auslastung zu erfassen sind, recht komplex und die Pflege aktueller Statistiken zeitaufwendig.

Mithilfe der automatischen Statistikerfassung, die Teil der Funktion zur automatischen Tabellenverwaltung von DB2 ist, können Sie den Datenbankmanager bestim-

men lassen, ob Datenbankstatistiken aktualisiert werden müssen. Die automatische Statistikerfassung kann *synchron* bei der Anwendungskompilierung durch die Verwendung der Echtzeitstatistikfunktion (RTS, Real-Time Statistics) stattfinden. Alternativ kann das Dienstprogramm RUNSTATS für eine *asynchrone* Erfassung zur Ausführung im Hintergrund eingerichtet werden. Obwohl die Statistikerfassung im Hintergrund aktiviert werden kann, während die Echtzeitstatistikerfassung inaktiviert ist, muss die Statistikerfassung im Hintergrund aktiviert sein, damit eine Echtzeitstatistikerfassung erfolgen kann. Die automatische Statistikerfassung im Hintergrund (**auto_runstats**) und die automatische Echtzeitstatistikerfassung (**auto_stmt_stats**) werden standardmäßig aktiviert, wenn eine neue Datenbank erstellt wird.

Asynchron und in Echtzeit ausgeführte Statistikerfassung

Wenn die Echtzeitstatistikerfassung aktiviert ist, können Statistikdaten mithilfe bestimmter Metadaten konstruiert werden. *Konstruieren* bedeutet in diesem Fall, dass Statistiken abgeleitet oder generiert und nicht im Rahmen der normalen RUNSTATS-Aktivitäten erfasst werden. Zum Beispiel lässt sich die Anzahl von Zeilen in einer Tabelle aus der Kenntnis der Anzahl von Seiten in der Tabelle, der Seitengröße und der durchschnittlichen Zeilenbreite abgeleiten. In einigen Fällen werden die Statistikdaten jedoch nicht abgeleitet, sondern durch den Index- und den Datenmanager gepflegt, sodass sie direkt im Katalog gespeichert werden können. Zum Beispiel verwaltet der Indexmanager einen Zähler für die Blattseiten und Stufen in jedem Index.

Das Abfrageoptimierungsprogramm bestimmt auf der Basis des Bedarfs der Abfrage und des Volumens an Tabellenaktualisierungsaktivitäten (Anzahl der UPDATE-, INSERT- oder DELETE-Operationen), wie die Statistiken erfasst werden sollen.

Die Echtzeitstatistikerfassung stellt zeitgerechtere und präzisere Statistikdaten zur Verfügung. Präzise Statistikdaten können bessere Abfrageausführungspläne und eine höhere Leistung zur Folge haben. Wenn die Echtzeitstatistikerfassung nicht aktiviert ist, erfolgt die asynchrone Statistikerfassung in Intervallen von je zwei Stunden. Dies ist möglicherweise nicht häufig genug, um präzise Statistiken für bestimmte Anwendungen bereitzustellen.

Wenn die Echtzeitstatistikerfassung aktiviert ist, erfolgt die asynchrone Statistikerfassungsprüfung dennoch in Intervallen von je zwei Stunden. Die Echtzeitstatistikerfassung leitet in folgenden Fällen auch asynchrone Erfassungsanforderungen ein:

- Wenn das Tabellenaktivitätsvolumen nicht ausreicht, um eine synchrone Erfassung zu erfordern, jedoch ausreicht, um eine asynchrone Erfassung zu erfordern.
- Wenn die synchrone Statistikerfassung mit Stichproben gearbeitet hat, weil die Tabelle sehr groß war.
- Wenn die synchronen Statistikdaten konstruiert wurden.
- Wenn die synchrone Statistikerfassung fehlgeschlagen ist, weil die Erfassungszeit überschritten wurde.

Es können höchstens zwei asynchrone Anforderungen gleichzeitig verarbeitet werden, jedoch nur für verschiedene Tabellen. Die eine Anforderung muss durch die Echtzeitstatistikerfassung, die andere durch die Überprüfung der asynchronen Statistikerfassung eingeleitet worden sein.

Die Leistungsbeeinträchtigung durch die automatische Statistikerfassung wird auf mehrere Arten minimiert:

- Die asynchrone Statistikerfassung erfolgt durch eine gedrosselte Ausführung des Dienstprogramms RUNSTATS. Die Drosselung begrenzt je nach aktuellen Datenbankaktivitäten die Ressourcenkapazität, die vom Dienstprogramm RUNSTATS beansprucht wird: Wenn die Datenbankaktivitäten zunehmen, läuft das Dienstprogramm RUNSTATS langsamer, sodass sein Ressourcenbedarf sinkt.
- Die synchrone Statistikerfassung ist auf fünf Sekunden pro Abfrage begrenzt. Dieser Wert kann durch die RTS-Optimierungsrichtlinie gesteuert werden. Wenn die synchrone Erfassung das Zeitlimit überschreitet, wird eine Anforderung zur asynchronen Erfassung übergeben.
- Die synchrone Statistikerfassung speichert die Statistiken nicht im Systemkatalog. Stattdessen werden die Statistiken in einem Statistikcache gespeichert und später durch eine asynchrone Operation im Systemkatalog gespeichert. Dadurch werden Systemaufwand und mögliche Sperrenkonflikte vermieden, die beim Aktualisieren des Systemkatalogs auftreten können. Statistiken im Statistikcache sind für nachfolgende SQL-Kompilierungsanforderungen verfügbar.
- Pro Tabelle findet nur eine synchrone Statistikerfassungsoperation statt. Andere Agenten, die eine synchrone Statistikerfassung erfordern, konstruieren Statistikdaten, sofern möglich, und setzen die Anweisungskompilierung fort. Dieses Verhalten wird auch in einer Umgebung mit partitionierten Datenbanken realisiert, in der Agenten in verschiedenen Datenbankpartitionen möglicherweise synchrone Statistiken benötigen.
- Sie können den Typ der Statistiken, die erfasst werden, anpassen, indem Sie die Statistikprofilerstellung aktivieren, sodass anhand von Informationen zu früheren Datenbankaktivitäten bestimmt wird, welche Statistiken für die Auslastung der Datenbank erforderlich sind. Alternativ können Sie auch ein eigenes Statistikprofil für eine bestimmte Tabelle erstellen.
- Nur Tabellen mit fehlenden Statistiken oder hohen Graden an Aktivität (gemessen an der Anzahl der Aktualisierungs-, Einfüge- und Löschoperationen) werden für die Statistikerfassung in Betracht gezogen. Auch wenn die Tabelle die Bedingungen für die Statistikerfassung erfüllt, werden die synchronen Statistiken nicht erfasst, sofern sie nicht für die Abfrageoptimierung erforderlich sind. In einigen Fällen kann das Abfrageoptimierungsprogramm einen Plan ohne Statistiken auswählen.
- Bei der asynchronen Statistikerfassungsprüfung werden für große Tabellen (solche mit als 4000 Seiten) Stichproben erstellt, um festzustellen, ob die Statistiken durch intensive Tabellenaktivitäten geändert wurden. Statistiken für solch große Tabellen werden nur erfasst, wenn dies gerechtfertigt ist.
- Bei der asynchronen Statistikerfassung wird das Dienstprogramm RUNSTATS automatisch zur Ausführung während des Onlineverwaltungsfensters terminiert, das in 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.
- Die synchrone Statistikerfassung und die Konstruktion von Statistikdaten richten sich nicht nach dem Onlineverwaltungsfensters, das in Ihrer Verwaltungsrichtlinie angegeben ist, weil synchrone Anforderungen sofort ausgeführt werden müssen und nur über eine begrenzte Erfassungszeit verfügen. Die synchrone Statistikerfassung und die Konstruktion von Statistikdaten richten sich nach der Richtlinie, die die Gruppe von Tabellen angibt, die zum Umfang der automatischen Statistikerfassung gehören.
- Während der Ausführung der automatischen Statistikerfassung bleiben die betroffenen Tabellen weiterhin für reguläre Datenbankaktivitäten (Aktualisierungs-, Einfüge- und Löschoperationen) verfügbar.

- Echtzeitstatistiken (synchrone oder konstruierte) werden für Kurznamen nicht erfasst. Zur automatischen Aktualisierung von Statistikdaten für Kurznamen im Systemkatalog (durch asynchrone Statistikerfassung) rufen Sie die Prozedur SYSPROC.NNSTAT auf.

Die synchrone Echtzeitstatistikerfassung wird für reguläre Tabellen, MQTs (Materialized Query Tables) und globale temporäre Tabellen ausgeführt. Asynchrone Statistiken werden für globale temporäre Tabellen nicht erfasst.

Die automatische Statistikerfassung (synchron oder asynchron) wird für folgende Elemente nicht ausgeführt:

- Statistiksichten
- Tabellen, die als flüchtig markiert sind (Tabellen, deren Feld VOLATILE in der Katalogsicht SYSCAT.TABLES definiert ist)
- Tabellen, deren Statistiken durch UPDATE-Anweisungen direkt in den SYSSTAT-Katalogsichten manuell aktualisiert wurden

  Wenn Sie Tabellenstatistikdaten manuell modifizieren, geht der Datenbankmanager davon aus, dass nun Sie für die Pflege dieser Statistikdaten verantwortlich sind. Um den Datenbankmanager dazu zu veranlassen, die Statistikdaten für eine Tabelle, deren Statistiken manuell aktualisiert wurden, wieder zu pflegen, führen Sie den Befehl **RUNSTATS** zur Erfassung von Statistiken aus oder geben die Statistikerfassung bei der Verwendung des Befehls LOAD an. Tabellen, die vor Version 9.5 erstellt wurden und deren Statistikdaten vor dem Upgrade manuell aktualisiert wurden, sind davon nicht betroffen. Die Statistikdaten dieser Tabellen werden vom Datenbankmanager weiterhin automatisch gepflegt, bis sie manuell aktualisiert werden.

Für folgende Elemente erfolgt keine Konstruktion von Statistikdaten:

- Statistiksichten
- Tabellen, deren Statistiken durch UPDATE-Anweisungen direkt in den SYSSTAT-Katalogsichten manuell aktualisiert wurden. Beachten Sie, dass, wenn die Echtzeitstatistikerfassung nicht aktiviert ist, dennoch einige Statistiken für Tabellen konstruiert werden, deren Statistiken manuell aktualisiert wurden.

In einer Umgebung mit partitionierten Datenbanken werden die Statistikdaten in nur einer Datenbankpartition erfasst und extrapoliert. Der Datenbankmanager erfasst Statistikdaten (sowohl synchrone als auch asynchrone) stets in der ersten Datenbankpartition der Datenbankpartitionsgruppe.

Aktivitäten zur Echtzeitstatistikerfassung finden nicht vor Ablauf der ersten fünf Minuten nach der Datenbankaktivierung statt.

Sowohl für statisches als auch für dynamisches SQL wird eine Verarbeitung der Echtzeitstatistiken ausgeführt.

Eine Tabelle, die durch die Verwendung des Befehls **IMPORT** abgeschnitten wurde, wird automatisch als Tabelle mit veralteten Statistiken erkannt.

Eine automatische Statistikerfassung (synchron und asynchron) macht im Cache gespeicherte dynamische Anweisungen, die auf Tabellen verweisen, für die Statistiken erfasst wurden, ungültig. Dies geschieht, damit im Cache gespeicherte dynamische Anweisungen mit den aktuellsten Statistiken reoptimiert werden können.

Asynchrone automatische Statistikerfassungsoperationen können unterbrochen werden, wenn die Datenbank inaktiviert wird. Wenn die Datenbank nicht mit dem Befehl **ACTIVATE DATABASE** oder der API explizit aktiviert wurde, wird sie inaktiviert, wenn der letzte Benutzer die Verbindung zur Datenbank trennt. Wenn Operationen unterbrochen werden, können die Fehlernachrichten in der DB2-Diagnoseprotokolldatei aufgezeichnet werden. Aktivieren Sie die Datenbank explizit, um die Unterbrechung asynchroner automatischer Statistikerfassungsoperationen zu vermeiden.

Echtzeitstatistiken und EXPLAIN-Verarbeitung

Es erfolgt keine Echtzeitverarbeitung für eine Abfrage, die durch die EXPLAIN-Funktion nur bearbeitet, jedoch nicht ausgeführt wird. In der folgenden Tabelle sind die Verhaltensweisen unter verschiedenen Werten des Sonderregisters CURRENT EXPLAIN MODE zusammengefasst.

Tabelle 72. Echtzeitstatistikerfassung als Funktion des Werts des Sonderregisters CURRENT EXPLAIN MODE

Wert in CURRENT EXPLAIN MODE	Echtzeitstatistikerfassung in Betracht gezogen?
YES	Ja
EXPLAIN	Nein
NO	Ja
REOPT	Ja
RECOMMEND INDEXES	Nein
EVALUATE INDEXES	Nein

Automatische Statistikerfassung und Statistikcache

In DB2 Version 9.5 wurde ein Statistikcache eingeführt, um synchron erfasste Statistikdaten für alle Abfragen verfügbar zu machen. Dieser Cache ist Teil des Katalogcache. In einer Umgebung mit partitionierten Datenbanken befindet sich dieser Cache nur in der Katalogdatenbankpartition. Der Katalogcache kann mehrere Einträge für dasselbe SYSTABLES-Objekt speichern, wodurch sich das Volumen des Katalogcache in allen Datenbankpartitionen erhöht. Ziehen Sie in Betracht, den Wert des Datenbankkonfigurationsparameters **catalogcache_sz** zu erhöhen, wenn die Echtzeitstatistikerfassung aktiviert ist.

Seit DB2 Version 9 können Sie den Konfigurationsadvisor dazu verwenden, die Anfangskonfiguration für neue Datenbanken zu bestimmen. Der Konfigurationsadvisor empfiehlt, den Datenbankkonfigurationsparameter **auto_stmt_stats** auf den Wert ON zu setzen.

Automatische Statistikerfassung und Statistikprofile

Synchrone und asynchrone Statistiken werden entsprechend einem Statistikprofil erfasst, das für eine Tabelle in Kraft ist. Von dieser Regel gibt es folgende Ausnahmen:

- Zur Minimierung des Aufwands für die synchrone Statistikerfassung kann der Datenbankmanager Statistikdaten durch Stichprobenentnahme erfassen. In diesem Fall können die Stichprobenrate und die Methode von denen abweichen, die im Statistikprofil angegeben sind.
- Es ist möglich, dass die synchrone Statistikerfassung Statistiken konstruiert, obwohl es vielleicht nicht möglich ist, alle im Statistikprofil angegebenen Statisti-

ken zu konstruieren. Zum Beispiel können Spaltenstatistiken wie COLCARD, HIGH2KEY und LOW2KEY nicht konstruiert werden, wenn die Spalte nicht an führender Position in einem Index enthalten ist.

Wenn die synchrone Statistikerfassung nicht alle Statistiken erfassen kann, die im Statistikprofil angegeben sind, wird eine Anforderung zur asynchronen Statistikerfassung übergeben.

Obwohl die Echtzeitstatistikerfassung mit dem Ziel entwickelt wurde, den Aufwand für die Statistikerfassung zu minimieren, sollten Sie sie zunächst in einer Testumgebung erproben, um sicherzustellen, dass es zu keinen Leistungseinbußen kommt. Dies ist in einigen OLTP-Szenarios (OLTP - Onlinetransaktionsverarbeitung) möglich, insbesondere wenn die Ausführungsdauer einer Abfrage einer oberen Begrenzung unterliegt.

Aktivieren 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 Datenbankstatistiken zu aktualisieren und zu pflegen.

Informationen zu diesem Vorgang

In Umgebungen, in denen eine einzelne Datenbankpartition auf einem Einzelprozessor betrieben wird, können Sie diese Funktionalität erweitern, indem Sie Abfragedaten sammeln und Statistikprofile generieren, die den DB2-Server bei der automatischen Erfassung der exakten Gruppe der für Ihre Auslastung erforderlichen Statistiken unterstützen. Diese Option ist in Umgebungen mit partitionierten Datenbanken, in bestimmten Umgebungen mit föderierten Datenbanken oder in Umgebungen mit aktivierter partitionsinterner Parallelität nicht verfügbar.

Für die Aktivierung der automatischen Statistikerfassung müssen Sie zuerst die Datenbank entsprechend konfigurieren, indem Sie die Datenbankkonfigurationsparameter **auto_maint** und **auto_tbl_maint** auf den Wert ON setzen. Sie haben dann die nachfolgend beschriebenen Möglichkeiten.

Vorgehensweise

1. Zur Aktivierung der Statistikerfassung im Hintergrund setzen Sie den Datenbankkonfigurationsparameter **auto_runstats** auf den Wert ON.
2. Zur Aktivierung der Statistikerfassung in Echtzeit setzen Sie die sowohl den Datenbankkonfigurationsparameter **auto_stmt_stats** als auch den Datenbankkonfigurationsparameter **auto_runstats** auf den Wert ON.
3. Zur Aktivierung der automatischen Statistikprofilerstellung setzen Sie die Datenbankkonfigurationsparameter **auto_stats_prof** und **auto_prof_upd** auf den Wert ON. Wenn der Datenbankkonfigurationsparameter **auto_runstats** ebenfalls auf den Wert ON gesetzt wird, werden die Statistikdaten automatisch anhand des generierten Profils erfasst. Beachten Sie, dass **auto_stats_prof** nicht aktiviert werden kann, wenn der Datenbankkonfigurationsparameter **section_actuals** aktiviert ist (SQLCODE -5153).

Erfassen von Statistiken mit einem Statistikprofil:

Das Dienstprogramm RUNSTATS bietet die Option zur Registrierung und Verwendung eines Statistikprofils, das den Typ von Statistiken angibt, die für eine bestimmte Tabelle zu erfassen sind, wie zum Beispiel Tabellenstatistiken, Indexstatistiken oder Verteilungsstatistiken. Diese Funktion vereinfacht die Statistikerfassung dadurch, dass Sie RUNSTATS-Optionen zur künftigen Verwendung speichern können.

Um 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, führen Sie den Befehl **RUNSTATS** mit der Option SET PROFILE ONLY aus. Zur Erfassung von Statistiken unter Verwendung eines bereits registrierten Profils führen Sie den Befehl **RUNSTATS** mit der Option USE PROFILE aus.

Zur Prüfung, welche Optionen zurzeit in dem Statistikprofil für eine bestimmte Tabelle angegeben sind, führen Sie eine Abfrage auf die Katalogsicht SYSCAT.TABLES aus. Beispiel:

```
SELECT STATISTICS_PROFILE FROM SYSCAT.TABLES WHERE TABNAME = 'EMPLOYEE'
```

Automatische Statistikprofilerstellung

Statistikprofile können außerdem automatisch mithilfe der DB2-Funktion zur automatischen Statistikprofilerstellung generiert werden. Wenn diese Funktion aktiviert ist, werden Informationen zur Datenbankaktivität erfasst und im Data-Warehouse für Abfragefeedback gespeichert. Anschließend wird ein Statistikprofil auf der Basis dieser Daten generiert. Die Aktivierung dieser Funktion kann Abhilfe schaffen, wenn Unsicherheit darüber besteht, welche Statistiken für eine bestimmte Auslastung relevant sind.

Die automatische Statistikprofilerstellung kann zusammen mit der Funktion zur automatischen Statistikerfassung verwendet werden, die automatisch Statistikpflegeoperationen auf der Basis der Informationen terminiert, die in dem automatisch generierten Statistikprofil enthalten sind.

Stellen Sie zur Aktivierung der automatischen Statistikprofilerstellung sicher, dass die automatische Tabellenverwaltung bereits durch Einstellen der entsprechenden Datenbankkonfigurationsparameter aktiviert ist. Weitere Informationen finden Sie im Abschnitt „auto_maint - Automatische Verwaltung (Konfigurationsparameter)". 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.

Die automatische Statistikprofilgenerierung wird in Umgebungen mit partitionierten Datenbanken, in bestimmten Umgebungen mit föderierten Datenbanken und bei aktivierter partitionsinterner Parallelität nicht unterstützt. Die automatische Statistikprofilgenerierung kann nicht aktiviert werden, wenn für die Datenbank Ist-Daten für Abschnitte aktiviert sind (SQLCODE -5153).

Die automatische Statistikprofilerstellung eignet sich am besten für Systeme, die umfangreiche komplexe Abfragen ausführen, die viele Vergleichselemente enthalten, große Joins verwenden oder eine extensive Gruppierung angeben. Sie eignet sich weniger für Systeme, die in erster Linie durch transaktionsorientierte Auslastungen gekennzeichnet sind.

In einer Entwicklungsumgebung, in der die Leistungseinbuße durch eine Laufzeitüberwachung leicht tolerierbar ist, setzen Sie die Konfigurationsparameter

auto_stats_prof und **auto_prof_upd** auf den Wert ON. Wenn ein Testsystem mit realistischen Daten und Abfragen arbeitet, können entsprechende Statistikprofile auf das Produktionssystem übertragen werden, auf dem Abfragen davon profitieren können, ohne dass der zusätzliche Überwachungsaufwand anfällt.

Wenn in einer Produktionsumgebung für eine bestimmte Gruppe von Abfragen Leistungsprobleme (Probleme, die sich auf fehlerhafte Statistiken zurückführen lassen) festgestellt werden, können Sie den Konfigurationsparameter **auto_stats_prof** auf ON setzen und die Zielauslastung über einen Zeitraum hinweg ausführen. Die automatische Statistikprofilgenerierung analysiert das Abfragefeedback und erstellt Empfehlungen in den SYSTOOLS.OPT_FEEDBACK_RANKING-Tabellen. Sie können diese Empfehlungen einsehen und die Statistikprofile manuell nach Bedarf optimieren. Wenn der DB2-Server die Statistikprofile auf der Grundlage dieser Empfehlungen automatisch aktualisieren soll, aktivieren Sie den Konfigurationsparameter **auto_prof_upd**, wenn Sie den Konfigurationsparameter **auto_stats_prof** aktivieren.

Erstellen des Data-Warehouse für Abfragefeedback

Das Data-Warehouse für Abfragefeedback, das für die automatische Statistikprofilerstellung erforderlich ist, besteht aus fünf Tabellen im Schema SYSTOOLS. In diesen Tabellen werden Informationen zu den Vergleichselementen, die während der Abfrageausführung festgestellt werden, sowie Empfehlungen zur Statistikerfassung gespeichert. Es handelt sich um die folgenden fünf Tabellen:

- OPT_FEEDBACK_PREDICATE
- OPT_FEEDBACK_PREDICATE_COLUMN
- OPT_FEEDBACK_QUERY
- OPT_FEEDBACK_RANKING
- OPT_FEEDBACK_RANKING_COLUMN

Verwenden Sie die Prozedur SYSINSTALLOBJECTS, um das Data-Warehouse für Abfragefeedback zu erstellen. Weitere Informationen zu dieser Prozedur, die zum Erstellen oder Löschen von Objekten im Schema SYSTOOLS verwendet wird, finden Sie im Abschnitt „SYSINSTALLOBJECTS".

Bei der automatischen Erstellung von Statistikdaten und der Profilermittlung verwendeter Speicher:

Die automatischen Funktionen zur Statistikerfassung und Reorganisation speichern Arbeitsdaten in Tabellen, die Teil Ihrer Datenbank sind. Diese Tabellen werden im Tabellenbereich SYSTOOLSPACE erstellt.

Der Tabellenbereich SYSTOOLSPACE wird automatisch mit Standardoptionen erstellt, wenn die Datenbank aktiviert wird. Der Speicherbedarf für diese Tabellen ist proportional zur Anzahl der Tabellen in der Datenbank und kann auf ca. 1 KB pro Tabelle geschätzt werden. Wenn dies für Ihre Datenbank eine beträchtliche Größe ist, kann es sinnvoll sein, den Tabellenbereich zu löschen und erneut zu erstellen, um eine angemessene Speicherkapazität zuzuordnen. Obwohl die Tabellen für die automatische Verwaltung und den Diagnosemonitor im Tabellenbereich automatisch erneut erstellt werden, gehen alle Protokolldaten, die in diesen Tabellen erfasst wurden, verloren, wenn Sie den Tabellenbereich löschen.

Aktivitätsprotokollierung für die automatische Statistikerfassung:

Das Statistikprotokoll ist eine Aufzeichnung aller Aktivitäten zur Statistikerfassung (manuell und automatisch), die in einer bestimmten Datenbank stattgefunden haben.

Der Standardname des Statistikprotokolls lautet `db2optstats.nummer.log`. Es befindet sich im Verzeichnis `$diagpath/events`. Das Statistikprotokoll ist ein Umlaufprotokoll. Das Protokollierungsverhalten wird durch die Registrierdatenbankvariable **DB2_OPTSTATS_LOG** gesteuert.

Das Statistikprotokoll kann direkt angezeigt oder mithilfe der Tabellenfunktion SYSPROC.PD_GET_DIAG_HIST abgefragt werden. Diese Tabellenfunktion gibt eine Reihe von Spalten zurück, die Standardinformationen zu jedem protokollierten Ereignis enthalten. Dies sind zum Beispiel die Zeitmarke, der DB2-Instanzname, der Datenbankname, die Prozess-ID, der Prozessname und die Thread-ID. Das Protokoll enthält außerdem generische Spalten zur Verwendung durch verschiedene Protokolleinrichtungen. In der folgenden Tabelle werden die generischen Spalten und ihre Verwendung im Statistikprotokoll beschrieben.

Tabelle 73. Generische Spalten in der Statistikprotokolldatei

Spaltenname	Datentyp	Beschreibung
OBJTYPE	VARCHAR(64)	Der Typ von Objekt, auf den sich das Ereignis bezieht. Bei der Statistikprotokollierung ist dies der Typ von Statistik, der zu erfassen ist. OBJTYPE kann sich auch auf einen Hintergrundprozess zur Statistikerfassung beziehen, wenn der Prozess gestartet oder gestoppt wird. Ferner kann sich OBJTYPE auf Aktivitäten beziehen, die von der automatischen Statistikerfassung ausgeführt werden, zum Beispiel Stichprobentests, Anfangsstichproben und Tabellenbewertung. Mögliche Werte für Aktivitäten zur Statistikerfassung: **TABLE STATS** Tabellenstatistiken werden erfasst. **INDEX STATS** Indexstatistiken werden erfasst. **TABLE AND INDEX STATS** Tabellen- und Indexstatistiken werden erfasst.

Tabelle 73. Generische Spalten in der Statistikprotokolldatei (Forts.)

Spaltenname	Datentyp	Beschreibung
(Fortsetzung)		Mögliche Werte für die automatische Statistikerfassung: **EVALUATION** Der automatische, im Hintergrund ausgeführte Statistikerfassungsprozess hat mit der Bewertungsphase begonnen. In dieser Phase werden Tabellen überprüft, um festzustellen, ob für sie aktualisierte Statistikdaten erforderlich sind. Ist dies der Fall, werden Statistikdaten erfasst. **INITIAL SAMPLING** Statistikdaten werden für eine Tabelle durch Stichprobenentnahme erfasst. Die Stichprobenstatistiken werden im Systemkatalog gespeichert. Dies ermöglicht der automatischen Statistikerfassung ein rasches Fortfahren für eine Tabelle, die keine Statistiken hat. Nachfolgende Operationen erfassen Statistikdaten ohne Stichprobenentnahme. Die anfängliche Stichprobenentnahme wird während der Bewertungsphase der automatischen Statistikerfassung ausgeführt. **SAMPLING TEST** Statistikdaten werden für eine Tabelle durch Stichprobenentnahme erfasst. Die Stichprobenstatistiken werden nicht im Systemkatalog gespeichert. Die Stichprobenstatistiken werden mit den aktuellen Katalogstatistiken verglichen, um festzustellen, ob und wann vollständige Statistikdaten für die betroffene Tabelle erfasst werden sollten. Die Stichprobenentnahme wird während der Bewertungsphase der automatischen Statistikerfassung durchgeführt. **STATS DAEMON** Der Statistikdämon ist ein Hintergrundprozess zur Verarbeitung von Anforderungen, die durch die Echtzeitstatistikerfassung übergeben werden. Dieser Objekttyp wird protokolliert, wenn der Hintergrundprozess gestartet und gestoppt wird.
OBJNAME	VARCHAR(255)	Der Name des Objekts, auf das sich das Ereignis bezieht, falls verfügbar. Für die Statistikprotokollierung ist dies der Name der Tabelle oder des Index. Wenn OBJTYPE den Wert STATS DAEMON oder EVALUATION hat, ist OBJNAME der Datenbankname und OBJNAME_QUALIFIER hat den Wert NULL.
OBJNAME_QUALIFIER	VARCHAR(255)	Für die Statistikprotokollierung ist dies das Schema der Tabelle oder des Index.

Tabelle 73. Generische Spalten in der Statistikprotokolldatei (Forts.)

Spaltenname	Datentyp	Beschreibung
EVENTTYPE	VARCHAR(24)	Der Ereignistyp ist die Aktion, die diesem Ereignis zugeordnet ist. Die folgenden Werte für die Statistikprotokollierung sind möglich: **COLLECT** Diese Aktion wird für eine Operation zur Statistikerfassung protokolliert. **START** Diese Aktion wird protokolliert, wenn der Hintergrundprozess der Echtzeitstatistikerfassung (OBJTYPE = STATS DAEMON) oder eine Bewertungsphase der automatischen Statistikerfassung (OBJTYPE = EVALUATION) gestartet wird. **STOP** Diese Aktion wird protokolliert, wenn der Hintergrundprozess der Echtzeitstatistikerfassung (OBJTYPE = STATS DAEMON) oder eine Bewertungsphase der automatischen Statistikerfassung (OBJTYPE = EVALUATION) gestoppt wird. **ACCESS** Diese Aktion wird protokolliert, wenn versucht wurde, auf eine Tabelle zu Zwecken der Statistikerfassung zuzugreifen. Dieser Ereignistyp dient zur Protokollierung eines fehlgeschlagenen Zugriffsversuchs, wenn das Objekt nicht verfügbar ist. **WRITE** Diese Aktion wird protokolliert, wenn zuvor erfasste Statistikdaten, die im Statistikcache gespeichert sind, in den Systemkatalog geschrieben werden.
FIRST_EVENTQUALIFIERTYPE	VARCHAR(64)	Der Typ des ersten Ereignisqualifikationsmerkmals. Ereignisqualifikationsmerkmale dienen zur Beschreibung der von dem Ereignis betroffenen Bereiche. Für die Statistikprotokollierung ist das erste Ereignisqualifikationsmerkmal die Zeitmarke für den Zeitpunkt, zu dem das Ereignis aufgetreten ist. Der Wert für den Typ des ersten Ereignisqualifikationsmerkmals ist AT.
FIRST_EVENTQUALIFIER	CLOB(16K)	Das erste Qualifikationsmerkmal für das Ereignis. Für die Statistikprotokollierung ist das erste Ereignisqualifikationsmerkmal die Zeitmarke für den Zeitpunkt, zu dem das Statistikereignis aufgetreten ist. Die Zeitmarke des Statistikereignisses kann sich von der Zeitmarke des Protokollsatzes, wie er in der Spalte TIMESTAMP dargestellt wird, unterscheiden.
SECOND_EVENTQUALIFIERTYPE	VARCHAR(64)	Der Typ des zweiten Ereignisqualifikationsmerkmals. Für die Statistikprotokollierung kann dieser Wert BY oder NULL sein. Dieses Feld wird für andere Ereignistypen nicht verwendet.

Tabelle 73. Generische Spalten in der Statistikprotokolldatei (Forts.)

Spaltenname	Datentyp	Beschreibung
SECOND_EVENTQUALIFIER	CLOB(16K)	Das zweite Qualifikationsmerkmal für das Ereignis. Bei der Statistikprotokollierung stellt diese Spalte für COLLECT-Ereignistypen dar, wie die Statistiken erfasst wurden. Mögliche Werte: **User** Die Statistikerfassung wurde durch einen DB2-Benutzer ausgeführt, der den Befehl **LOAD**, **REDISTRIBUTE** oder **RUNSTATS** aufgerufen oder die Anweisung CREATE INDEX abgesetzt hat. **Synchronous** Die Statistikerfassung wurde bei der SQL-Anweisungskompilierung durch den DB2-Server ausgeführt. Die Statistiken werden im Statistikcache, jedoch nicht im Systemkatalog gespeichert. **Synchronous sampled** Die Statistikerfassung wurde durch Stichprobenentnahme bei der SQL-Anweisungskompilierung durch den DB2-Server ausgeführt. Die Statistiken werden im Statistikcache, jedoch nicht im Systemkatalog gespeichert. **Fabricate** Die Statistikdaten wurden bei der SQL-Anweisungskompilierung mithilfe von Informationen konstruiert, die vom Daten- und Indexmanager gepflegt werden. Die Statistiken werden im Statistikcache, jedoch nicht im Systemkatalog gespeichert. **Fabricate partial** Nur einige Statistikdaten wurden bei der SQL-Anweisungskompilierung mithilfe von Informationen konstruiert, die vom Daten- und Indexmanager gepflegt werden. Insbesondere wurden nur die HIGH2KEY- und LOW2KEY-Werte für bestimmte Spalten konstruiert. Die Statistiken werden im Statistikcache, jedoch nicht im Systemkatalog gespeichert. **Asynchronous** Die Statistiken wurden durch einen DB2-Hintergrundprozess erfasst und im Systemkatalog gespeichert. Dieses Feld wird für andere Ereignistypen nicht verwendet.
THIRD_EVENTQUALIFIERTYPE	VARCHAR(64)	Der Typ des dritten Ereignisqualifikationsmerkmals. Für die Statistikprotokollierung kann dieser Wert DUE TO oder NULL sein.

Tabelle 73. Generische Spalten in der Statistikprotokolldatei (Forts.)

Spaltenname	Datentyp	Beschreibung
THIRD_EVENTQUALIFIER	CLOB(16K)	Das dritte Qualifikationsmerkmal für das Ereignis. Für die Statistikprotokollierung stellt diese Spalte die Ursache dar, aus der die Statistikaktivität nicht abgeschlossen werden konnte. Mögliche Werte: **Timeout** Die synchrone Statistikerfassung hat das Zeitbudget überschritten. **Error** Die Statistikaktivität ist aufgrund eines Fehlers fehlgeschlagen. **RUNSTATS error** Die synchrone Statistikerfassung ist aufgrund eines **RUNSTATS**-Fehlers fehlgeschlagen. Bei einigen Fehlern wurde die SQL-Anweisungskompilierung möglicherweise erfolgreich abgeschlossen, auch wenn die Statistikdaten nicht erfasst werden konnten. Wenn zum Beispiel nicht genügend Speicherplatz für die Statistikerfassung verfügbar ist, wird die SQL-Anweisungskompilierung fortgesetzt. **Object unavailable** Die Statistiken konnten für das Datenbankobjekt nicht erfasst werden, weil kein Zugriff auf das Objekt möglich war. Einige mögliche Ursachen: • Das Objekt ist im Z-Modus (Super Exclusive) gesperrt. • Der Tabellenbereich, in dem sich das Objekt befindet, ist nicht verfügbar. • Die Tabellenindizes müssen erneut erstellt werden. **Conflict** Die synchrone Statistikerfassung wurde nicht ausgeführt, weil eine andere Anwendung bereits synchrone Statistiken erfasste. Überprüfen Sie die Spalte FULLREC oder die **db2diag**-Protokolldateien auf Details zu diesem Fehler.
EVENTSTATE	VARCHAR(255)	Der Status des Objekts oder der Aktion infolge des Ereignisses. Für die Statistikprotokollierung gibt diese Spalte den Status der Statistikoperation an. Mögliche Werte: • Start • Success • Failure

Beispiel

In diesem Beispiel gibt die Abfrage Protokollsätze für Ereignisse bis zu einem Jahr vor der aktuellen Zeitmarke zurück, indem sie die Tabellenfunktion PD_GET_DIAG_HIST aufruft.

```
select pid, tid,
    substr(eventtype, 1, 10),
    substr(objtype, 1, 30) as objtype,
    substr(objname_qualifier, 1, 20) as objschema,
    substr(objname, 1, 10) as objname,
    substr(first_eventqualifier, 1, 26) as event1,
    substr(second_eventqualifiertype, 1, 2) as event2_type,
    substr(second_eventqualifier, 1, 20) as event2,
    substr(third_eventqualifiertype, 1, 6) as event3_type,
    substr(third_eventqualifier, 1, 15) as event3,
    substr(eventstate, 1, 20) as eventstate
  from table(sysproc.pd_get_diag_hist
    ('optstats', 'EX', 'NONE',
      current_timestamp - 1 year, cast(null as timestamp))) as sl
  order by timestamp(varchar(substr(first_eventqualifier, 1, 26), 26));
```

Die Ergebnisse werden nach der Zeitmarke in der Spalte FIRST_EVENTQUALIFIER geordnet, die den Zeitpunkt des Statistikereignisses angibt.

```
PID   TID          EVENTTYPE OBJTYPE              OBJSCHEMA OBJNAME    EVENT1                    EVENT2_ EVENT2              EVENT3_ EVENT3   EVENTSTATE
                                                                                                  TYPE                        TYPE
----- ------------ --------- -------------------- --------- -------- ------------------------- ------- ------------------- ------- -------- ----------
28399    1082145120 START     STATS DAEMON         -         PROD_DB  2007-07-09-18.37.40.398905 -       -                   -       -        success
28389  183182027104 COLLECT   TABLE AND INDEX STATS DB2USER   DISTRICT 2007-07-09-18.37.43.261222 BY      Synchronous         -       -        start
28389  183182027104 COLLECT   TABLE AND INDEX STATS DB2USER   DISTRICT 2007-07-09-18.37.43.407447 BY      Synchronous         -       -        success
28399    1082145120 COLLECT   TABLE AND INDEX STATS DB2USER   CUSTOMER 2007-07-09-18.37.43.471614 BY      Asynchronous        -       -        start
28399    1082145120 COLLECT   TABLE AND INDEX STATS DB2USER   CUSTOMER 2007-07-09-18.37.43.524496 BY      Asynchronous        -       -        success
28399    1082145120 STOP      STATS DAEMON         -         PROD_DB  2007-07-09-18.37.43.526212 -       -                   -       -        success
28389  183278496096 COLLECT   TABLE STATS          DB2USER   ORDER_LINE 2007-07-09-18.37.48.676524 BY      Synchronous sampled -       -        start
28389  183278496096 COLLECT   TABLE STATS          DB2USER   ORDER_LINE 2007-07-09-18.37.53.677546 BY      Synchronous sampled DUE TO  Timeout  failure
28389    1772561034 START     EVALUATION           -         PROD_DB  2007-07-10-12.36.11.092739 -       -                   -       -        success
28389    8231991291 COLLECT   TABLE AND INDEX STATS DB2USER   DISTRICT 2007-07-10-12.36.30.737603 BY      Asynchronous        -       -        start
28389    8231991291 COLLECT   TABLE AND INDEX STATS DB2USER   DISTRICT 2007-07-10-12.36.34.029756 BY      Asynchronous        -       -        success
28389    1772561034 STOP      EVALUATION           -         PROD_DB  2007-07-10-12.36.39.685188 -       -                   -       -        success
28399    1504428165 START     STATS DAEMON         -         PROD_DB  2007-07-10-12.37.43.319291 -       -                   -       -        success
28399    1504428165 COLLECT   TABLE AND INDEX STATS DB2USER   CUSTOMER 2007-07-10-12.37.43.471614 BY      Asynchronous        -       -        start
28399    1504428165 COLLECT   TABLE AND INDEX STATS DB2USER   CUSTOMER 2007-07-10-12.37.44.524496 BY      Asynchronous        -       -        failure
28399    1504428165 STOP      STATS DAEMON         -         PROD_DB  2007-07-10-12.37.45.905975 -       -                   -       -        success
28399    4769515044 START     STATS DAEMON         -         PROD_DB  2007-07-10-12.48.33.319291 -       -                   -       -        success
28389    4769515044 WRITE     TABLE AND INDEX STATS DB2USER   CUSTOMER 2007-07-10-12.48.33.969888 BY      Asynchronous        -       -        start
28389    4769515044 WRITE     TABLE AND INDEX STATS DB2USER   CUSTOMER 2007-07-10-12.48.34.215230 BY      Asynchronous        -       -        success
```

Verbessern der Abfrageleistung für große Statistikprotokolle:

Wenn die Statistikprotokolldateien groß sind, können Sie die Abfrageleistung verbessern, indem Sie die Protokollsätze in eine Tabelle kopieren, Indizes erstellen und anschließend Statistiken erfassen.

Vorgehensweise

1. Erstellen Sie eine Tabelle mit den entsprechenden Spalten für die Protokollsätze.

```
create table db2user.stats_log (
  pid       bigint,
  tid       bigint,
  timestamp timestamp,
  dbname    varchar(128),
  retcode   integer,
  eventtype varchar(24),
  objtype   varchar(30),
  objschema varchar(20),
  objname   varchar(30),
  event1_type varchar(20),
  event1      timestamp,
  event2_type varchar(20),
  event2      varchar(40),
  event3_type varchar(20),
  event3      varchar(40),
  eventstate  varchar(20))
```

2. Deklarieren Sie einen Cursor für eine Abfrage für SYSPROC.PD_GET_DIAG_HIST.

```
declare c1 cursor for
  select pid, tid, timestamp, dbname, retcode, eventtype,
      substr(objtype, 1, 30) as objtype,
      substr(objname_qualifier, 1, 20) as objschema,
      substr(objname, 1, 30) as objname,
      substr(first_eventqualifiertype, 1, 20),
      substr(first_eventqualifier, 1, 26),
      substr(second_eventqualifiertype, 1, 20),
      substr(second_eventqualifier, 1, 40),
      substr(third_eventqualifiertype, 1, 20),
      substr(third_eventqualifier, 1, 40),
      substr(eventstate, 1, 20)
    from table (sysproc.pd_get_diag_hist
      ( 'optstats', 'EX', 'NONE',
        current_timestamp - 1 year, cast(null as timestamp ))) as sl
```

3. Laden Sie die Statistikprotokollsätze in die Tabelle.

```
load from c1 of cursor replace into db2user.stats_log
```

4. Erstellen Sie Indizes und erfassen Sie dann Statistiken zu der Tabelle.

```
create index sl_ix1 on db2user.stats_log(eventtype, event1);
create index sl_ix2 on db2user.stats_log(objtype, event1);
create index sl_ix3 on db2user.stats_log(objname);
runstats on table db2user.stats_log with distribution and sampled
detailed indexes all;
```

Richtlinien für die Erfassung und Aktualisierung von Statistiken

Das Dienstprogramm RUNSTATS erfasst Statistikdaten für Tabellen, Indizes und Statistiksichten, um präzise Informationen für das Optimierungsprogramm zur Auswahl von Zugriffsplänen bereitzustellen.

Verwenden Sie das Dienstprogramm RUNSTATS zur Erfassung von Statistiken in folgenden Situationen:

- Nachdem Daten in eine Tabelle geladen und geeignete Indizes erstellt wurden.
- Nachdem ein neuer Index für eine Tabelle erstellt wurde.
- Nachdem eine Tabelle mit dem Dienstprogramm REORG reorganisiert wurde.
- Nachdem eine Tabelle und die zugehörigen Indizes durch UPDATE-, INSERT- oder DELETE-Operationen in erheblichem Umfang geändert wurden.
- Vor dem Binden von Anwendungsprogrammen, deren Leistung von kritischer Bedeutung ist.
- Wenn Sie aktuelle und frühere Statistiken vergleichen wollen.
- Wenn der Wert für die Größe des Vorablesezugriffs (PREFETCHSIZE) geändert wurde.
- Nachdem der Befehl REDISTRIBUTE DATABASE PARTITION GROUP ausgeführt wurde.
- Wenn XML-Spalten vorhanden sind. Wenn RUNSTATS nur zum Sammeln von Statistikdaten für XML-Spalten verwendet wird, werden Statistikdaten für Nicht-XML-Spalten, die während einer Ladeoperation oder durch eine frühere Ausführung des Dienstprogramms RUNSTATS gesammelt wurden, behalten. Wenn zuvor Statistikdaten für einige XML-Spalten gesammelt wurden, werden diese Statistikdaten ersetzt oder, falls die aktuelle RUNSTATS-Operation diese Spalten nicht umfasst, gelöscht.

Zur Verbesserung der Leistung von RUNSTATS und zur Einsparung von Plattenspeicherplatz für Statistiken sollten Sie in Betracht ziehen, nur die Spalten anzugeben, für die Datenverteilungsstatistiken erfasst werden sollten.

Sie sollten Anwendungsprogramme nach der Ausführung von RUNSTATS erneut binden (REBIND). Das Abfrageoptimierungsprogramm könnte einen anderen Zugriffsplan auswählen, wenn neue Statistikdaten verfügbar sind.

Wenn bei einem Mal nicht die gesamte Gruppe von Statistikdaten erfasst werden kann, verwenden Sie das Dienstprogramm RUNSTATS für Teilgruppen der Objekte. Wenn infolge laufender Aktivitäten an diesen Objekten Inkonsistenzen auftreten, wird während der Abfrageoptimierung eine Warnung (SQL0437W, Ursachencode 6) zurückgegeben. In diesem Fall führen Sie das Dienstprogramm RUNSTATS erneut aus, um die Verteilungsstatistiken zu aktualisieren.

Um sicherzustellen, dass Indexstatistikdaten mit der entsprechenden Tabelle synchron sind, erfassen Sie Tabellen- und Indexstatistiken zur selben Zeit. Wenn seit der letzten Erfassung von Tabellenstatistikdaten umfangreiche Änderungen an einer Tabelle vorgenommen wurden, geht durch die Aktualisierung nur der Indexstatistikdaten für diese Tabelle die Synchronisierung zwischen den beiden Arten von Statistikdaten verloren.

Die Verwendung des Dienstprogramms RUNSTATS auf einem Produktionssystem kann sich negativ auf die Auslastungsleistung auswirken. Das Dienstprogramm unterstützt jetzt eine Drosselungsoption, die zur Begrenzung des Leistungseinflusses der Ausführung von RUNSTATS während Zeiten hoher Auslastung durch Datenbankaktivitäten verwendet werden kann.

Wenn Sie Statistikdaten für eine Tabelle in einer Umgebung mit partitionierten Datenbanken erfassen, operiert das Dienstprogramm RUNSTATS nur in der Datenbankpartition, von der aus das Dienstprogramm ausgeführt wird. Die Ergebnisse aus dieser Datenbankpartition werden für die anderen Datenbankpartitionen extrapoliert. Wenn diese Datenbankpartition keinen erforderlichen Teil der Tabelle enthält, wird die Anforderung an die erste Datenbankpartition in der Datenbankpartitionsgruppe gesendet, die die erforderlichen Daten enthält.

Statistikdaten für eine Statistiksicht werden in allen Datenbankpartitionen erfasst, die Basistabellen enthalten, auf die von der Sicht verwiesen wird.

Beachten Sie die folgenden Hinweise zur Verbesserung der Effizienz von RUNSTATS und zur Nützlichkeit der Statistikdaten:

- Erfassen Sie Statistikdaten nur für Spalten, die für den Join von Tabellen verwendet werden, oder für Spalten, die in den Klauseln WHERE, GROUP BY oder ähnlichen von Klauseln von Abfragen verwendet werden. Wenn die Spalten indexiert sind, können Sie diese Spalten mit der Klausel ONLY ON KEY COLUMNS im Befehl **RUNSTATS** angeben.
- Passen Sie die Werte der Datenbankkonfigurationsparameter **num_freqvalues** und **num_quantiles** für bestimmte Tabellen und Spalten 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, verwenden Sie die Klausel COLLECT STATISTICS, 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, führen Sie RUNSTATS erneut aus, um die Statistiken zu aktualisieren.

- Da RUNSTATS Statistikdaten nur in einer Datenbankpartition erfasst, sind die Statistikdaten weniger genau, wenn die Daten nicht gleichmäßig über alle Datenbankpartitionen verteilt sind. Wenn Sie eine ungleichmäßige Datenverteilung vermuten, ziehen Sie in Betracht, die Daten vor der Ausführung des Dienstprogramms RUNSTATS mithilfe des Befehls **REDISTRIBUTE DATABASE PARTITION GROUP** erneut auf Datenbankpartitionen zu verteilen.
- Für DB2 Version 9.7 Fixpack 1 und spätere Releases können Verteilungsstatistiken für eine XML-Spalte erfasst werden. Verteilungsstatistiken werden für jeden Index zu XML-Daten erfasst, der für die XML-Spalte angegeben ist. Standardmäßig werden maximal 250 Quantile für die Verteilungsstatistikdaten für jeden Index zu XML-Daten verwendet.

  Bei der Erfassung von Verteilungsstatistiken zu einer XML-Spalte können Sie die maximale Anzahl von Quantilen angeben. Sie können die maximale Anzahl von Quantilen senken, um den Speicherplatzbedarf für XML-Verteilungsstatistiken auf der Basis Ihres jeweiligen Datenvolumens zu reduzieren. Sie können die maximale Anzahl von Quantilen auch heraufsetzen, wenn 250 Quantile nicht ausreichen, um die Verteilungsstatistiken des Datenbestands für einen Index zu XML-Daten zu erfassen.

Erfassen von Katalogstatistiken:

Mithilfe des Dienstprogramms **RUNSTATS** können Sie Statistiken zu Tabellen, Indizes und Statistiksichten erfassen. Das Abfrageoptimierungsprogramm verwendet diese Informationen zur Auswahl der optimalen Zugriffspläne für Abfragen.

Informationen zu diesem Vorgang

Informationen zu den Zugriffsrechten und Berechtigungen, die zur Verwendung dieses Dienstprogramms erforderlich sind, finden Sie in der Beschreibung zum Befehl **RUNSTATS**. Gehen Sie wie folgt vor, um Katalogstatistiken zu erfassen:

Vorgehensweise

1. Stellen Sie eine Verbindung zu der Datenbank her, in der die Tabellen, Indizes und Statistiksichten enthalten sind, für die Sie statistische 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 Statistiksichten ausgeführt werden.
3. Setzen Sie nach Abschluss der RUNSTATS-Operation eine Anweisung COMMIT ab, um Sperren freizugeben.
4. Binden Sie alle Pakete erneut (Rebind), die auf Tabellen, Indizes und Statistiksichten zugreifen, für die Sie statistische Informationen aktualisiert haben.

Ergebnisse

Anmerkung:

1. Der Befehl **RUNSTATS** unterstützt keine Verwendung von Kurznamen. Wenn Abfragen auf eine föderierte Datenbank zugreifen, verwenden Sie **RUNSTATS** zur Aktualisierung von Statistiken für Tabellen in allen Datenbanken, 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. Wenn Sie Statistikdaten für eine Tabelle in einer Umgebung mit partitionierten Datenbanken erfassen, operiert das Dienstprogramm **RUNSTATS** nur in der Datenbankpartition, von der aus das Dienstprogramm ausgeführt wird. Die Ergebnisse aus dieser Datenbankpartition werden für die anderen Datenbankpartitionen extrapoliert. Wenn diese Datenbankpartition keinen erforderlichen Teil der Tabelle enthält, wird die Anforderung an die erste Datenbankpartition in der Datenbankpartitionsgruppe gesendet, die die erforderlichen Daten enthält.

   Statistikdaten für eine Statistiksicht werden in allen Datenbankpartitionen erfasst, die Basistabellen enthalten, auf die von der Sicht verwiesen wird.
3. In DB2 Version 9.7 Fixpack 1 und späteren Releases gelten die folgenden Punkte für die Erfassung von Verteilungsstatistiken für eine Spalte des Typs XML:
   - Verteilungsstatistiken werden für jeden Index zu XML-Daten erfasst, der für eine XML-Spalte angegeben ist.
   - Der Befehl **RUNSTATS** muss Verteilungsstatistiken und Tabellenstatistiken sammeln, um Verteilungsstatistiken für einen Index zu XML-Daten zu erfassen.
   - Standardmäßig erfasst der Befehl **RUNSTATS** maximal 250 Quantile für die Verteilungsstatistik für jeden Index zu XML-Daten. Die maximale Anzahl von Quantilen für eine Spalte kann bei der Ausführung des Befehls **RUNSTATS** angegeben werden.
   - Verteilungsstatistikdaten werden für Indizes zu XML-Daten vom Typ VARCHAR, DOUBLE, TIMESTAMP und DATE erfasst. XML-Verteilungsstatistikdaten werden nicht für Indizes zu XML-Daten vom Typ VARCHAR HASHED erfasst.
   - Verteilungsstatistikdaten werden nicht für partitionierte Indizes zu XML-Daten erfasst, die für eine partitionierte Tabelle definiert sind.

Erfassen von Statistiken an einer Stichprobe der Tabellendaten:

Tabellenstatistiken werden vom Abfrageoptimierungsprogramm zur Auswahl des besten Zugriffsplans für eine Abfrage verwendet. Daher ist es wichtig, dass die Statistiken aktuell bleiben. Bei der ständig wachsenden Größe von Datenbanken wird eine effiziente Erfassung von Statistiken immer mehr zu einer Herausforderung.

Eine effektive Strategie besteht darin, Statistikdaten aus einer Zufallsstichprobe von Tabellendaten zu erfassen. Für Systeme, die von der E/A- oder Prozessorleistung abhängig sind, können die Leistungsvorteile enorm sein.

Das DB2-Produkt bietet Ihnen die Möglichkeit, effiziente Datenstichproben für die Statistikerfassung zu erstellen, durch die eine potenzielle Leistungsverbesserung für das Dienstprogramm RUNSTATS um mehrere Größenordnungen erzielt und gleichzeitig ein hoher Grad an Genauigkeit aufrechterhalten werden kann.

Es sind zwei Methoden zur Stichprobenentnahme verfügbar: die Stichprobenentnahme auf Zeilenebene und die Stichprobenentnahme auf Seitenebene. Eine Beschreibung dieser Stichprobenmethoden finden Sie im Abschnitt „Datenstichproben in Abfragen".

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 Zeile-

nebene erhebliche Leistungsverbesserungen, selbst wenn der Umfang der E/A-Operationen nicht kleiner wird, weil die Erfassung von Statistiken sehr prozessorintensiv ist.

Stichproben auf Zeilenebene bieten im Vergleich zu Stichproben auf Seitenebene Vorteile, wenn die Datenwerte eine hohe Clusterbildung aufweisen. Gegenüber der Stichprobe auf Seitenebene vermittelt die Stichprobe auf Zeilenebene einen genaueren Eindruck von den Daten, da sie *P* Prozent der Zeilen von jeder Datenseite umfasst. Bei Stichproben auf Seitenebene befinden jeweils alle Zeilen von *P* Prozent der Seiten in der Stichprobenmenge. Wenn die Zeilen eine zufällige Verteilung über die Tabelle aufweisen, ist die Genauigkeit von Statistiken über Zeilenstichproben und über Seitenstichproben ähnlich.

Jede Stichprobe wird über wiederholte Aufrufe des Befehls **RUNSTATS** hinweg nach dem Zufallsprinzip generiert, sofern nicht die Option REPEATABLE verwendet wird. In diesem Fall wird die vorherige Stichprobe erneut generiert. Diese Option kann in Fällen nützlich sein, in denen konsistente Statistiken für Tabellen erforderlich sind, deren Daten konstant bleiben.

Unterelementstatistiken:

Wenn Sie LIKE-Vergleichselemente mit dem Platzhalterzeichen % an einer beliebigen anderen Position als am Ende des Musters angeben, sollten Sie Basisinformationen zur Unterelementstruktur erfassen.

Ebenso wie das LIKE-Vergleichselement mit Platzhalterzeichen (z. B. `SELECT...FROM DOCUMENTS WHERE KEYWORDS LIKE '%simulation%'`) müssen die Spalten und die Abfrage bestimmte Kriterien erfüllen, um von Unterelementstatistiken profitieren zu können.

Tabellenspalten sollten durch Leerzeichen getrennte Unterfelder bzw. Unterelemente enthalten. Zum Beispiel könnte eine Tabelle DOCUMENTS mit vier Zeilen zu Textrecherchezwecken eine Spalte KEYWORDS mit Listen relevanter Schlüsselwörter enthalten. Die Werte in der Spalte KEYWORDS sind zum Beispiel:

```
'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 fünf Unterelementen, von denen jedes ein Wort (Schlüsselwort) ist und durch ein Leerzeichen von den anderen Wörtern getrennt ist.

Die Abfrage sollte auf diese Spalten in WHERE-Klauseln verweisen.

Das Optimierungsprogramm schätzt immer die Anzahl der Zeilen ab, die den einzelnen Vergleichselementen entsprechen. Bei solchen LIKE-Vergleichselementen mit Platzhaltern nimmt das Optimierungsprogramm an, dass die verglichene Spalte eine Reihe miteinander verketteter Elemente enthält, und schätzt die Länge jedes Elements ausgehend von der Länge der Zeichenfolge ohne führende und abschließende %-Zeichen ab. Wenn Sie Unterelementstatistiken erfassen, verfügt das Optimierungsprogramm über Informationen zur Länge der einzelnen Unterelemente und der Begrenzungszeichen. Es kann mithilfe dieser zusätzlichen Informationen präziser abschätzen, wie viele Zeilen das Vergleichselement erfüllen werden.

Zur Erfassung von Unterelementstatistiken führen Sie den Befehl **RUNSTATS** mit der Option LIKE STATISTICS aus.

Statistiken des Dienstprogramms RUNSTATS zu Unterelementen:

Das Dienstprogramm RUNSTATS versucht, Statistikdaten für Spalten der Typen CHAR und VARCHAR mit dem Codepageattribut für einen Einzelbytezeichensatz (SBCS), für FOR BIT DATA oder für UTF-8 zu erfassen, wenn Sie die Klausel LIKE STATISTICS angeben. Das Dienstprogramm RUNSTATS kann auch festlegen, dass keine Statistikdaten erfasst werden sollen, wenn es nach der Analyse der Spaltenwerte feststellt, dass solche Statistikdaten nicht angemessen sind.

SUB_COUNT
Die durchschnittliche Anzahl der Unterelemente.

SUB_DELIM_LENGTH
Durchschnittslänge jedes Begrenzers, der die Unterelemente trennt. Unter einem Begrenzer sind in diesem Kontext ein Leerzeichen oder mehrere aufeinander folgende Leerzeichen zu verstehen.

Die Registrierdatenbankvariable **DB2_LIKE_VARCHAR** beeinflusst die Art und Weise, wie das Optimierungsprogramm ein Vergleichselement der folgenden Form behandelt: `<spalte > like '%<zeichenfolge>%'`. Weitere Informationen zu dieser Registrierdatenbankvariablen finden Sie im Abschnitt „Abfragecompilervariablen".

Sie können die Werte der Unterelementstatistiken durch eine Abfrage der Katalogsicht SYSCAT.COLUMNS untersuchen. Beispiel:

```
select substr(colname, 1, 16), sub_count, sub_delim_length
  from syscat.columns where tabname = 'DOCUMENTS'
```

Bei Verwendung der Klausel LIKE STATISTICS kann die Ausführung des Dienstprogramms RUNSTATS länger dauern. Wenn Sie diese Option in Betracht ziehen, wägen Sie die Verbesserungen in der Abfrageleistung gegen diesen zusätzlichen Systemaufwand ab.

Allgemeine Regeln zur manuellen Aktualisierung von Katalogstatistiken:

Die wichtigste allgemeine Regel, die bei einer Aktualisierung von Katalogstatistiken zu beachten ist, besteht darin sicherzustellen, dass gültige Werte, Wertebereiche und Formate der verschiedenen Statistikdaten in den Sichten für diese 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 in einer Tabelle 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 SYSSTAT.TABLES zuerst aktualisieren, wird ein Fehler zurückgegeben, weil der CARD-Wert kleiner als der COLCARD-Wert wäre.

In einigen Fällen ist ein Konflikt jedoch schwer zu erkennen und es wird möglicherweise kein Fehler zurückgegeben. Dies kann insbesondere der Fall sein, wenn die betroffenen Statistiken in verschiedenen Katalogtabellen gespeichert sind.

Stellen Sie vor einer Aktualisierung von Katalogstatistiken (zumindest) Folgendes sicher:

- Numerische Statistikwerte sind entweder -1 oder größer oder gleich null (0).
- Numerische Statistikwerte, die Prozentsätze darstellen (z. B. CLUSTERRATIO in der Katalogsicht SYSSTAT.INDEXES), liegen zwischen 0 und 100.

Wenn eine Tabelle erstellt wird, werden die Katalogstatistikwerte auf -1 gesetzt, um anzugeben, dass die Tabelle keine Statistikdaten hat. Bis Statistikdaten erfasst werden, verwendet der DB2-Server Standardwerte für die Kompilierung und Optimierung von SQL- oder XQuery-Anweisungen. Eine Aktualisierung der Tabellen- oder Indexstatistiken schlägt möglicherweise fehl, wenn die neuen Werte mit den Standardwerten nicht konsistent sind. Daher wird empfohlen, nach der Erstellung einer Tabelle und vor dem Versuch, Statistiken für die Tabelle oder ihre Indizes zu aktualisieren, das Dienstprogramm RUNSTATS auszuführen.

Anmerkung:

1. Bei Zeilentypen sind die Statistikdaten auf Tabellenebene für NPAGES, FPAGES und OVERFLOW für eine untergeordnete Tabelle nicht aktualisierbar.
2. Tabellen- und Indexstatistiken auf Partitionsebene können nicht aktualisiert werden.

Regeln zur manuellen Aktualisierung von Spaltenstatistiken:

Bei der Aktualisierung von Statistiken in der Katalogsicht SYSSTAT.COLUMNS sind bestimmte Richtlinien zu beachten.

- Stellen Sie bei der manuellen Aktualisierung von HIGH2KEY- oder LOW2KEY-Werten Folgendes sicher:
  - Die Werte sind für den Datentyp der entsprechenden Benutzerspalte gültig.
  - Die Länge der Werte 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 oder LOW2KEY in Betracht gezogen werden.
  - Die Werte werden so gespeichert, dass sie in der SET-Klausel einer UPDATE-Anweisung sowie für 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 in den Spalten HIGH2KEY oder LOW2KEY finden Sie in Tabelle 74.

Tabelle 74. HIGH2KEY- und LOW2KEY-Werte nach Datentyp

Datentyp in Benutzerspalte	Benutzerdaten	Entsprechender HIGH2KEY- oder LOW2KEY-Wert
INTEGER	-12	-12
CHAR	abc	'abc'
CHAR	ab'c	'ab''c'

  - Der Wert für HIGH2KEY ist größer als der Wert für LOW2KEY, wenn es mehr als drei unterschiedliche Werte in der entsprechenden Spalte gibt.

- Die Kardinalität einer Spalte (COLCARD in der Katalogsicht SYSSTAT.COLUMNS) kann nicht größer sein als die Kardinalität der entsprechenden Tabelle oder Statistiksicht (CARD in der Katalogsicht SYSSTAT.TABLES).
- Die Anzahl der Nullwerte in einer Spalte (NUMNULLS in der Katalogsicht SYSSTAT.COLUMNS) kann nicht größer sein als die Kardinalität der entsprechenden Tabelle oder Statistiksicht (CARD in der Katalogsicht SYSSTAT.TABLES).
- Statistiken werden für Spalten, die mit LONG- oder LOB-Datentypen definiert sind, nicht unterstützt.

Regeln zur manuellen Aktualisierung von Tabellen- und Kurznamenstatistiken:

Bei der Aktualisierung von Statistiken in der Katalogsicht SYSSTAT.TABLES sind bestimmte Richtlinien zu beachten.

- Die einzigen statistischen Werte, die Sie in der Katalogsicht SYSSTAT.TABLES aktualisieren können, sind CARD, FPAGES, NPAGES und OVERFLOW. Für Tabellen mit multidimensionalem Clustering (MDC) können zudem die Werte für ACTIVE_BLOCKS aktualisiert werden.
- Der Wert für CARD muss größer als oder gleich allen COLCARD-Werten für die entsprechende Tabelle in der Katalogsicht SYSSTAT.COLUMNS sein.
- Der Wert für CARD muss größer als der Wert für NPAGES sein.
- Der Wert für FPAGES muss größer als der Wert für NPAGES sein.
- Der Wert für NPAGES muss kleiner oder gleich einem beliebigen Wert für den Abrufteil („Fetch-Teil") der Wertepaare in der Spalte PAGE_FETCH_PAIRS eines Index sein (unter der Annahme, dass diese Statistik für den Index relevant ist).
- Der Wert für CARD darf nicht kleiner oder gleich irgendeinem Wert für den Abrufteil („Fetch-Teil") der Wertepaare in der Spalte PAGE_FETCH_PAIRS eines Index sein (unter der Annahme, dass diese Statistik für den Index relevant ist).

Gehen Sie in einem System föderierter Datenbanken vorsichtig vor, wenn Sie Statistiken für einen Kurznamen über eine ferne Sicht manuell aktualisieren. Statistische Information, wie die Anzahl von Zeilen, die von diesem Kurznamen zurückgegeben werden, entsprechen möglicherweise nicht dem realen Aufwand zur Auswertung dieser fernen Sicht und können das DB2-Optimierungsprogramm irreführen. In bestimmten Fällen können ferne Sichten jedoch von Aktualisierungen der Statistiken profitieren. Dies sind zum Beispiel ferne Sichten, die für eine einzelne Basistabelle ohne Anwendung von Spaltenfunktionen auf die SELECT-Liste definiert wurden. Komplexe Sichten machen möglicherweise einen komplexen Optimierungsprozess erforderlich, bei dem jede Abfrage optimiert wird. Ziehen Sie die Erstellung lokaler Sichten über Kurznamen in Betracht, sodass das DB2-Optimierungsprogramm in der Lage ist, den Aufwand für solche Sichten präziser abzuschätzen.

Detaillierte Indexstatistiken

Durch eine Ausführung des Dienstprogramms RUNSTATS für Indizes mit der Option DETAILED werden statistische Informationen erfasst, mit deren Hilfe das Optimierungsprogramm abschätzen kann, wie viele Datenseitenabrufe abhängig von der Pufferpoolgröße erforderlich werden. Diese Zusatzinformationen unterstützen das Optimierungsprogramm bei einer besseren Abschätzung des Aufwands für den Zugriff auf eine Tabelle über einen Index.

Detaillierte Statistiken stellen kurze Informationen über die Anzahl der physischen E/A-Operationen bereit, die für den Zugriff auf die Datenseiten einer Tabelle erforderlich werden, wenn eine vollständige Indexsuche bei verschiedenen Puffergrößen ausgeführt wird. Das Dienstprogramm RUNSTATS durchsucht die Seiten eines In-

dex und modelliert dabei die verschiedenen Puffergrößen und schätzt ab, 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. Im ungünstigsten Fall könnte jede Zeile auf eine andere Seite verweisen, was höchstens 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 auf 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 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.

Erfassen Sie detaillierte Indexstatistiken, wenn Folgendes zutrifft:

- Abfragen verweisen auf Spalten, die nicht im Index enthalten sind.
- Die Tabelle hat mehrere Indizes ohne Clustering mit verschiedenen Graden von Clusterbildung.
- Der Grad der Clusterbildung unter den Schlüsselwerten ist nicht gleichmäßig.
- Indexwerte werden in nicht gleichmäßiger Weise aktualisiert.

Eine Erkennung 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 Ermittlung, ob eine dieser Bedingungen vorliegt, besteht darin, die detaillierten Statistikdaten für einen Index zu erfassen und zu untersuchen und sie zu behalten, wenn die resultierenden Wertepaare für PAGE_FETCH_PAIRS nicht linear sind.

Wenn Sie detaillierte Indexstatistiken erfassen, dauert die Ausführung des Dienstprogramms RUNSTATS länger und erfordert mehr Speicher und Verarbeitungszeit. Für die Option SAMPLED DETAILED werden zum Beispiel 2 MB des Statistikzwischenspeichers benötigt. Ordnen Sie wegen dieses Speicherbedarfs dem Datenbankkonfigurationsparameter **stat_heap_sz** zusätzliche 488 4-KB-Seiten zu. Wenn der Zwischenspeicher zu klein ist, gibt das Dienstprogramm RUNSTATS einen Fehler zurück, bevor es versucht, Statistikdaten zu erfassen.

CLUSTERFACTOR und PAGE_FETCH_PAIRS werden nur dann erfasst, wenn die Tabelle eine ausreichende Größe (mehr als ca. 25 Seiten) aufweist. In diesem Fall hat CLUSTERFACTOR einen Wert zwischen 0 und 1, während CLUSTERRATIO den Wert -1 (nicht erfasst) hat. Wenn die Tabelle relativ klein ist, werden nur Daten für die Spalte CLUSTERRATIO mit einem Wert zwischen 0 und 100 vom Dienstprogramm RUNSTATS erfasst, während für die Spalten CLUSTERFACTOR und PAGE_FETCH_PAIRS keine Daten erfasst werden. Wenn die Klausel DETAILED nicht angegeben wird, werden nur die Daten für CLUSTERRATIO erfasst.

Erfassen von Indexstatistiken:

Erfassen Sie Indexstatistiken, um das Optimierungsprogramm bei der Entscheidung zu unterstützen, ob ein bestimmter Index zur Erfüllung einer Abfrage verwendet werden sollte.

Informationen zu diesem Vorgang

Das folgende Beispiel basiert auf einer Datenbank mit dem Namen SALES, die eine Tabelle CUSTOMERS mit den Indizes CUSTIDX1 und CUSTIDX2 enthält.

Informationen zu den Zugriffsrechten und Berechtigungen, die zur Verwendung des Dienstprogramms RUNSTATS erforderlich sind, finden Sie in der Beschreibung zum Befehl **RUNSTATS**.

Gehen Sie wie folgt vor, um detaillierte Statistikdaten für einen Index zu erfassen:

Vorgehensweise

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 detaillierter Statistiken für CUSTIDX1 und CUSTIDX2:
     ```
     runstats on table sales.customers and detailed indexes all
     ```
   - Zur Erfassung detaillierter Statistiken für beide Indizes, jedoch mit Stichprobenentnahme anstelle detaillierter Berechnungen für jeden Indexeintrag:
     ```
     runstats on table sales.customers and sampled detailed indexes all
     ```

     Für die Option SAMPLED DETAILED werden 2 MB des Statistikzwischenspeichers benötigt. Ordnen Sie wegen dieses Speicherbedarfs dem Datenbankkonfigurationsparameter **stat_heap_sz** zusätzliche 488 4-KB-Seiten zu. Wenn der Zwischenspeicher zu klein ist, gibt das Dienstprogramm RUNSTATS einen Fehler zurück, bevor es versucht, Statistikdaten zu erfassen.
   - Zur Erfassung detaillierter Statistiken für Indizes mit Stichprobenentnahme sowie von Verteilungsstatistiken für die Tabelle, sodass Index- und Tabellenstatistiken konsistent sind:
     ```
     runstats on table sales.customers
       with distribution on key columns
         and sampled detailed indexes all
     ```

Regeln zur manuellen Aktualisierung von Indexstatistiken:

Bei der Aktualisierung von Statistiken in der Katalogsicht SYSSTAT.INDEXES sind bestimmte Richtlinien zu beachten.

- Die folgenden Regeln gelten für PAGE_FETCH_PAIRS:
  - Einzelne Werte in der PAGE_FETCH_PAIRS-Statistik dürfen eine Länge von 10 Stellen nicht überschreiten und müssen kleiner als der größte ganzzahlige Wert (2.147.483.647) sein.
  - Einzelne Werte in der PAGE_FETCH_PAIRS-Statistik müssen durch Leerzeichenbegrenzer getrennt werden.
  - Es muss immer eine gültige PAGE_FETCH_PAIRS-Statistik geben, wenn für CLUSTERFACTOR ein Wert größer null (0) angegeben ist.
  - Es müssen sich genau 11 Wertepaare in einer einzelnen PAGE_FETCH_PAIRS-Statistik befinden.
  - Die Puffergrößenwerte in einer PAGE_FETCH_PAIRS-Statistik (jeweils der erste Wert in einem Paar) müssen in aufsteigender Reihenfolge angegeben werden.
  - Kein Wert für die Puffergröße in einer PAGE_FETCH_PAIRS-Statistik darf den Wert MIN(NPAGES, 524.287) bei einem 32-Bit-Betriebssystem bzw. MIN(NPA-

GES, 2.147.483.647) bei einem 64-Bit-Betriebssystem überschreiten. Dabei ist NPAGES (gespeichert in der Katalogsicht SYSSTAT.TABLES) die Anzahl der Seiten in der entsprechenden Tabelle.

– Die Seitenabrufwerte (FETCH-Werte) in einer PAGE_FETCH_PAIRS-Statistik (jeweils der zweite Wert in einem Paar) müssen in absteigender Reihenfolge angegeben werden, wobei kein einzelner Wert kleiner als der NPAGES-Wert oder größer als der CARD-Wert für die entsprechende Tabelle sein darf.
– Wenn der Wert für die Puffergröße in zwei aufeinander folgenden Paaren identisch ist, müssen auch die Seitenabrufwerte in beiden Paaren identisch sein.

Das folgende Beispiel zeigt eine gültige PAGE_FETCH_PAIRS-Statistik:

```
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'
```

Dabei gilt Folgendes:

```
NPAGES = 300
CARD   = 10000
CLUSTERRATIO  =  -1
CLUSTERFACTOR = 0.9
```

- Die folgenden Regeln gelten für die Werte von CLUSTERRATIO und CLUSTERFACTOR:
  – 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, muss ein gültiger Wert in der Spalte PAGE_FETCH_PAIRS enthalten sein.
- Bei relationalen Indizes gelten die folgenden Regeln für FIRSTKEYCARD, FIRST2KEYCARD, FIRST3KEYCARD, FIRST4KEYCARD, FULLKEYCARD und INDCARD:
  – Für einen einspaltigen Index sein muss der Wert für FIRSTKEYCARD gleich dem Wert für FULLKEYCARD sein.
  – Der Wert für FIRSTKEYCARD muss gleich dem Wert SYSSTAT.COLUMNS.COLCARD für die entsprechende Spalte sein.
  – Wenn einige dieser Indexstatistikwerte nicht relevant sind, setzen Sie diese auf den Wert -1. 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:

    ```
    FIRSTKEYCARD <= FIRST2KEYCARD <= FIRST3KEYCARD <= FIRST4KEYCARD
      <= FULLKEYCARD <= INDCARD == CARD
    ```

- Für Indizes zu XML-Daten gilt die folgende Beziehung zwischen den Werten für FIRSTKEYCARD, FIRST2KEYCARD, FIRST3KEYCARD, FIRST4KEYCARD, FULLKEYCARD und INDCARD:

  ```
  FIRSTKEYCARD <= FIRST2KEYCARD <= FIRST3KEYCARD <= FIRST4KEYCARD
    <= FULLKEYCARD <= INDCARD
  ```

- 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.

Verteilungsstatistiken

Sie können zwei Arten von Statistiken über die Datenverteilung erfassen: Statistiken zu Wertehäufigkeiten und Quantilstatistiken.

- *Statistiken zu Wertehäufigkeiten* stellen Informationen über eine Spalte und den Datenwert mit der höchsten Anzahl mehrfacher Vorkommen, den Wert mit der zweithöchsten Anzahl Vorkommen usw. bis zu der Stufe bereit, die durch den Wert des Datenbankkonfigurationsparameters **num_freqvalues** angegeben wird. Wenn die Erfassung von Wertehäufigkeitsstatistiken inaktiviert werden soll, setzen Sie den Parameter **num_freqvalues** auf den Wert 0. Sie können außerdem die Klausel NUM_FREQVALUES im Befehl **RUNSTATS** für eine bestimmte Tabelle, Statistiksicht oder Spalte verwenden.
- *Quantilstatistiken* 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" (Quantile) 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 maximalen Schätzfehler durch das Optimierungsprogramm von 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 eine bestimmte Tabelle, Statistiksicht oder Spalte definieren.

Anmerkung: Das Dienstprogramm RUNSTATS beansprucht mehr Verarbeitungsressourcen und Hauptspeicher (angegeben durch den Datenbankkonfigurationsparameter **stat_heap_sz**), wenn höhere Werte für die Parameter **num_freqvalues** und **num_quantiles** verwendet werden.

Wann sind Verteilungsstatistiken zu erfassen

Bevor Sie entscheiden, ob Verteilungsstatistiken für eine Tabelle oder eine Statistiksicht hilfreich wären, stellen Sie zunächst Folgendes fest:

- Ob die Abfragen in einer Anwendung Hostvariablen verwenden.

  Verteilungsstatistiken sind am besten für dynamische und statische Abfragen geeignet, die keine Hostvariablen verwenden. Das Optimierungsprogramm nutzt Verteilungsstatistiken bei der Bewertung von Abfragen, die Hostvariablen enthalten, nur in begrenztem Umfang.
- Ob die Daten in Spalten gleichmäßig verteilt sind.

  Erstellen Sie 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;
```

Es können zwei Typen von ungleichmäßiger Datenverteilung auftreten, und dies möglicherweise zu gleicher Zeit.

- Datenwerte können sich häufen, 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:

```
  0,0
  5,1
  6,3
  7,1
  8,2
  8,4
  8,5
  9,1
 93,6
100,0
```

  Quantilstatistiken helfen dem Optimierungsprogramms bei der Einschätzung dieser Art von Datenverteilung.

  Bei der Ermittlung, ob Spaltendaten ungleichmäßig verteilt sind, können Abfragen helfen. Beispiel:

```
select c1, count(*) as occurrences
  from t1
group by c1
order by occurrences desc
```

- Es können häufig mehrfach auftretende Datenwerte vorkommen. Betrachten Sie eine Spalte, in der die Daten mit den folgenden Häufigkeiten verteilt sind:

Tabelle 75. Häufigkeit von Datenwerten in einer Spalte

Datenwert	**Häufigkeit**
20	5
30	10
40	10
50	25
60	25
70	20
80	5

  Bei der Verarbeitung zahlreicher mehrfach vorkommender Werte können dem Optimierungsprogramm Statistiken sowohl zur Wertehäufigkeit als auch zu Quantilen helfen.

Wann sind nur Indexstatistiken zu erfassen

Sie können eine Erfassung von Statistiken, die nur auf Indexdaten basieren, in folgenden Situationen in Betracht ziehen:

- Seit der letzten Ausführung des Dienstprogramms RUNSTATS wurde ein neuer Index erstellt und Sie möchten nicht erneut Statistiken für die Tabellendaten erfassen.
- Es wurden viele Änderungen an den Daten vorgenommen, die die erste Spalte eines Index betreffen.

Welche Stufe an statistischer Genauigkeit ist anzugeben

Verwenden Sie die Datenbankkonfigurationsparameter **num_quantiles** und **num_freqvalues**, um die Genauigkeit anzugeben, mit der Verteilungsstatistiken gespeichert werden sollen. Sie können die Genauigkeit auch mit den entsprechenden Optionen des Befehls **RUNSTATS** angeben, wenn Sie Statistiken für eine Tabelle oder für Spalten erfassen. Je höher Sie diese Werte setzen, desto höher ist die Genauigkeit, mit der das Dienstprogramm RUNSTATS Verteilungsstatistiken erstellt und aktualisiert. Allerdings erfordert eine größere Genauigkeit auch mehr Ressourcen, sowohl während der Ausführung von RUNSTATS selbst als auch für die Speicherung von mehr Daten 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 Angabe der Anzahl von Quantilen setzen Sie den Datenbankkonfigurationsparameter **num_quantiles** auf einen Wert zwischen 20 und 50.

- Bestimmen Sie zuerst den maximalen Fehler, der bei der Schätzung der Anzahl von Zeilen für jede Bereichsabfrage akzeptabel ist, als Prozentsatz *P*.
- Die Anzahl der Quantile sollte ca. 100/*P* für BETWEEN-Vergleichselemente und 50/*P* für alle anderen Typen von Bereichsvergleichselementen (<, <=, > oder >=) sein.

Zum Beispiel ergeben 25 Quantile einen maximalen Schätzfehler von 4 % für BETWEEN-Vergleichselemente und 2 % für Vergleichselemente mit dem Operator „>". Im Allgemeinen sollten Sie mindestens 10 Quantile angeben. Mehr als 50 Quantile sind nur bei extrem ungleichmäßig verteilten Daten erforderlich. Wenn Sie nur Häufigkeitsstatistiken 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 ein Quantil passt.

Verwendung der Verteilungsstatistiken durch das Optimierungsprogramm:

Das Optimierungsprogramm nutzt Verteilungsstatistiken zu besseren Abschätzungen des Aufwands für verschiedene Abfragezugriffspläne.

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: Zur Auswahl des günstigsten Zugriffsplans muss das Optimierungsprogramm die Anzahl der Zeilen mit einem Spaltenwert abschätzen, der ein Gleichheits- oder Bereichsvergleichselement erfüllt. Je genauer die Schätzung ist, desto größer ist auch die Wahrscheinlichkeit, dass das Optimierungsprogramm den optimalen Zugriffsplan wählt. Betrachten Sie die folgende Abfrage:

```
select c1, c2
  from table1
  where c1 = 'NEW YORK'
  and c2 <= 10
```

Nehmen Sie an, dass es einen Index für die beiden Spalten C1 und C2 gibt. Ein möglicher Zugriffsplan besteht darin, über den Index für C1 alle Zeilen mit `C1 = 'NEW YORK'` abzurufen und anschließend für jede abgerufene Zeile zu überprüfen, ob `C2 <= 10` gilt. Ein alternativer Plan wäre, über den Index für C2 alle Zeilen mit `C2 <= 10` abzurufen und anschließend für jede abgerufene Zeile zu überprüfen, ob `C1 = 'NEW YORK'` gilt. Da der Hauptaufwand der Ausführung einer Abfrage in der Regel durch das Abrufen der Zeilen verursacht wird, ist der beste Plan der Plan, der die wenigsten Abrufe erfordert. Zur Auswahl dieses Plans ist eine Abschätzung der Anzahl der Zeilen erforderlich, die das jeweilige Vergleichselement erfüllen.

Wenn keine Verteilungsstatistiken verfügbar sind, jedoch das Dienstprogramm RUNSTATS für eine Tabelle oder eine Statistiksicht 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 in einer Spalte (CARD). Die Anzahl der Zeilen, die ein Gleichheitsvergleichselement oder ein Bereichsvergleichselement erfüllen, wird unter der Annahme abgeschätzt, dass die Datenwerte in der Spalte gleiche Häufigkeiten haben und dass die Datenwerte gleichmäßig zwischen LOW2KEY und HIGH2KEY verteilt sind. Insbesondere wird die Anzahl der Zeilen, die ein Gleichheitsvergleichselement (`C1 = KEY`) erfüllen, mit dem Wert CARD/COLCARD geschätzt. Die Anzahl der Zeilen, die ein Bereichsvergleichselement (`C1 BETWEEN KEY1 AND KEY2`) erfüllen, kann mithilfe der folgenden Formel geschätzt werden:

```
    KEY2 - KEY1
------------------  x CARD
HIGH2KEY - LOW2KEY
```

Diese Schätzwerte sind nur dann realistisch, wenn die tatsächliche Verteilung der Datenwerte in einer Spalte entsprechend gleichmäßig ist. Wenn keine Verteilungsstatistikdaten verfügbar sind und entweder die Häufigkeit von Datenwerten stark variiert oder die Datenwerte sehr ungleichmäßig verteilt sind, können die Schätzungen um Größenordnungen von der Realität abweichen, sodass das Optimierungsprogramm möglicherweise einen suboptimalen Zugriffsplan auswählt.

Wenn Verteilungsstatistikdaten verfügbar sind, kann die Wahrscheinlichkeit solcher Fehler wesentlich verringert werden, indem die Statistikdaten zu Wertehäufigkeiten zur Abschätzung der Anzahl von Zeilen, die ein Gleichheitsvergleichselement erfüllen, und die Statistikdaten zu Wertehäufigkeiten in Kombination mit den Quantilstatistiken zur Abschätzung der Anzahl von Zeilen, die ein Bereichsvergleichselement erfüllen, herangezogen werden.

Erfassen von Verteilungsstatistiken für bestimmte Spalten:

Zur Gewährleistung der Effizienz sowohl von RUNSTATS-Operationen als auch der nachfolgenden Abfrageplananalyse ist es sinnvoll, Verteilungsstatistikdaten nur für die Spalten zu erfassen, auf die von Abfragen in Klauseln wie WHERE, GROUP BY und ähnlichen verwiesen wird. Darüber hinaus können Sie auch Kardinalitätsstatistiken für kombinierte Gruppen von Spalten erfassen. Das Optimierungsprogramm verwendet solche Informationen zur Erkennung von Spaltenkorrelationen bei der Abschätzung der Selektivität von Abfragen, die auf die Spalten in einer Gruppe verweisen.

Informationen zu diesem Vorgang

Das folgende Beispiel basiert auf einer Datenbank mit dem Namen SALES, die eine Tabelle CUSTOMERS mit den Indizes CUSTIDX1 und CUSTIDX2 enthält.

Informationen zu den Zugriffsrechten und Berechtigungen, die zur Verwendung des Dienstprogramms RUNSTATS erforderlich sind, finden Sie in der Beschreibung zum Befehl **RUNSTATS**.

Wenn Sie Statistikdaten für eine Tabelle in einer Umgebung mit partitionierten Datenbanken erfassen, operiert das Dienstprogramm RUNSTATS nur in der Datenbankpartition, von der aus das Dienstprogramm ausgeführt wird. Die Ergebnisse aus dieser Datenbankpartition werden für die anderen Datenbankpartitionen extrapoliert. Wenn diese Datenbankpartition keinen erforderlichen Teil der Tabelle enthält, wird die Anforderung an die erste Datenbankpartition in der Datenbankpartitionsgruppe gesendet, die die erforderlichen Daten enthält.

Gehen Sie wie folgt vor, um Statistiken für bestimmte Spalten zu erfassen:

Vorgehensweise

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 von Verteilungsstatistiken für die Spalten ZIP und YTDTOTAL:
     ```
     runstats on table sales.customers
       with distribution on columns (zip, ytdtotal)
     ```
   - Zur Erfassung von Verteilungsstatistiken für die gleichen Spalten, jedoch mit anderen Verteilungsoptionen:
     ```
     runstats on table sales.customers
       with distribution on columns (
         zip, ytdtotal num_freqvalues 50 num_quantiles 75)
     ```
   - Zur Erfassung von Verteilungsstatistiken für die Spalten, die in CUSTIDX1 und CUSTIDX2 indexiert sind:
     ```
     runstats on table sales.customer
       on key columns
     ```
   - Zur Erfassung von Statistiken für die Spalten ZIP und YTDTOTAL sowie für eine Spaltengruppe, die REGION und TERRITORY enthält:
     ```
     runstats on table sales.customers
       on columns (zip, (region, territory), ytdtotal)
     ```
   - Nehmen Sie an, dass Statistiken für Nicht-XML-Spalten zuvor durch den Befehl **LOAD** mit der Option STATISTICS erfasst wurden. Geben Sie zur Erfassung von Statistiken für die XML-Spalte MISCINFO den folgenden Befehl ein:
     ```
     runstats on table sales.customers
       on columns (miscinfo)
     ```
   - Zur Erfassung nur für die Nicht-XML-Spalten:
     ```
     runstats on table sales.customers
       excluding xml columns
     ```

     Die Klausel EXCLUDING XML COLUMNS hat Vorrang vor allen anderen Klauseln, die XML-Spalten angeben.

- In DB2 Version 9.7 Fixpack 1 und späteren Releases erfasst der folgende Befehl Verteilungsstatistikdaten mit maximal 50 Quantilen für die XML-Spalte MISCINFO. Für alle anderen Spalten in der Tabelle wird der Standardwert von 20 Quantilen verwendet:

  ```
  runstats on table sales.customers
     with distribution on columns ( miscinfo num_quantiles 50 )
     default num_quantiles 20
  ```

  Anmerkung: Die Erfassung von Verteilungsstatistiken für die XML-Spalte MISCINFO unterliegt folgenden Voraussetzungen:
  - Es müssen Tabellen- und Verteilungsstatistiken erfasst werden.
  - Es muss ein Index zu XML-Daten für die Spalte definiert sein und der für den Index angegebene Datentyp muss VARCHAR, DOUBLE, TIMESTAMP oder DATE sein.

Erweiterte Beispiele für die Verwendung von Verteilungsstatistiken:

Verteilungsstatistiken stellen Informationen über die Häufigkeit und Verteilung von Tabellendaten bereit, die das Optimierungsprogramm bei der Generierung von Abfragezugriffsplänen unterstützen, wenn die Daten nicht gleichmäßig verteilt sind und viele gleiche Werte vorhanden sind.

Die folgenden Beispiele sollen Ihnen helfen, sich mit der möglichen Verwendung von Verteilungsstatistiken durch das Optimierungsprogramm vertraut zu machen.

Beispiel mit Statistiken zur Wertehäufigkeit

Betrachten Sie eine Abfrage, die ein Gleichheitsvergleichselement der Form `C1 = KEY` enthält. Wenn Statistiken zur Wertehäufigkeit verfügbar sind, kann das Optimierungsprogramm diese wie folgt zur Auswahl eines geeigneten Zugriffsplans verwenden:

- Wenn `KEY` einer der *N* häufigsten Werte ist, 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 (COLCARD - *N*) Werte eine gleichmäßige Verteilung aufweisen. Das heißt, die Anzahl der Zeilen wird nach der folgenden Formel (1) abgeschätzt:

  ```
  CARD - NUM_FREQ_ROWS
  --------------------
      COLCARD - N
  ```

Dabei ist CARD die Anzahl von Zeilen in der Tabelle, COLCARD ist die Kardinalität der Spalte und NUM_FREQ_ROWS ist die Gesamtanzahl von Zeilen mit einem Wert, der gleich einem der *N* häufigsten Werte ist.

Betrachten Sie zum Beispiel eine Spalte C1, deren Datenwerte die folgenden Häufigkeiten aufweisen:

Datenwert	Häufigkeit
1	2
2	3
3	40
4	4

Datenwert	Häufigkeit
5	1

Die Anzahl der Zeilen in der Tabelle ist 50, die Spaltenkardinalität 5. Genau 40 Zeilen erfüllen das Vergleichselement `C1 = 3`. Wenn angenommen wird, dass die Daten gleichmäßig verteilt wären, würde das Optimierungsprogramm die Anzahl der Zeilen, die das Vergleichselement erfüllen mit 50/5 = 10 abschätzen, was einen Fehler von -75 % ausmacht. Wenn Statistikdaten zur Wertehäufigkeit auf der Basis nur des häufigsten Werte (d. h. *N* = 1) verfügbar sind, wird die Anzahl der Zeilen mit 40 abgeschätzt, sodass kein Fehler entsteht.

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 %:

```
geschätzte Zeilen - tatsächliche Zeilen
---------------------------------------  X 100
         tatsächliche Zeilen
10 - 2
------  X 100 = 400%
   2
```

Bei Verwendung der Häufigkeitsstatistik (*N* = 1) schätzt das Optimierungsprogramm die Anzahl der Zeilen, die diesen Wert enthalten, anhand der oben gezeigten Formel (1) wie folgt ab:

```
(50 - 40)
--------- = 3
 (5 - 1)
```

Der Fehler wird dabei um eine Größenordnung verringert:

```
 3 - 2
 ----- = 50%
   2
```

Beispiel mit Quantilstatistiken

In den folgenden Ausführungen zu Quantilstatistiken wird der Begriff „K-Quantile" verwendet. Das *K-Quantil* für eine Spalte ist der kleinste Datenwert *V*, sodass mindestens *K* Zeilen Datenwerte enthalten, die kleiner oder gleich *V* sind. Ein K-Quantil lässt sich berechnen, indem die Zeilen in der Spalte aufsteigenden 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 C1, die folgende Werte enthält:

```
  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

- Genau sieben Zeilen erfüllen das Vergleichselement `C <= 8,5`. Bei einer angenommenen gleichmäßigen Datenverteilung wird durch die folgende Formel (2):

  ```
     KEY2 - KEY1
  ------------------  X CARD
  HIGH2KEY - LOW2KEY
  ```

  mit LOW2KEY anstelle von KEY1 die Anzahl der Zeilen, die das Vergleichselement erfüllen wie folgt abgeschätzt:

  ```
   8,5 - 5,1
  ----------  X 10 ≈ 0
  93,6 - 5,1
  ```

  Die Notation '≈' bedeutet „annähernd gleich". Der Fehler bei dieser Schätzung ist annähernd -100 %.

  Wenn Quantilstatistiken verfügbar sind, schätzt das Optimierungsprogramm die Anzahl der Zeilen, die dieses Vergleichselement erfüllen, durch den Wert *K* ab, der dem Wert 8,5 (dem höchsten Wert in einem der Quantile) entspricht. Dies ist 7. In diesem Fall reduziert sich der Fehler auf 0.

- Genau acht Zeilen erfüllen das Vergleichselement `C <= 10`. Wenn das Optimierungsprogramm von einer gleichmäßigen Datenwertverteilung ausgeht und Formel (2) verwendet, 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 mithilfe von Quantilen die Anzahl von Zeilen, die das Vergleichselement erfüllen, als `r_1 + r_2` abschätzen, wobei `r_1` die Anzahl der Zeilen ist, die das Vergleichselement `C <= 8,5` erfüllen, und `r_2` die Anzahl der Zeilen ist, die das Vergleichselement `C > 8,5 AND C <= 10` erfüllen. Wie im vorigen Beispiel gilt `r_1 = 7`. Zur Abschätzung von `r_2` verwendet das Optimierungsprogramm die lineare Interpolation:

  ```
         10 - 8,5
  r_2 ≈ ----------  X (Anzahl Zeilen mit Wert > 8,5 und <= 100,0)
         100 - 8,5

         10 - 8,5
  r_2 ≈ ----------  X (10 - 7)
         100 - 8,5

         1,5
  r_2 ≈ -----   X (3)
         91,5

  r_2 ≈ 0
  ```

  Die abschließende Schätzung ist r_1 + r_2 ≈ 7, wobei der Fehler nur -12,5 % beträgt.

Quantile verbessern die Schätzgenauigkeit in den obigen Beispielen deswegen, weil die realen Datenwerte „Häufungen" im Bereich von 5 bis 10 aufweisen, 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

Nehmen Sie an, dass *K*-Quantile für *K* = 5, 25, 75, 95 und 100 verfügbar sind:

K	***K*-Quantil**
5	20
25	40
75	50
95	70
100	80

Nehmen Sie außerdem an, dass Statistiken zu Wertehäufigkeiten auf der Basis der drei häufigsten Werte verfügbar sind.

Genau zehn Zeilen erfüllen das Vergleichselement `C BETWEEN 20 AND 30`. Unter der Annahme einer gleichmäßigen Datenverteilung und unter Verwendung der Formel (2) wird die Anzahl der Zeilen, die das Vergleichselement erfüllen, wie folgt abgeschätzt:

```
30 - 20
-------  X 100 = 25
70 - 30
```

Diese Abschätzung enthält einen Fehler von 150 %.

Unter Verwendung der Häufigkeitsstatistik und der Quantilstatistik 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 (1) wird `r_1` folgendermaßen abgeschätzt:

```
100 - 80
-------- = 5
  7 - 3
```

`r_2` wird mit linearer Interpolation folgendermaßen abgeschätzt:

```
  30 - 20
  -------  X (Anzahl Zeilen mit Wert > 20 und <= 40)
  40 - 20

  30 - 20
= -------  X (25 - 5)
  40 - 20

= 10
```

Dies ergibt einen endgültigen Schätzwert von 15, wodurch der Schätzfehler um den Faktor 3 verringert wird.

Regeln zur manuellen Aktualisierung von Verteilungsstatistiken:

Bei der Aktualisierung von Statistiken in der Katalogsicht SYSSTAT.COLDIST sind bestimmte Richtlinien zu beachten.

- Statistiken zu Wertehäufigkeiten:
  - VALCOUNT-Werte müssen bei steigenden Werten für SEQNO gleich bleiben oder abnehmen.
  - Die Anzahl von COLVALUE-Werten muss kleiner oder gleich der Anzahl der unterschiedlichen (distinkten) Werte in der Spalte sein. Diese Anzahl wird in SYSSTAT.COLUMNS.COLCARD 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 SYSSTAT.TABLES.CARD gespeichert.
  - In der Regel sollten COLVALUE-Werte zwischen dem zweithöchsten und dem zweitniedrigsten Datenwert für die Spalte liegen. Diese beiden Werte werden in der Spalte HIGH2KEY bzw. LOW2KEY in der Sicht SYSSTAT.COLUMNS gespeichert. Es kann nur einen häufigen Wert geben, der größer als der HIGH2KEY-Wert ist, und einen häufigen Wert, der kleiner als der LOW2KEY-Wert ist.
- Quantilstatistiken:
  - COLVALUE-Werte müssen bei steigenden Werten für SEQNO gleich bleiben oder abnehmen.
  - VALCOUNT-Werte müssen bei steigenden Werten für SEQNO zunehmen.
  - Der größte COLVALUE-Wert muss einen entsprechenden Eintrag in der Spalte VALCOUNT haben, der gleich der Anzahl von Zeilen in der Spalte ist.
  - In der Regel sollten COLVALUE-Werte zwischen dem zweithöchsten und dem zweitniedrigsten Datenwert für die Spalte liegen. Diese beiden Werte werden in der Spalte HIGH2KEY bzw. LOW2KEY in der Sicht SYSSTAT.COLUMNS gespeichert.

Nehmen Sie an, dass Verteilungsstatistikdaten für eine Spalte C1 mit *Z* Zeilen verfügbar sind, und Sie möchten die Statistiken so modifizieren, dass sie einer Spalte entsprechen, die dieselben relativen Proportionen von Datenwerten, jedoch mit (*F* x *Z*) Zeilen hat. Um die Werte der Häufigkeits- oder Quantilstatistiken um einen Faktor *F* zu erhöhen, multiplizieren Sie jeden VALCOUNT-Eintrag mit *F*.

Statistiken für benutzerdefinierte Funktionen

Zur Erstellung statistischer Informationen für benutzerdefinierte Funktionen (UDFs) bearbeiten Sie die Katalogsicht SYSSTAT.ROUTINES.

Das Dienstprogramm RUNSTATS erfasst keine Statistiken für benutzerdefinierte Funktionen. Wenn UDF-Statistiken verfügbar sind, können sie vom Optimierungs-

programm zur Abschätzung des Aufwands für verschiedene Zugriffspläne verwendet werden. Sind keine Statistiken verfügbar, verwendet das Optimierungsprogramm Standardwerte, die von einer einfachen benutzerdefinierten Funktion ausgehen.

In Tabelle 76 sind die Katalogsichtspalten aufgeführt, für die Sie Schätzwerte zur Verbesserung der Leistung bereitstellen können. Beachten Sie, dass nur Spaltenwerte in der Katalogsicht SYSSTAT.ROUTINES (nicht in SYSCAT.ROUTINES) von Benutzern geändert werden können.

Tabelle 76. Funktionsstatistiken (SYSCAT.ROUTINES und SYSSTAT.ROUTINES)

Statistik	Beschreibung
IOS_PER_INVOC	Geschätzte Anzahl der Schreib- oder Leseanforderungen, die jedes Mal ausgeführt werden, wenn eine Funktion aufgerufen wird
INSTS_PER_INVOC	Geschätzte Anzahl der Maschineninstruktionen, die jedes Mal ausgeführt werden, wenn eine Funktion aufgerufen wird
IOS_PER_ARGBYTE	Geschätzte Anzahl der Schreib- oder Leseanforderungen, die für jedes Eingabeargumentbyte ausgeführt werden
INSTS_PER_ARGBYTE	Geschätzte Anzahl der Maschineninstruktionen, die für jedes Eingabeargumentbyte ausgeführt werden
PERCENT_ARGBYTES	Geschätzter Durchschnittsprozentwert der Eingabeargumentbyte, die von einer Funktion tatsächlich verarbeitet werden
INITIAL_IOS	Geschätzte Anzahl der Schreib- oder Leseanforderungen, die ausgeführt werden, wenn eine Funktion zum ersten oder letzten Mal aufgerufen wird
INITIAL_INSTS	Geschätzte Anzahl der Maschineninstruktionen, die ausgeführt werden, wenn eine Funktion zum ersten oder letzten Mal aufgerufen wird
CARDINALITY	Geschätzte Anzahl von Zeilen, die von einer Tabellenfunktion generiert werden

Betrachten Sie zum Beispiel die benutzerdefinierte Funktion EU_SHOE, die eine amerikanische Schuhgröße in die entsprechende europäische Schuhgröße umwandelt. Für diese UDF könnten Sie die Werte der Statistikspalten in der Katalogsicht SYSSTAT.ROUTINES mit folgenden Werten versehen:

- INSTS_PER_INVOC: Geben Sie die geschätzte Anzahl der Maschineninstruktionen an, die zu folgenden Operationen erforderlich sind:
  - Aufrufen von EU_SHOE
  - Initialisieren der Ausgabezeichenfolge
  - Rückgabe des Ergebnisses
- INSTS_PER_ARGBYTE: Geben Sie die geschätzte Anzahl der Maschineninstruktionen an, die zur Umwandlung der Eingabezeichenfolge in eine europäische Schuhgröße erforderlich ist.

- PERCENT_ARGBYTES: Geben Sie den Wert 100 an, der anzeigt, dass die gesamte Eingabezeichenfolge umzuwandeln ist.
- INITIAL_INSTS, IOS_PER_INVOC, IOS_PER_ARGBYTE und INITIAL_IOS: Geben Sie für diese Spalten jeweils den Wert 0 an, da diese UDF nur Berechnungen ausführt.

PERCENT_ARGBYTES würde für eine Funktion verwendet, die nicht immer die gesamte Eingabezeichenfolge verarbeitet. Ein Beispiel wäre die 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 dass im Durchschnitt 75 % des zweiten Arguments zum Auffinden des ersten durchsucht werden. Aufgrund dieser Informationen und der folgenden Annahmen sollte PERCENT_ARGBYTES auf den Wert 75 gesetzt werden:

- 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 oder INITIAL_IOS zur Eintragung der geschätzten Anzahl von Maschineninstruktionen bzw. Schreib- oder Leseanforderungen verwenden, die beim ersten bzw. letzten Aufruf einer Funktion ausgeführt werden. Diese Werte können zum Beispiel den Aufwand zur Einrichtung eines Arbeitspufferbereichs darstellen.

Informationen über Ein-/Ausgaben und die Instruktionen, die von einer benutzerdefinierten Funktion (UDF) verwendet werden, erhalten Sie über die Ausgaben des Compilers für die Programmiersprache oder durch Überwachungstools, die für Ihr Betriebssystem verfügbar sind.

Katalogstatistiken für Modellierung und Fallstudien

Sie können die Auswirkungen von Änderungen an bestimmten Statistikdaten im Systemkatalog auf die Datenbankleistung zu Planungszwecken untersuchen.

Diese Aktualisierbarkeit ausgewählter Systemkatalogstatistiken bietet Ihnen folgende Möglichkeiten:

- Sie können eine Abfrageleistung auf einem Entwicklungssystem unter Verwendung von Systemstatistiken eines Produktionssystems modellieren.
- 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.

Zur Änderung von Statistiken für Tabellen und Indizes und der zugehörigen Komponenten müssen Sie über eine explizite DBADM-Berechtigung für die Datenbank verfügen. Benutzer mit der Berechtigung DATAACCESS können UPDATE-Anweisungen an Sichten ausführen, die im Schema SYSSTAT definiert sind, um Werte in den Statistikspalten zu ändern.

Benutzer ohne die Berechtigung DATAACCESS können nur die Zeilen sehen, die Statistikdaten für Objekte enthalten, für die sie das Zugriffsrecht CONTROL haben. Wenn Sie nicht über die Berechtigung DATAACCESS verfügen, 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 diese Aktualisierungen nur lokale Metadaten betreffen (Tabellenstatistiken zu Datenquellen werden nicht geändert) und sich nur auf die globale Zugriffsstrategie auswirken, die vom DB2-Optimierungsprogramm generiert wird.
- Eigentumsrecht für benutzerdefinierte Funktionen (UDFs).

Der folgende Code ist ein Beispiel dafür, wie Tabellenstatistiken für die Tabelle EMPLOYEE aktualisiert werden können:

```
update sysstat.tables
  set
    card = 10000,
    npages = 1000,
    fpages = 1000,
    overflow = 2
  where tabschema = 'MELNYK'
    and tabname = 'EMPLOYEE'
```

Gehen Sie beim manuellen Aktualisieren von Katalogstatistiken mit besonderer Sorgfalt vor. Willkürliche Änderungen können die Leistung nachfolgender Abfragen ernsthaft beeinflussen. Sie können eine beliebige der folgenden Methoden verwenden, um die Statistiken in Ihrem Entwicklungssystem wieder auf einen konsistenten Zustand zurückzusetzen:

- Sie können die UOW (Unit of Work, Arbeitseinheit), in der Ihre manuellen Änderungen ausgeführt wurden, durch ein Rollback rückgängig machen (unter der Annahme, dass die UOW noch nicht mit Commit festgeschrieben wurde).
- Sie können das Dienstprogramm RUNSTATS ausführen, um die Katalogstatistiken zu aktualisieren.
- Sie können die Katalogstatistiken aktualisieren, um anzugeben, dass keine Statistiken erfasst wurden. Wenn zum Beispiel der Wert der Spalte NPAGES auf -1 gesetzt wird, gibt dies an, dass diese Statistik nicht erfasst wurde.
- Sie können die Änderungen, die Sie ausgeführt haben, zurücknehmen. Diese Methode ist nur möglich, wenn Sie mit dem Befehl **db2look** die Statistiken erfasst haben, bevor Sie Änderungen vorgenommen haben.

Wenn das Optimierungsprogramm feststellt, dass ein Wert oder eine Kombination von Werten nicht gültig ist, verwendet es die entsprechenden Standardwerte und gibt eine Warnung zurück. Dies geschieht jedoch nur sehr selten, da die meisten Prüfungen stattfinden, wenn die Statistiken aktualisiert werden.

Statistiken zur Modellierung von Produktionsdatenbanken:

Manchmal ist es wünschenswert, auf einem Entwicklungssystem einen Teil der Daten eines Produktionssystems zu haben. Jedoch sind die Zugriffspläne, die auf Entwicklungssystemen ausgewählt werden, nicht unbedingt dieselben wie die, die auf dem Produktionssystem ausgewählt würden.

In einigen Fällen ist es erforderlich, die Katalogstatistiken und die Konfiguration des Entwicklungssystems so zu aktualisieren, dass sie dem Produktionssystem entsprechen.

Der Befehl **db2look** kann im Mimic-Modus (mit der Option '-m') zur Generierung von DML-Anweisungen (DML, Data Manipulation Language, Datenbearbeitungssprache) verwendet werden, die erforderlich sind, um die Katalogstatistiken einer Entwicklungsdatenbank und einer Produktionsdatenbank in Übereinstimmung zu bringen.

Nach der Ausführung der von **db2look** generierten UPDATE-Anweisungen im Entwicklungssystem, kann dieses System zur Prüfung der Zugriffspläne verwendet werden, die in der Produktionsdatenbank generiert werden. Da das Optimierungsprogramm den Ein-/Ausgabeaufwand mithilfe der Konfiguration von Tabellenbereichen abschätzt, müssen die Tabellenbereiche des Entwicklungssystems vom selben Typ (SMS oder DMS) sein und die gleiche Anzahl von Containern besitzen wie die Tabellenbereiche des Produktionssystems. Auf dem Testsystem ist unter Umständen weniger physischer Hauptspeicher verfügbar als auch dem Produktionssystem. Eventuell ist es nicht möglich, für die speicherbezogenen Konfigurationsparameter auf dem Testsystem dieselben Werte wie auf dem Produktionssystem festzulegen. Mit dem Befehl **db2fopt** können Sie Werte zuordnen, die während der Anweisungskompilierung vom Optimierungsprogramm verwendet werden sollen. Falls das Produktionssystem beispielsweise mit der Einstellung **sortheap**=20000 ausgeführt wird und das Testsystem nur mit der Einstellung **sortheap**=5000 ausgeführt werden kann, können Sie mit dem Befehl **db2fopt** auf dem Testsystem den Wert 20000 für **opt_sortheap** festlegen. Bei der Auswertung von Zugriffsplänen verwendet das Abfrageoptimierungsprogramm während der Anweisungskompilierung **opt_sortheap** anstelle von **sortheap**.

Vermeiden von manuellen Aktualisierungen der Katalogstatistiken

Der DB2-Datenserver unterstützt das manuelle Aktualisieren von Katalogstatistiken durch die Ausführung von UPDATE-Anweisungen für Sichten im Schema SYSSTAT.

Diese Möglichkeit kann nützlich sein, wenn eine Produktionsdatenbank auf einem Testsystem simuliert werden soll, um Abfragezugriffspläne zu untersuchen. Das Dienstprogramm **db2look** leistet nützliche Dienste bei der Erfassung der DDL- und UPDATE-Anweisungen für Sichten im Schema SYSSTAT, die anschließend auf einem anderen System ausgeführt werden können.

Vermeiden Sie eine Beeinflussung des Abfrageoptimierungsprogramms, indem Sie manuell falsche Statistikdaten angeben, um einen bestimmten Abfragezugriffsplan zu erzwingen. Dieses Verfahren kann zwar für einige Abfragen zu einer besseren Leistung führen, jedoch für andere die Leistung auch verschlechtern. Ziehen Sie zunächst andere Optimierungsoptionen in Betracht (z. B. die Verwendung von Optimierungsrichtlinien und -profilen), bevor Sie auf dieses Verfahren zurückgreifen. Wenn dieses Verfahren doch erforderlich werden sollte, stellen Sie sicher, dass Sie die ursprünglichen Statistikdaten für den Fall aufzeichnen, dass sie wiederhergestellt werden müssen.

Minimieren der Leistungsbeeinträchtigung durch das Dienstprogramm RUNSTATS

Die Leistung des Dienstprogramms RUNSTATS kann auf verschiedene Arten verbessert werden.

Gehen Sie wie folgt vor, um die Leistungsbeeinträchtigung durch dieses Dienstprogramm zu minimieren:

- Begrenzen Sie die Spalten, für die Statistiken erfasst werden sollen, indem Sie die Klausel COLUMNS verwenden. Viele Spalten werden nie von Vergleichselementen in der Abfrageauslastung verwendet, sodass keine Statistiken für sie erforderlich sind.
- Begrenzen Sie die Spalten, für die Verteilungsstatistiken erfasst werden, wenn die Daten eher gleichmäßig verteilt sind. Die Erfassung von Verteilungsstatistiken erfordert mehr CPU-Zeit und Hauptspeicher als die Erfassung von Basisspaltenstatistiken. Allerdings ist es zur Ermittlung, ob die Werte einer Spalte gleichmäßig verteilt sind, erforderlich, bereits vorhandene Statistiken zur Hand zu haben oder die Daten abzufragen. Diese Methode geht zudem davon aus, dass die Daten gleichmäßig verteilt bleiben, wenn die Tabelle geändert wird.
- Begrenzen Sie die Anzahl der Seiten und Zeilen, die verarbeitet werden, indem Sie eine Stichprobenentnahme auf Seiten- oder Zeilenebene (durch Angabe der Klausel TABLESAMPLE SYSTEM bzw. BERNOULLI) verwenden. Beginnen Sie mit einer Stichprobenentnahme von 10 % auf Seitenebene, indem Sie `TABLESAMPLE SYSTEM(10)` angeben. Prüfen Sie die Genauigkeit der Statistiken und stellen Sie fest, ob sich die Systemleistung aufgrund von Änderungen im Zugriffsplan verschlechtert hat. Wenn sie sich verschlechtert hat, versuchen Sie eine Stichprobenentnahme von 10 % auf Zeilenebene, indem Sie `TABLESAMPLE BERNOULLI(10)` angeben. Wenn die Genauigkeit der Statistiken nicht ausreicht, erhöhen Sie den Stichprobenbetrag. Wenn Sie die Stichprobenentnahme auf Seiten- oder Zeilenebene mit dem Befehl **RUNSTATS** verwenden, verwenden Sie denselben Stichprobensatz für Tabellen, die durch Join verknüpft werden. Dies ist wichtig, um sicherzustellen, dass die Joinspaltenstatistiken den gleichen Genauigkeitsgrad besitzen.
- Erfassen Sie Indexstatistiken bei der Indexerstellung, indem Sie die Option COLLECT STATISTICS in der Anweisung CREATE INDEX verwenden. Diese Methode ist schneller als die Ausführung einer separaten RUNSTATS-Operation nach der Erstellung des Index. Sie stellt außerdem sicher, dass für den neuen Index sofort nach der Erstellung Statistiken generiert werden, sodass das Optimierungsprogramm den Aufwand bei Verwendung des Index genau abschätzen kann.
- Erfassen Sie Statistiken, wenn Sie den Befehl **LOAD** mit der Option REPLACE ausführen. Diese Methode ist schneller als die Ausführung einer separaten RUNSTATS-Operation nach Abschluss der Ladeoperation. Sie stellt außerdem sicher, dass für die Tabelle die aktuellsten Statistikdaten sofort nach dem Laden der Daten verfügbar sind, sodass das Optimierungsprogramm den Aufwand bei Verwendung der Tabelle genau abschätzen kann.

In einer Umgebung mit partitionierten Datenbanken erfasst das Dienstprogramm RUNSTATS Statistikdaten nur aus einer Datenbankpartition. Wenn der Befehl **RUNSTATS** für eine Datenbankpartition ausgeführt wird, in der sich die Tabelle befindet, werden die Statistiken dort erfasst. Ist dies nicht der Fall, werden die Statistiken in der ersten Datenbankpartition in der Datenbankpartitionsgruppe für die Tabelle erfasst. Stellen Sie zur Gewährleistung konsistenter Statistiken sicher, dass Statistiken für durch einen Join verknüpfte Tabellen in derselben Datenbankpartition erfasst werden.

Datenkomprimierung und Leistung

Die Datenkomprimierung kann verwendet werden, um das Volumen der Daten zu verringern, das von der Platte gelesen oder auf die Platte geschrieben werden muss, sodass sich der Ein-/Ausgabeaufwand reduziert.

Zwei Formen von Datenkomprimierung sind gegenwärtig verfügbar:

- Bei der *Wertkomprimierung* werden doppelte Einträge für einen Wert entfernt, sodass nur eine Kopie gespeichert und die Positionen aller Verweise auf den gespeicherten Wert aufgezeichnet werden.
- Bei der *Zeilenkomprimierung* werden sich wiederholende Muster, die sich über mehrere Spaltenwerte innerhalb einer Zeile erstrecken, durch kürzere Symbolzeichenfolgen ersetzt. Die Logik der Zeilenkompression durchsucht eine zu komprimierende Tabelle nach sich wiederholenden und doppelten Daten. Ein Komprimierungswörterverzeichnis (Compression Dictionary) enthält kurze, numerische Schlüssel für diese Daten. In einer komprimierten Zeile ersetzen diese Schlüssel die tatsächlichen Daten.

Vor DB2 Version 9.1 mussten Sie ein Komprimierungswörterverzeichnis durch die Ausführung einer Offlinetabellenreorganisation manuell erstellen. Ab Version 9.5 wird automatisch ein Komprimierungswörterverzeichnis für Tabellen erstellt, für die die Datenkomprimierung aktiviert wird.

Mit diesem Release wird die autonome Zeilenkomprimierung, die in Version 9.5 für permanente Tabellen eingeführt wurde, erweitert, sodass sie nun auch alle temporären Tabellen mit einschließt. Die Datenkomprimierung für temporäre Tabellen ist durch folgende Merkmale gekennzeichnet:

- Sie verringert den Bedarf an temporären Plattenspeicherplatz für große und komplexe Abfragen.
- Sie erhöht die Abfrageleistung.

Die Datenkomprimierung für temporäre Tabellen wird unter DB2 Storage Optimization Feature automatisch aktiviert.

Für jede temporäre Tabelle, für die die Zeilenkomprimierung infrage kommt, sind 2 - 3 MB an zusätzlichem Speicher zur Erstellung des Komprimierungswörterverzeichnis erforderlich. Dieser Speicher bleibt zugeordnet, bis das Komprimierungswörterverzeichnis erstellt ist.

Indexobjekte und Indizes für komprimierte temporäre Tabellen können ebenfalls komprimiert werden, um Speicherkosten zu reduzieren. Dies ist insbesondere für große OLTP-Umgebungen (OLTP - Onlinetransaktionsverarbeitung) und Data-Warehouse-Umgebungen nützlich, in denen gewöhnlich viele umfangreiche Indizes vorhanden sind. In beiden Fällen kann die Indexkomprimierung zu beträchtlichen Leistungsverbesserungen in Umgebungen, die von der E/A-Leistung abhängig sind, sowie zu geringen bis keinen Leistungseinbußen in Umgebungen, die von der CPU-Leistung abhängig sind, führen.

Ist die Komprimierung für eine Tabelle mit einer XML-Spalte aktiviert, werden die in dem XDA-Objekt gespeicherten XML-Daten ebenfalls komprimiert. Es wird ein separates Komprimierungswörterverzeichnis für die XML-Daten in dem XDA-Objekt gespeichert. Dies gilt auch für Tabellen, deren XML-Spalten in der aktuellen Version des DB2-Produkts hinzugefügt wurden. Die XDA-Komprimierung wird für

Tabellen, deren XML-Spalten mit Versionen vor dieser Version erstellt wurden, nicht unterstützt. Bei derartigen Tabellen wird lediglich das Datenobjekt komprimiert.

Verringern des Protokollierungsaufwands zur Verbesserung der DML-Leistung

Der Datenbankmanager verwaltet Protokolldateien, in denen alle Datenbankänderungen aufgezeichnet werden. Es sind zwei Protokollierungsstrategien verfügbar: die Umlaufprotokollierung und die Archivprotokollierung.

- Bei der *Umlaufprotokollierung* werden Protokolldateien (beginnend mit der ersten Protokolldatei) wiederverwendet, wenn die verfügbaren Dateien voll sind. Die überschriebenen Protokollsätze sind nicht wiederherstellbar.
- Bei der *Archivprotokollierung* werden Protokolldateien archiviert, wenn sie mit Protokollsätzen gefüllt sind. Die Protokollspeicherung ermöglicht die aktualisierende Recovery (Rollforward), bei der Änderungen an der Datenbank (abgeschlossene UOWs oder Transaktionen), die in den Protokolldateien aufgezeichnet sind, bei einer Wiederherstellung erneut angewendet werden können.

Alle Änderungen an regulären Daten- und Indexseiten werden vom Protokollprozess in den Protokollpuffer geschrieben, bevor sie auf den Plattenspeicher geschrieben werden. Die Verarbeitung von SQL-Anweisungen muss in folgenden Situationen darauf warten, dass Protokolldaten auf die Platte geschrieben werden:

- Bei einer Commitoperation.
- Bis die entsprechenden Datenseiten auf Platte geschrieben wurden, weil der DB2-Server mit einer Vorabschreibprotokollierung (write ahead) arbeitet, bei der nicht alle geänderten Daten- und Indexseiten auf Platte geschrieben werden müssen, wenn eine Transaktion mit einer COMMIT-Anweisung abgeschlossen wird.
- Bis Änderungen (zumeist infolge der Ausführung von DDL-Anweisungen (Datendefinitionssprache)) an den Metadaten ausgeführt wurden.
- Wenn der Protokollpuffer voll ist.

Der Datenbankmanager schreibt Protokolldaten auf diese Weise auf Platte, um Verarbeitungsverzögerungen zu minimieren. Wenn viele kurze Transaktionen gleichzeitig verarbeitet werden, ist der größte Teil der Verzögerung auf COMMIT-Anweisungen zurückzuführen, die darauf warten müssen, dass Protokolldaten auf Platte geschrieben werden. Infolgedessen schreibt der Protokollprozess häufig kleine Mengen von Protokolldaten auf Platte. Zusätzliche Verzögerungen werden durch Protokoll-E/A-Operationen verursacht. Um die Anwendungsantwortzeit gegenüber Protokollierungsverzögerungen auszugleichen, setzen Sie den Datenbankkonfigurationsparameter **mincommit** auf einen Wert größer 1. Diese Einstellung kann zwar längere Verzögerungen für Commitoperationen für einige Anwendungen verursachen, jedoch können möglicherweise 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. Inline-LOB-Werte werden wie Werte des Typs

VARCHAR behandelt und somit bei der Protokollierung von Aktualisierungen, Einfügungen oder Löschungen vollständig berücksichtigt.

Inline-LOB-Daten zur Leistungsverbesserung

Einige Anwendungen sind durch eine extensive Nutzung großer Objekte (LOB-Daten) gekennzeichnet. In vielen Fällen sind diese großen Objekte nicht enorm, sondern nur einige KB groß. Die Leistung des LOB-Datenzugriffs kann jetzt verbessert werden, indem solche LOB-Daten innerhalb der formatierten Zeilen von Datenseiten und nicht im LOB-Speicherobjekt gespeichert werden.

Solche integrierten LOB-Daten werden als *Inline-LOB-Daten* bezeichnet. Früher konnte die Verarbeitung solcher LOB-Daten Engpässe für Anwendungen zur Folge haben. Inline-LOB-Daten verbessern die Leistung von Abfragen, die auf LOB-Daten zugreifen, weil keine zusätzlichen Ein-/Ausgabeoperationen zum Abrufen, Einfügen oder Aktualisieren dieser Daten erforderlich sind. Darüber hinaus kommen Inline-LOB-Daten für die Zeilenkomprimierung infrage.

Diese Funktion wird durch die Option INLINE LENGTH in der Anweisung CREATE TABLE oder ALTER TABLE aktiviert. Die Option INLINE LENGTH gilt für strukturierte Typen, für den Typ XML oder für LOB-Spalten. Im Fall einer LOB-Spalte gibt die Inline-Länge die maximale Bytegröße eines LOB-Werts (einschließlich vier Byte Systemaufwand) an, der in einer Basistabellenzeile gespeichert werden kann.

Diese Funktion ist implizit für alle LOB-Spalten in neuen oder vorhandenen Tabellen aktiviert (wenn LOB-Spalten hinzugefügt werden) sowie für alle vorhandenen LOB-Spalten bei einem Datenbankupgrade. Jede einzelne LOB-Spalte verfügt über reservierten Speicherbereich für Zeilen, der sich nach der definierten Maximalgröße richtet. Der implizite Wert INLINE LENGTH für die einzelnen LOB-Spalten wird automatisch definiert und gespeichert, als wäre er explizit angegeben worden.

LOB-Werte, die nicht integriert (inline) gespeichert werden können, werden separat im LOB-Speicherobjekt gespeichert.

Dabei ist zu beachten, dass bei einer Tabelle, die Spalten mit Inline-LOB-Daten enthält, weniger Zeilen auf eine Seite passen und die Leistung von Abfragen, die nur Daten zurückgeben, bei denen es sich nicht um LOB-Daten handelt, beeinträchtigt werden kann. LOB-Inlining ist bei Workloads sinnvoll, bei denen die Mehrzahl der Anweisungen LOB-Spalten beinhalten.

LOB-Daten werden zwar nicht unbedingt aufgezeichnet, bei Inline-LOB-Daten erfolgt jedoch immer eine Aufzeichnung. Inline-LOB-Daten können somit zu einem zusätzlichen Protokollierungsaufwand führen.

Kapitel 4. Entwickeln einer Strategie zur Leistungsoptimierung

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 erweisen. 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 Clusterindizes
- Neue MQTs
- Umwandlung in MDC-Tabellen (mit mehrdimensionalem Clustering)
- Umverteilen von Tabellen

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

Mit dem Befehl **db2advis** können Sie den Designadvisor starten.

Der Designadvisor kann die folgenden Aufgaben vereinfachen:

Planen und Einrichten einer neuen Datenbank
Beim Entwurf der Datenbank können Sie den Designadvisor verwenden, um Entwurfsalternativen in einer Testumgebung für die Indexierung, MQTs, MDC-Tabellen oder die Datenbankpartitionierung zu erstellen.

In Umgebungen mit partitionierten Datenbanken können Sie den Designadvisor für folgende Aufgaben verwenden:

- Ermitteln einer geeigneten Strategie für eine Datenbankpartitionierung vor dem Laden von Daten in eine Datenbank.
- Unterstützung beim Upgrade von einer Einzelpartitionsdatenbank auf eine Mehrpartitionsdatenbank.
- Unterstützung bei der Migration von einem anderen Datenbankprodukt auf eine DB2-Mehrpartitionsdatenbank.

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 Datenbankleistung 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 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

IBM InfoSphere Optim Query Workload Tuner stellt Tools für die Leistungsverbesserung einzelner SQL-Anweisungen und die Leistung von Gruppen von SQL-Anweisungen bereit, die als Abfrageworkloads bezeichnet werden. Weitere Informationen zu diesem Produkt finden Sie auf der Seite mit der Produktübersicht unter http://www.ibm.com/software/data/optim/query-workload-tuner-db2-luw/index.html. In Version 3.1.1 oder späteren Versionen des Produkts können Sie auch den Designadvisor für Workloads verwenden, um zahlreiche Operationen auszuführen, die im Assistenten für den DB2-Designadvisor verfügbar waren. Weitere Informationen finden Sie in der Dokumentation des Designadvisors für Workloads unter http://publib.boulder.ibm.com/infocenter/dstudio/v3r1/topic/com.ibm.datatools.qrytune.workloadtunedb2luw.doc/topics/genrecsdsgn.html.

Ausgabe des Designadvisors

Die Ausgabe des Designadvisors wird standardmäßig in die Standardausgabe geschrieben und in den ADVISE_*-Tabellen gespeichert:

- Die Tabelle ADVISE_INSTANCE wird mit einer neuen Zeile pro Ausführung des Designadvisors aktualisiert:
  - Die Felder START_TIME und END_TIME zeigen die Start- und Stoppzeiten des Dienstprogramms.
  - Das Feld STATUS enthält den Wert COMPLETED, wenn das Dienstgramm erfolgreich beendet wurde.
  - Das Feld MODE gibt an, ob die Option '-m' im Befehl **db2advis** verwendet wurde.
  - Das Feld COMPRESSION gibt den Typ der verwendeten Komprimierung an.
- Die Spalte USE_TABLE in der Tabelle ADVISE_TABLE enthält den Wert Y, wenn Empfehlungen zu MQTs, MDC-Tabellen oder Strategien für die Datenbankpartitionierung gegeben wurden.

  MQT-Empfehlungen sind in der Tabelle ADVISE_MQT, MDC-Empfehlungen in der Tabelle ADVISE_TABLE und Empfehlungen zu Strategien für die Datenbankpartitionierung in der Tabelle ADVISE_PARTITION zu finden. Die Spalte RUN_ID in diesen Tabellen enthält einen Wert, der dem Wert der Spalte START_TIME einer Zeile in der Tabelle ADVISE_INSTANCE entspricht, sodass diese Zeile der jeweiligen Ausführung des Designadvisors zugeordnet wird.

  Wenn MQT-, MDC- oder Datenbankpartitionierungsempfehlungen bereitgestellt werden, wird der relevante ALTER TABLE-Aufruf für die entsprechende gespeicherte Prozedur in die Spalte ALTER_COMMAND der Tabelle ADVISE_TABLE eingefügt. Der ALTER TABLE-Aufruf der gespeicherten Prozedur wird möglicherweise aufgrund von Einschränkungen in Bezug auf die Tabelle für die gespeicherte Prozedur ALTOBJ nicht erfolgreich ausgeführt.
- Die Spalte USE_INDEX in der Tabelle ADVISE_INDEX enthält den Wert Y (Index empfohlen oder ausgewertet) oder R (ein vorhandener Cluster-Satz-ID-Index wurde zur Aufhebung des Clusterings empfohlen), wenn Indexempfehlungen gegeben wurden.
- Die Spalte COLSTATS in der Tabelle ADVISE_MQT enthält Spaltenstatistiken für eine MQT. Diese Statistiken sind in einer XML-Struktur wie der folgenden enthalten:

```
<?xml version=\"1.0\" encoding=\"USASCII\"?>
<colstats>
    <column>
         <name>COLNAME1</name>
         <colcard>1000</colcard>
         <high2key>999</high2key>
         <low2key>2</low2key>
    </column>
    ....

    <column>
         <name>COLNAME100</name>
         <colcard>55000</colcard>
         <high2key>49999</high2key>
         <low2key>100</low2key>
    </column>
</colstats>
```

Sie können Empfehlungen des Designadvisors mithilfe der Option '-o' im Befehl **db2advis** in einer Datei speichern. Die gespeicherte Ausgabe des Designadvisors besteht aus den folgenden Elementen:

- CREATE-Anweisungen für empfohlene neue Indizes MQTs, MDC-Tabellen oder Datenbankpartitionierungsstrategien
- REFRESH-Anweisungen für MQTs
- **RUNSTATS**-Befehle für neue Objekte

Ein Beispiel dieser Ausgabe sieht folgendermaßen aus:

```
--<?xml version="1.0"?>
--<design-advisor>
--<mqt>
--<identifier>
--<name>MQT612152202220000</name>
--<schema>ZILIO2 </schema>
--</identifier>
--<statementlist>3</statementlist>
--<benefit>1013562.481682</benefit>
--<overhead>1468328.200000</overhead>
--<diskspace>0.004906</diskspace>
--</mqt>
.....
--<index>
--<identifier>
--<name>IDX612152221400000</name>
--<schema>ZILIO2 </schema>
--</identifier>
--<table><identifier>
--<name>PART</name>
--<schema>TPCD </schema>
--</identifier></table>
--<statementlist>22</statementlist>
--<benefit>820160.000000</benefit>
--<overhead>0.000000</overhead>
--<diskspace>9.063500</diskspace>
--</index>
.....
--<statement>
--<statementnum>11</statementnum>
--<statementtext>
--
-- select
-- c_name, c_custkey, o_orderkey, o_orderdate, o_totalprice,
-- sum(l_quantity) from tpcd.customer, tpcd.orders,
-- tpcd.lineitem where o_orderkey in( select
-- l_orderkey from tpcd.lineitem group by l_orderkey
```

```
-- having sum(l_quantity) > 300 ) and c_custkey
-- = o_custkey and o_orderkey = l_orderkey group by
-- c_name, c_custkey, o_orderkey, o_orderdate, o_totalprice
-- order by o_totalprice desc, o_orderdate fetch first
-- 100 rows only
--</statementtext>
--<objects>
--<identifier>
--<name>MQT612152202490000</name>
--<schema>ZILIO2 </schema>
--</identifier>
--<identifier>
--<name>ORDERS</name>
--<schema>TPCD </schema>
--</identifier>
--<identifier>
--<name>CUSTOMER</name>
--<schema>TPCD </schema>
--</identifier>
--<identifier>
--<name>IDX612152235020000</name>
--<schema>ZILIO2 </schema>
--</identifier>
--<identifier>
--<name>IDX612152235030000</name>
--<schema>ZILIO2 </schema>
--</identifier>
--<identifier>
--<name>IDX612152211360000</name>
--<schema>ZILIO2 </schema>
--</identifier>
--</objects>
--<benefit>2091459.000000</benefit>
--<frequency>1</frequency>
--</statement>
```

Diese XML-Struktur kann mehr als eine Spalte enthalten. Die Spaltenkardinalität (d. h. die Anzahl der Werte in jeder Spalte) sowie optional die Werte für HIGH2KEY und LOW2KEY werden mit eingeschlossen.

Die Basistabelle, für die ein Index definiert wird, wird ebenfalls mit eingeschlossen. Eine Rangordnung für von Indizes und MQTs kann anhand des Werts 'benefit' bestimmt werden. Sie können die Rangordnung für Indizes auch durch (benefit - overhead) und für MQTs durch (benefit - 0,5 * overhead) ermitteln.

Nach der Liste von Indizes und MQTs folgt die Liste von Anweisungen in der Auslastung, die den SQL-Text, die Anweisungsnummer der Anweisung, die geschätzte Leistungsverbesserung (benefit) aus den Empfehlungen sowie die Liste von Tabellen, Indizes und MQTs enthält, die von der jeweiligen Anweisung verwendet wurden. Die ursprünglichen Abstände im SQL-Text wurden in diesem Ausgabebeispiel beibehalten, jedoch wird der SQL-Text normalerweise aus Gründen der besseren Lesbarkeit in Kommentarzeilen von 80 Zeichen aufgeteilt.

Vorhandene Indizes oder MQTs werden in der Ausgabe angegeben, wenn sie zur Ausführung einer Auslastung verwendet werden.

Empfehlungen zu MDC-Tabellen und zur Datenbankpartitionierung werden in diesem XML-Ausgabebeispiel nicht explizit angezeigt.

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**-Befehle zu einem einzigen **RUNSTATS**-Aufruf für die neuen oder modifizierten Objekte
- Angeben geeigneterer Objektnamen anstelle der vom System generierten IDs
- Entfernen aller DDL-Anweisungen (Data Definition Language, Datendefinitionssprache) für Objekte, die nicht sofort implementiert werden sollen, durch Löschen oder Verwenden von Kommentarzeichen

Verwenden des Designadvisors

Der Designadvisor kann durch Aufrufen des Befehls **db2advis** ausgeführt werden.

Informationen zu diesem Vorgang

Vorgehensweise

1. Definieren Sie Ihre Auslastung. Siehe „Definieren einer Auslastung für den Designadvisor".
2. Führen Sie den Befehl **db2advis** für diese 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 **db2advis** und nehmen Sie alle erforderlichen Modifikationen vor.
4. Implementieren Sie die Empfehlungen des Designadvisors je nach Bedarf.

Definieren einer Auslastung für den Designadvisor

Wenn der Designadvisor eine bestimmte Auslastung analysiert, zieht er Faktoren wie den Typ der Anweisungen, die in der Auslastung enthalten sind, die Häufigkeit, mit der eine bestimmte Anweisung auftritt, sowie Merkmale der Datenbank in Betracht, um Empfehlungen zu generieren, die den Gesamtaufwand zur Ausführung der Auslastung minimieren.

Informationen zu diesem Vorgang

Eine *Auslastung* ist eine Gruppe von SQL-Anweisungen, die der Datenbankmanager in einem bestimmten Zeitraum verarbeiten muss. Der Designadvisor kann für folgende Arten von Anweisungen ausgeführt werden:

- Eine einzelne SQL-Anweisung, die Sie in der Befehlszeile mit dem Befehl **db2advis** angeben
- Eine Gruppe dynamischer SQL-Anweisungen, die in einer DB2-Momentaufnahme erfasst wurden
- Eine Gruppe von SQL-Anweisungen, die in einer Auslastungsdatei enthalten sind

Sie können eine neue Auslastungsdatei erstellen oder eine bereits vorhandene Auslastungsdatei ändern. Sie können Anweisungen in die Datei aus verschiedenen Quellen importieren, wie zum Beispiel:

- Eine Textdatei mit Trennzeichen
- Eine Ereignismonitortabelle
- Query Patroller-Protokolldatentabellen durch Angabe der Option -qp über die Befehlszeile
- Mit EXPLAIN bearbeitete Anweisungen in der Tabelle EXPLAIN_STATEMENT

- Kürzlich ausgeführte SQL-Anweisungen, die mit einer DB2-Momentaufnahme erfasst wurden
- Workload-Manager-Aktivitätstabellen
- Workload-Manager-Ereignismonitortabellen durch Angabe der Option -wlm über die Befehlszeile

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

Vorgehensweise

- 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 ab, sodass dynamische SQL-Anweisungen für die Datenbank ausgeführt werden können.
  3. Rufen Sie den Befehl **db2advis** 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 in einer Auslastungsdatei auszuführen:
  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 einer 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 könnten Sie der ersten SELECT-Anweisung eine Häufigkeit von 10 zuordnen. Die zweite SELECT-Anweisung hat eine Häufigkeit von 1. 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. Rufen Sie den Befehl **db2advis** mit der Option -i gefolgt vom Namen der Auslastungsdatei auf.
- Zur Ausführung des Designadvisors für eine Auslastung, die in der Tabelle ADVISE_WORKLOAD enthalten ist, rufen Sie den Befehl **db2advis** mit der Option -w gefolgt vom Namen der Auslastung ein.

Verwenden des Designadvisors für die Konvertierung von einer Einzelpartitions- in eine Mehrpartitionsdatenbank

Sie können den Designadvisor zur Unterstützung der Konvertierung einer Datenbank mit einer Einzelpartition in eine Datenbank mit mehreren Partitionen verwenden.

Informationen zu diesem Vorgang

Neben der Ermittlung von Empfehlungen zu neuen Indizes, MQTs (Materialized Query Tables) und Tabellen mit mehrdimensionalem Clustering (MDC-Tabellen) kann der Designadvisor auch Empfehlungen zur Verteilung von Daten ausgeben.

Vorgehensweise

1. Mit dem Befehl **db2licm** können Sie den DPF-Lizenzschlüssel (DPF = Database Partitioning Feature, Datenbankpartitionierungsfunktion) registrieren.
2. Erstellen Sie mindestens einen Tabellenbereich in einer Partitionsgruppe einer Datenbank mit mehreren Partitionen.

   Anmerkung: Der Designadvisor kann eine Datenumverteilung nur auf vorhandene Tabellenbereiche empfehlen.
3. Führen Sie den Designadvisor unter Angabe der Partitionierungsoption im Befehl **db2advis** aus.
4. Nehmen Sie an der Ausgabedatei des Befehls **db2advis** geringfügige Änderungen vor, bevor Sie die DDL-Anweisungen (DDL - Data Definition Language, Datendefinitionssprache) ausführen, die vom Designadvisor generiert wurden. Da die Datenbankpartitionierung konfiguriert werden muss, bevor Sie das DDL-Script ausführen können, das vom Designadvisor generiert wird, sind die Empfehlungen in dem Script, das zurückgegeben wird, durch Kommentarzeichen inaktiviert. Es bleibt Ihnen überlassen, die Tabellen den Empfehlungen entsprechend umzuwandeln.

Begrenzungen und Einschränkungen des Designadvisors

Bei den Empfehlungen des Designadvisors zu Indizes, MQTs (Materialized Query Tables), Tabellen mit mehrdimensionalem Clustering (MDC-Tabellen) sowie zur Datenbankpartitionierung sind bestimmte Begrenzungen und Einschränkungen zu beachten.

Einschränkungen bei Indexempfehlungen

- Indizes, die für MQTs empfohlen werden, dienen zur Verbesserung der Auslastungsleistung und nicht der REFRESH-Leistung für MQTs.
- Ein RID-Clusterindex wird für MDC-Tabellen nur empfohlen. Der Designadvisor berücksichtigt RID-Clusterindizes nur als Option und erstellt keine MDC-Struktur für die Tabelle.
- Design Advisor Version 9.7 empfiehlt keine partitionierten Indizes für partitionierte Tabellen. Es wird für alle Indizes die explizite Klausel NOT PARTITIONED empfohlen.

Einschränkungen bei MQT-Empfehlungen

- Der Designadvisor empfiehlt keine inkrementellen MQTs. Wenn Sie inkrementelle MQTs erstellen wollen, können Sie mit REFRESH IMMEDIATE definierte MQTs in inkrementelle MQTs mit einer eigenen Auswahl von 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 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.

Einschränkungen bei MDC-Empfehlungen

- Eine vorhandene Tabelle muss mit ausreichend Daten gefüllt sein, bevor der Designadvisor ein mehrdimensionales Clustering (MDC) für die Tabelle in Betracht zieht. Ein Minimum von 20 bis 30 MB Daten wird empfohlen. Für Tabellen, die kleiner als 12 EXTENTSIZE große Speicherbereiche sind, wird MDC nicht berücksichtigt.
- MDC-Empfehlungen für neue MQTs werden nicht in Betracht gezogen, sofern nicht die Stichprobenoption '-r' im Befehl **db2advis** verwendet wird.
- Der Designadvisor gibt keine MDC-Empfehlungen für typisierte, temporäre oder föderierte Tabellen.
- Es muss ausreichend Speicherplatz (ca. 1 % der Tabellendaten bei großen Tabellen) für die Stichprobendaten verfügbar sein, die bei der Ausführung des Befehls **db2advis** verwendet werden.
- Tabellen, für die keine Statistiken erfasst wurden, werden aus der Betrachtung ausgeschlossen.
- Der Designadvisor gibt keine Empfehlungen für mehrspaltige Dimensionen.

Einschränkungen bei Datenbankpartitionierungsempfehlungen

Der Designadvisor kann eine Datenbankpartitionierung nur für DB2 Enterprise Server Edition empfehlen.

Weitere Einschränkungen

Während der Ausführung des Designadvisors werden temporäre Simulationskatalogtabellen erstellt. Eine unvollständige Ausführung 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 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 nur die DROP-Anweisungen, die zum Entfernen der Tabellen benötigt werden, ohne jedoch die Tabellen zu entfernen.

Anmerkung: In Version 9.5 ist die Option -f die Standardoption. Das bedeutet, dass der Datenbankmanager bei Ausführung von **db2advis** mit der MQT-Auswahl automatisch alle lokalen Simulationskatalogtabellen mit derselben Benutzer-ID als Schemanamen löscht.

Sie sollten einen getrennten Tabellenbereich in der Katalogdatenbankpartition zum Speichern dieser simulierten Katalogtabellen erstellen und die Option DROPPED TABLE RECOVERY in der Anweisung CREATE oder ALTER TABLESPACE auf den Wert OFF setzen. Dies ermöglicht eine einfachere Bereinigung und eine schnellere Ausführung des Designadvisors.

Teil 2. Fehlerbehebung

Der erste Schritt einer jeden guten Fehleranalyse ist die vollständige Beschreibung des vorliegenden Fehlers. Ohne eine Fehlerbeschreibung werden Sie nicht wissen, wo Sie mit der Suche nach der Fehlerursache beginnen sollen.

Im Rahmen der Fehlerbeschreibung sollten Sie sich unter anderem die folgenden grundlegenden Fragen stellen:

- Welche Symptome sind vorhanden?
- Wo tritt der Fehler auf?
- Wann tritt der Fehler auf?
- Unter welchen Umständen tritt der Fehler auf?
- Ist der Fehler reproduzierbar?

Durch Beantwortung dieser und anderer Fragen lassen sich die meisten Fehler gut beschreiben, und dies ist der beste erste Schritt auf dem Weg zur Fehlerbeseitigung.

Welche Symptome sind vorhanden?

Zu Beginn einer Fehlerbeschreibung ist die offensichtlichste Frage "Welche Symptome sind vorhanden?". Diese Frage mag auf den ersten Blick recht einfach erscheinen, kann jedoch in eine Reihe weiterer Fragen unterteilt werden, um ein Bild zu erhalten, das das Problem genauer beschreibt. Hierbei kann es sich unter anderem um folgende Fragen handeln:

- Wer oder was meldet den Fehler?
- Wie lauten die Fehlercodes und Fehlernachrichten?
- Wie äußert sich der Fehler? Beispiel: Schleife, Blockierung, Stopp, Leistungseinbußen, falsche Ergebnisse.
- Wie wirkt sich der Fehler auf die Geschäftsabläufe aus?

Wo tritt der Fehler auf?

Es ist nicht immer einfach zu ermitteln, woher der Fehler ursprünglich stammt. Dennoch ist dies einer der wichtigsten Schritte bei der Fehlerbehebung. Zwischen der Komponente, die den Fehler meldet, und der Komponente, die fehlschlägt, können viele Technologieschichten liegen. Netze, Platten und Treiber sind nur einige der Komponenten, die bei der Untersuchung des Problems zu berücksichtigen sind.

- Ist der Fehler plattformspezifisch oder tritt er auf mehreren Plattformen auf?
- Besteht Unterstützung für die aktuelle Umgebung und Konfiguration?
- Wird die Anwendung auf dem Datenbankserver lokal ausgeführt oder auf einem fernen Server?
- Ist ein Gateway beteiligt?
- Ist die Datenbank auf einzelnen Platten oder auf einem RAID-Plattenstapel gespeichert?

Fragen wie diese helfen Ihnen dabei, die Ebene des Problems einzugrenzen, und sind erforderlich, um den Ursprung des Fehlers zu ermitteln. Bedenken Sie, dass der Fehler seine Wurzeln nicht immer auf der Ebene hat, die den Fehler meldet.

Um ermitteln zu können, wo ein Fehler auftritt, muss man unter anderem die Umgebung kennen, in der das Problem liegt. Sie sollten immer ein wenig Zeit darauf verwenden, die Umgebung des Fehlers vollständig zu beschreiben, einschließlich des Betriebssystems und seiner Version, der gesamten zugehörigen Software und entsprechender Versionen sowie der Hardware. Vergewissern Sie sich, dass Sie in einer Umgebung arbeiten, bei der es sich um eine unterstützte Konfiguration handelt, da sich viele Fehler dadurch erklären lassen, dass Softwareversionen vorliegen, die nicht für die gemeinsame Ausführung gedacht sind oder nicht ausreichend zusammen getestet wurden.

Wann tritt der Fehler auf?

Ein weiterer erforderlicher Schritt bei der Fehleranalyse besteht darin, den detaillierten Zeitablauf der Ereignisse zu ermitteln, die zu dem Fehler geführt haben. Dies gilt insbesondere für Fehler, die nur ein einziges Mal auftreten. Am einfachsten lässt sich dies bewerkstelligen, wenn Sie rückwärts arbeiten. Beginnen Sie hierbei mit dem Zeitpunkt, zu dem der Fehler auftrat (und zwar so präzise wie möglich, selbst wenn es sich um Millisekunden handelt), und arbeiten Sie sich dann rückwärts durch die verfügbaren Protokolle und Informationen. Normalerweise braucht man lediglich das erste verdächtige Ereignis in den Protokollen der Diagnoseprogramme zu finden. Allerdings ist dies nicht immer einfach und setzt einige praktische Erfahrung voraus. Es ist insbesondere dann schwierig zu wissen, wie weit man die Protokolle rückverfolgen soll, wenn mehrere Technologieschichten vorliegen, die jeweils eigene Diagnoseinformationen bereitstellen.

- Tritt der Fehler nur zu einer bestimmten Tages- oder Nachtzeit auf?
- Wie häufig tritt der Fehler auf?
- Welche Folge von Ereignissen führt zu dem Zeitpunkt, zu dem der Fehler gemeldet wird?
- Tritt der Fehler nach einer Änderung an der Umgebung auf, beispielsweise nach einem Upgrade bereits vorhandener Software- oder Hardwarekomponenten bzw. nach der Installation neuer Software- oder Hardwarekomponenten?

Die Beantwortung solcher Fragen hilft Ihnen dabei, den zeitlichen Ablauf der Ereignisse genau zu bestimmen und so einen Bezugsrahmen für die Analyse zu schaffen.

Unter welchen Umständen tritt der Fehler auf?

Eine vollständige Fehlerbeschreibung setzt voraus, dass man weiß, welche anderen Komponenten zum Zeitpunkt des Fehlers sonst noch ausgeführt werden. Wenn ein Problem in einer bestimmten Umgebung oder unter bestimmten Umständen auftritt, kann dies ein wesentlicher Hinweis auf die Fehlerursache sein.

- Tritt der Fehler immer bei der Ausführung derselben Task auf?
- Ist eine bestimmte Folge von Ereignissen erforderlich, damit das Problem auftritt?
- Schlagen gleichzeitig noch andere Anwendungen fehl?

Die Beantwortung dieser Art von Fragen hilft Ihnen dabei, die Umgebung zu beschreiben, in der das Problem auftritt, und eventuelle Zusammenhänge und Abhängigkeiten zu erkennen. Bedenken Sie, dass verschiedene Probleme nicht unbedingt miteinander zusammenhängen müssen, nur weil sie zu derselben Zeit auftreten.

Ist der Fehler reproduzierbar?

Für die Fehlerbeschreibung und -analyse ist es "ideal", wenn sich ein Fehler reproduzieren lässt. Bei reproduzierbaren Fehlern steht fast immer eine größere Anzahl an Tools oder Prozeduren zur Verfügung, die bei der Untersuchung helfen können. Daher sind reproduzierbare Fehler normalerweise leichter zu beheben.

Allerdings können reproduzierbare Fehler auch einen Nachteil haben: Wenn das Problem signifikante Auswirkungen auf die Geschäftsabläufe hat, will man eine Wiederholung des Problems natürlich vermeiden. In diesem Fall ist die Reproduktion des Fehlers in einer Test- oder Entwicklungsumgebung häufig vorzuziehen.

- Kann der Fehler auf einer Testmaschine reproduziert werden?
- Haben mehrere Benutzer oder Anwendungen die gleiche Art von Problem?
- Lässt sich der Fehler reproduzieren, indem ein einzelner Befehl, eine Befehlsgruppe oder eine bestimmte Anwendung bzw. eine eigenständige Anwendung ausgeführt wird?
- Kann der Fehler reproduziert werden, indem der/die äquivalente Befehl/Abfrage über eine DB2-Befehlszeile ausgegeben wird?

Die Reproduktion eines einmalig auftretenden Fehlers in einer Test- oder Entwicklungsumgebung ist häufig vorzuziehen, da sich die Untersuchung in einer solchen Umgebung normalerweise flexibler gestalten und besser steuern lässt.

Kapitel 5. Tools für die Fehlerbehebung

Mit den folgenden Tools können Sie Diagnosedaten erfassen, formatieren und analysieren.

- db2dart

  Mit dem Befehl **db2dart** kann überprüft werden, ob die Architektur von Datenbanken und den in ihnen enthaltenen Objekten einwandfrei ist. Mit diesem Tool kann auch der Inhalt der Datenbanksteuerdateien angezeigt werden, um Daten aus Tabellen zu extrahieren, auf die ansonsten möglicherweise kein Zugriff besteht.

- db2diag

  Das Tool **db2diag** dient zum Filtern und Formatieren der in den **db2diag**-Protokolldateien bereitgestellten Informationsmenge. Das Filtern von **db2diag**-Protokolldateien kann die erforderliche Zeit für die Suche nach den benötigten Datensätzen bei der Fehlerbehebung reduzieren.

- db2greg

  Die globale Registrierdatenbank kann mit dem Tool **db2greg** angezeigt und bearbeitet werden.

- db2level

  Mit dem Befehl **db2level** können Sie die Version und die Servicestufe (Buildstufe und Fixpacknummer) der verwendeten DB2-Instanz ermitteln.

- db2look

  In vielen Fällen ist es von Vorteil, wenn eine Datenbank erstellt werden kann, die die gleiche Struktur wie eine andere Datenbank aufweist. Anstatt neue Anwendungen oder Recoverypläne auf einem Produktionssystem zu testen, ist es beispielsweise sinnvoller, ein Testsystem mit der gleichen Struktur und gleichen Daten zu erstellen und die Tests dort ohne eine Beeinträchtigung des Produktionssystems auszuführen. Mithilfe des Tools **db2look** können Sie die entsprechenden DDL-Anweisungen extrahieren, die erforderlich sind, um die Datenbankobjekte einer Datenbank in einer anderen Datenbank zu reproduzieren. Mit diesem Tool können auch die entsprechenden SQL-Anweisungen generiert werden, die erforderlich sind, um die Statistikdaten aus der einen Datenbank in der anderen zu replizieren, sowie die Anweisungen, die benötigt werden, um die Datenbankkonfiguration, die Datenbankmanagerkonfiguration und die Registrierdatenbankvariablen zu replizieren.

- db2ls

  Da Sie mehrere Kopien von DB2-Produkten auf dem System installieren können und für die Installation dieser DB2-Produkte und -Features einen Pfad Ihrer Wahl festlegen können, benötigen Sie ein Tool, das protokolliert, was an welcher Position installiert ist. Unter den unterstützten Linux- und UNIX-Betriebssystemen können Sie mit dem Befehl **db2ls** die DB2-Produkte und -Features auflisten, die auf Ihrem System installiert sind, einschließlich der HTML-Dokumentation zu DB2 Version 9.

- db2pd

  Das Tool **db2pd** wird für die Fehlerbehebung verwendet, da es schnell und unmittelbar Informationen aus den DB2-Speichersätzen bereitstellen kann.

- db2support

  Das wichtigste DB2-Dienstprogramm, das Sie bei der Erfassung von Informationen zu einem DB2-Problem bzw. -Fehler ausführen müssen, ist **db2support**. Das

Dienstprogramm **db2support** dient zur automatischen Erfassung aller verfügbaren DB2- und Systemdiagnoseinformationen. Darüber hinaus bietet es eine optionale interaktive Frage-und-Antwort-Sitzung, in der Fragen zu den jeweiligen Umständen des aufgetretenen Problems gestellt werden.

- db2val

  Das Tool **db2val** verifiziert die Kernfunktion einer DB2-Kopie, indem es Installationsdateien, Instanzen, die Erstellung einer Datenbank, Verbindungen zu dieser Datenbank und den Status von Umgebungen mit partitionierten Datenbanken auf Gültigkeit überprüft.

- Traces

  Wenn bei DB2 ein wiederholt auftretendes und reproduzierbares Problem vorkommt, können mit einem Trace bisweilen zusätzliche Informationen zu diesem Problem erfasst werden. Unter normalen Umständen sollten Sie einen Trace nur dann verwenden, wenn Sie von IBM Software Support dazu aufgefordert werden. Der Prozess der Durchführung eines Trace umfasst das Einstellen der Tracefunktion, das Reproduzieren des Fehlers und das Sammeln der Daten.

- Plattformspezifische Tools (Windows) (Linux und UNIX)

  Mit den von den Betriebssystemen Windows, Linux und UNIX bereitgestellten nützlichen Diagnosetools können Daten erfasst und verarbeitet werden, die das Ermitteln der Ursache für ein auf Ihrem System festgestelltes Problem erleichtern.

Prüfen von Archivprotokolldateien mit dem Tool 'db2cklog'

Durch das Prüfen der Archivprotokolldateien wird sichergestellt, dass Protokolldateien, deren fehlerfreier Status verifiziert wurde, verfügbar sind, falls eine aktualisierende Recovery erforderlich wird, und dass die Recoveryoperation nicht aufgrund eines Protokolldateifehlers fehlschlägt. Die Informationen in diesem Abschnitt beschreiben, wie Sie Protokolldateien mithilfe des Tools 'db2cklog' prüfen und wie Sie vorgehen, wenn eine Protokolldatei die Prüfung nicht besteht.

Vorbereitende Schritte

Sie benötigen Leseberechtigung für die Archivprotokolldateien, damit das Tool 'db2cklog' die Protokolldateien lesen und die entsprechenden Prüfungen durchführen kann. Nur geschlossene Protokolldateien, wie zum Beispiel Archivprotokolldateien, können erfolgreich geprüft werden. Wenn Sie das Tool für eine aktive Protokolldatei ausführen, kann es diese Datei nicht präzise prüfen; Sie erhalten eine Warnung, dass diese Datei noch aktiv ist.

Informationen zu diesem Vorgang

Das Tool 'db2cklog' liest einzelne Protokolldateien oder einen Bereich mit bestimmten Protokolldateinummern und prüft die interne Gültigkeit der Dateien. Protokolldateien, die die Prüfung ohne Fehlernachrichten oder Warnungen bestehen, sind Dateien mit verifiziertem fehlerfreiem Status, die Sie bei einer aktualisierende Recovery verwenden können. Wenn die Prüfung einer Archivprotokolldatei mit einer Fehlernachricht oder Warnung fehlschlägt, darf diese Protokolldatei nicht bei einer aktualisierenden Recovery verwendet werden. Eine Archivprotokolldatei, deren Prüfung fehlschlägt, kann nicht korrigiert werden. Die in dieser Task beschriebene Benutzeraktion enthält eine Erläuterung der in diesem Fall auszuführenden Schritte.

Eine Prüfung der Archivprotokolldateien ist in den folgenden Szenarios nützlich:

- Unmittelbar vor dem Starten einer aktualisierenden Recovery: Wenn eine aktualisierende Recovery erforderlich wird, können Sie zuvor das Tool 'db2cklog' für alle Archivprotokolldateien ausführen, die zur Durchführung der aktualisierende Recovery benötigt werden, um sicherzustellen, dass die Protokolldateien gültig sind. Indem Sie das Tool 'db2cklog' vorab für die Protokolldateien ausführen, können Sie eine Situation vermeiden, in der die Recovery vor dem Abschluss der Ausführung aufgrund eines Protokolldateifehlers fehlschlägt und eine weitere Recovery erforderlich macht.
- Jedesmal, wenn eine Protokolldatei geschlossen und in das Protokollarchivverzeichnis kopiert wird: Eine Prüfung der Archivprotokolldateien kann im Rahmen der Routineoperationen als zusätzliche Sicherheitsmaßnahme durchgeführt werden, um sicherzustellen, dass stets Dateien mit verifiziertem fehlerfreiem Status zur Verfügung stehen. Durch diese vorbeugende Maßnahme kann sofort festgestellt werden, ob eine Kopie einer Protokolldatei gesucht werden muss oder ob ein Datenbankgesamtbackup erforderlich ist, um einen neuen Recoverypunkt zu bestimmen. Falls eine aktualisierende Recovery notwendig wird, kann so eine mögliche Verzögerung reduziert werden.

Vorgehensweise

Geben Sie zum Prüfen der Archivprotokolldateien den Befehl 'db2cklog' über die Befehlszeile ein und geben Sie die zu prüfende(n) Protokolldatei(en) an. Beachten Sie, dass im Befehl 'db2cklog' keine vollständigen Protokolldateinamen verwendet werden dürfen, sondern nur die numerischen Kennungen, die Teil der Protokolldateinamen sind. Die numerische Kennung der Protokolldatei `S0000001.LOG` ist beispielsweise 1; zur Prüfung dieser Protokolldatei geben Sie `db2cklog 1` an. Wenn sich die Archivprotokolldateien nicht im aktuellen Verzeichnis befinden, geben Sie den relativen oder absoluten Pfad für die Protokolldateien mit dem optionalen Parameter **ARCHLOGPATH** an.

1. Wenn Sie die Gültigkeit einer einzelnen Archivprotokolldatei prüfen möchten, geben Sie die numerische Kennung dieser Protokolldatei als *protokolldateinummer1* im Befehl an. Beispiel: Zur Prüfung der Gültigkeit der Protokolldatei `S0000000.LOG` im Verzeichnis `/home/amytang/tests` geben Sie den Befehl `db2cklog 0 ARCHLOGPATH /home/amytang/tests` ein.
2. Wenn Sie die Gültigkeit eines Bereichs von Archivprotokolldateien prüfen möchten, geben Sie die erste und letzte numerische Kennung dieses Bereichs in Befehl an (*protokolldateinummer1* bis *protokolldateinummer2*). Alle Protokolldateien innerhalb des Bereichs werden geprüft, es sei denn, das mit *protokolldateinummer2* angegebene obere Ende des Bereichs liegt numerisch unterhalb des mit *protokolldateinummer1* angegebenen Bereichsanfangs. In diesem Fall wird nur *protokolldateinummer1* geprüft. Beispiel: Zur Prüfung der Gültigkeit der Protokolldateien im Bereich von `S0000000.LOG` bis `S0000005.LOG` im Verzeichnis `/home/nrichers/tests` geben Sie den folgenden Befehl ein: `db2cklog 0 TO 5 ARCHLOGPATH /home/nrichers/tests`.

Ergebnisse

Das Tool 'db2cklog' gibt den Rückkehrcode null für alle Dateien zurück, die die Gültigkeitsprüfung bestehen. Bei der Angabe eines Bereichs mit nummerierten Archivprotokolldateien liest das Tool 'db2cklog' eine Datei nach der anderen, führt die Prüfung durch und gibt für jede Datei einen Rückkehrcode zurück. Das Tool wird beim ersten festgestellten Fehler gestoppt, auch wenn ein Bereich mit Protokolldateien angegeben wurde und weitere Dateien vorhanden sind, die vom Tool noch nicht geprüft wurden. Die DBT-Nachricht, die beim Feststellen eines Fehlers zurückgegeben wird, kann Ihnen zusätzliche Informationen dazu liefern, warum

die Prüfung einer Archivprotokolldatei fehlgeschlagen ist. Es ist jedoch nicht möglich, eine ungültige Protokolldatei zu korrigieren. Wenn Sie eine DBT-Warnung erhalten, dass eine Protokolldatei möglicherweise noch aktiv ist, Sie jedoch genau wissen, dass die Datei eine Archivprotokolldatei ist, behandeln Sie die Archivprotokolldatei als ungültig und führen Sie die Schritte aus, die in der Benutzeraktion in dieser Task beschrieben sind.

Beispiel

Das folgende Beispiel zeigt die typische Ausgabe des Befehls 'db2cklog' beim Parsing einer Protokolldatei, in diesem Fall der Datei S0000002.LOG. Diese Datei besteht die Gültigkeitsprüfung mit dem Rückkehrcode null.

```
$ db2cklog 2
```

```
_____     D B 2 C K L O G     _____

                DB2-Tool zum Prüfen von Protokolldateien
                     I    B    M


          Das Tool db2cklog ist ein Dienstprogramm, mit dem Sie die
        Integrität einer Archivprotokolldatei testen und ermitteln können,
                  ob die Protokolldatei im Befehl 'rollforward database'
                              verwendet werden kann.
```

```
=====================================================
"db2cklog": Protokolldateiheader von "S0000002.LOG" wird verarbeitet.

"db2cklog": Protokollseiten von "S0000002.LOG" werden verarbeitet
(Gesamtzahl Protokollseiten: "316840").
            ==> Seite "1" ...
            ==> Seite "25001" ...
            ==> Seite "50001" ...
            ==> Seite "75001" ...
            ==> Seite "100001" ...
            ==> Seite "125001" ...
            ==> Seite "150001" ...
            ==> Seite "175001" ...
            ==> Seite "200001" ...
            ==> Seite "225001" ...
            ==> Seite "250001" ...
            ==> Seite "275001" ...
            ==> Seite "300001" ...

"db2cklog": Verarbeitung der Protokolldatei "S0000002.LOG" ist abgeschlossen.
Rückkehrcode: "0".
=====================================================
```

Nächste Schritte

Wenn die Gültigkeitsprüfung einer Archivprotokolldatei fehlschlägt, ist die Benutzeraktion davon abhängig, ob eine Kopie der Protokolldatei vorhanden ist, die die Gültigkeitsprüfung durch das Tool 'db2cklog' bestehen kann. Wenn Sie nicht sicher sind, ob eine Kopie der Protokolldatei vorhanden ist, überprüfen Sie die Einstellung für den Konfigurationsparameter **logarchmeth2**, der festlegt, ob Ihr Daten-

bankserver eine sekundäre Kopie jeder Protokolldatei archiviert. Wenn Sie Protokolle bei der Archivierung prüfen und die Protokollspiegelung ebenfalls auf Ihrem Datenserver konfiguriert ist, finden Sie möglicherweise eine Kopie der Protokolldatei im Protokollspiegelpfad, da der Datenserver die Protokolldateien nicht sofort nach der Archivierung wiederverwendet.

- Wenn Sie über eine Kopie der Archivprotokolldatei verfügen, führen Sie den Befehl 'db2cklog' für diese Kopie aus. Wenn die Kopie der Protokolldatei die Gültigkeitsprüfung besteht, ersetzen Sie die Protokolldatei, die nicht gelesen werden kann, durch die gültige Kopie der Protokolldatei.
- Wenn Sie nur über eine Kopie der Archivprotokolldatei verfügen und diese Kopie die Gültigkeitsprüfung nicht besteht, kann die Protokolldatei nicht korrigiert werden und ist damit nicht für die aktualisierende Recovery verwendbar. In diesem Fall müssen Sie schnellstmöglich ein Datenbankgesamtbackup durchführen, um einen neuen, späteren Recoverypunkt zu bestimmen, bei dem die aktualisierende Recovery nicht von der unbrauchbaren Protokolldatei abhängt.

Übersicht über das Tool 'db2dart'

Mit dem Befehl **db2dart** kann überprüft werden, ob die Architektur von Datenbanken und den in ihnen enthaltenen Objekten einwandfrei ist. Mit diesem Tool kann auch der Inhalt der Datenbanksteuerdateien angezeigt werden, um Daten aus Tabellen zu extrahieren, auf die ansonsten möglicherweise kein Zugriff besteht.

Setzen Sie den Befehl **db2dart** ohne Parameter ab, um alle Optionen anzuzeigen, die möglich sind. Einige Optionen, die Parameter erfordern, wie beispielsweise die Tabellenbereichs-ID, werden angefordert, wenn sie in der Befehlszeile nicht explizit angegeben werden.

Standardmäßig erstellt das Dienstprogramm **db2dart** eine Berichtsdatei namens `databaseName.RPT`. In Einzelpartitionsdatenbankumgebungen wird die Datei im aktuellen Verzeichnis erstellt. In Datenbankumgebungen mit mehreren Partitionen wird die Datei in einem Unterverzeichnis des Diagnoseverzeichnisses erstellt. Das Unterverzeichnis hat den Namen `DART####`, wobei `####` für die Datenbankpartitionsnummer steht.

Die Daten und Metadaten in einer Datenbank werden vom Dienstprogramm **db2dart** direkt von der Platte gelesen. Aus diesem Grund sollte das Tool nie für Datenbanken ausgeführt werden, die noch über aktive Verbindungen verfügen. Sind Verbindungen vorhanden, erkennt das Tool beispielsweise keine Seiten im Pufferpool und keine Steuerstrukturen im Speicher und meldet infolge dessen möglicherweise Fehler, die gar nicht vorliegen. Wenn Sie **db2dart** für eine Datenbank ausführen, für die eine Recovery nach einem Systemabsturz erforderlich ist oder für die noch keine aktualisierende Recovery durchgeführt wurde, kann es aufgrund der inkonsistenten Daten auf der Platte ebenso zu solchen Inkonsistenzen kommen.

Vergleich zwischen INSPECT und db2dart

Der Befehl **INSPECT** überprüft eine Datenbank auf Architekturintegrität, wobei die Seiten der Datenbank auf Seitenkonsistenz überprüft werden. Mit dem Befehl **INSPECT** wird überprüft, ob die Strukturen von Tabellenobjekten und von Tabellenbereichen gültig sind. Die Prüfung über Objektgrenzen hinweg führt eine Onlineindex-zu-Datenkonsistenz-Prüfung durch. Mit dem Befehl **db2dart** wird die korrekte Architektur der Datenbank überprüft und eventuell gefundene Fehler werden berichtet.

Wie der Befehl **db2dart** ermöglicht auch der Befehl **INSPECT** das Überprüfen von Datenbanken, Tabellenbereichen und Tabellen. Ein wesentlicher Unterschied zwischen den beiden Befehlen besteht darin, dass die Datenbank vor der Ausführung von **db2dart** inaktiviert sein muss, wogegen INSPECT eine Datenbankverbindung benötigt und ausgeführt werden kann, während gleichzeitig andere Verbindungen zur Datenbank aktiv sind.

Wenn Sie die Datenbank nicht inaktivieren, liefert **db2dart** unzuverlässige Ergebnisse.

Die folgende Tabelle enthält eine Übersicht zu den Unterschieden zwischen den Tests, die über die Befehle **db2dart** und **INSPECT** ausgeführt werden können.

Tabelle 77. Funktionsvergleich zwischen db2dart und INSPECT für Tabellenbereiche

Durchgeführte Tests	db2dart	INSPECT
SMS-Tabellenbereiche		
Tabellenbereichsdateien prüfen	JA	NEIN
Inhalt von internen Seitenkopffeldern vergleichen	JA	JA
DMS-Tabellenbereiche		
Prüfen auf Speicherbereichsmasken, auf die von mehr als einem Objekt verwiesen wird	JA	NEIN
Jede Speicherbereichsmaskenseite auf Konsistenzbitfehler prüfen	NEIN	JA
Jede Speicherabbildseite auf Konsistenzbitfehler prüfen	NEIN	JA
Inhalt von internen Seitenkopffeldern vergleichen	JA	JA
Prüfen, ob die Speicherbereichsmasken den Speicherabbildern entsprechen	JA	NEIN

Tabelle 78. Funktionsvergleich zwischen db2dart und INSPECT für Datenobjekte

Durchgeführte Tests	db2dart	INSPECT
Datenobjekte auf Konsistenzbitfehler prüfen	JA	JA
Inhalt bestimmter Steuerzeilen prüfen	JA	NEIN
Länge und Position von Spalten mit variabler Länge prüfen	JA	NEIN
LONG VARCHAR-, LONG VARGRAPHIC- und LOB-Deskriptoren in Tabellenzeilen prüfen	JA	NEIN

Tabelle 78. Funktionsvergleich zwischen db2dart und INSPECT für Datenobjekte (Forts.)

Durchgeführte Tests	db2dart	INSPECT
Gesamtseitenzahl, Anzahl der belegten Seiten und Prozentsatz des freien Speicherbereichs prüfen	NEIN	JA
Inhalt von internen Seitenkopffeldern vergleichen	JA	JA
Jeden Zeilensatztyp und dessen Länge prüfen	JA	JA
Prüfen, ob sich Zeilen überschneiden	JA	JA

Tabelle 79. Funktionsvergleich zwischen db2dart und INSPECT für Indexobjekte

Durchgeführte Tests	db2dart	INSPECT
Auf Konsistenzbitfehler prüfen	JA	JA
Position und Länge des Indexschlüssels prüfen und prüfen, ob eine Überschneidung vorliegt	JA	JA
Reihenfolge der Schlüssel im Index prüfen	JA	NEIN
Gesamtseitenzahl und Anzahl der belegten Seiten prüfen	NEIN	JA
Inhalt von internen Seitenkopffeldern vergleichen	JA	JA
Eindeutigkeit von eindeutigen Schlüssel prüfen	JA	NEIN
Vorhandensein der Datenzeile für einen bestimmten Indexeintrag prüfen	NEIN	JA
Jeden Schlüssel anhand eines Datenwerts überprüfen	NEIN	JA

Tabelle 80. Funktionsvergleich zwischen db2dart und INSPECT für Blockzuordnungsobjekte

Durchgeführte Tests	db2dart	INSPECT
Auf Konsistenzbitfehler prüfen	JA	JA
Gesamtseitenzahl und Anzahl der belegten Seiten prüfen	NEIN	JA
Inhalt von internen Seitenkopffeldern vergleichen	JA	JA

Tabelle 81. Funktionsvergleich zwischen db2dart und INSPECT für Langfelder und LOB-Objekte

Durchgeführte Tests	db2dart	INSPECT
Zuordnungsstrukturen prüfen	JA	JA
Gesamtseitenzahl und Anzahl der belegten Seiten prüfen (nur für LOB-Objekte)	NEIN	JA

Darüber hinaus können folgende Aktionen mit dem Befehl **db2dart** ausgeführt werden:

- Datenseiten formatieren und Speicherauszüge für Datenseiten erstellen
- Indexseiten formatieren und Speicherauszüge für Indexseiten erstellen
- Datenzeilen in ASCII-Format mit Begrenzern formatieren
- Index als ungültig markieren

Der Befehl **INSPECT** kann für diese Zwecke nicht eingesetzt werden.

Analysieren der db2diag-Protokolldateien mit dem Tool 'db2diag'

Datenbank- und Systemadministratoren wird empfohlen, als primäre Protokolldatei das Protokoll mit Benachrichtigungen für die Systemverwaltung (Administration Notification) zu verwenden. Die **db2diag**-Protokolldateien sind für die Fehlerbehebung durch den IBM Software Support vorgesehen.

Die Nachrichten für das Protokoll mit Benachrichtigungen für die Systemverwaltung werden in einem standardisierten Nachrichtenformat ebenfalls in den **db2diag**-Protokolldateien aufgezeichnet.

Das Tool **db2diag** dient zum Filtern und Formatieren der in den **db2diag**-Protokolldateien bereitgestellten Informationsmenge. Das Filtern von Datensätzen der **db2diag**-Protokolldateien kann die erforderliche Zeit für die Suche nach den benötigten Datensätzen bei der Fehlerbehebung reduzieren.

Beispiel 1: db2diag-Protokolldateien nach Datenbankname filtern

Enthält eine Instanz mehrere Datenbanken und wollen Sie nur die Nachrichten anzeigen, die sich auf die Datenbank SAMPLE beziehen, können Sie die **db2diag**-Protokolldateien wie folgt filtern:

```
db2diag -g db=SAMPLE
```

Auf diese Weise würden Sie nur die Datensätze in Datensätzen der **db2diag**-Protokolldateien sehen, die 'DB: SAMPLE' enthalten, wie zum Beispiel der folgende Datensatz:

```
2006-02-15-19.31.36.114000-300 E21432H406         LEVEL: Error
PID     : 940                  TID  : 660         PROC : db2syscs.exe
INSTANCE: DB2                  NODE : 000         DB   : SAMPLE
APPHDL  : 0-1056               APPID: *LOCAL.DB2.060216003103
FUNCTION: DB2 UDB, base sys utilities, sqleDatabaseQuiesce, probe:2
MESSAGE : ADM7507W  Die Anforderung, ein Quiesce für die Datenbank durchzuführen,
war erfolgreich.
```

Beispiel 2: db2diag-Protokolldateien nach Prozess-ID filtern

Mithilfe des folgenden Befehls können alle Nachrichten zu schwer wiegenden Fehlern angezeigt werden, die von Prozessen in den Partitionen 0, 1, 2 oder 3 mit der Prozess-ID (PID) 2200 generiert werden:

```
db2diag -g level=Severe,pid=2200 -n 0,1,2,3
```

Bitte beachten Sie, dass dieser Befehl auf verschiedene Arten und Weisen geschrieben werden könnte, einschließlich **db2diag -l severe -pid 2200 -n 0,1,2,3**. Des Weiteren ist zu beachten, dass mit der Option **-g** eine von der Groß-/Kleinschreibung abhängige Suche angegeben wird, sodass in diesem Fall 'Severe' funktionieren, die Angabe von 'severe' jedoch fehlschlagen würde. Mit diesen Befehlen können erfolgreich die Datensätze in den **db2diag**-Protokolldateien abgerufen werden, die diese Kriterien erfüllen, wie beispielsweise folgender Datensatz:

```
2006-02-13-14.34.36.027000-300 I18366H421         LEVEL: Severe
PID     : 2200                 TID  : 660         PROC : db2syscs.exe
INSTANCE: DB2                  NODE : 000         DB   : SAMPLE
APPHDL  : 0-1433               APPID: *LOCAL.DB2.060213193043
FUNCTION: DB2 UDB, data management, sqldPoolCreate, probe:273
RETCODE : ZRC=0x8002003C=-2147352516=SQLB_BAD_CONTAINER_PATH
          "Der Pfad des Containers ist ungültig"
```

Beispiel 3: Ausgabe des Tools db2diag formatieren

Mit dem folgenden Befehl werden alle Datensätze herausgefiltert, die nach dem 1. Januar 2006 erstellt wurden, nicht schwer wiegende sowie schwer wiegende Fehler enthalten und in den Partitionen 0, 1 oder 2 gespeichert sind. Die Ausgabe der übereinstimmenden Datensätze (wie beispielsweise der Zeitmarke, der Partitionsnummer und der Fehlerkategorie) erfolgt in der ersten Zeile, die Ausgabe der PID, TID und des Instanznamens erfolgt in der zweiten Zeile und die Ausgabe der Fehlernachricht im Anschluss daran:

```
db2diag -time 2006-01-01 -node "0,1,2" -level "Severe, Error" | db2diag -fmt "Time: %{ts}
Partition: %node Message Level: %{level} \nPid: %{pid}  Tid: %{tid}
Instance: %{instance}\nMessage: @{msg}\n"
```

Beispielausgabe:

```
Time: 2006-02-15-19.31.36.099000 Partition: 000 Message Level: Error
Pid: 940 Tid:940 Instance: DB2
Message: ADM7506W  Es besteht eine Anforderung, ein Quiesce für die Datenbank
durchzuführen.
```

Wenn Sie weitere Informationen benötigen, geben Sie die folgenden Befehle aus:

- **db2diag -help** (liefert eine Kurzbeschreibung aller verfügbarer Optionen)
- **db2diag -h brief** (liefert eine Beschreibung aller Optionen ohne Beispiele)
- **db2diag -h notes** (liefert Hinweise zur Verwendung und Informationen zu Einschränkungen)
- **db2diag -h examples** (liefert eine Reihe von Beispielen für die ersten Schritte)
- **db2diag -h tutorial** (liefert Beispiele zu allen verfügbaren Optionen)
- **db2diag -h all** (liefert die umfangreichste Liste von Optionen)

Beispiel 4: Nachrichten von unterschiedlichen Funktionen filtern

Die folgenden Beispiele zeigen, wie Sie nur Nachrichten einer bestimmten Funktion (oder aller Funktionen) innerhalb des Datenbankmanagers anzeigen können. Die unterstützten Funktionen sind:

- ALL, wodurch Datensätze aller Funktionen zurückgegeben werden
- MAIN, wodurch Datensätze von allgemeinen DB2-Protokollen der Diagnoseprogramme wie den **db2diag**-Protokolldateien und vom Protokoll mit Benachrichtigungen für die Systemverwaltung zurückgegeben werden
- OPTSTATS, wodurch Datensätze zurückgegeben werden, die zu Optimierungsprogrammstatistiken gehören

Gehen Sie wie folgt vor, um Nachrichten der Funktion MAIN zu lesen:

```
db2diag -facility MAIN
```

Gehen Sie wie folgt vor, um Nachrichten der Funktion OPTSTATS anzuzeigen und die Datensätze mit einer schwer wiegenden (severe) Einstufung herauszufiltern:

```
db2diag -fac OPTSTATS -level Severe
```

Gehen Sie wie folgt vor, um Nachrichten aller verfügbarer Funktionen anzuzeigen und die Datensätze mit instance=harmistr und level=Error herauszufiltern:

```
db2diag -fac all -g instance=harmistr,level=Error
```

Gehen Sie wie folgt vor, um alle Nachrichten der Funktion OPTSTATS anzuzeigen, die als Fehler (Error) eingestuft wurden, und anschließend die Felder 'Timestamp' und 'PID' in einem bestimmten Format auszugeben:

```
db2diag -fac optstats -level Error -fmt " Time :%{ts} Pid :%{pid}"
```

Beispiel 5: Dateien zusammenfügen und Datensätze anhand der Zeitmarken sortieren

Dieses Beispiel zeigt, wie zwei oder mehr **db2diag**-Protokolldateien zusammengefügt und die Datensätze anhand von Zeitmarken sortiert werden können.

Die folgenden zwei **db2diag**-Protokolldateien sollen zusammengefügt werden:

- db2diag.0.log. Sie enthält Datensätze der Stufe 'Error' mit den folgenden Zeitmarken:
  - 2009-02-26-05.28.49.822637
  - 2009-02-26-05.28.49.835733
  - 2009-02-26-05.28.50.258887
  - 2009-02-26-05.28.50.259685
- db2diag.1.log. Sie enthält Datensätze der Stufe 'Error' mit den folgenden Zeitmarken:
  - 2009-02-26-05.28.11.480542
  - 2009-02-26-05.28.49.764762
  - 2009-02-26-05.29.11.872184
  - 2009-02-26-05.29.11.872968

Führen Sie den folgenden Befehl aus, um die beiden Diagnoseprotokolldateien zusammenzufügen und anhand der Zeitmarken zu sortieren:

```
db2diag -merge db2diag.0.log db2diag.1.log -fmt %{ts} -level error
```

Das Zusammenfügen und Sortieren der Datensätze führt zu folgendem Ergebnis:

- 2009-02-26-05.28.11.480542
- 2009-02-26-05.28.49.764762
- 2009-02-26-05.28.49.822637

- 2009-02-26-05.28.49.835733
- 2009-02-26-05.28.50.258887
- 2009-02-26-05.28.50.259685
- 2009-02-26-05.29.11.872184
- 2009-02-26-05.29.11.872968

Dabei werden die Zeitmarken beim Zusammenfügen chronologisch sortiert.

Beispiel 6: Auf mehrere Verzeichnispfade verteilte Diagnosedateien eines einzelnen Hosts zusammenfügen und die Datensätze nach Zeitmarken sortieren

Dieses Beispiel zeigt, wie Dateien aus drei Datenbankpartitionen auf dem aktuellen Host zusammengefügt werden können. Um die Verzeichnispfade der verteilten Diagnosedateien zu erhalten, wurde der Konfigurationsparameter **diagpath** des Datenbankmanagers wie folgt definiert:

```
db2 update dbm cfg using diagpath '"$n"'
```

Die drei **db2diag**-Protokolldateien, die zusammengefügt werden sollen, sind nachfolgend aufgeführt:

- ~/sqllib/db2dump/NODE0000/db2diag.log
- ~/sqllib/db2dump/NODE0001/db2diag.log
- ~/sqllib/db2dump/NODE0002/db2diag.log

Führen Sie den folgenden Befehl aus, um die drei Diagnoseprotokolldateien zusammenzufügen und anhand der Zeitmarken zu sortieren:

```
db2diag -merge
```

Beispiel 7: Auf mehrere Verzeichnispfade verteilte Diagnosedateien mehrerer Hosts und Datenbankpartitionen zusammenfügen

In diesem Beispiel wurde der Standardverzeichnispfad für Diagnosedaten auf der Basis des physischen Hosts und der Datenbankpartition aufgeteilt, indem der Konfigurationsparameter **diagpath** des Datenbankmanagers definiert und der folgende Befehl verwendet wurde:

```
db2 update dbm cfg using diagpath '"$h$n"'
```

Dieses Beispiel veranschaulicht das Abrufen der Ausgabe aller Datensätze von allen Diagnoseprotokollen sowie das Zusammenfügen der Diagnoseprotokolldateien von jeweils drei Datenbankpartitionen auf den beiden Hosts `bower` und `horton`. Die folgende Liste enthält die 6 **db2diag**-Protokolldateien:

- ~/sqllib/db2dump/HOST_bower/NODE0000/db2diag.log
- ~/sqllib/db2dump/HOST_bower/NODE0001/db2diag.log
- ~/sqllib/db2dump/HOST_bower/NODE0002/db2diag.log
- ~/sqllib/db2dump/HOST_horton/NODE0003/db2diag.log
- ~/sqllib/db2dump/HOST_horton/NODE0004/db2diag.log
- ~/sqllib/db2dump/HOST_horton/NODE0005/db2diag.log

Zur Ausgabe der Datensätze aller 6 **db2diag**-Protokolldateien führen Sie den folgenden Befehl aus:

```
db2diag -global
```

Führen Sie den folgenden Befehl aus, um alle 6 **db2diag**-Protokolldateien im Verzeichnispfad für Diagnosedaten von den jeweils drei Datenbankpartitionen auf den Hosts bower und horton zusammenzufügen und die Ausgabe auf der Basis der Zeitmarkte zu formatieren:

```
db2diag -global -merge -sdir /temp/keon -fmt %{ts}
```

Dabei ist /temp/keon ein von den Hosts bower und horton gemeinsam verwendetes Verzeichnis, in dem die temporären zusammengefügten Dateien vom jeweiligen Host während der Verarbeitung gespeichert werden.

Beispiel 8: Filtern und Zusammenfügen ausschließlich neuer Diagnoseprotokolleinträge

In diesem Beispiel werden die **db2diag**-Protokolldateieinträge so gefiltert, dass nur eine bestimmte Anzahl der neuesten Einträge angezeigt wird. Geben Sie Folgendes ein, um die letzten 5 formatierten Einträge für jede der 3 Partitionen in einer Umgebung mit partitionierten Datenbanken nach Zeitmarke zusammengefügt und formatiert anzuzeigen:

```
db2diag -lastrecords 5 -global -merge -sdir /home/vbmithun -fmt %{ts}

2010-10-08-04.46.02.092192
2010-10-08-04.46.02.092821
2010-10-08-04.46.02.093497
2010-10-08-04.46.02.094431
2010-10-08-04.46.02.095317
2010-10-08-04.46.05.068648
2010-10-08-04.46.05.069212
2010-10-08-04.46.05.069900
2010-10-08-04.46.05.071008
2010-10-08-04.46.05.071831
2010-10-08-04.46.07.302051
2010-10-08-04.46.07.302727
2010-10-08-04.46.07.303544
2010-10-08-04.46.07.304647
2010-10-08-04.46.07.305391
```

Sie können die neuesten Diagnoseprotokolleinträge zusätzlich so filtern, dass nur Nachrichten einer bestimmten Nachrichtenstufe zurückgegeben werden. Geben Sie zum Beispiel Folgendes ein, um von den letzten 10 Einträgen nur die Einträge zurückzugeben, die die Nachrichtenstufe 'schwerwiegend' aufweisen:

```
$ db2diag db2diag.log -lastrecords 10 -level Severe -fmt %{ts}

2010-08-11-04.11.33.733807
2010-08-11-04.11.33.735398
```

Beispiel 9: Archivieren der db2diag-Protokolldateien auf einem Client ohne Instanzen

Ab Version 9.7 Fixpack 4 ist die Option **db2diag –archive** (oder **–A**) in IBM Data Server Driver Package und IBM Data Server for ODBC and CLI verfügbar. Mit dieser Option können Sie die Diagnoseprotokolldatei auf einem Client ohne Instanzen archivieren. Beispiel:

```
$ db2diag -A
db2diag: Moving "/home/usr1/clidriver/db2dump/db2diag.log"
         to     "/home/usr1/clidriver/db2dump/db2diag.log_2010-09-14-01.16.26"
```

Wenn Sie andere Optionen als **–archive** bzw. **–A** angeben, wird eine Fehlernachricht zurückgegeben. Beispiel:

```
$ db2diag –x
db2diag: Unrecognized option: -x

$ db2diag –pid 1234
db2diag: Unrecognized option: -pid
```

Anzeigen und Ändern der globalen Registrierdatenbank (UNIX) mit 'db2greg'

Sie können die globale Registrierdatenbank auf UNIX- und Linux-Plattformen mit dem Befehl **db2greg** anzeigen.

Seit DB2 Version 9.7 werden die globalen DB2-Profilregistrierdatenbanken nicht mehr in der Textdatei <DB2DIR>/default.env aufgezeichnet. Stattdessen werden die globalen DB2-Profileinstellungen für die aktuelle DB2-Installation nun in der Datei global.reg für globale Registrierung eingetragen.

Die globale Registrierdatenbank (Global Registry) ist nur auf UNIX- und Linux-Plattformen vorhanden:

- Bei Rootinstallationen befindet sich die Datei für die globale Registrierdatenbank (Global Registry) im Verzeichnis /var/db2/global.reg (/var/opt/db2/global.reg für HP-UX).
- Bei Nicht-Rootinstallationen befindet sich die Datei für die globale Registrierdatenbank im Verzeichnis $HOME/sqllib/global.reg, wobei $HOME das Ausgangsverzeichnis des Nicht-Rootbenutzers ist.

Die globale Registrierdatenbank besteht aus drei verschiedenen Datensatztypen:

- "Service": Servicesätze enthalten Informationen auf Produktebene, beispielsweise zur Produktversion und zum Installationspfad.
- "Instanz": Instanzsätze enthalten Informationen auf Instanzebene, wie beispielsweise den Instanznamen, den Installationspfad, die Version und die Markierung zum Start beim Booten (start-at-boot).
- "Variable": Variablensätze enthalten Informationen auf Variablenebene, wie beispielsweise den Variablennamen und den Variablenwert.
- Kommentar.

Die globale Registrierdatenbank kann mit dem Tool **db2greg** angezeigt werden. Dieses Tool befindet sich im Verzeichnis sqllib/bin sowie im Installationsverzeichnis install unter bin (zur Verwendung bei Anmeldung als Root).

Die globale Registrierdatenbank kann mit dem Tool **db2greg** bearbeitet werden. Für die Bearbeitung der globalen Registrierdatenbank in Rootinstallationen ist Rootberechtigung erforderlich.

Sie sollten das Tool **db2greg** nur dann verwenden, wenn Sie von IBM Software Support dazu aufgefordert werden.

Identifizieren der Version und Servicestufe von Produkten

Mit dem Befehl **db2level** können Sie die Version und die Servicestufe (Buildstufe und Fixpacknummer) der verwendeten DB2-Instanz ermitteln.

Um festzustellen, ob Ihre DB2-Instanz die neueste Servicestufe aufweist, vergleichen Sie die 'db2level'-Ausgabe mit den Informationen auf den Fixpack-Downloadseiten der DB2-Unterstützungswebsite: www.ibm.com/support/docview.wss?rs=71&uid=swg27007053.

Ein typisches Ergebnis der Ausführung des Befehls **db2level** auf einem Windows-Systems würde in etwa wie folgt aussehen:

```
DB21085I  Instanz "DB2" verwendet "32" Bit und DB2-Codefreigabe "SQL09010" mit
Aktualitäts-ID "01010107".
Informationstoken: "DB2 v9.1.0.189", "n060119", "" und Fixpack "0".
Produkt ist in "c:\SQLLIB" mit DB2-Kopienamen "db2build" installiert.
```

Durch die Kombination der vier Informationstoken wird die genaue Servicestufe der verwendeten DB2-Instanz eindeutig identifiziert. Diese Informationen müssen unbedingt vorliegen, wenn Sie sich an IBM Software Support wenden.

Für JDBC- oder SQLJ-Anwendungen: Wenn Sie den IBM DB2-Treiber für SQLJ und JDBC verwenden, können Sie die Version des Treibers ermitteln, indem Sie das Dienstprogramm 'db2jcc' ausführen.

```
db2jcc -version

IBM DB2 JDBC Driver Architecture 2.3.63
```

Nachahmen von Datenbanken mit 'db2look'

In vielen Fällen ist es von Vorteil, wenn eine Datenbank erstellt werden kann, die die gleiche Struktur wie eine andere Datenbank aufweist. Anstatt neue Anwendungen oder Recoverypläne auf einem Produktionssystem zu testen, ist es beispielsweise sinnvoller, ein Testsystem mit der gleichen Struktur und gleichen Daten zu erstellen und die Tests dort auszuführen.

Auf diese Weise wirken sich die negativen Leistungseinflüsse der Tests nicht auf das Produktionssystem aus, und es ist auch nicht von einer eventuellen versehentlichen Vernichtung von Daten durch eine fehlerhafte Anwendung betroffen. Auch bei der Untersuchung von Problemen (wie beispielsweise ungültigen Ergebnissen, Leistungsproblemen etc.) ist es unter Umständen einfacher, die Probleme in einem Testsystem zu untersuchen, das mit dem Produktionssystem identisch ist.

Mithilfe des Tools **db2look** können Sie die entsprechenden DDL-Anweisungen extrahieren, die erforderlich sind, um die Datenbankobjekte einer Datenbank in einer anderen Datenbank zu reproduzieren. Mit diesem Tool können auch die entsprechenden SQL-Anweisungen generiert werden, die erforderlich sind, um die Statistikdaten aus der einen Datenbank in der anderen zu replizieren, sowie die Anweisungen, die benötigt werden, um die Datenbankkonfiguration, die Datenbankmanagerkonfiguration und die Registrierdatenbankvariablen zu replizieren. Dies ist wichtig, da die neue Datenbank möglicherweise nicht genau die gleiche Datengruppe enthält wie die ursprüngliche Datenbank, Sie aber weiterhin die gleichen Zugriffspläne für beide Systeme verwenden wollen. Der Befehl **db2look** darf nur für Datenbanken verwendet werden, die auf DB2-Servern ab Version 9.5 ausgeführt werden.

Das Tool **db2look** wird im Handbuch *DB2 Command Reference* ausführlich beschrieben. Sie können jedoch eine Liste der Optionen anzeigen, indem Sie das Tool ohne Parameter ausführen. Detailliertere Informationen zur Verwendung erhalten Sie, wenn Sie die Option **-h** angeben.

Verwenden von 'db2look' zum Nachahmen der Tabellen in einer Datenbank

Um die DDL für die Tabellen in der Datenbank zu extrahieren, verwenden Sie die Option **-e**. Erstellen Sie beispielsweise wie folgt von der Datenbank SAMPLE eine Kopie namens SAMPLE2, in der alle Objekte erstellt werden, die auch in der ersten Datenbank vorhanden sind:

```
C:\>db2 create database sample2
DB20000I Der Befehl CREATE DATABASE wurde erfolgreich ausgeführt.
C:\>db2look -d sample -e > sample.ddl
-- USER ist:
-- DLL für Tabelle(n) wird erstellt
-- Automatisches Binden des Pakets ...
-- Binden war erfolgreich
-- Automatisches Binden des Pakets ...
-- Binden war erfolgreich
```

Anmerkung: Soll die DDL für die benutzerdefinierten Speicherbereiche, Datenbankpartitionsgruppen und Pufferpools ebenfalls erzeugt werden, fügen Sie im obigen Befehl nach der Markierung **-e** die Markierung **-l** hinzu. Die standardmäßigen Datenbankpartitionsgruppen, Pufferpools und Tabellenbereiche werden nicht extrahiert. Dies liegt daran, dass sie in jeder Datenbank bereits standardmäßig vorhanden sind. Wenn Sie diese Elemente ebenfalls nachahmen wollen, müssen Sie sie manuell ändern.

Rufen Sie die Datei `sample.ddl` in einem Texteditor auf. Da die DDL in dieser Datei für die neue Datenbank ausgeführt werden soll, müssen Sie die Anweisung CONNECT TO SAMPLE in CONNECT TO SAMPLE2 ändern. Wurde die Option **-l** verwendet, müssen Sie unter Umständen die Pfade der Tabellenbereichsbefehle so ändern, dass sie ebenfalls auf die entsprechenden Pfade zeigen. Betrachten Sie bei dieser Gelegenheit auch den Rest des Inhalts der Datei. Die Anweisungen CREATE TABLE, ALTER TABLE und CREATE INDEX sollten für alle Benutzertabellen in der Beispieldatenbank vorhanden sein:

```
...
-----------------------------------------------
-- DDL-Anweisungen für Tabelle "DB2"."ORG"
-----------------------------------------------

  CREATE TABLE "DB2"."ORG"  (
    "DEPTNUMB" SMALLINT NOT NULL ,
    "DEPTNAME" VARCHAR(14) ,
    "MANAGER" SMALLINT ,
    "DIVISION" VARCHAR(10) ,
    "LOCATION" VARCHAR(13) )
   IN "USERSPACE1" ;
...
```

Nachdem Sie die Verbindungsanweisung CONNECT geändert haben, führen Sie die Anweisungen wie folgt aus:

```
C:\>db2 -tvf sample.ddl > sample2.out
```

Werfen Sie einen Blick auf die Ausgabedatei `sample2.out` - alle Operationen müssen erfolgreich ausgeführt worden sein. Traten Fehler auf, müssen diese in entsprechenden Fehlernachrichten angegeben sein. Beheben Sie eventuelle Fehler und führen Sie die Anweisungen anschließend erneut aus.

Wie Sie der Ausgabe entnehmen können, wurde die DDL für alle Benutzertabellen exportiert. Dies ist das Standardverhalten. Es stehen jedoch weitere Optionen zur

Verfügung, um genauer anzugeben, welche Tabellen eingeschlossen werden sollen. Um beispielsweise nur die Tabellen STAFF und ORG einzuschließen, verwenden Sie die Option **-t** wie folgt:

```
C:\>db2look -d sample -e -t staff org > staff_org.ddl
```

Um nur Tabellen mit dem Schema DB2 einzuschließen, verwenden Sie die Option **-z** wie folgt:

```
C:\>db2look -d sample -e -z db2 > db2.ddl
```

Nachahmen von Statistikdaten für Tabellen

Soll mithilfe der Testdatenbank die Leistung überprüft oder ein Leistungsproblem behoben werden, müssen die für beide Datenbanken generierten Zugriffspläne unbedingt identisch sein. Das Optimierungsprogramm generiert Zugriffspläne auf der Grundlage von Statistikdaten, Konfigurationsparametern, Registrierdatenbankvariablen und Umgebungsvariablen. Sind diese Elemente auf beiden Systemen identisch, trifft dies sehr wahrscheinlich auch auf die Zugriffspläne zu.

Wurden in beide Datenbanken genau die gleichen Daten geladen und werden für beide Datenbanken die gleichen Optionen von RUNSTATS ausgeführt, sollten die Statistikdaten identisch sein. Enthalten die Datenbanken jedoch unterschiedliche Daten oder wird in der Testdatenbank nur eine Untergruppe von Daten verwendet, weisen die Statistikdaten wahrscheinlich große Unterschiede auf. In diesem Fall können Sie mithilfe von **db2look** die Statistikdaten aus der Produktionsdatenbank zusammenstellen und in die Testdatenbank stellen. Dazu erstellen Sie UPDATE-Anweisungen für die Gruppe SYSSTAT der aktualisierbaren Katalogtabellen sowie RUNSTATS-Befehle für alle Tabellen.

Die Option zum Erstellen der Statistikanweisungen lautet **-m**. Gehen Sie noch einmal zum Beispiel SAMPLE/SAMPLE2 zurück, stellen Sie die Statistikdaten aus der Datenbank SAMPLE zusammen, und fügen Sie diese der Datenbank SAMPLE2 hinzu:

```
C:\>db2look -d sample -m > stats.dml
-- USER ist:
-- db2look wird im Nachahmungsmodus ausgeführt
```

Wie zuvor muss in der Ausgabedatei die Anweisung CONNECT TO SAMPLE in CONNECT TO SAMPLE2 geändert werden. Betrachten Sie wiederum auch im Rest der Datei den Inhalt der RUNSTATS- und UPDATE-Anweisungen:

```
...
-- Tabelle ORG nachahmen
RUNSTATS ON TABLE "DB2"."ORG" ;

UPDATE SYSSTAT.INDEXES
SET NLEAF=-1,
    NLEVELS=-1,
    FIRSTKEYCARD=-1,
    FIRST2KEYCARD=-1,
    FIRST3KEYCARD=-1,
    FIRST4KEYCARD=-1,
    FULLKEYCARD=-1,
    CLUSTERFACTOR=-1,
    CLUSTERRATIO=-1,
    SEQUENTIAL_PAGES=-1,
    PAGE_FETCH_PAIRS='',
    DENSITY=-1,
    AVERAGE_SEQUENCE_GAP=-1,
    AVERAGE_SEQUENCE_FETCH_GAP=-1,
    AVERAGE_SEQUENCE_PAGES=-1,
```

```
    AVERAGE_SEQUENCE_FETCH_PAGES=-1,
    AVERAGE_RANDOM_PAGES=-1,
    AVERAGE_RANDOM_FETCH_PAGES=-1,
    NUMRIDS=-1,
    NUMRIDS_DELETED=-1,
    NUM_EMPTY_LEAFS=-1
WHERE TABNAME = 'ORG' AND TABSCHEMA = 'DB2   ';
...
```

Ähnlich der Option **-e**, mit der die DDL extrahiert wird, können die Optionen **-t** und **-z** verwendet werden, um eine Gruppe von Tabellen anzugeben.

Extrahieren von Konfigurationsparametern und Umgebungsvariablen

Das Optimierungsprogramm wählt Zugriffspläne auf der Grundlage von Statistikdaten, Konfigurationsparametern, Registrierdatenbankvariablen und Umgebungsvariablen aus. Wie bei den Statistikdaten kann **db2look** verwendet werden, um die erforderlichen UPDATE- und SET-Anweisungen für die Konfiguration zu generieren. Hierfür wird die Option **-f** verwendet. Beispiel:

```
c:\>db2look -d sample -f>config.txt
-- USER ist: DB2INST1
-- Automatisches Binden des Pakets ...
-- Binden war erfolgreich
-- Automatisches Binden des Pakets ...
-- Binden war erfolgreich
```

Die Ausgabe in der Datei `config.txt` sieht etwa wie folgt aus:

```
-- Diese CLP-Datei wurde erstellt mit DB2LOOK Version 9.1
-- Zeitmarke: 2/16/2006 7:15:17 PM
-- Datenbankname: SAMPLE
-- Datenbankmanagerversion: DB2/NT Version 9.1.0
-- Codepage der Datenbank: 1252
-- Sortierfolge für Datenbank lautet: UNIQUE


CONNECT TO SAMPLE;

------------------------------------------------------------
-- Konfigurationsparameter für Datenbank und Datenbankmanager
------------------------------------------------------------

UPDATE DBM CFG USING cpuspeed 2.991513e-007;
UPDATE DBM CFG USING intra_parallel NO;
UPDATE DBM CFG USING comm_bandwidth 100.000000;
UPDATE DBM CFG USING federated NO;

...

-------------------------------------
-- Einstellungen der Umgebungsvariablen
-------------------------------------


COMMIT WORK;

CONNECT RESET;
```

Anmerkung: Es werden nur die Parameter und Variablen eingeschlossen, die sich auf den DB2-Compiler auswirken. Wenn eine Registrierdatenbankvariable, die sich auf den Compiler auswirkt, auf ihren Standardwert gesetzt wird, wird sie unter den Einstellungen der Umgebungsvariablen nicht aufgeführt.

Auflisten der auf Ihrem System installierten DB2-Datenbankprodukte (Linux und UNIX)

Unter den unterstützten Linux- und UNIX-Betriebssystemen können Sie mit dem Befehl **db2ls** die DB2-Datenbankprodukte und -Features auflisten, die auf Ihrem System installiert sind, einschließlich der HTML-Dokumentation zu DB2 Version 9.7.

Vorbereitende Schritte

Mindestens ein Datenbankprodukt von DB2 Version 9 (oder einer späteren Version) muss bereits von einem Benutzer mit Rootberechtigung installiert worden sein, damit ein symbolischer Link zum Befehl **db2ls** im Verzeichnis `/usr/local/bin` verfügbar ist.

Informationen zu diesem Vorgang

Da Sie mehrere Kopien von DB2-Datenbankprodukten auf dem System installieren können und für die Installation dieser DB2-Datenbankprodukte und -Features einen Pfad Ihrer Wahl festlegen können, benötigen Sie ein Tool, das protokolliert, was an welcher Position installiert ist. Unter den unterstützten Linux- und UNIX-Betriebssystemen können Sie mit dem Befehl **db2ls** die DB2-Produkte und -Features auflisten, die auf Ihrem System installiert sind, einschließlich der HTML-Dokumentation zu DB2.

Der Befehl **db2ls** befindet sich sowohl auf den Installationsmedien als auch in einer DB2-Installationskopie auf dem System. Der Befehl **db2ls** kann von beiden Positionen aus ausgeführt werden. Der Befehl **db2ls** kann von den Installationsmedien für alle Produkte mit Ausnahme von IBM Data Server Driver Package ausgeführt werden.

Mit dem Befehl **db2ls** können Sie Folgendes auflisten:

- Die Installationspfade der auf dem System installierten DB2-Datenbankprodukte und ihre DB2-Datenbankproduktstufe
- Alle oder bestimmte DB2-Datenbankprodukte und -Features in einem bestimmten Installationspfad

Einschränkungen

Die Ausgabe des Befehls **db2ls** hängt von der verwendeten ID ab:

- Wird der Befehl **db2ls** von einem Benutzer mit Rootberechtigung ausgeführt, werden nur DB2-Rootinstallationen abgerufen.
- Wird der Befehl **db2ls** von einer Benutzer-ID ohne Rootberechtigung ausgeführt, werden DB2-Rootinstallationen abgerufen und die nicht als Root ausgeführte Installation, deren Eigner die übereinstimmende Benutzer-ID ohne Rootberechtigung ist. DB2-Installationen, deren Eigner andere Benutzer-IDs ohne Rootberechtigung sind, werden nicht abgerufen.

Der Befehl **db2ls** ist die einzige Möglichkeit zum Abfragen eines DB2-Datenbankprodukts. Sie können *keine* Abfrage von DB2-Datenbankprodukten mithilfe von nativen Dienstprogrammen der Betriebssysteme Linux oder UNIX (z. B. **pkginfo**, **rpm**, **SMIT** oder **swlist**) durchführen. Alle vorhandenen Scripts, in denen native Installationsprogramme enthalten sind, die Sie für Schnittstellen oder Abfragen im Zusammenhang mit DB2-Installationen verwenden, müssen geändert werden.

Den Befehl **db2ls** können Sie *nicht* unter Windows-Betriebssystemen verwenden.

Vorgehensweise

- Wenn Sie den Pfad, in dem die DB2-Datenbankprodukte auf Ihrem System installiert sind, und die Version dieser DB2-Datenbankprodukte auflisten möchten, geben Sie Folgendes ein:

  ```
  db2ls
  ```

  Der Befehl listet die folgenden Informationen für jedes DB2-Datenbankprodukt auf, das auf dem System installiert ist:
  - Installationspfad
  - Version
  - Fixpack
  - Spezielle Installationsnummer. Diese Spalte wird von der IBM DB2-Unterstützung verwendet.
  - Installationsdatum. Diese Spalte gibt an, wann das DB2-Datenbankprodukt zuletzt modifiziert wurde.
  - Benutzer-ID für Installationsprogramm. Diese Spalte gibt die Benutzer-ID an, mit der das DB2-Datenbankprodukt installiert wurde.
- Wenn Sie Informationen zu DB2-Datenbankprodukten oder -Features in einem bestimmten Installationspfad auflisten möchten, müssen Sie den Parameter **q** angeben:

  ```
  db2ls -q -p -b basisinstallationsverzeichnis
  ```

  Dabei gilt Folgendes:
  - **q** gibt an, dass Sie ein Produkt oder Feature abfragen. Dieser Parameter ist verbindlich. Wenn ein DB2-Produkt der Version 8 abgefragt wird, wird ein Leerwert zurückgegeben.
  - **p** gibt an, dass in der Liste Produkte, aber keine Features angezeigt werden.
  - **b** gibt das Installationsverzeichnis des Produkts oder des Features an, das abgefragt wird. Dieser Parameter ist verbindlich, wenn Sie den Befehl nicht im Installationsverzeichnis ausführen.

Ergebnisse

Je nach den angegebenen Parametern werden auf den Befehl hin die folgenden Informationen aufgelistet:

- Installationspfad. Diese Angabe erfolgt nur einmal, nicht für jede Funktion.
- Die folgenden Informationen werden angezeigt:
  - Die ID der Antwortdatei für das installierte Feature bzw. (bei Angabe der Option **p**) die ID der Antwortdatei für das installierte Produkt. Beispiel: `ENTERPRISE_SERVER_EDITION`.
  - Der Name des Features bzw. (bei Angabe der Option **p**) der Name des Produkts.
  - Produktversion, Release, Modifikationsstufe, Fixpack-Stufe (VRMF). Beispiel: `9.5.0.0`.
  - Fixpack (sofern anwendbar). Ist beispielsweise Fixpack 1 installiert, wird der Wert 1 angezeigt. Dies schließt nachgeordnete Versionen wie Fixpack 1a ein.
- Wenn eine der VRMF-Informationen des Produkts nicht übereinstimmen, wird eine Warnung am Ende der Ausgabeliste angezeigt. In dieser Nachricht wird die Anwendung eines Fixpacks empfohlen.

Überwachung und Fehlerbehebung mit dem Befehl 'db2pd'

Der Befehl **db2pd** wird für die Fehlerbehebung verwendet, da dieser Befehl umgehend Informationen aus den DB2-Speichersätzen bereitstellen kann.

Übersicht

Das Tool erfasst Informationen, ohne Verriegelungen zu verwenden oder Steuerkomponentenressourcen zu nutzen. Es ist daher möglich (und wird erwartet), dass Informationen abgerufen werden, die sich während des Erfassens von Informationen durch **db2pd** ändern; deshalb sind die Daten unter Umständen nicht vollkommen exakt. Wenn geänderte Speicherzeiger festgestellt werden, wird eine Signalroutine verwendet, um zu verhindern, dass **db2pd** abnormal beendet wird. Dies kann dazu führen, dass Nachrichten wie 'Changing data structure forced command termination' ('Befehl aufgrund geänderter Datenstruktur beendet') in der Ausgabe enthalten sind. Dennoch kann das Tool für die Fehlerbehebung nützlich sein. Zu den Vorteilen beim Erfassen von Informationen ohne Verriegelung gehören beispielsweise das schnellere Abrufen und das Vermeiden von Konkurrenzsituationen bei Steuerkomponentenressourcen.

Wenn Sie Informationen zum Datenbankverwaltungssystem erfassen wollen, wenn ein bestimmter SQLCODE, ZRC-Code oder ECF-Code auftritt, können Sie dies mithilfe des Befehls **db2pdcfg -catch** bewerkstelligen. Wenn die Fehler erfasst werden, wird das Script 'db2cos' (Aufrufscript) gestartet. Das Script 'db2cos' kann dynamisch so geändert werden, dass ein beliebiger Befehl **db2pd**, Betriebssystembefehl oder ein anderer, zur Problemlösung erforderlicher Befehl ausgeführt werden kann. Die Schablonenscriptdatei für 'db2cos' befindet sich unter UNIX und Linux im Verzeichnis `sqllib/bin`. Unter Windows befindet sich 'db2cos' im Verzeichnis `$DB2PATH\bin`.

Wenn Sie einen neuen Knoten hinzufügen, können Sie den Fortschritt der Operation auf dem Datenbankpartitionsserver überwachen, bei der der Knoten mit dem Befehl **db2pd -addnode** und den optionalen Parametern für ausführlichere Informationen `oldviewapps` und `detail` hinzugefügt wird.

Wenn Sie eine Liste der Ereignismonitore benötigen, die zurzeit aktiv sind oder zuvor aus einem beliebigen Grund inaktiviert wurden, können Sie diese Liste mit dem Befehl **db2pd -gfw** aufrufen. Dieser Befehl gibt auch für jede schnelle Ausgabeprogramm-EDU Statistikdaten und Informationen zu den Zielen zurück, in die Ereignismonitore Daten schreiben.

Beispiele

Es folgt eine Reihe von Beispielen, in denen der Befehl **db2pd** zur Beschleunigung der Fehlerbehebung eingesetzt werden kann:

- Beispiel 1: Wartestatus für Sperren diagnostizieren
- Beispiel 2: Mit dem Parameter **-wlocks** alle Sperren erfassen, auf die gewartet wird
- Beispiel 3: Mit dem Parameter **-apinfo** detaillierte Laufzeitinformationen zum Eigner der Sperre und zum wartenden Prozess erfassen
- Beispiel 4: Aufrufscripts bei der Bearbeitung eines Sperrenfehlers verwenden
- Beispiel 5: Anwendung zu einer dynamischen SQL-Anweisung zuordnen
- Beispiel 6: Speicherbelegung überwachen
- Beispiel 7: Anwendung ermitteln, die den Tabellenbereich belegt

- Beispiel 8: Recovery überwachen
- Beispiel 9: Menge der von einer Transaktion genutzten Ressourcen ermitteln
- Beispiel 10: Protokollbelegung überwachen
- Beispiel 11: Sysplex-Liste anzeigen
- Beispiel 12: Stack-Traces generieren
- Beispiel 13: Speicherstatistik für eine Datenbankpartition anzeigen
- Beispiel 14: Überwachen des Fortschritts bei der Indexreorganisation
- Beispiel 15: Anzeigen der obersten EDUs, sortiert nach benötigter Prozessorzeit, sowie Anzeigen von EDU-Stackinformationen
- Beispiel 16: Anzeigen der Messdaten für Agentenereignisse

Beispiel 1: Wartestatus für Sperren diagnostizieren

Der Befehl **db2pd -db** *datenbankname* **-locks -transactions -applications -dynamic** liefert Ergebnisse ähnlich den folgenden:

```
Locks:
Address            TranHdl Lockname                   Type Mode Sts Owner Dur HldCnt Att    ReleaseFlg
0x07800000202E5238 3       00020002000000040000000052 Row  ..X  G   3     1   0      0x0000 0x40000000
0x07800000202E4668 2       00020002000000040000000052 Row  ..X  W*  2     1   0      0x0000 0x40000000
```

Die ersten Ergebnisse für die Datenbank, die Sie mithilfe der Datenbanknamensoption **-db** angegeben haben, zeigen die Sperren für diese Datenbank an. Die Ergebnisse zeigen, dass TranHdl 2 bei einer Sperre wartet, die von TranHdl 3 gehalten wird.

```
Transactions:
Address            AppHandl [nod-index] TranHdl Locks State Tflag      Tflag2     Firstlsn       Lastlsn        LogSpace SpaceReserved TID           AxRegCnt GXID
0x0780000020251B80 11       [000-00011] 2       4     READ  0x00000000 0x00000000 0x000000000000 0x000000000000 0        0             0x000000000B7 1        0
0x0780000020252900 12       [000-00012] 3       4     WRITE 0x00000000 0x00000000 0x000000FA000C 0x000000FA000C 113      154           0x000000000B8 1        0
```

Es wird deutlich, dass TranHdl 2 mit AppHandl 11 verknüpft ist und TranHdl 3 mit AppHandl 12.

```
Applications:
Address            AppHandl [nod-index] NumAgents CoorPid Status        C-AnchID C-StmtUID L-AnchID L-StmtUID Appid

0x07800000006879E0 12       [000-00012] 1         1073336 UOW-Waiting   0        0         17       1         *LOCAL.burford.060303225602
0x0780000000685E80 11       [000-00011] 1         1040570 UOW-Executing 17       1         94       1         *LOCAL.burford.060303225601
```

Es wird deutlich, dass AppHandl 12 zuletzt die dynamische Anweisung 17, 1 ausführte, ApplHandl 11 momentan die dynamische Anweisung 17, 1 ausführt und zuletzt die Anweisung 94, 1 ausführte.

```
Dynamic SQL Statements:
Address            AnchID StmtUID NumEnv NumVar NumRef NumExe Text
0x07800000209FD800 17     1       1      1      2      2      update pdtest set c1 = 5
0x07800000209FCCC0 94     1       1      1      2      2      set lock mode to wait 1
```

Es wird deutlich, dass die Textspalte SQL-Anweisungen anzeigt, die mit dem Zeitlimit für Sperre verknüpft sind.

Beispiel 2: Mit dem Parameter **-wlocks** alle Sperren erfassen, auf die gewartet wird

Der Befehl **db2pd -wlocks -db pdtest** liefert Ergebnisse ähnlich den folgenden. Diese Ergebnisse zeigen, dass die erste Anwendung (AppHandl 47) eine Einfügung (INSERT) in eine Tabelle vornimmt und die zweite Anwendung (AppHandl 46) eine SELECT-Anweisung für die betreffende Tabelle ausführt:

```
venus@boson:/home/venus =>db2pd -wlocks -db pdtest

Database Partition 0 -- Database PDTEST -- Active -- Up 0 days 00:01:22

Locks being waited on :
```

```
AppHandl [nod-index] TranHdl    Lockname                  Type    Mode Conv Sts CoorEDU    AppName  AuthID   AppID
47       [000-00047] 8          00020004000000000840000652 Row     ..X       G   5160       db2bp    VENUS    *LOCAL.venus.071207213730
46       [000-00046] 2          00020004000000000840000652 Row     .NS       W   5913       db2bp    VENUS    *LOCAL.venus.071207213658
```

Beispiel 3: Mit dem Parameter **-apinfo** detaillierte Laufzeitinformationen zum Eigner der Sperre und zum wartenden Prozess erfassen

Die folgende Beispielausgabe wurde unter denselben Bedingungen wie die Ausgabe für Beispiel 2 generiert:

```
venus@boson:/home/venus =>db2pd -apinfo 47 -db pdtest

Database Partition 0 -- Database PDTEST -- Active -- Up 0 days 00:01:30

Application :
  Address :                0x0780000001676480
  AppHandl [nod-index] :   47       [000-00047]
  Application PID :        876558
  Application Node Name :  boson
  IP Address:              n/a
  Connection Start Time :  (1197063450)Fri Dec  7 16:37:30 2007
  Client User ID :         venus
  System Auth ID :         VENUS
  Coordinator EDU ID :     5160
  Coordinator Partition :  0
  Number of Agents :       1
  Locks timeout value :    4294967294 seconds
  Locks Escalation :       No
  Workload ID :            1
  Workload Occurrence ID : 2
  Trusted Context :        n/a
  Connection Trust Type :  non trusted
  Role Inherited :         n/a
  Application Status :     UOW-Waiting
  Application Name :       db2bp
  Application ID :         *LOCAL.venus.071207213730

  ClientUserID :           n/a
  ClientWrkstnName :       n/a
  ClientApplName :         n/a
  ClientAccntng :          n/a

  List of inactive statements of current UOW :
    UOW-ID :          2
    Activity ID :     1
    Package Schema :  NULLID
    Package Name :    SQLC2G13
    Package Version :
    Section Number :  203
    SQL Type :        Dynamic
    Isolation :       CS
    Statement Type :  DML, Insert/Update/Delete
    Statement :       insert into pdtest values 99


venus@boson:/home/venus =>db2pd -apinfo 46 -db pdtest

Database Partition 0 -- Database PDTEST -- Active -- Up 0 days 00:01:39

Application :
  Address :                0x0780000000D77A60
  AppHandl [nod-index] :   46       [000-00046]
  Application PID :        881102
  Application Node Name :  boson
  IP Address:              n/a
  Connection Start Time :  (1197063418)Fri Dec  7 16:36:58 2007
  Client User ID :         venus
```

```
System Auth ID :         VENUS
Coordinator EDU ID :     5913
Coordinator Partition :  0
Number of Agents :       1
Locks timeout value :    4294967294 seconds
Locks Escalation :       No
Workload ID :            1
Workload Occurrence ID : 1
Trusted Context :        n/a
Connection Trust Type :  non trusted
Role Inherited :         n/a
Application Status :     Lock-wait
Application Name :       db2bp
Application ID :         *LOCAL.venus.071207213658

ClientUserID :           n/a
ClientWrkstnName :       n/a
ClientApplName :         n/a
ClientAccntng :          n/a

List of active statements :
 *UOW-ID :          3
  Activity ID :     1
  Package Schema :  NULLID
  Package Name :    SQLC2G13
  Package Version :
  Section Number :  201
  SQL Type :        Dynamic
  Isolation :       CS
  Statement Type :  DML, Select (blockable)
  Statement :       select * from pdtest
```

Beispiel 4: Aufrufscripts bei der Bearbeitung eines Sperrenfehlers verwenden

Suchen Sie zum Verwenden der Aufrufscripts die db2cos-Ausgabedateien. Die Speicherposition der Dateien wird vom Konfigurationsparameter **diagpath** des Datenbankmanagers gesteuert. Der Inhalt der Ausgabedateien ist unterschiedlich und hängt davon ab, welche Befehle Sie in die db2cos-Scriptdatei eingeben. Es folgt ein Beispiel für die Ausgabe, die bereitgestellt wird, wenn die db2cos-Scriptdatei den Befehl **db2pd -db sample -locks** enthält:

```
Lock Timeout Caught
Thu Feb 17 01:40:04 EST 2006
Instance DB2
Database: SAMPLE
Partition Number: 0
PID: 940
TID: 2136
Function: sqlplnfd
Component: lock manager
Probe: 999
Timestamp: 2006-02-17-01.40.04.106000
AppID: *LOCAL.DB2...
AppHdl:
...
Database Partition 0 -- Database SAMPLE -- Active -- Up 0 days 00:06:53
Locks:
Address TranHdl Lockname Type Mode Sts Owner Dur HldCnt Att Rlse
0x402C6B30 3 00020003000000040000000052 Row ..X W* 3 1 0 0 0x40
```

In der Ausgabe gibt `W*` die Sperre an, bei der ein Zeitlimit überschritten wurde. In diesem Fall ist ein Wartestatus für Sperren aufgetreten. Eine Zeitlimit für Sperren kann auch auftreten, wenn eine Sperre in einen höheren Modus konvertiert wird. Dies wird in der Ausgabe durch `C*` angegeben.

Sie können die Ergebnisse einer Transaktion, einer Anwendung, einem Agenten oder sogar einer SQL-Anweisung mit der von anderen **db2pd**-Befehlen in der db2cos-Datei bereitgestellten Ausgabe zuordnen. Sie können die Ausgabe einschränken oder andere Befehle verwenden, um die gewünschten Informationen zu erfassen. Sie können beispielsweise die Parameter **-locks wait** von **db2pd** verwenden, um nur Sperren mit einem Wartestatus auszugeben. Sie können auch die Parameter **-app** und **-agent** verwenden.

Beispiel 5: Anwendung zu einer dynamischen SQL-Anweisung zuordnen

Der Befehl **db2pd -applications -dynamic** listet die aktuelle und die letzte Anker-ID sowie die eindeutige Anweisungs-ID für dynamische SQL-Anweisungen auf. Dies ermöglicht eine direkte Zuordnung einer Anwendung zu einer dynamischen SQL-Anweisung.

```
Applications:
Address            AppHandl [nod-index] NumAgents  CoorPid Status
0x00000002006D2120 780      [000-00780] 1          10615    UOW-Executing

C-AnchID C-StmtUID  L-AnchID L-StmtUID  Appid
163      1          110      1          *LOCAL.burford.050202200412

Dynamic SQL Statements:
Address            AnchID StmtUID NumEnv NumVar NumRef NumExe Text
0x0000000220A02760 163    1       2      2      2      1      CREATE VIEW MYVIEW
0x0000000220A0B460 110    1       2      2      2      1      CREATE VIEW YOURVIEW
```

Beispiel 6: Speicherbelegung überwachen

Wie die folgende Beispielausgabe verdeutlicht, kann Ihnen der Befehl **db2pd -memblock** dabei helfen, sich einen Überblick über die Speicherbelegung zu verschaffen:

```
All memory blocks in DBMS set.

Address            PoolID     PoolName     BlockAge   Size(Bytes) I LOC   File
0x0780000000740068 62         resynch      2          112         1 1746  1583816485
0x0780000000725688 62         resynch      1          108864      1 127   1599127346
0x07800000001F4348 57         ostrack      6          5160048     1 3047  698130716
0x07800000001B5608 57         ostrack      5          240048      1 3034  698130716
0x07800000001A0068 57         ostrack      1          80          1 2970  698130716
0x07800000001A00E8 57         ostrack      2          240         1 2983  698130716
0x07800000001A0208 57         ostrack      3          80          1 2999  698130716
0x07800000001A0288 57         ostrack      4          80          1 3009  698130716
0x0780000000700068 70         apmh         1          360         1 1024  3878879032
0x07800000007001E8 70         apmh         2          48          1 914   1937674139
0x0780000000700248 70         apmh         3          32          1 1000  1937674139
...
```

Anschließend folgt die sortierte Ausgabe 'pro Pool':

```
Memory blocks sorted by size for ostrack pool:
PoolID     PoolName   TotalSize(Bytes)     TotalCount LOC   File
57         ostrack    5160048              1          3047  698130716
57         ostrack    240048               1          3034  698130716
57         ostrack    240                  1          2983  698130716
57         ostrack    80                   1          2999  698130716
57         ostrack    80                   1          2970  698130716
57         ostrack    80                   1          3009  698130716
Total size for ostrack pool: 5400576 bytes

Memory blocks sorted by size for apmh pool:
PoolID     PoolName   TotalSize(Bytes)     TotalCount LOC   File
70         apmh       40200                2          121   2986298236
70         apmh       10016                1          308   1586829889
70         apmh       6096                 2          4014  1312473490
70         apmh       2516                 1          294   1586829889
```

```
70          apmh        496                 1           2192  1953793439
70          apmh        360                 1           1024  3878879032
70          apmh        176                 1           1608  1953793439
70          apmh        152                 1           2623  1583816485
70          apmh        48                  1           914   1937674139
70          apmh        32                  1           1000  1937674139
Total size for apmh pool: 60092 bytes
...
```

Im letzten Abschnitt der Ausgabe werden die Speicherkonsumenten für die gesamte Speichergruppe sortiert:

```
All memory consumers in DBMS memory set:
PoolID      PoolName    TotalSize(Bytes)     %Bytes TotalCount %Count LOC   File
57          ostrack     5160048              71.90  1          0.07   3047  698130716
50          sqlch       778496               10.85  1          0.07   202   2576467555
50          sqlch       271784               3.79   1          0.07   260   2576467555
57          ostrack     240048               3.34   1          0.07   3034  698130716
50          sqlch       144464               2.01   1          0.07   217   2576467555
62          resynch     108864               1.52   1          0.07   127   1599127346
72          eduah       108048               1.51   1          0.07   174   4210081592
69          krcbh       73640                1.03   5          0.36   547   4210081592
50          sqlch       43752                0.61   1          0.07   274   2576467555
70          apmh        40200                0.56   2          0.14   121   2986298236
69          krcbh       32992                0.46   1          0.07   838   698130716
50          sqlch       31000                0.43   31         2.20   633   3966224537
50          sqlch       25456                0.35   31         2.20   930   3966224537
52          kerh        15376                0.21   1          0.07   157   1193352763
50          sqlch       14697                0.20   1          0.07   345   2576467555
...
```

Sie können Hauptspeicherblöcke auch für privaten Speicher unter den Betriebssystemen UNIX und Linux auflisten. Der Befehl **db2pd -memb pid=159770** generiert beispielsweise Ergebnisse ähnlich den folgenden:

```
All memory blocks in Private set.

Address            PoolID     PoolName     BlockAge   Size(Bytes) I LOC   File
0x0000000110469068 88         private      1          2488        1 172   4283993058
0x0000000110469A48 88         private      2          1608        1 172   4283993058
0x000000011046A0A8 88         private      3          4928        1 172   4283993058
0x000000011046B408 88         private      4          7336        1 172   4283993058
0x000000011046D0C8 88         private      5          32          1 172   4283993058
0x000000011046D108 88         private      6          6728        1 172   4283993058
0x000000011046EB68 88         private      7          168         1 172   4283993058
0x000000011046EC28 88         private      8          24          1 172   4283993058
0x000000011046EC68 88         private      9          408         1 172   4283993058
0x000000011046EE28 88         private      10         1072        1 172   4283993058
0x000000011046F288 88         private      11         3464        1 172   4283993058
0x0000000110470028 88         private      12         80          1 172   4283993058
0x00000001104700A8 88         private      13         480         1 1534  862348285
0x00000001104702A8 88         private      14         480         1 1939  862348285
0x0000000110499FA8 88         private      80         65551       1 1779  4231792244
Total set size: 94847 bytes

Memory blocks sorted by size:
PoolID      PoolName    TotalSize(Bytes)     TotalCount LOC   File
88          private     65551                1          1779  4231792244
88          private     28336                12         172   4283993058
88          private     480                  1          1939  862348285
88          private     480                  1          1534  862348285
Total set size: 94847 bytes
```

Beispiel 7: Anwendung ermitteln, die den Tabellenbereich belegt

Mit dem Befehl **db2pd -tcbstats** können Sie die Anzahl der Einfügungen für eine Tabelle ermitteln. Es folgen Beispielinformationen zu der benutzerdefinierten, globalen, temporären Tabelle TEMP1:

```
TCB Table Information:
Address            TbspaceID TableID PartID MasterTbs MasterTab TableName  SchemaNm ObjClass DataSize  LfSize LobSize XMLSize
0x0780000020B62AB0 3         2       n/a    3         2         TEMP1      SESSION  Temp     966       0      0       0
```

```
TCB Table Stats:
Address            TableName   Scans   UDI   PgReorgs   NoChgUpdts   Reads   FscrUpdates Inserts   Updates   Deletes   OvFlReads OvFlCrtes
0x0780000020B62AB0 TEMP1       0       0     0          0            0       0           43968     0         0          0        0
```

Sie können anschließend die Informationen zu Tabellenbereich 3 mithilfe des Befehls **db2pd -tablespaces** abrufen. Die Beispielausgabe sieht wie folgt aus:

```
Tablespace 3 Configuration:
Address            Type Content PageSz ExtentSz Auto Prefetch BufID BufIDDisk FSC NumCntrs MaxStripe  LastConsecPg Name
0x0780000020B1B5A0 DMS  UsrTmp  4096   32       Yes  32       1     1         On  1        0          31           TEMPSPACE2

Tablespace 3 Statistics:
Address            TotalPgs   UsablePgs  UsedPgs    PndFreePgs FreePgs    HWM        State      MinRecTime NQuiescers
0x0780000020B1B5A0 5000       4960       1088       0          3872       1088       0x00000000 0          0

Tablespace 3 Autoresize Statistics:
Address            AS  AR  InitSize    IncSize     IIP MaxSize     LastResize               LRF
0x0780000020B1B5A0 No  No  0           0           No  0           None            No

Containers:
Address            ContainNum Type    TotalPgs   UseablePgs StripeSet  Container
0x0780000020B1DCC0 0          File    5000       4960       0          /home/db2inst1/tempspace2a
```

Die Spalte FreePgs zeigt, dass der Speicherbereich gefüllt wird. Wenn der Wert für freie Seiten kleiner wird, steht weniger Speicherbereich zur Verfügung. Beachten Sie auch, dass die Summe aus FreePgs und UsedPgs dem Wert für UsablePgs entspricht.

Auf der Basis dieser Informationen können Sie mit dem Befehl **db2pd -db sample -dyn** die dynamische SQL-Anweisung identifizieren, die die Tabelle TEMP1 verwendet.

```
Database Partition 0 -- Database SAMPLE -- Active -- Up 0 days 00:13:06

Dynamic Cache:
Current Memory Used           1022197
Total Heap Size               1271398
Cache Overflow Flag           0
Number of References          237
Number of Statement Inserts   32
Number of Statement Deletes   13
Number of Variation Inserts   21
Number of Statements          19

Dynamic SQL Statements:
Address          AnchID StmtUID  NumEnv  NumVar  NumRef  NumExe  Text
0x0000000220A08C40 78     1        2       2       3       2       declare global temporary table temp1 (c1 char(6)) not logged
0x0000000220A8D960 253    1        1       1       24      24      insert into session.temp1 values('TEST')
```

Jetzt können Sie die Informationen aus der vorangehenden Ausgabe zu der Ausgabe der Anwendungen zuordnen, um die Anwendung mit dem Befehl **db2pd -db sample -app** zu identifizieren.

```
Applications:
Address            AppHandl [nod-index] NumAgents  CoorPid Status
0x0000000200661840 501      [000-00501] 1          11246   UOW-Waiting

C-AnchID C-StmtUID  L-AnchID L-StmtUID  Appid
0        0          253      1          *LOCAL.db2inst1.050202160426
```

Sie können mit dem Wert für die Anker-ID (AnchID), der die dynamische SQL-Anweisung kennzeichnet, die zugehörige Anwendung identifizieren. Die Ergebnisse zeigen, dass der Wert für die letzte Anker-ID (L-AnchID) dem Wert für die Anker-ID (AnchID) entspricht. Die Ergebnisse zu der Ausführung von **db2pd** verwenden Sie für die nächste Ausführung von **db2pd**.

Die Ausgabe von **db2pd -agent** zeigt die Anzahl der von der Anwendung gelesenen Zeilen (Spalte Rowsread) und die Anzahl der von der Anwendung geschriebenen Zeilen (Spalte Rowswrtn) an. Diese Werte geben Ihnen, wie die folgende Beispielausgabe verdeutlicht, einen Eindruck davon, welche Aufgaben die Anwendung abgeschlossen hat und welche sie noch ausführen muss.

```
Address            AppHandl [nod-index] AgentPid  Priority  Type DBName
0x0000000200698080 501      [000-00501] 11246     0         Coord SAMPLE

State       ClientPid  Userid   ClientNm Rowsread   Rowswrtn   LkTmOt
Inst-Active 26377      db2inst1 db2bp    22         9588       NotSet
```

Sie können die Werte für AppHandl und AgentPid, die sich bei der Ausführung des Befehls **db2pd -agent** ergeben haben, zu den entsprechenden Werten für AppHandl und CoorPid zuordnen, die sich bei der Ausführung des Befehls **db2pd -app** ergeben haben.

Die Schritte sind geringfügig anders, wenn Sie vermuten, dass eine interne, temporäre Tabelle den Tabellenbereich auffüllt. Sie verwenden in diesem Fall jedoch weiterhin den Befehl **db2pd -tcbstats**, um Tabellen mit einer großen Anzahl an Einfügungen (Inserts) zu ermitteln. Es folgen Beispielinformationen zu einer impliziten temporären Tabelle:

```
TCB Table Information:
Address            TbspaceID TableID PartID MasterTbs MasterTab TableName         SchemaNm ObjClass DataSize   ...
0x0780000020CC0D30 1         2       n/a    1         2         TEMP (00001,00002) <30>
<JMC Temp     2470 ...
0x0780000020CC14B0 1         3       n/a    1         3         TEMP (00001,00003) <31>
<JMC Temp     2367 ...
0x0780000020CC21B0 1         4       n/a    1         4         TEMP (00001,00004) <30>
<JMC Temp     1872 ...

TCB Table Stats:
Address            TableName          Scans   UDI   PgReorgs   NoChgUpdts Reads   FscrUpdates Inserts ...
0x0780000020CC0D30 TEMP (00001,00002) 0       0     0          0          0       0           43219 ...
0x0780000020CC14B0 TEMP (00001,00003) 0       0     0          0          0       0           42485 ...
0x0780000020CC21B0 TEMP (00001,00004) 0       0     0          0          0       0           0     ...
```

In diesem Beispiel liegt eine große Anzahl an Einfügungen in Tabellen mit der Namenskonvention TEMP (TbspaceID, TableID) vor. Hierbei handelt es sich um implizite temporäre Tabellen. Die Werte in der Spalte SchemaNm folgen der Namenskonvention des Wertes für AppHandl verknüpft mit dem Wert für SchemaNm, sodass es möglich ist, die Anwendung zu identifizieren, die die Verarbeitung durchführt.

Sie können diese Informationen nun zu der Ausgabe zu **db2pd -tablespaces** zuordnen, um den belegten Speicherbereich für Tabellenbereich 1 zu ermitteln. Achten Sie bei der Tabellenbereichsstatistik in der folgenden Ausgabe auf die Beziehung zwischen den Werten UsedPgs und UsablePgs:

```
Tablespace Configuration:
Address            Id    Type Content PageSz ExtentSz Auto Prefetch BufID BufIDDisk FSC NumCntrs MaxStripe LastConsecPg Name
0x07800000203FB5A0 1     SMS  SysTmp  4096   32       Yes  320      1     1         On  10       0         31           TEMPSPACE1

Tablespace Statistics:
Address            Id    TotalPgs   UsablePgs  UsedPgs    PndFreePgs FreePgs    HWM    State      MinRecTime NQuiescers
0x07800000203FB5A0 1     6516       6516       6516       0          0          0      0x00000000 0          0

Tablespace Autoresize Statistics:
Address            Id    AS  AR  InitSize    IncSize     IIP MaxSize     LastResize                LRF
0x07800000203FB5A0 1     No  No  0           0           No  0           None        No

Containers:
...
```

Anschließend können Sie mithilfe des Befehls **db2pd -app** die Anwendungskennungen 30 und 31 identifizieren (da diese in der von **-tcbstats** generierten Ausgabe enthalten waren):

```
Applications:
Address            AppHandl [nod-index] NumAgents  CoorPid    Status          C-AnchID C-StmtUID  L-AnchID   L-StmtUID  Appid
0x07800000006FB880 31       [000-00031] 1          4784182    UOW-Waiting     0        0          107        1          *LOCAL.db2inst1.051215214142
0x07800000006F9CE0 30       [000-00030] 1          8966270    UOW-Executing   107      1          107        1          *LOCAL.db2inst1.051215214013
```

Ordnen Sie nun die Informationen aus der vorangehenden Ausgabe zu der Ausgabe für dynamisches SQL zu, indem Sie den Befehl **db2pd -dyn** ausführen:

```
Dynamic SQL Statements:
Address            AnchID StmtUID NumEnv NumVar NumRef NumExe Text
0x0780000020B296C0 107    1          1          1          43         43         select c1, c2 from test group by c1,c2
```

Beispiel 8: Recovery überwachen

Wenn Sie den Befehl **db2pd -recovery** ausführen, werden, wie die folgende Beispielausgabe verdeutlicht, in der Ausgabe mehrere Zähler angezeigt, mit denen Sie den Fortschritt der Recovery überprüfen können. Die Werte `Current Log` (aktuelles Protokoll) und `Current LSN` (aktuelle Protokollfolgenummer) geben die Protokollposition an. Der Wert für `CompletedWork` (abgeschlossene Arbeit) gibt die Menge der bis zu diesem Zeitpunkt abgeschlossenen Arbeit in Byte an.

```
Recovery:
Recovery Status     0x00000401
Current Log         S0000005.LOG
Current LSN         000002551BEA
Job Type            ROLLFORWARD RECOVERY
Job ID              7
Job Start Time      (1107380474) Wed Feb  2 16:41:14 2005
Job Description     Database Rollforward Recovery
Invoker Type        User
Total Phases        2
Current Phase       1

Progress:
Address       PhaseNum Description StartTime               CompletedWork  TotalWork
0x0000000200667160 1   Forward     Wed Feb  2 16:41:14 2005 2268098 bytes  Unknown
0x0000000200667258 2   Backward    NotStarted              0 bytes        Unknown
```

Beispiel 9: Menge der von einer Transaktion verwendeten Ressourcen ermitteln

Die Ausgabe zum Befehl **db2pd -transactions** enthält, wie die folgende Beispielausgabe verdeutlicht, die Anzahl der Sperren, die erste Protokollfolgenummer, die letzte Protokollfolgenummer, den verwendeten Speicherbereich sowie den reservierten Speicherplatz. Diese Informationen können für die Analyse des Verhaltens einer Transaktion von Nutzen sein.

```
Transactions:
Address            AppHandl [nod-index] TranHdl  Locks  State  Tflag
0x0000000022026D980 797      [000-00797] 2        108    WRITE  0x00000000
0x0000000022026E600 806      [000-00806] 3        157    WRITE  0x00000000
0x0000000022026F280 807      [000-00807] 4        90     WRITE  0x00000000

Tflag2     Firstlsn       Lastlsn        LogSpace   SpaceReserved
0x00000000 0x000001072262 0x0000010B2C8C 4518       95450
0x00000000 0x000001057574 0x0000010B3340 6576       139670
0x00000000 0x00000107CF0C 0x0000010B2FDE 3762       79266

TID            AxRegCnt   GXID
0x000000000451 1          0
0x0000000003E0 1          0
0x000000000472 1          0
```

Beispiel 10: Protokollbelegung überwachen

Mit dem Befehl **db2pd -logs** kann die Protokollbelegung für eine Datenbank überwacht werden. Mit dem Wert `Pages Written` (geschriebene Seiten) können Sie, wie die folgende Beispielausgabe verdeutlicht, ermitteln, ob die Protokollbelegung zunimmt:

```
Logs:
Current Log Number            0
Pages Written                 0
Cur Commit Disk Log Reads     0
Cur Commit Total Log Reads    0
Method 1 Archive Status       n/a
Method 1 Next Log to Archive  n/a
Method 1 First Failure        n/a
Method 2 Archive Status       n/a
Method 2 Next Log to Archive  n/a
```

```
Method 2 First Failure        n/a
Log Chain ID                  0
Current LSN                   0x0000000001F40010

Address            StartLSN       State      Size  Pages  Filename
0x00002B75E9E3D2D0 0000000001F40010 0x00000000 1000      1000      S0000000.LOG
0x00002B75E9E53D70 0000000002328010 0x00000000 1000      1000      S0000001.LOG
0x00002B75E9E545D0 0000000002710010 0x00000000 1000      1000      S0000002.LOG
```

Mithilfe dieser Ausgabe können Sie zwei Typen von Problemen erkennen:

- Wenn das letzte Protokollarchiv fehlschlägt, wird für `Archive Status` (Archivstatus) der Wert `Failure` (Fehler) angegeben. Wenn der Archivierungsfehler andauert, sodass keine Protokolle archiviert werden können, wird für `Archive Status` (Archivstatus) der Wert `First Failure` (Erstes Auftreten eines Fehlers) angegeben.
- Wenn die Protokollarchivierung sehr langsam abläuft, liegt der Wert für `Next Log to Archive` (nächstes zu archivierendes Protokoll) unter dem Wert für `Current Log Number` (aktuelle Protokollnummer). Wenn die Archivierung verlangsamt ist, reicht der Speicherplatz für die aktiven Protokolldateien möglicherweise nicht mehr aus, was wiederum dazu führen kann, dass in der Datenbank keine Datenänderungen mehr ausgeführt werden können.

Beispiel 11: Sysplex-Liste anzeigen

Abgesehen von der Verwendung des Befehls **db2pd -sysplex** (siehe Beispielausgabe) besteht die einzige Möglichkeit, die Sysplex-Liste zu dokumentieren, in der Verwendung eines DB2-Traces.

```
Sysplex List:
Alias:         HOST
Location Name: HOST1
Count:         1

IP Address      Port       Priority   Connections Status     PRDID
 1.2.34.56      400        1          0           0
```

Beispiel 12: Stack-Traces generieren

Sie können mit dem Befehl **db2pd -stack all** (Windows-Betriebssysteme) oder dem Befehl **-stack** (UNIX-Betriebssysteme) Stack-Traces für alle Prozesse in der aktuellen Datenbankpartition erzeugen. Wenn Sie vermuten, dass bei einem Prozess oder Thread eine Schleife oder Blockierung vorliegt, können Sie diesen Befehl iterativ verwenden.

Sie können den aktuellen Aufrufstack für eine bestimmte Engine-Dispatchable-Unit (EDU) abrufen, wie das folgende Beispiel verdeutlicht, indem Sie den Befehl **db2pd -stack** *edu-id* absetzen:

```
Attempting to dump stack trace for eduid 137.
  See current DIAGPATH for trapfile.
```

Um die Aufrufstacks für alle DB2-Prozesse abzurufen, verwenden Sie z. B. den Befehl **db2pd -stack all** (unter Windows-Betriebssystemen):

```
Attempting to dump all stack traces for instance.
See current DIAGPATH for trapfiles.
```

Wenn Sie eine Umgebung mit partitionierten Datenbanken und mehreren physischen Knoten verwenden, können Sie die Informationen von allen Partitionen mit dem folgenden Befehl abrufen: **db2_all "; db2pd -stack all"**. Wenn es sich je-

doch bei allen diesen Partitionen um logische Partitionen auf derselben Maschine handelt, ist die folgende Methode schneller: **db2pd -alldbp -stacks**.

Es ist auch möglich, die Ausgabe des Befehls **db2pdb -stacks** mit dem Parameter **dumpdir** in eine bestimmten Verzeichnispfad umzuleiten und mit dem Parameter **timeout** nur die Ausgabe für einen bestimmten Zeitraum umzuleiten. Beispiel: Geben Sie den folgenden Befehl ein, um die Ausgabe von Stack-Traces für alle Prozesse für einen Zeitraum von 30 Sekunden in /home/waleed/mydir umzuleiten:

```
db2pd -alldbp -stack all dumpdir=/home/waleed/mydir timeout=30
```

Beispiel 13: Speicherstatistik für eine Datenbankpartition anzeigen

Mit dem Befehl **db2pd -dbptnmem** wird die Menge an Speicher angezeigt, den der DB2-Server momentan belegt, sowie eine allgemeine Übersicht über die Serverbereiche, die diesen Speicher verwenden.

Das folgende Beispiel zeigt die Ausgabe des Befehls **db2pd -dbptnmem** auf einer AIX-Maschine:

```
Database Partition Memory Controller Statistics

Controller Automatic: Y
Memory Limit:   122931408 KB
Current usage:     651008 KB
HWM usage:         651008 KB
Cached memory:     231296 KB
```

Die Datenfelder und -Spalten sind im Folgenden beschrieben:

Controller Automatic
Gibt die Einstellung des Speichercontrollers an. Der Wert lautet "Y", wenn der Konfigurationsparameter **instance_memory** auf AUTOMATIC gesetzt ist. Dies bedeutet, dass der Datenbankmanager automatisch die Obergrenze für die Speicherbelegung bestimmt.

Memory Limit
Wenn eine Begrenzung des Instanzspeichers besteht, stellt der Wert des Konfigurationsparameters **instance_memory** die obere Begrenzung des DB2-Serverspeichers dar, der belegt werden kann.

Current usage
Die Speichermenge, die der Server momentan belegt.

HWM usage
Die obere Grenze (High Water Mark, HWM) bzw. der Höchstwert für die Speicherbelegung seit der Aktivierung der Datenbankpartition (seit der Ausführung des Befehls **db2start**).

Cached memory
Der Umfang der unter 'Current Usage' angegebenen Speichermenge, die momentan nicht verwendet wird, sondern aus Leistungsgründen für zukünftige Speicheranforderungen im Cache zwischengespeichert ist.

Es folgt die Fortsetzung zu der Beispielausgabe für den Befehl **db2pd -dbptnmem** bei einer Ausführung unter dem Betriebssystem AIX:

```
Individual Memory Consumers:
Name           Mem Used (KB)   HWM Used (KB)   Cached (KB)
============================================================
APPL-DBONE     160000          160000          159616
DBMS-name       38528           38528            3776
FMP_RESOURCES   22528           22528               0
```

```
PRIVATE         13120           13120             740
FCM_RESOURCES   10048           10048               0
LCL-p606416       128             128               0
DB-DBONE       406656          406656           67200
```

Alle registrierten Speicher„konsumenten" innerhalb des DB2-Servers sind zusammen mit der Menge des Gesamtspeichers aufgeführt, den sie belegen. Die Spalten sind im Folgenden beschrieben:

Name Ein kurzer, eindeutiger Name des Speicherkonsumenten, z. B.:

APPL-*datenbankname*
Für die Datenbank *datenbankname* verwendeter Anwendungsspeicher

DBMS-*name*
Globaler Speicherbedarf des Datenbankmanagers

FMP_RESOURCES
Für die Kommunikation mit **db2fmps** erforderlicher Speicher

PRIVATE
Verschiedene Speicheranforderungen für den privaten Speicher

FCM_RESOURCES
FCM-Ressourcen (Fast Communications Manager)

LCL-*prozess-id*
Speichersegment, das zur Kommunikation mit lokalen Anwendungen verwendet wird

DB-*datenbankname*
Für die Datenbank *datenbankname* verwendeter Datenbankspeicher

Mem Used (KB)
Die dem Konsumenten zurzeit zugeordnete Speichermenge

HWM Used (KB)
Die obere Grenze (High Water Mark, HWM) bzw. der Spitzenwert für den vom Konsumenten belegten Speicher

Cached (KB)
Anteil der unter 'Mem Used (KB)' angegebenen Speichermenge, die momentan nicht verwendet wird, jedoch für zukünftige Speicherzuordnungen sofort verfügbar ist.

Beispiel 14: Überwachen des Fortschritts bei der Indexreorganisation

Ab DB2 Version 9.7 Fixpack 2 weist der Fortschrittsbericht einer Indexreorganisation die folgenden Merkmale auf:

- Der Befehl **db2pd -reorgs index** dokumentiert den Indexreorganisationsprozess für partitionierte Indizes (in Fixpack 1 wurde die Unterstützung ausschließlich für nicht partitionierte Indizes eingeführt).
- Der Befehl **db2pd -reorgs index** unterstützt die Überwachung der Indexreorganisation auf Partitionsebene (d. h., während der Reorganisation einer einzelnen Partition).
- Der Reorganisationsprozess für nicht partitionierte Indizes und der Reorganisationsprozess für partitionierte Indizes werden in getrennten Ausgaben dokumentiert. Eine Ausgabe enthält die Informationen zum Reorganisationsprozess für nicht partitionierte Indizes; die nachfolgenden Ausgaben zeigen den Reorganisa-

tionsprozess für partitionierte Indizes in jeder Tabellenpartition. In jeder Ausgabe werden die Statistikdaten der Indexreorganisation für nur jeweils eine Partition dokumentiert.

- Nicht partitionierte Indizes werden zuerst verarbeitet; anschließend erfolgt die serielle Verarbeitung der partitionierten Indizes.
- Mit dem Befehl **db2pd -reorgs index** werden die folgenden zusätzlichen Informationsfelder in der Ausgabe für partitionierte Indizes angezeigt:
  - `MaxPartition` - Gesamtzahl der Partitonen für die verarbeitete Tabelle. Bei der Reorganisation auf Partitionsebene weist `MaxPartition` stets den Wert 1 auf, da nur eine einzige Partition reorganisiert wird.
  - `PartitionID` - Die Datenpartitions-ID für die verarbeitete Partition.

Im Folgenden ist ein Beispiel für eine Ausgabe dargestellt, die mit dem Befehl **db2pd -reorgs index** erstellt wurde. Hierbei wird der Fortschritt der Indexreorganisation für eine bereichspartitionierte Tabelle mit 2 Partitionen dokumentiert.

Anmerkung: Im ersten Teil der Ausgabe ist die Indexreorganisationstatistik des nicht partitionierten Index dokumentiert. Die nachfolgenden Ausgabeteile enthalten die Indexreorganisationsstatistik der partitionierten Indizes für die einzelnen Partitionen.

```
Index Reorg Stats:
Retrieval Time: 02/08/2010 23:04:21
TbspaceID: -6       TableID: -32768
Schema: ZORAN    TableName: BIGRPT
Access: Allow none
Status: Completed
Start Time: 02/08/2010 23:03:55   End Time: 02/08/2010 23:04:04
Total Duration: 00:00:08
Prev Index Duration:  -
Cur Index Start: -
Cur Index: 0            Max Index: 2            Index ID: 0
Cur Phase: 0          ( -     )   Max Phase: 0
Cur Count: 0                      Max Count: 0
Total Row Count: 750000

Retrieval Time: 02/08/2010 23:04:21
TbspaceID: 2        TableID: 5
Schema: ZORAN    TableName: BIGRPT
PartitionID: 0      MaxPartition: 2
Access: Allow none
Status: Completed
Start Time: 02/08/2010 23:04:04   End Time: 02/08/2010 23:04:08
Total Duration: 00:00:04
Prev Index Duration:  -
Cur Index Start: -
Cur Index: 0            Max Index: 2            Index ID: 0
Cur Phase: 0          ( -     )   Max Phase: 0
Cur Count: 0                      Max Count: 0
Total Row Count: 375000

Retrieval Time: 02/08/2010 23:04:21
TbspaceID: 2        TableID: 6
Schema: ZORAN    TableName: BIGRPT
PartitionID: 1      MaxPartition: 2
Access: Allow none
Status: Completed
Start Time: 02/08/2010 23:04:08   End Time: 02/08/2010 23:04:12
Total Duration: 00:00:04
Prev Index Duration:  -
Cur Index Start: -
```

```
Cur Index: 0            Max Index: 2            Index ID: 0
Cur Phase: 0          ( -     )   Max Phase: 0
Cur Count: 0                      Max Count: 0
Total Row Count: 375000
```

Beispiel 15: Anzeigen der obersten EDUs, sortiert nach benötigter Prozessorzeit, sowie Anzeigen von EDU-Stackinformationen

Wenn Sie den Befehl **db2pd** mit der Parameteroption **-edus** eingeben, werden in der Ausgabe alle EDUs (Engine Dispatchable Unit, von der Steuerkomponente zuteilbare Einheit) aufgeführt. Die Ausgabe für EDUs kann mit dem von Ihnen angegebenen Grad an Detailliertheit zurückgegeben werden, z. B. auf Instanz- oder Memberebene. Ausschließlich unter Linux- und UNIX-Betriebssystemen kann auch die Parameterunteroption **interval** angegeben werden, die bewirkt, dass zwei Momentaufnahmen aller EDUs im Abstand eines von Ihnen angegebenen Intervalls erstellt werden. Wenn der Parameter **interval** angegeben wird, enthalten zwei zusätzliche Spalten in der Ausgabe die Deltaprozessorbenutzerzeit (Spalte USR DELTA) und die Deltaprozessorsystemzeit (Spalte SYS DELTA) für das angegebene Intervall.

Im folgenden Beispiel werden die Deltawerte für die Prozessorbenutzerzeit und die Prozessorsystemzeit für ein Intervall von 5 Sekunden angegeben:

```
$ db2pd -edus interval=5

Database Partition 0 -- Active -- Up 0 days 00:53:29 -- Date 06/04/2010 03:34:59

List of all EDUs for database partition 0

db2sysc PID: 1249522
db2wdog PID: 2068678

EDU ID    TID            Kernel TID     EDU Name                 USR          SYS         USR DELTA    SYS DELTA
================================================================================================================================
6957      6957           13889683       db2agntdp (SAMPLE  ) 0              58.238506     0.820466    1.160726     0.014721
6700      6700           11542589       db2agent (SAMPLE) 0                 52.856696     0.754420    1.114821     0.015007
5675      5675           4559055        db2agntdp (SAMPLE  ) 0              60.386779     0.854234    0.609233     0.014304
3088      3088           13951225       db2agntdp (SAMPLE  ) 0              80.073489     2.249843    0.499766     0.006247
3615      3615           2887875        db2loggw (SAMPLE) 0                 0.939891      0.410493    0.011694     0.004204
4900      4900           6344925        db2pfchr (SAMPLE) 0                 1.748413      0.014378    0.014343     0.000103
7986      7986           13701145       db2agntdp (SAMPLE  ) 0              1.410225      0.025900    0.003636     0.000074
2571      2571           8503329        db2ipccm 0                          0.251349      0.083787    0.002551     0.000857
7729      7729           14168193       db2agntdp (SAMPLE  ) 0              1.717323      0.029477    0.000998     0.000038
7472      7472           11853991       db2agnta (SAMPLE) 0                 1.860115      0.032926    0.000860     0.000012
3358      3358           2347127        db2loggr (SAMPLE) 0                 0.151042      0.184726    0.000387     0.000458
515       515            13820091       db2aiothr 0                         0.405538      0.312007    0.000189     0.000178
7215      7215           2539753        db2agntdp (SAMPLE  ) 0              1.165350      0.019466    0.000291     0.000008
6185      6185           2322517        db2wlmd (SAMPLE) 0                  0.061674      0.034093    0.000169     0.000100
6442      6442           2756793        db2evmli (DB2DETAILDEADLOCK) 0      0.072142      0.052436    0.000092     0.000063
4129      4129           15900799       db2glock (SAMPLE) 0                 0.013239      0.000741    0.000064     0.000001
2         2              11739383       db2alarm 0                          0.036904      0.028367    0.000009     0.000009
4386      4386           13361367       db2dlock (SAMPLE) 0                 0.015653      0.001281    0.000014     0.000003
1029      1029           15040579       db2fcms 0                           0.041929      0.016598    0.000010     0.000004
5414      5414           14471309       db2pfchr (SAMPLE) 0                 0.000093      0.000002    0.000000     0.000000
258       258            13656311       db2sysc 0                           8.369967      0.263539    0.000000     0.000000
5157      5157           7934145        db2pfchr (SAMPLE) 0                 0.027598      0.000177    0.000000     0.000000
1543      1543           2670647        db2fcmr 0                           0.004191      0.000079    0.000000     0.000000
1286      1286           8417339        db2extev 0                          0.000312      0.000043    0.000000     0.000000
2314      2314           14360813       db2licc 0                           0.000371      0.000051    0.000000     0.000000
5928      5928           3137537        db2taskd (SAMPLE) 0                 0.004903      0.000572    0.000000     0.000000
3872      3872           2310357        db2lfr (SAMPLE) 0                   0.000126      0.000007    0.000000     0.000000
4643      4643           11694287       db2pclnr (SAMPLE) 0                 0.000094      0.000002    0.000000     0.000000
1800      1800           5800175        db2extev 0                          0.001212      0.002137    0.000000     0.000000
772       772            7925817        db2thcln 0                          0.000429      0.000072    0.000000     0.000000
2057      2057           6868993        db2pdbc 0                           0.002423      0.001603    0.000000     0.000000
2828      2828           10866809       db2resync 0                         0.016764      0.003098    0.000000     0.000000
```

Um nur Informationen zu den EDUs abzurufen, die am meisten Prozessorzeit in Anspruch nehmen, und um die Menge der zurückgegebenen Daten zu begrenzen, können Sie darüber hinaus die Parameteroption **top** angeben. Im folgenden Beispiel werden nur die ersten fünf EDU für einen Zeitraum von 5 Sekunden zurückgegeben. Stackinformationen werden ebenfalls zurückgegeben und separat in dem durch DUMPDIR angegebenen Verzeichnispfad gespeichert, standardmäßig **diagpath**.

```
$ db2pd -edus interval=5 top=5 stacks

Database Partition 0 -- Active -- Up 0 days 00:54:00 -- Date 06/04/2010 03:35:30

List of all EDUs for database partition 0

db2sysc PID: 1249522
db2wdog PID: 2068678

EDU ID    TID           Kernel TID     EDU Name                USR          SYS         USR DELTA    SYS DELTA
============================================================================================================
3358      3358          2347127        db2loggr (SAMPLE) 0     0.154906     0.189223    0.001087     0.001363
3615      3615          2887875        db2loggw (SAMPLE) 0     0.962744     0.419617    0.001779     0.000481
515       515           13820091       db2aiothr 0             0.408039     0.314045    0.000658     0.000543
258       258           13656311       db2sysc 0               8.371388     0.264812    0.000653     0.000474
6700      6700          11542589       db2agent (SAMPLE) 0     54.814420    0.783323    0.000455     0.000310


$ ls -ltr
total 552
drwxrwxr-t    2 vbmithun build         256 05-31 09:59 events/
drwxrwxr-t    2 vbmithun build         256 06-04 03:17 stmmlog/
-rw-r--r--    1 vbmithun build       46413 06-04 03:35 1249522.3358.000.stack.txt
-rw-r--r--    1 vbmithun build       22819 06-04 03:35 1249522.3615.000.stack.txt
-rw-r--r--    1 vbmithun build       20387 06-04 03:35 1249522.515.000.stack.txt
-rw-r--r--    1 vbmithun build       50426 06-04 03:35 1249522.258.000.stack.txt
-rw-r--r--    1 vbmithun build      314596 06-04 03:35 1249522.6700.000.stack.txt
-rw-r--r--    1 vbmithun build       94913 06-04 03:35 1249522.000.processObj.txt
```

Beispiel 16: Anzeigen der Messdaten für Agentenereignisse

Der Befehl **db2pd** unterstützt die Rückgabe von Ereignismessdaten für Agenten. Wenn Sie feststellen müssen, ob sich der Status eines Agenten während eines bestimmten Zeitraums geändert hat, dann verwenden Sie die Option event zusammen mit dem Parameter **-agents**. Die Spalte AGENT_STATE_LAST_UPDATE_TIME(Tick Value), die zurückgegeben wird, zeigt den letzten Zeitpunkt an, zu dem das vom Agenten verarbeitete Ereignis geändert wurde. Zusammen mit einem zuvor angeforderten Wert für AGENT_STATE_LAST_UPDATE_TIME(Tick Value) können Sie feststellen, ob ein Agent die Verarbeitung bei einer neuen Task fortgesetzt hat oder ob mit der Verarbeitung derselben Task über einen längeren Zeitraum hinweg fortgefahren wird.

```
db2pd –agents event
Database Partition 0 -- Active -- Up 0 days 03:18:52 -- Date 06/27/2011 11:47:10

Agents:
Current agents:      12
Idle agents:         0
Active coord agents: 10
Active agents total: 10
Pooled coord agents: 2
Pooled agents total: 2

AGENT_STATE_LAST_UPDATE_TIME(Tick Value)         EVENT_STATE  EVENT_TYPE  EVENT_OBJECT  EVENT_OBJECT_NAME
2011-06-27-14.44.38.859785(5622972377924968075)  IDLE         WAIT        REQUEST       n/a
```

Erfassen von Informationen zur Umgebung mit dem Befehl 'db2support'

Das wichtigste DB2-Dienstprogramm, das Sie bei der Erfassung von Informationen zu einem DB2-Problem bzw. -Fehler ausführen müssen, ist **db2support**. Das Dienstprogramm **db2support** dient zur automatischen Erfassung aller verfügbaren DB2- und Systemdiagnoseinformationen. Darüber hinaus bietet es eine optionale interaktive Frage-und-Antwort-Sitzung, in der Fragen zu den jeweiligen Umständen des aufgetretenen Problems gestellt werden.

Informationen zu diesem Vorgang

Durch die Verwendung des Dienstprogramms **db2support** können mögliche Benutzerfehler vermieden werden, da Befehle wie beispielsweise GET DATABASE CONFIGURATION FOR *datenbankname* oder **LIST TABLESPACES SHOW DETAIL** nicht manuell eingegeben werden müssen. Darüber hinaus nimmt das Erfassen von Daten weniger Zeit in Anspruch, da keine speziellen Anweisungen bezüglich der auszuführenden Befehle und der zu sammelnden Dateien erforderlich sind.

Vorgehensweise

- Führen Sie den Befehl **db2support -h** aus, um eine vollständige Liste der Befehlsoptionen aufzurufen.
- Erfassen Sie die Daten mithilfe des entsprechenden **db2support**-Befehls.

  db2support sollte von einem Benutzer mit SYSADM-Berechtigung ausgeführt werden, wie beispielsweise einem Instanzeigner, sodass das Dienstprogramm alle erforderlichen Informationen erfassen kann, ohne dass Fehler auftreten. Wenn **db2support** von einem Benutzer ohne die Berechtigung SYSADM ausgeführt wird, können SQL-Fehler (z. B. SQL1092N) auftreten, wenn das Dienstprogramm Befehle wie beispielsweise **QUERY CLIENT** oder **LIST ACTIVE DATABASES** ausführt.

  Wenn Sie das Dienstprogramm **db2support** verwenden, um IBM Software Support Informationen zu übermitteln, führen Sie den Befehl **db2support** aus, während auf dem System der betreffende Fehler auftritt. Auf diese Weise erfasst das Tool zeitnahe Informationen wie beispielsweise Daten zur Leistung des Betriebssystems. Wenn es nicht möglich ist, das Dienstprogramm zum Zeitpunkt des Fehlers auszuführen, können Sie den Befehl **db2support** auch nach Auftreten des Fehlers absetzen, da einige FODC-Diagnosedateien (FODC = First Occurence Data Capture) automatisch generiert werden.

  Wenn ein FODC-Paket in einem Verzeichnispfad gespeichert wird, bei dem es sich nicht um den Standarddiagnosepfad handelt, bzw. wenn es nicht in einem durch die FODCPATH-Einstellung angegebenen Pfad gespeichert wird, müssen Sie im Befehl **db2support** den FODC-Pfad mithilfe des Parameters **-fodcpath** angeben, damit das FODC-Paket in die Datei `db2support.zip` aufgenommen werden kann.

  Der folgende Basisaufruf ist normalerweise ausreichend zur Erfassung eines Großteils der Informationen, die für die Fehlerbehebung erforderlich sind, mit Ausnahme der Fälle, in denen der Pfad für ein FODC-Paket mit dem Parameter **-fodcpath** angegeben werden muss (wenn die Option **-c** verwendet wird, stellt das Dienstprogramm eine Verbindung zur Datenbank her):

  ```
  db2support <ausgabepfad> -d <datenbankname> -c
  ```

  Die so erfasste Ausgabe wird benutzerfreundlich in einem komprimierten Archiv (ZIP), `db2support.zip`, gespeichert und kann so an ein beliebiges System übertraten und dort extrahiert werden.

Ergebnisse

Die Art der von **db2support** erfassten Informationen hängt davon ab, mit welchen Optionen der Befehl aufgerufen wird, ob der Datenbankmanager gestartet ist oder nicht und ob eine Verbindung zur Datenbank hergestellt werden kann.

Das Dienstprogramm **db2support** erfasst in allen Fällen die folgenden Informationen:

- **db2diag**-Protokolldateien
- Alle Trapdateien
- Sperrenlistendateien
- Speicherauszugsdateien
- Verschiedene systembezogene Dateien
- Ausgabe von verschiedenen Systembefehlen
- `db2cli.ini`

In Abhängigkeit von den jeweiligen Bedingungen erfasst das Dienstprogramm **db2support** möglicherweise auch die folgenden Informationen:

- Aktive Protokolldateien
- Steuerdateien für Pufferpools und Tabellenbereiche (`SQLSPCS.1` und `SQLSPCS.2`) (mit Option **-d**)
- Inhalt des db2dump-Verzeichnisses
- Erweiterte Systeminformationen (mit Option **-s**)
- Datenbankkonfigurationseinstellungen (mit Option **-d**)
- Konfigurationseinstellungsdateien des Datenbankmanagers
- FODC-Informationen (mit den Optinen -fodc und -fodcpath)
- Datei mit Protokolldateikopfdaten (mit Option **-d**)
- Datei des Recoveryprotokolls (mit Option **-d**)
- Formatierte Daten für die Systemkatalogtabellen SYSIBM.SYSTABLES, SYSIBM-.SYSINDEXES und SYSIBM.SYSDATAPARTITIONS (mit Option `-d` und dem Tool **db2support** nicht im Optimierungsprogrammmodus)

Der HTML-Bericht `db2support.html` enthält stets die folgenden Informationen:

- PMR-Nummer (PMR - Problem Management Record) (bei Angabe von **-n**)
- Betriebssystem und Version (beispielsweise AIX 5.1)
- DB2-Release-Informationen
- Angabe des Umgebungstyps (32-Bit- oder 64-Bit-Umgebung)
- DB2-Installationspfadinformation
- Inhalt der Datei `db2nodes.cfg`
- Anzahl der CPUs und Platten sowie Menge des Speicherplatzes
- Liste der Datenbanken in der Instanz
- Registrierdatenbankinformationen und Umgebung, einschließlich PATH und LIB-PATH
- Freier Plattenspeicherplatz für aktuelles Dateisystem und I-Nodes für UNIX
- Java SDK-Version
- Java JCC-Version
- Java JCC-Konfiguration
- Datenbankmanagerkonfiguration

- Auflistung der Datei des Recoveryprotokolls für die Datenbank
- Ausgabe von **ls -lR** (oder Windows-Äquivalent) des Verzeichnisses 'sqllib'
- Ergebnis des Befehls **LIST NODE DIRECTORY**
- Ergebnis des Befehls **LIST ADMIN NODE DIRECTORY**
- Ergebnis des Befehls **LIST DCS DIRECTORY**
- Ergebnis des Befehls **LIST DCS APPLICATIONS EXTENDED**
- Liste der gesamten installierten Software

Die folgenden Informationen werden in der Datei db2support.html aufgezeichnet, wenn die Option **-s** angegeben wird:

- Ausführliche Datenträgerinformationen (Partitionsaufbau, Typ, LVM-Informationen etc.)
- Ausführliche Netzinformationen
- Kernelstatistikdaten
- Firmwareversionen
- Sonstige für das Betriebssystem spezifische Befehle

Wenn DB2 gestartet ist, enthält die Datei db2support.html die folgenden zusätzlichen Informationen:

- Clientverbindungsstatus
- Datenbank- und Datenbankmanagerkonfiguration (für die Datenbankkonfiguration ist die Option **-d** erforderlich)
- CLI-Konfiguration
- Informationen zum Speicherpool (Größe und Belegung). Bei Verwendung der Option **-d** werden sämtliche Daten erfasst.
- Ergebnis des Befehls **LIST ACTIVE DATABASES**
- Ergebnis des Befehls **LIST DCS APPLICATIONS**

Wenn die Option **-c** angegeben ist und eine Verbindung zur Datenbank erfolgreich hergestellt wurde, enthält die Datei db2support.html die folgenden Informationen:

- Anzahl der Benutzertabellen
- Ungefähre Größe der Datenbankdaten
- Momentaufnahme der Datenbank
- Momentaufnahme der Anwendung
- Pufferpoolinformationen
- Ergebnis des Befehls **LIST APPLICATIONS**
- Ergebnis des Befehls **LIST COMMAND OPTIONS**
- Ergebnis des Befehls **LIST DATABASE DIRECTORY**
- Ergebnis des Befehls **LIST INDOUBT TRANSACTIONS**
- Ergebnis des Befehls **LIST DATABASE PARTITION GROUPS**
- Ergebnis des Befehls **LIST DBPARTITIONNUMS**
- Ergebnis des Befehls **LIST ODBC DATA SOURCES**
- Ergebnis des Befehls **LIST PACKAGES/TABLES**
- Ergebnis des Befehls **LIST TABLESPACE CONTAINERS**
- Ergebnis des Befehls **LIST TABLESPACES**
- Ergebnis des Befehls **LIST DRDA IN DOUBT TRANSACTIONS**
- Informationen des DB2-Workload-Managers

Beispielinhalt der Datei 'db2support.zip'

Sie können den Befehl **db2support** mit dem Parameter **-unzip** verwenden, um den Inhalt der Datei db2support.zip lokal zu extrahieren, und dabei optional den Verzeichnispfad angeben, in dem der Inhalt extrahiert werden soll. Darüber hinaus können Sie die Option **-unzip** angeben, um den Inhalt archivierter Diagnosedaten zu extrahieren, ohne dass Sie zusätzliche Software benötigen. Wenn Sie lediglich die in der Datei db2support.zip enthaltenen Dateien auflisten möchten, ohne den eigentlichen Inhalt zu extrahieren, können Sie stattdessen den Parameter **-unzip list** im Befehl **db2support** verwenden.

Für den Beispielinhalt der Datei db2support.zip wurde der folgende Befehl ausgeführt:

```
db2support . -d sample -c -f -st "select * from staff"
```

Beim Extrahieren der Datei db2support.zip wurden die folgenden Dateien und Verzeichnisse erfasst:

- DB2CONFIG/ - Konfigurationsinformationen (z. B. Datenbank, Datenbankmanager, Pufferpool, CLI und Java Developer Kit)
- DB2DUMP/ - Inhalt der Datei db2diag.log für die letzten 3 Tage
- DB2MISC/ - Liste des Verzeichnisses sqllib
- DB2SNAP/ - Ausgabe der DB2-Befehle (z. B. **db2set**, **LIST TABLES**, **LIST INDOUBT TRANSACTIONS** und **LIST APPLICATIONS**)
- db2supp_opt.zip - Diagnoseinformationen für beim Optimierungsprogramm aufgetretene Fehler
- db2supp_system.zip - Betriebssysteminformationen
- db2support.html - In HTML-Abschnitten formatierte Diagnoseinformationen
- db2support.log - Diagnoseprotokollinformationen für die Datenerfassung mit **db2support**
- db2support_options.in - Befehlszeilenoptionen, die zum Starten der Erfassung durch **db2support** verwendet wurden

Informationen zum Optimierungsprogramm finden Sie in der Datei db2supp_opt.zip. Nach dem Extrahieren dieser Datei finden Sie folgende Verzeichnisse:

- OPTIMIZER/ - Diagnoseinformationen für beim Optimierungsprogramm aufgetretene Fehler
- OPTIMIZER/optimizer.log - Datei mit einem Protokoll aller Aktivitäten
- OPTIMIZER/CATALOGS - Alle Kataloge mit LOBs in den folgenden Unterverzeichnissen (nur generiert, wenn die LOB-Spalte in der Katalogtabelle nicht leer ist):
  - FUNCTIONS
  - INDEXES
  - NODEGROUPS
  - ROUTINES
  - SEQUENCES
  - TABLES
  - VIEWS
- OPTIMIZER/DB2DUMP - Ausgabe zu 'db2serv' (Ausgabedateien 'serv.*' und 'serv2.*')

Systeminformationen finden Sie in der Datei db2supp_system.zip. Nach dem Extrahieren dieser Datei finden Sie folgende Dateien und Verzeichnisse:

- DB2CONFIG/ - db2cli.ini (Dateien aus dem Verzeichnis ~/sqllib/cfg)
- DB2MISC/ - Datei DB2SYSTM (binär) etc.
- OSCONFIG/ - Verschiedene Dateien mit Betriebssysteminformationen (z. B. 'netstat', 'services', 'vfs', 'ulimit' und 'hosts')
- OSSNAP/ - Momentaufnahmen des Betriebssystems (z. B. 'iostat', 'netstat', 'uptime', 'vmstat' und 'ps_elf')
- SQLDBDIR/ - wichtige Pufferpoolmetadateien (~/sqllib/sqldbdir)
- SQLGWDIR/ - DCS-Verzeichnis (Dateien aus dem Verzeichnis ~/sqllib/sqlgwdir)
- SQLNODIR/ - Knotenverzeichnis (Dateien aus dem Verzeichnis ~/sqllib/sqlnodir)
- SPMLOG/ - Dateien aus dem Verzeichnis ~/sqllib/spmlog
- report.log - Protokoll der gesamten Erfassungsaktivitäten

DB2-Kopie überprüfen

Mit dem Befehl **db2val** wird sichergestellt, dass die DB2-Kopie ordnungsgemäß funktioniert.

Informationen zu diesem Vorgang

Das Tool **db2val** verifiziert die Kernfunktion einer DB2-Kopie, indem es Installationsdateien, Instanzen, die Erstellung einer Datenbank, Verbindungen zu dieser Datenbank und den Status von Umgebungen mit partitionierten Datenbanken auf Gültigkeit überprüft. Diese Überprüfung kann dann nützlich sein, wenn Sie eine DB2-Kopie unter Linux- und UNIX-Betriebssystemen mit tar.gz-Dateien manuell implementiert haben. Mit dem Befehl **db2val** kann auf rasche Weise sichergestellt werden, ob die gesamte Konfiguration ordnungsgemäß durchgeführt wurde und die DB2-Kopie Ihren Vorstellungen entspricht. Sie können Instanzen und Datenbanken angeben oder den Befehl **db2val** für alle Instanzen ausführen. Der Befehl **db2val** ist in den Verzeichnissen *DB2-installationspfad*\bin und sqllib/bin zu finden.

Beispiel

Wenn Sie beispielsweise alle Instanzen für die DB2-Kopie überprüfen möchten, müssen Sie den folgenden Befehl ausführen:

```
db2val -a
```

Ausführliche Details zum Befehl **db2val** sowie weitere Beispiele finden Sie im Abschnitt "db2val - Überprüfungstool für DB2-Kopien (Befehl)".

Grundlegende Tracediagnose

Wenn bei DB2 ein wiederholt auftretendes und reproduzierbares Problem vorkommt, können mit einem Trace bisweilen zusätzliche Informationen zu diesem Problem erfasst werden. Unter normalen Umständen sollten Sie einen Trace nur dann verwenden, wenn Sie von IBM Software Support dazu aufgefordert werden. Der Prozess der Durchführung eines Trace umfasst das Einstellen der Tracefunktion, das Reproduzieren des Fehlers und das Sammeln der Daten.

Die vom Trace erfassten Informationsmengen wachsen sehr schnell. Wenn Sie den Trace durchführen, erfassen Sie nur die Fehlersituation, und vermeiden Sie möglichst alle anderen Aktivitäten. Verwenden Sie dabei das kleinstmögliche Szenario, um den Fehler zu reproduzieren.

Die Traceerfassung wirkt sich häufig als Leistungsverschlechterung bei der DB2-Instanz aus. Der Grad der Leistungsverschlechterung hängt vom Problemtyp sowie der Anzahl der Ressourcen ab, die zum Zusammenstellen der Trace-Informationen verwendet werden.

Wenn Traces erforderlich sind, erhalten Sie von IBM Software Support in der Regel die folgenden Informationen:

- Erläuterungen zu einfachen, schrittweise auszuführenden Prozeduren
- Eine Erläuterung zur jeweiligen Position, an der die einzelnen Traces erstellt werden sollen
- Eine Erläuterung zum Gegenstand der Traces
- Eine Erläuterung zum Zweck der angeforderten Traces
- Erläuterungen zu Rücksetzungsverfahren (z. B. zum Inaktivieren aller Traces)

Sie erhalten von IBM Software Support zwar Informationen dazu, welche Traces erstellt werden sollen, es folgen an dieser Stelle dennoch einige zusätzliche allgemeine Richtlinien dazu, wann bestimmte Traces zu erstellen sind:

- Wenn der Fehler während der Installation auftritt und die Standardinstallationsprotokolle nicht ausreichen, um die Fehlerursache zu ermitteln, sollten Installationstraces erstellt werden.
- Wenn der Fehler in einem der GUI-Tools (GUI = Graphical User Interface, grafische Benutzerschnittstelle) auftritt, dieselben Aktionen jedoch bei der Ausführung über explizite Befehle im DB2-Befehlsfenster erfolgreich sind, sollte ein Trace der Steuerzentrale erstellt werden. Beachten Sie, dass hierdurch nur Fehler bei den Tools erfasst werden, die über die Steuerzentrale gestartet werden können.
- Wenn der Fehler in einer CLI-Anwendung auftritt und nicht außerhalb der Anwendung reproduziert werden kann, sollte ein CLI-Trace erstellt werden.
- Wenn der Fehler in einer JDBC-Anwendung auftritt und nicht außerhalb der Anwendung reproduziert werden kann, sollte ein JDBC-Trace erstellt werden.
- Wenn sich der Fehler unmittelbar auf Informationen bezieht, die auf DRDA-Ebene übertragen werden, sollte ein DRDA-Trace erstellt werden.
- In allen anderen Situationen, in denen ein Trace durchführbar ist, ist ein DB2-Trace in der Regel am besten geeignet.

Trace-Informationen sind nicht immer für die Diagnose eines Fehlers hilfreich. So wird in den folgenden Situationen die Fehlerbedingung möglicherweise nicht erfasst:

- Die von Ihnen angegebene Tracepuffergröße war nicht groß genug, einen vollständigen Satz von Trace-Ereignissen aufzunehmen, sodass nützliche Informationen verloren gegangen sind, als der Trace das Schreiben in die Datei beendet hat oder ein Umlauf stattgefunden hat.
- Das Traceszenario hat die Fehlersituation nicht erneut produziert.
- Die Fehlersituation wurde reproduziert, doch die Annahme, wo der Fehler aufgetreten ist, war nicht korrekt. Beispielsweise wurde der Trace auf einer Client-Workstation erfasst, während der tatsächliche Fehler auf einem Server auftrat.

DB2-Traces

Abrufen eines DB2-Trace mit 'db2trc'

Der Befehl **db2trc** steuert die mit DB2 zur Verfügung gestellte Tracefunktion. Mit der Tracefunktion können Informationen zu Operationen aufgezeichnet und in ein lesbares Format konvertiert werden.

Es ist zu beachten, dass während der Durchführung eines Trace zusätzlicher Systemaufwand anfällt. Demnach kann die Aktivierung der Tracefunktion die Systemleistung beeinträchtigen.

Im Allgemeinen verwenden IBM Software Support und IBM Entwicklungsteams DB2-Traces für die Fehlerbehebung. Sie können einen Trace durchführen, um Informationen zu einem Problem zu erhalten, das untersucht wird. Allerdings ist der Nutzen eines solchen Trace ohne Kenntnisse des DB2-Quellcodes relativ begrenzt.

Dennoch sollten Sie mit der korrekten Aktivierung der Tracefunktion vertraut sein und wissen, wie Speicherauszüge für Tracedateien erstellt werden, falls Sie gebeten werden, diese abzurufen.

Anmerkung: Sie benötigen eine der Berechtigungen SYSADM, SYSCTRL oder SYSMAINT, um **db2trc** zu verwenden.

Führen Sie den Befehl **db2trc** ohne Parameter aus, um eine Übersicht über die verfügbaren Optionen zu erhalten:

```
C:\>db2trc
Syntax: db2trc (chg|clr|dmp|flw|fmt|inf|off|on) optionen
```

Weitere Informationen zu einem bestimmten Parameter des Befehls **db2trc** erhalten Sie, wenn Sie die Option `-u` verwenden. Führen Sie beispielsweise den folgenden Befehl aus, um weitere Informationen zur Aktivierung der Tracefunktion zu erhalten:

```
db2trc on -u
```

Hierdurch werden Informationen zu allen zusätzlichen Optionen (als 'facilities' bezeichnet) aufgerufen, die bei der Aktivierung der DB2-Tracefunktion angegeben werden können.

Beim Einschalten der Tracefunktion ist die wichtigste Option `-L`. Sie gibt die Größe des Speicherpuffers an, der zum Speichern der Trace-Informationen verwendet wird. Die Puffergröße kann in Byte oder MB angegeben werden. (Für die Angabe von Megabyte hängen Sie an den Wert 'M' oder 'm' an.) Die Tracepuffergröße muss eine Potenz von 2 MB sein. Wenn Sie eine Größe angeben, die diese Voraussetzungen nicht erfüllt, wird die Puffergröße automatisch auf die nächste Potenz von 2 abgerundet.

Wenn der Puffer zu klein ist, können Informationen verloren gehen. Standardmäßig werden nur die neuesten Trace-Informationen gespeichert, wenn der Puffer voll ist. Ist der Puffer zu groß, können beim Senden der Datei an die Mitarbeiter von IBM Software Support Probleme auftreten.

Wenn Sie einen Trace für eine Operation erstellen, die relativ kurz ist (wie beispielsweise eine Datenbankverbindung), reicht eine Größe von ca. 8 MB normalerweise aus:

```
C:\> db2trc on -l 8M Trace is turned on
```

Wenn Sie jedoch einen Trace für eine größere Operation erstellen oder wenn viele Aktionen gleichzeitig stattfinden, ist möglicherweise ein größerer Tracepuffer erforderlich.

Auf den meisten Plattformen kann die Tracefunktion zu jedem beliebigen Zeitpunkt aktiviert werden und funktioniert wie oben beschrieben. Bitte beachten Sie jedoch die folgenden Situationen:

1. Auf Systemen mit mehreren Datenbankpartitionen müssen Sie einen Trace für jede physische (im Gegensatz zur logischen) Datenbankpartition ausführen.
2. Wenn auf HP-UX-, Linux- und Solaris-Plattformen die Tracefunktion inaktiviert wird, nachdem die Instanz gestartet wurde, wird unabhängig von der angegebenen Größe beim nächsten Start der Tracefunktion ein sehr kleiner Puffer verwendet. Ein Beispiel: Gestern haben Sie die Tracefunktion mit **db2trc on -l 8m** eingeschaltet, einen Trace erfasst und anschließend die Tracefunktion ausgeschaltet (**db2trc off**). Heute möchten Sie einen Trace mit einem Hauptspeicherpuffer von 32 MB durchführen (**db2trc on -l 32m**), ohne die Instanz herunterzufahren und neu zu starten. In diesem Fall wird der Trace nur mit einem kleinen Puffer durchgeführt werden. Um einen Trace auf diesen Plattformen effektiv durchzuführen, müssen Sie die Tracefunktion mit der benötigten Puffergröße einschalten, bevor Sie die Instanz starten und den Inhalt des Puffers gegebenenfalls zu einem späteren Zeitpunkt „löschen".

Um die Menge der erfassten bzw. formatierten Daten zu reduzieren, unterstützt der Befehl **db2trc** mehrere Maskenoptionen. Die Reduzierung der erfassten Datenmenge ist sinnvoll, da hierdurch auch der Aufwand reduziert wird, der durch eine fortlaufende Traceerfassung anfällt, und da die Daten so selektiver erfasst werden können. Eine selektivere Datenerfassung ermöglicht auch eine schnellere Problemdiagnose.

Normalerweise wird die Maskenoption **-m** unter Anleitung von IBM Support angewendet. Sie können jedoch die Maskenoption **-p** verwenden, um einen Trace ausschließlich für bestimmte Prozess-IDs (und wahlweise auch Thread-IDs) zu erstellen. Beispiel: Für die Traceerstellung für Prozess 77 mit den Threads 1, 2, 3 und 4 sowie für Prozess 88 mit den Threads 5, 6, 7 und 8 lautet die Syntax wie folgt:

```
db2trc on -p 77.1.2.3.4,88.5.6.7.8
```

Verwenden der Scripts 'trcon' und 'troff' für die Steuerung der Traceerstellung

Es stehen zwei Scripts zur Verfügung, mit denen die Traceerstellung vereinfacht werden kann, indem mehrere manuell abgesetzte Befehle durch einen einzelnen Scriptaufruf ersetzt werden.

Das Script **db2trcon** aktiviert die Traceerstellung und unterstützt mehrere Optionen. Mit diesem Script können Sie 'db2trc' für einen bestimmten Zeitraum aktivieren, angeben, dass die Tracedaten nur für die EDUs (Engine Dispatchable Units) erfasst werden sollen, die am meisten Prozessorzeit verbrauchen, und Speicherauszugs-, Ablauf- und Formatdateien automatisch generieren. Beispiel: Geben Sie den folgenden Befehl ein, um die Traceerstellung für einen Zeitraum von 45 Sekunden für die 5 am meisten Prozessorzeit verbrauchenden EDUs zu aktivieren, wobei die Daten in 15-Sekunden-Intervallen vom Zeitpunkt der Scriptausführung an erfasst werden sollen:

```
db2trcon -duration 45 -top 5 -interval 15 -flw -fmt
```

Wenn 'db2trc' nach dem angegebenen Zeitraum inaktiviert wird, generiert 'db2trcon' automatisch die Speicherauszugs-, Ablauf-, und Formatdateien.

db2trcoff inaktiviert die Traceerstellung und generiert bei Bedarf Speicherauszugs-, Ablauf-, und Formatdateien automatisch mit einem einzigen Befehl. Beispiel: Geben Sie den folgenden Befehl ein, um 'db2trc' mit der Option '-force' zu inaktivieren und die Ablauf-, Format- und Speicherauszugsdateien zu generieren:

```
db2trcoff -flw -fmt -force
```

Beachten Sie Folgendes: Wenn Sie die Traceerstellung mit dem Script 'db2trcon' aktiviert und dabei einen Zeitraum für die Dauer angegeben haben, müssen Sie den Befehl 'db2troff' nicht separat eingeben.

Erstellen eines Speicherauszugs einer DB2-Tracedatei

Nachdem die Tracefunktion mit der Option `ON` aktiviert wurde, wird für die gesamte nachfolgende Arbeit, die von dieser Instanz ausgeführt wird, ein Trace erstellt.

Während die Tracefunktion aktiv ist, können Sie mit der Option `clr` den Tracepuffer bereinigen. Alle im Tracepuffer vorhandenen Informationen werden entfernt.

```
C:\>db2trc clr
Trace has been cleared
```

Wenn die Operation, für die ein Trace erstellt wird, abgeschlossen ist, können Sie die Option `dmp` gefolgt von einem Tracedateinamen verwenden, um für den Speicherpuffer einen Speicherauszug auf der Platte zu erstellen. Beispiel:

```
C:\>db2trc dmp trace.dmp
Trace has been dumped to file
```

Nachdem für den Tracepuffer ein Speicherauszug auf der Platte erstellt wurde, ist die Tracefunktion weiterhin aktiv. Verwenden Sie die Option `OFF`, um die Tracefunktion zu inaktivieren:

```
C:\>db2trc off
Trace is turned off
```

Formatieren einer DB2-Tracedatei

Die mit dem Befehl **db2trc dmp** erstellte Speicherauszugsdatei liegt in Binärformat vor und ist nicht lesbar. Prüfen sie, ob eine Tracedatei gelesen werden kann, indem Sie die binäre Tracedatei formatieren, damit die Fluss-Steuerung angezeigt wird, und senden Sie die formatierte Ausgabe an eine Nulleinheit.

Das folgende Beispiel zeigt den Befehl für die Ausführung dieser Task:

```
db2trc flw example.trc nul
```

Hierbei ist `example.trc` eine Binärdatei, die mit der Option **dmp** erstellt wurde.

Die Ausgabe dieses Befehls gibt Ihnen explizit an, wenn ein Problem beim Lesen der Datei besteht oder ob beim Trace ein Umlauf stattgefunden hat.

Zu diesem Zeitpunkt kann die Speicherauszugsdatei an IBM Software Support gesendet werden. Dort wird sie auf der Basis der verwendeten DB2-Servicestufe formatiert. In manchen Fällen werden Sie jedoch dazu aufgefordert, die Speicherauszugsdatei in ASCII-Format zu konvertieren, bevor Sie sie senden. Hierzu verwenden Sie die Optionen **flw** und **fmt**. Sie müssen den Namen der binären Speicherauszugsdatei und den Namen der zu erstellenden ASCII-Datei angeben:

```
C:\>db2trc flw trace.dmp trace.flw
C:\Temp>db2trc flw trace.dmp trace.flw
Total number of trace records     : 18854
Trace truncated                   : NO
Trace wrapped                     : NO
Number of trace records formatted : 1513 (pid: 2196 tid 2148 node: -1)
Number of trace records formatted : 100 (pid: 1568 tid 1304 node: 0)
...

C:\>db2trc fmt trace.dmp trace.fmt
C:\Temp>db2trc fmt trace.dmp trace.fmt
Trace truncated                   : NO
Trace wrapped                     : NO
Total number of trace records     : 18854
Number of trace records formatted : 18854
```

Wenn diese Ausgabe für "Trace wrapped" die Angabe "YES" enthält, bedeutet dies, dass der Tracepuffer nicht ausreichte, um alle während des Tracezeitraums erfassten Daten aufzuzeichnen. Abhängig von der Situation kann ein solcher Trace akzeptabel sein. Wenn Sie die neuesten Informationen benötigen (dies sind die Informationen, die standardmäßig aufbewahrt werden, es sei denn, die Option **-i** wird angegeben), ist der Inhalt der Tracedatei möglicherweise ausreichend. Wenn Sie jedoch Informationen zu den Ereignissen zu Beginn des Tracezeitraums oder Informationen zu allen Ereignissen benötigen, sollten Sie die Operation mit einem größeren Tracepuffer wiederholen.

Bei der Formatierung einer Binärdatei in eine lesbare Textdatei stehen Ihnen Optionen zur Verfügung. Sie können zum Beispiel `db2trc fmt -xml trace.dmp trace.fmt` verwenden, um die Binärdaten zu konvertieren und das Ergebnis in einem Format auszugeben, das durch XML geparst werden kann. Weitere Optionen werden in der ausführlichen Beschreibung der Tracebefehle (**db2trc**) dargestellt.

Darüber hinaus ist zu beachten, dass DB2 unter Linux- und UNIX-Betriebssystemen automatisch einen Speicherauszug des Tracepuffers auf der Platte erstellt, wenn es die Instanz aufgrund eines schwer wiegenden Fehlers beendet. Wenn also bei der abnormalen Beendigung einer Instanz die Tracefunktion aktiviert ist, wird eine Datei im Diagnoseverzeichnis erstellt, die den Namen `db2trdmp.###` hat, wobei ### die Nummer der Datenbankpartition angibt. Auf Windows-Plattformen findet dies nicht statt. In diesen Fällen muss ein Speicherauszug für den Trace manuell erstellt werden.

Abschließend folgt als Zusammenfassung ein Beispiel der üblichen Folge von **db2trc**-Befehlen:

```
db2trc on -l 8M
db2trc clr
<Befehle zur Fehlerreproduktion ausführen>
db2trc dump db2trc.dmp
db2trc off
db2trc flw db2trc.dmp <dateiname>.flw
db2trc fmt db2trc.dmp <dateiname>.fmt
db2trc fmt -c db2trc.dmp <dateiname>.fmtc
```

DRDA-Tracedateien

Vor der Analyse von DRDA-Traces sollten Sie sich darüber im Klaren sein, dass es sich bei DRDA um einen offenen Standard für die Definition von Daten- und Kommunikationsstrukturen handelt. So umfasst DRDA beispielsweise eine Reihe von Regeln zur Strukturierung von Daten für die Übertragung sowie zur Durchführung dieser Datenübertragung.

Diese Regeln sind in den folgenden Referenzhandbüchern definiert:

- DRDA V3 Vol. 1: Distributed Relational Database Architecture
- DRDA V3 Vol. 2: Formatted Data Object Content Architecture
- DRDA V3 Vol. 3: Distributed Data Management Architecture

PDF-Versionen dieser Handbücher sind unter www.opengroup.org verfügbar.

Das Dienstprogramm **db2drdat** zeichnet den Datenaustausch zwischen einem DRDA-Anwendungsrequester (AR) und einem DB2 DRDA-Anwendungsserver (AS) auf (zum Beispiel zwischen DB2 Connect und einem Host oder einem Datenbankserver aus der Produktreihe Power Systems).

Dienstprogramm für Trace

Das Dienstprogramm **db2drdat** zeichnet die Daten auf, die zwischen dem DB2 Connect-Server (für den IBM Data Server-Client) und dem IBM Mainframe-Datenbankserver ausgetauscht werden.

Die Kenntnis dieses Datenstroms ist für den Datenbankadministrator und den Anwendungsentwickler oft sehr hilfreich, da anhand dieses Wissens die Ursachen bestimmter Fehler gefunden werden können. Nehmen Sie die folgende Situation als Beispiel: Eine Datenbankanweisung `CONNECT TO` für einen IBM Mainframe-Datenbankserver wird abgesetzt, der Befehl schlägt jedoch fehl und Sie erhalten einen Rückkehrcode, der auf einen Fehler hinweist. Wenn genau bekannt ist, welche Informationen an das Verwaltungssystem des IBM Mainframe-Datenbankservers übertragen wurden, kann die Fehlerursache auch dann ermittelt werden, wenn die Informationen des Rückkehrcodes allgemein sind. Häufig schlägt ein Befehl aufgrund eines einfachen Benutzerfehlers fehl.

In der Ausgabe von `db2drdat` werden die zwischen der DB2 Connect-Workstation und dem Verwaltungssystem des IBM Mainframe-Datenbankservers ausgetauschten Datenströme aufgelistet. An den IBM Mainframe-Datenbankserver übertragene Daten werden unter SEND BUFFER (Sendepuffer), vom IBM Mainframe-Datenbankserver empfangene Daten unter RECEIVE BUFFER (Empfangspuffer) aufgeführt.

Wenn ein Empfangspuffer Informationen zum SQL-Kommunikationsbereich enthält, folgt auf diese eine formatierte Interpretation dieser Daten unter der Bezeichnung SQLCA. Das SQLCODE-Feld eines SQL-Kommunikationsbereichs ist der *nicht zugeordnete* Wert, so wie er vom IBM Mainframe-Datenbankserver zurückgegeben wurde. Die Sende- und Empfangspuffer sind von den ältesten zu den neuesten innerhalb der Datei sortiert. Jeder Puffer verfügt über folgende Angaben:

- Die Prozess-ID
- Eine Bezeichnung SEND BUFFER, RECEIVE BUFFER oder SQLCA. Der erste DDM-Befehl oder das erste DDM-Objekt in einem Puffer wird als DSS TYPE bezeichnet.

Die weiteren Daten in Sende- und Empfangspuffern werden in den folgenden fünf Spalten dargestellt:

- Die Byteanzahl
- Spalte 2 und 3 stellen den zwischen den beiden Systemen ausgetauschten DRDA-Datenstrom in ASCII oder EBCDIC dar.
- Eine ASCII-Darstellung der Spalten 2 und 3
- Eine EBCDIC-Darstellung der Spalten 2 und 3

Traceausgabe

Das Dienstprogramm **db2drdat** schreibt die folgenden Informationen in die *Tracedatei*:

- -r
  - Art der/des DRDA-Antwort/Objekts
  - Empfangspuffer
- -s
  - Art der DRDA-Anforderung
  - Sendepuffer
- -c
  - SQLCA
- TCP/IP-Fehlerinformationen
  - Rückkehrcode der Empfangsfunktion
  - Bewertung
  - Verwendetes Protokoll
  - Verwendete API
  - Funktion
  - Fehlernummer

Anmerkung:

1. Der Wert null für den Endecode zeigt an, dass der Befehl erfolgreich ausgeführt wurde. Ein Wert ungleich null zeigt an, dass der Befehl nicht erfolgreich ausgeführt wurde.
2. Die zurückgegebenen Felder hängen von der verwendeten API ab.
3. Welche Felder zurückgegeben werden, hängt von der Plattform ab, auf der DB2 Connect ausgeführt wird. Es können daher für dieselbe API unterschiedliche Felder zurückgegeben werden.
4. Wenn der Befehl **db2drdat** die Ausgabe an eine bereits existierende Datei leitet, wird die alte Datei gelöscht, sofern die Berechtigungen für die Datei dies zulassen.

Analyse der Traceausgabedatei

Die folgenden Informationen werden bei einem **db2drdat**-Trace erfasst:

- Die Prozess-ID (PID) der Client-Anwendung
- Der RDB_NAME, der im DCS-Verzeichnis katalogisiert ist
- Die ID(s) für den codierten Zeichensatz von DB2 Connect
- Die CCSID(s) für IBM Mainframe-Datenbankserver
- Das Verwaltungssystem des IBM Mainframe-Datenbankservers, mit dem das DB2 Connect-System kommuniziert

Der erste Puffer enthält die Befehle EXCSAT (Exchange Server Attributes) und ACCRDB (Access RDB), die an das Verwaltungssystem des IBM Mainframe-Datenbankservers gesendet werden. Diese Befehle werden als Ergebnis des Datenbankbefehls `CONNECT TO` gesendet. Der nächste Puffer enthält die Antwort, die DB2 Connect vom Verwaltungssystem des IBM Mainframe-Datenbankservers empfangen hat. Sie enthält EXCSATRD-Daten (Exchange Server Attributes Reply Data) und eine ACCRDBRM-Nachricht (Access RDB Reply Message).

EXCSAT

Der Befehl **EXCSAT** enthält den Workstationnamen des Clients, der vom Ob-

jekt SRVNAM (Servername) angegeben wird; dieser Name entspricht gemäß der DDM-Spezifikation dem Codepunkt X'116D'. Der Befehl **EXCSAT** befindet sich im ersten Puffer. Im Befehl **EXCSAT** werden die Werte X'9481A292' (die in CCSID 500 codiert sind) in *mask* umgesetzt, sobald X'116D' entfernt ist.

Der Befehl **EXCSAT** enthält außerdem das Objekt EXTNAM (externer Name), das oft in Diagnoseinformationen im IBM Mainframe-Datenbankverwaltungssystem zu finden ist. Es besteht aus einer 20 Byte langen Anwendungs-ID, gefolgt von einer 8 Byte langen Prozess-ID (oder einer 4 Byte langen Prozess-ID und einer 4 Byte langen Thread-ID). Es wird durch den Codepunkt X'115E' dargestellt und hat in diesem Beispiel den Wert `db2bp`, der durch Leerzeichen aufgefüllt ist und an den sich `000C50CC` anschließt. Auf einem IBM Data Server-Client unter Linux oder UNIX kann dieser Wert mit dem Befehl **ps** korreliert werden, der Prozessstatusinformationen zu aktiven Prozessen an die Standardausgabe übergibt.

ACCRDB

Der Befehl **ACCRDB** enthält den RDB_NAME im Objekt RDBNAM (Codepunkt X'2110'). Der Befehl **ACCRDB** folgt auf den Befehl **EXCSAT** im ersten Puffer. Im Befehl **ACCRDB** werden die Werte X'E2E3D3C5C3F1' in STLEC1 umgesetzt, sobald X'2110' entfernt ist. Dies entspricht dem Feld für den Zieldatenbanknamen im DCS-Verzeichnis.

Die Abrechnungszeichenfolge hat den Codepunkt X'2104'.

Der codierte Zeichensatz für die DB2 Connect-Workstation kann ermittelt werden, indem das CCSID-Objekt CCSIDSBC (CCSID für Einzelbytezeichen) mit Codepunkt X'119C' im **ACCRDB** angegeben wird. In diesem Beispiel ist der Wert für CCSIDSBC X'0333', d. h. 819.

Die zusätzlichen Objekte CCSIDDBC (CCSID für Doppelbytezeichen) und CCSIDMBC (CCSID für Mischbytezeichen) mit dem Codepunkt X'119D' und X'119E' sind im Befehl **ACCRDB** ebenfalls vorhanden. In diesem Beispiel ist der Wert für CCSIDDBC X'04B0', d. h. 1200. Der Wert für CCSIDMBC ist X'0333', d. h. 819.

EXCSATRD und ACCRDBRM

CCSID-Werte werden auch vom IBM Mainframe-Datenbankserver in der ACCRDBRM-Nachricht (Access RDB Reply Message) im zweiten Puffer zurückgegeben. Dieser Puffer enthält die EXCSATRD-Daten, gefolgt von den ACCRDBRM-Daten. Die Beispielausgabedatei enthält zwei CCSID-Werte für das IBM Mainframe-Datenbankserversystem. Die Werte sind 1208 (für Einzelbyte- und Mischbytezeichen) bzw. 1200 (für Doppelbytezeichen).

Wenn DB2 Connect die Codepage, die vom IBM Mainframe-Datenbankserver zurückgegeben wird, nicht erkennt, wird SQLCODE -332 mit der Quellen- und Zielcodepage an den Benutzer zurückgegeben. Wenn der IBM Mainframe-Datenbankserver den von DB2 Connect gesendeten codierten Zeichensatz nicht erkennt, gibt er VALNSPRM (Parameterwert nicht unterstützt, DDM-Codepunkt X'1252') zurück. Diese Angaben werden für den Benutzer in SQLCODE -332 umgesetzt.

Der Befehl ACCRDBRM enthält auch den Parameter PRDID (produktspezifische Kennung, Codepunkt X'112E'). Der Wert ist X'C4E2D5F0F8F0F1F5', d. h. DSN08015 in EBCDIC. Gemäß den Standards entspricht DSN der Angabe DB2 for z/OS. Die Versionsnummer ist ebenfalls angegeben. ARI bezeichnet DB2 Server für VSE & VM, SQL bezeichnet die DB2-Datenbank oder DB2 Connect und QSQ bezeichnet IBM DB2 for IBM i.

Beispiele für die Traceausgabedatei

Die folgenden Abbildungen zeigen ein Beispiel einer Ausgabe, das einige DRDA-Datenströme darstellt, die zwischen DB2 Connect-Workstations und einem Host- oder System i-Datenbankserver ausgetauscht werden. Vom Standpunkt des Benutzers wurde über den Befehlszeilenprozessor (CLP) ein Befehl `CONNECT TO` für eine Datenbank abgesetzt.

Abb. 35 auf Seite 535 verwendet DB2 Connect Enterprise Edition Version 9.1 und DB2 for z/OS Version 8 über eine TCP/IP-Verbindung.

```
1 data DB2 UDB DRDA Communication Manager sqljcSend fnc (3.3.54.5.0.100)
 pid 807116 tid 1 cpid -1 node 0 sec 0 nsec 0 probe 100
 bytes 16

 Data1  (PD_TYPE_UINT,8) unsigned integer:
 233

2 data DB2 UDB DRDA Communication Manager sqljcSend fnc (3.3.54.5.0.1177)
 pid 807116 tid 1 cpid -1 node 0 sec 0 nsec 19532 probe 1177
 bytes 250

 SEND BUFFER(AR):

           EXCSAT RQSDSS                    (ASCII)           (EBCDIC)
          0 1 2 3 4 5 6 7  8 9 A B C D E F   0123456789ABCDEF  0123456789ABCDEF
  0000  00C3D041000100BD 1041007F115E8482   ...A.....A...^..  .C}........".;db
    0010  F282974040404040 4040404040404040   ...@@@@@@@@@@@@  2bp
  0020  4040F0F0F0C3F5F0 C3C3F0F0F0000000   @@............    000C50CC000...
    0030  0000000000000000 0000000000000000   ................  ................
    0040  0000000000000000 000000000060F0F0   ............`..  ............-00
    0050  F0F1A2A495404040 4040404040404040   .....@@@@@@@@@@  01sun
  0060  4040404040404040 4040404040404040   @@@@@@@@@@@@@@@@
  0070  C4C5C3E5F8404040 F0A2A49540404040   .....@@@....@@@@  DECV8   0sun
  0080  4040404040404040 4000181404140300   @@@@@@@@.......           .......
    0090  0724070008147400 05240F0008144000   .$....t..$....@.  .............. .
    00A0  08000E1147D8C4C2 F261C1C9E7F6F400   ....G....a......  .....QDB2/AIX64.
    00B0  08116D9481A29200 0C115AE2D8D3F0F9   ..m.......Z.....  .._mask...]SQL09
    00C0  F0F0F0                              ...               000

           ACCSEC RQSDSS                    (ASCII)           (EBCDIC)
           0 1 2 3 4 5 6 7  8 9 A B C D E F   0123456789ABCDEF  0123456789ABCDEF
  0000  0026D00100020020 106D000611A20003   .&..... .m......  ..}....._...s..
    0010  00162110E2E3D3C5 C3F1404040404040   ..!.......@@@@@@  ....STLEC1
  0020  404040404040                        @@@@@@

3 data DB2 UDB DRDA Communication Manager sqljcReceive fnc (3.3.54.3.0.100)
 pid 807116 tid 1 cpid -1 node 0 sec 0 nsec 110546200 probe 100
 bytes 12

 Data1  (PD_TYPE_UINT,4) unsigned integer:
 105

4 data DB2 UDB DRDA Communication Manager sqljcReceive fnc (3.3.54.3.0.1178)
 pid 807116 tid 1 cpid -1 node 0 sec 0 nsec 110549755 probe 1178
 bytes 122

 RECEIVE BUFFER(AR):

           EXCSATRD OBJDSS                  (ASCII)           (EBCDIC)
           0 1 2 3 4 5 6 7  8 9 A B C D E F   0123456789ABCDEF  0123456789ABCDEF
  0000  0059D04300010053 1443000F115EE5F8   .Y.C...S.C...^..  ..}.........;V8
    0010  F1C14BE2E3D3C5C3 F100181404140300   ..K.............  1A.STLEC1.......
    0020  0724070007147400 05240F0007144000   .$....t..$....@.  .............. .
    0030  0700081147D8C4C2 F20014116DE2E3D3   ....G.......m...  .....QDB2..._STL
  0040  C5C3F14040404040 4040404040000C11   ...@@@@@@@@@@...  EC1          ...
  0050  5AC4E2D5F0F8F0F1 F5                Z........         ]DSN08015

                ACCSECRD OBJDSS                 (ASCII)           (EBCDIC)
                0 1 2 3 4 5 6 7  8 9 A B C D E F  0123456789ABCDEF  0123456789ABCDEF
    0000  0010D0030002000A 14AC000611A20003   ..............  ..}.........s..

5 data DB2 UDB DRDA Communication Manager sqljcSend fnc (3.3.54.5.0.100)
 pid 807116 tid 1 cpid -1 node 0 sec 0 nsec 110656806 probe 100
 bytes 16

 Data1  (PD_TYPE_UINT,8) unsigned integer:
 233
```

Abbildung 35. Beispiel einer Traceausgabe (TCP/IP-Verbindung)

```
6 data DB2 UDB DRDA Communication Manager sqljcSend fnc (3.3.54.5.0.1177)
 pid 807116 tid 1 cpid -1 node 0 sec 0 nsec 110659711 probe 1177
 bytes 250

 SEND BUFFER(AR):

           SECCHK RQSDSS                    (ASCII)           (EBCDIC)
           0 1 2 3 4 5 6 7  8 9 A B C D E F  0123456789ABCDEF  0123456789ABCDEF
  0000  003CD04100010036 106E000611A20003   .<.A...6.n......  ..}......>...s..
  0010  00162110E2E3D3C5 C3F1404040404040   ..!.......@@@@@@  ....STLEC1
  0020  404040404040000C 11A1D9858799F485   @@@@@@..........        ....Regr4e
    0030  A599000A11A09585 A6A39695           ............      vr....newton


           ACCRDB RQSDSS                    (ASCII)           (EBCDIC)
           0 1 2 3 4 5 6 7  8 9 A B C D E F  0123456789ABCDEF  0123456789ABCDEF
  0000  00ADD001000200A7 20010006210F2407   ........ ...!.$.  ..}....x........
    0010  00172135C7F9F1C1 F0C4F3C14BD7C1F8   ..!5........K...  ....G91A0D3A.PA8
  0020  F806030221064600 162110E2E3D3C5C3   ....!.F..!......  8..........STLEC
    0030  F140404040404040 4040404040000C11   .@@@@@@@@@@@@...  1            ...
  0040  2EE2D8D3F0F9F0F0 F0000D002FD8E3C4   ............/...  .SQL09000....QTD
  0050  E2D8D3C1E2C30016 00350006119C0333   .........5.....3  SQLASC..........
  0060  0006119D04B00006 119E0333003C2104   ...........3.

7 data DB2 UDB DRDA Communication Manager sqljcReceive fnc (3.3.54.3.0.100)
 pid 807116 tid 1 cpid -1 node 0 sec 0 nsec 259908001 probe 100
 bytes 12

 Data1  (PD_TYPE_UINT,4) unsigned integer:
 176

8 data DB2 UDB DRDA Communication Manager sqljcReceive fnc (3.3.54.3.0.1178)
 pid 807116 tid 1 cpid -1 node 0 sec 0 nsec 259911584 probe 1178
 bytes 193

 RECEIVE BUFFER(AR):

            SECCHKRM RPYDSS                  (ASCII)           (EBCDIC)
            0 1 2 3 4 5 6 7  8 9 A B C D E F  0123456789ABCDEF  0123456789ABCDEF
  0000  0015D0420001000F 1219000611490000   ...B.........I..  ..}.............
    0010  000511A400                         .....             ...u.


           ACCRDBRM RPYDSS                  (ASCII)           (EBCDIC)
           0 1 2 3 4 5 6 7  8 9 A B C D E F  0123456789ABCDEF  0123456789ABCDEF
  0000  009BD00200020095 2201000611490000   ........"....I..  ..}....n........
    0010  000D002FD8E3C4E2 D8D3F3F7F0000C11   .../............  ....QTDSQL370...
    0020  2EC4E2D5F0F8F0F1 F500160035000611   ............5...  .DSN08015.......
  0030  9C04B80006119E04 B80006119D04B000   ................  ................
    0040  0C11A0D5C5E6E3D6 D540400006212524   .........@@..!%$  ...NEWTON  .....
  0050  34001E244E000624 4C00010014244D00   4..$N..$L....$M.  ....+...<.....(.
    0060  06244FFFFF000A11 E8091E768301BE00   .$O........v....  ..!.....Y...c...
    0070  2221030000000005 68B3B8C7F9F1C1F0   "!......h.......  ...........G91A0
    0080  C4F3C1D7C1F8F840 4040400603022106   .......@@@@...!.  D3APA88    .....
    0090  46000A11E8091E76 831389             F......v...       ....Y...c.i

9 data DB2 UDB DRDA Communication Manager sqljcSend fnc (3.3.54.5.0.100)
 pid 807116 tid 1 cpid -1 node 0 sec 2 nsec 364420503 probe 100
 bytes 16

 Data1  (PD_TYPE_UINT,8) unsigned integer:
 10
```

Abbildung 36. Beispiel einer Trace-Ausgabe (TCP/IP-Verbindung) - Fortsetzung

```
10 data DB2 UDB DRDA Communication Manager sqljcSend fnc (3.3.54.5.0.1177)
 pid 807116 tid 1 cpid -1 node 0 sec 2 nsec 364440751 probe 1177
 bytes 27

 SEND BUFFER(AR):

           RDBCMM RQSDSS                    (ASCII)           (EBCDIC)
           0 1 2 3 4 5 6 7  8 9 A B C D E F   0123456789ABCDEF  0123456789ABCDEF
  0000  000AD00100010004 200E               ........ .        ..}.......
11 data DB2 UDB DRDA Communication Manager sqljcReceive fnc (3.3.54.3.0.100)
 pid 807116 tid 1 cpid -1 node 0 sec 2 nsec 475009631 probe 100
 bytes 12

 Data1  (PD_TYPE_UINT,4) unsigned integer:
 54
12 data DB2 UDB DRDA Communication Manager sqljcReceive fnc (3.3.54.3.0.1178)
 pid 807116 tid 1 cpid -1 node 0 sec 2 nsec 475014579 probe 1178
 bytes 71

 RECEIVE BUFFER(AR):

           ENDUOWRM RPYDSS                  (ASCII)           (EBCDIC)
           0 1 2 3 4 5 6 7  8 9 A B C D E F   0123456789ABCDEF  0123456789ABCDEF
  0000  002BD05200010025 220C000611490004   .+.R...%"....I..  ..}.............
    0010  00162110E2E3D3C5 C3F1404040404040   ..!.......@@@@@@  ....STLEC1
  0020  4040404040400005 211501             @@@@@@..!..             .....


           SQLCARD OBJDSS                   (ASCII)           (EBCDIC)
           0 1 2 3 4 5 6 7  8 9 A B C D E F   0123456789ABCDEF  0123456789ABCDEF
  0000  000BD00300010005 2408FF             ........$..       ..}........
13 data DB2 UDB DRDA Communication Manager sqljcSend fnc (3.3.54.5.0.100)
 pid 807116 tid 1 cpid -1 node 0 sec 5 nsec 721710319 probe 100
 bytes 16

 Data1  (PD_TYPE_UINT,8) unsigned integer:
 126
14 data DB2 UDB DRDA Communication Manager sqljcSend fnc (3.3.54.5.0.1177)
 pid 807116 tid 1 cpid -1 node 0 sec 5 nsec 721727276 probe 1177
 bytes 143

 SEND BUFFER(AR):

           EXCSQLIMM RQSDSS                 (ASCII)           (EBCDIC)
           0 1 2 3 4 5 6 7  8 9 A B C D E F   0123456789ABCDEF  0123456789ABCDEF
  0000  0053D0510001004D 200A00442113E2E3   .S.Q...M ..D!...  ..}....(......ST
  0010  D3C5C3F140404040 4040404040404040   ....@@@@@@@@@@@@  LEC1
  0020  D5E4D3D3C9C44040 4040404040404040   ......@@@@@@@@@@  NULLID
  0030  4040E2D8D3C3F2C6 F0C1404040404040   @@........@@@@@@    SQLC2F0A
  0040  4040404041414141 41484C5600CB0005   @@@@AAAAAAHLV....      ......<.....
    0050  2105F1                             !..               ..1


           SQLSTT OBJDSS                    (ASCII)           (EBCDIC)
           0 1 2 3 4 5 6 7  8 9 A B C D E F   0123456789ABCDEF  0123456789ABCDEF
  0000  002BD00300010025 2414000000001B64   .+.....%$......d  ..}.............
  0010  656C657465206672 6F6D206464637375   elete from ddcsu  .%......?_......
  0020  73312E6D79746162 6C65FF             s1.mytable.       ..._`./.%..
15 data DB2 UDB DRDA Communication Manager sqljcReceive fnc (3.3.54.3.0.100)
 pid 807116 tid 1 cpid -1 node 0 sec 5 nsec 832901261 probe 100
 bytes 12

 Data1  (PD_TYPE_UINT,4) unsigned integer:
 102
```

Abbildung 37. Beispiel einer Trace-Ausgabe (TCP/IP-Verbindung) - Fortsetzung

```
16 data DB2 UDB DRDA Communication Manager sqljcReceive fnc (3.3.54.3.0.1178)
 pid 807116 tid 1 cpid -1 node 0 sec 5 nsec 832906528 probe 1178
 bytes 119

 RECEIVE BUFFER(AR):

           SQLCARD OBJDSS                   (ASCII)           (EBCDIC)
           0 1 2 3 4 5 6 7  8 9 A B C D E F   0123456789ABCDEF  0123456789ABCDEF
  0000  0066D00300010060 240800FFFFFF3434   .f.....`$.....44  ..}....-........
  0010  3237303444534E58 4F544C2000FFFFFE   2704DSNXOTL ....  ......+.!.<.....
    0020  0C00000000000000 00FFFFFFFF000000   ................  ................
    0030  0000000000572020 2057202020202020   .....W   W       ................
  0040  001053544C454331 2020202020202020   ..STLEC1          ....<...........
  0050  2020000F44444353 5553312E4D595441     ..DDCSUS1.MYTA  ............(...
  0060  424C450000FF                        BLE...            .<....
17 data DB2 UDB DRDA Communication Manager sqljcSend fnc (3.3.54.5.0.100)
 pid 807116 tid 1 cpid -1 node 0 sec 5 nsec 833156953 probe 100
 bytes 16

 Data1  (PD_TYPE_UINT,8) unsigned integer:
 10
18 data DB2 UDB DRDA Communication Manager sqljcSend fnc (3.3.54.5.0.1177)
 pid 807116 tid 1 cpid -1 node 0 sec 5 nsec 833159843 probe 1177
 bytes 27

 SEND BUFFER(AR):

           RDBRLLBCK RQSDSS                 (ASCII)           (EBCDIC)
           0 1 2 3 4 5 6 7  8 9 A B C D E F   0123456789ABCDEF  0123456789ABCDEF
  0000  000AD00100010004 200F               ........ .         ..}.......
19 data DB2 UDB DRDA Communication Manager sqljcReceive fnc (3.3.54.3.0.100)
 pid 807116 tid 1 cpid -1 node 0 sec 5 nsec 943302832 probe 100
 bytes 12

 Data1  (PD_TYPE_UINT,4) unsigned integer:
 54
20 data DB2 UDB DRDA Communication Manager sqljcReceive fnc (3.3.54.3.0.1178)
 pid 807116 tid 1 cpid -1 node 0 sec 5 nsec 943306288 probe 1178
 bytes 71

 RECEIVE BUFFER(AR):

           ENDUOWRM RPYDSS                  (ASCII)           (EBCDIC)
           0 1 2 3 4 5 6 7  8 9 A B C D E F   0123456789ABCDEF  0123456789ABCDEF
  0000  002BD05200010025 220C000611490004   .+.R...%"....I..  ..}.............
    0010  00162110E2E3D3C5 C3F1404040404040   ..!.......@@@@@@  ....STLEC1
  0020  4040404040400005 211502             @@@@@@..!..             .....


           SQLCARD OBJDSS                   (ASCII)           (EBCDIC)
           0 1 2 3 4 5 6 7  8 9 A B C D E F   0123456789ABCDEF  0123456789ABCDEF
  0000  000BD00300010005 2408FF             ........$..       ..}........
```

Abbildung 38. Beispiel einer Trace-Ausgabe (TCP/IP-Verbindung) - Fortsetzung

Informationen zu nachfolgenden Puffern für DRDA-Traces

Auch die nachfolgenden Sende- und Empfangspuffer können auf zusätzliche Informationen hin analysiert werden. Die nächste Anforderung enthält einen Befehl COMMIT. Der Befehl **commit** weist das Verwaltungssystem des IBM Mainframe-Datenbankservers an, die aktuelle UOW (Unit of Work) festzuschreiben. Der vierte Puffer wird vom Verwaltungssystem des IBM Mainframe-Datenbankservers als Er-

gebnis eines Commits oder eines Rollbacks empfangen. Er enthält die ENDUOWRM-Nachricht (End Unit of Work Reply Message), die anzeigt, dass die aktuelle UOW beendet wurde.

In diesem Beispiel enthält der Trace-Eintrag 12 einen leeren SQLCA, angegeben durch den DDM-Codepunkt X'2408' gefolgt von X'FF'. Ein leerer SQL-Kommunikationsbereich (X'2408FF') zeigt die erfolgreiche Ausführung an (SQLCODE 0).

Abb. 35 auf Seite 535 zeigt ein Beispiel eines Empfangspuffers mit einem Fehler-SQLCA bei Trace-Eintrag 16.

Traces der Steuerzentrale

Bevor für den Fehler ein Trace in der Steuerzentrale durchgeführt wird, empfiehlt es sich, zunächst sicherzustellen, dass nicht der gleiche Fehler auftritt, wenn die entsprechenden Aktionen mittels expliziter Befehle über die DB2-Eingabeaufforderung ausgeführt werden.

Wenn Sie eine Task in der Steuerzentrale (oder einem der anderen GUI-Tools, die über die Steuerzentrale gestartet werden können) ausführen, wird häufig der Knopf 'Befehl anzeigen' angezeigt, der die genaue Syntax für den Befehl liefert, den das Tool verwenden wird. Wenn genau dieser Befehl über die DB2-Eingabeaufforderung erfolgreich ausgeführt wird, im GUI-Tool jedoch fehlschlägt, ist es angemessen, einen Trace über die Steuerzentrale abzurufen.

Um einen Trace für einen Fehler durchzuführen, der nur in der Steuerzentrale reproduzierbar ist, starten Sie die Steuerzentrale wie folgt:

```
db2cc -tf dateiname
```

Mit diesem Befehl wird die Tracefunktion der Steuerzentrale aktiviert und die Traceausgabe in die angegebene Datei geschrieben. Die Ausgabedatei wird unter Windows im Verzeichnis `<db2-installationspfad>\sqllib\tools` und unter UNIX und Linux im Verzeichnis `/home/<benutzer-id>/sqllib/tools` gespeichert.

Anmerkung: Wenn Sie die Steuerzentrale mit aktivierter Tracefunktion gestartet haben, reproduzieren Sie den Fehler anhand von möglichst wenigen Schritten. Versuchen Sie zu vermeiden, im Tool auf Elemente zu klicken, die nicht erforderlich sind oder mit dem Fehler nicht in Zusammenhang stehen. Schließen Sie nach der Reproduktion des Fehlers sowohl die Steuerzentrale als auch alle anderen GUI-Tools, die Sie zwecks Fehlerreproduktion geöffnet haben.

Die neue Tracedatei muss zur Analyse an IBM Software Support gesendet werden.

JDBC-Tracedateien

Abrufen von Traces von Anwendungen, die den DB2 JDBC Type 2-Treiber für Linux, UNIX und Windows verwenden

In diesem Abschnitt wird beschrieben, wie Sie einen Trace von einer Anwendung abrufen können, die den DB2 JDBC Type 2-Treiber für Linux-, UNIX- und Windows-Systeme verwendet.

Informationen zu diesem Vorgang

Dieser Tracetyp wird in folgenden Fehlersituationen eingesetzt:

- Fehler in einer JDBC-Anwendung, die den DB2 JDBC Type 2-Treiber für Linux, UNIX oder Windows (DB2 JDBC Type 2-Treiber) verwendet.

- Fehler in gespeicherten DB2-JDBC-Prozeduren.

Anmerkung: Es gibt viele Schlüsselwörter, die der Datei db2cli.ini hinzugefügt werden können und die sich auf das Anwendungsverhalten auswirken können. Diese Schlüsselwörter können Anwendungsfehler entweder beheben oder auch verursachen. Einige Schlüsselwörter werden in der CLI-Dokumentation nicht behandelt. Sie sind nur über die DB2-Unterstützungsfunktion erhältlich. Befinden sich in Ihrer Datei db2cli.ini Schlüsselwörter, die nicht dokumentiert sind, wurden sie wahrscheinlich von der DB2-Unterstützungsfunktion empfohlen. Intern verwendet der DB2 JDBC Type 2-Treiber den CLI-Treiber für den Datenbankzugriff. Die Java-Methode 'getConnection()' beispielsweise wird intern vom DB2 JDBC Type 2-Treiber der CLI-Funktion 'SQLConnect()' zugeordnet. Daher kann für Java-Entwickler zusätzlich zum DB2-JDBC-Trace auch ein CLI-Trace von Nutzen sein.

Vorgehensweise

1. Erstellen Sie einen Pfad für die Tracedateien. Es muss ein Pfad erstellt werden, auf den alle Benutzer Schreibzugriff haben.

   Verwenden Sie hierzu unter Windows beispielsweise folgenden Befehl:

   ```
   mkdir c:\temp\trace
   ```

   Unter Linux und UNIX:

   ```
   mkdir /tmp/trace
   chmod 777 /tmp/trace
   ```

2. Aktualisieren Sie die Schlüsselwörter der CLI-Konfiguration. Hierfür gibt es zwei Möglichkeiten:
   - Manuelle Bearbeitung der Datei db2cli.ini. Die Position der Datei db2cli.ini kann sich ändern, je nachdem, ob der Microsoft ODBC Driver Manager verwendet wird, welcher Typ von Datenquellennamen (DSN) verwendet wird, welcher Typ von Client oder Treiber installiert ist und ob die Registrierdatenbankvariable **DB2CLIINIPATH** definiert ist. Weitere Informationen finden Sie im Abschnitt zur „Initialisierungsdatei 'db2cli.ini'" in der Veröffentlichung *Call Level Interface Guide and Reference, Volume 1*.
     a. Öffnen Sie die Datei db2cli.ini in einem einfachen Texteditor.
     b. Fügen Sie der Datei den folgenden Abschnitt hinzu (bzw. hängen Sie die Variablen an, falls der Abschnitt COMMON bereits vorhanden ist):

        ```
        [COMMON]
                            JDBCTrace=1
                            JDBCTracePathName=<pfad>
                            JDBCTraceFlush=1
        ```

        Hierbei steht <pfad> beispielsweise für C:\temp\trace auf Windows-Plattformen bzw. für /tmp/trace auf Linux- oder UNIX-Plattformen.

     c. Speichern Sie die Datei mit mindestens einer Leerzeile am Dateiende. (Hierdurch werden einige Parsingfehler vermieden.)
   - Verwenden von UPDATE CLI CFG-Befehlen zum Aktualisieren der Datei db2cli.ini. Geben Sie die folgenden Befehle aus:

     ```
     db2 UPDATE CLI CFG FOR SECTION COMMON USING JDBCTrace 1
     db2 UPDATE CLI CFG FOR SECTION COMMON USING JDBCTracePathName <pfad>
     ```

     Hierbei steht <pfad> beispielsweise für C:\temp\trace auf Windows-Plattformen bzw. für /tmp/trace auf Linux- oder UNIX-Plattformen.

     ```
     db2 UPDATE CLI CFG FOR SECTION COMMON USING JDBCTraceFlush 1
     ```

   Wenn Sie die Tracefunktion zum Diagnostizieren von Anwendungsproblemen verwenden, ist zu bedenken, dass sich die Funktion auf die Anwendungsleis-

tung auswirkt und dass dies alle Anwendungen betrifft, nicht nur die Testanwendung. Daher sollte darauf geachtet werden, die Tracefunktion wieder zu inaktivieren, nachdem der Fehler ermittelt wurde.

3. Geben Sie den folgenden Befehl aus, um zu prüfen, ob die korrekten Schlüsselwörter eingerichtet wurden und berücksichtigt werden:

   ```
   db2 GET CLI CFG FOR SECTION COMMON
   ```

4. Starten Sie die Anwendung erneut.

   Die Datei `db2cli.ini` wird nur beim Start der Anwendung gelesen. Daher muss die Anwendung erneut gestartet werden, damit die Änderungen wirksam werden.

   Wird ein Trace für eine gespeicherte JDBC-Prozedur erstellt, bedeutet dies den Neustart der DB2-Instanz.

5. Erfassen Sie den Fehler. Führen Sie die Anwendung so lange aus, bis der Fehler generiert wird, und schließen Sie die Anwendung danach wieder. Grenzen Sie das Umfeld möglichst ein, sodass zum Zeitpunkt der Traceerstellung nur diejenigen JDBC-Anwendungen aktiv sind, die mit der erneuten Generierung des Fehlers in Zusammenhang stehen. Dadurch werden die Tracedateien leichter verständlich.

6. Inaktivieren Sie die JDBC-Tracefunktion.

   Setzen Sie manuell das Schlüsselwort 'JDBCTrace=0' im Abschnitt [COMMON] der Datei `db2cli.ini`, oder geben Sie folgende Befehle aus:

   ```
   db2 UPDATE CLI CFG FOR SECTION COMMON USING Trace 0
   db2 UPDATE CLI CFG FOR SECTION COMMON USING JDBCTrace 0
   ```

7. Starten Sie alle Anwendungen erneut, die aktiv sind und Traces erstellen.

8. Erfassen Sie die Tracedateien.

   Die JDBC-Tracedateien werden in den Pfad geschrieben, der im Schlüsselwort 'JDBCTracePathName' angegeben ist. Die Namen aller generierten Dateien enden mit der Erweiterung '.trc'. Es sind alle Dateien erforderlich, die zum Zeitpunkt der erneuten Fehlergenerierung im Tracepfad erstellt wurden.

Abrufen von Traces für Anwendungen, die den DB2 Universal JDBC-Treiber verwenden

In diesem Abschnitt wird beschrieben, wie Sie einen Trace für eine Anwendung abrufen können, die den DB2 Universal JDBC-Treiber verwendet.

Vorgehensweise

Wenn Sie über eine SQLJ- oder JDBC-Anwendung verfügen, die den DB2 Universal JDBC-Treiber verwendet, gibt es verschiedene Möglichkeiten, einen JDBC-Trace zu aktivieren.

- Wenn Sie die Schnittstelle 'DataSource' verwenden, um eine Verbindung zu einer Datenquelle herzustellen, aktivieren Sie die Tracefunktion mithilfe der Methoden 'DataSource.setTraceLevel()' und 'DataSource.setTraceFile()'.
- Wenn Sie die Schnittstelle 'DriverManager' verwenden, um eine Verbindung zu einer Datenquelle herzustellen, lässt sich die Tracefunktion am einfachsten aktivieren, indem für 'DriverManager' vor dem Verbindungsaufbau 'logWriter' gesetzt wird.

  Beispiel:

  ```
  DriverManager.setLogWriter(new PrintWriter(new FileOutputStream("trace.txt")));
  ```

- Wenn Sie die Schnittstelle 'DriverManager' verwenden, können Sie beim Laden des Treibers alternativ auch die Merkmale 'traceFile' und 'traceLevel' als Teil der URL angeben.

Beispiel:

```
String databaseURL =
"jdbc:db2://hal:50000/sample:traceFile=c:/temp/trace.txt;" ;
```

CLI-Tracedateien

Anwendungen, die auf den CLI-Treiber zugreifen, können das CLI-Tracedienstprogramm verwenden. Dieses Dienstprogramm zeichnet alle Funktionsaufrufe auf, die von den CLI-Treibern an eine Tracedatei abgesetzt wurden. Diese Daten sind bei der Problembestimmung nützlich.

Dieser Tracetyp wird in Situationen eingesetzt, in denen Probleme mit den folgenden Komponenten auftreten:

- CLI-Anwendung
- ODBC-Anwendung (da ODBC-Anwendungen die CLI-Schnittstelle für den Zugriff auf DB2 verwenden)
- Gespeicherte CLI-Prozeduren
- JDBC-Anwendungen und gespeicherten JDBC-Prozeduren

Bei der Diagnose von ODBC-Anwendungen lassen sich Probleme häufig am einfachsten mithilfe eines ODBC-Trace oder eines CLI-Trace bestimmen. Wenn Sie einen ODBC-Treibermanager verwenden, steht wahrscheinlich auch die Funktionalität zur Durchführung eines ODBC-Trace bereit. Die Dokumentation des Treibermanagers enthält Informationen dazu, wie die ODBC-Tracefunktion aktiviert wird. CLI-Traces sind speziell auf DB2 zugeschnitten und enthalten häufig mehr Informationen als ein generischer ODBC-Trace. Beide Traces weisen normalerweise viele Ähnlichkeiten auf. Sie listen die Eingangs- und Ausgangspunkte aller CLI-Aufrufe einer Anwendung einschließlich aller Parameter und Rückkehrcodes für diese Aufrufe auf.

Anmerkung: Der CLI-Trace bietet nur sehr wenige Informationen zu den internen Abläufen des CLI-Treibers.

Der DB2 JDBC Type 2-Treiber für Linux, UNIX und Windows (DB2 JDBC Type 2-Treiber) benötigt zum Zugriff auf die Datenbank den CLI-Treiber. Folglich müssen Java-Entwickler auch die CLI-Tracefunktion aktivieren, um zusätzliche Informationen darüber zu erhalten, wie ihre Anwendungen mit der Datenbank über die verschiedenen Softwareschichten interagieren. Die DB2-JDBC- und die CLI-Traceoptionen sind voneinander unabhängig, obwohl beide in der Datei 'db2cli.ini' festgelegt werden.

Im vorliegenden Abschnitt werden die folgenden Themen behandelt:

- DB2-CLI- und DB2-JDBC-Tracekonfiguration
- DB2-CLI-Traceoptionen und die Datei 'db2cli.ini'
- DB2-JDBC-Traceoptionen und die Datei 'db2cli.ini'
- DB2-CLI-Treibertrace versus Trace des ODBC-Treibermanagers
- DB2-CLI-Treiber, DB2 JDBC Type 2-Treiber und DB2-Traces
- DB2-CLI- und DB2-JDBC-Traces und gespeicherte CLI- oder Java-Prozeduren

Der CLI- und der DB2 JDBC Type 2-Treiber für Linux, UNIX und Windows bieten umfassende Tracefunktionen. Standardmäßig sind diese Funktionen inaktiviert und belegen keine zusätzlichen IT-Ressourcen. Werden sie aktiviert, dann generieren sie mindestens eine Textprotokolldatei, sobald eine Anwendung auf den entsprechen-

den Treiber (CLI- oder DB2 JDBC Type 2-Treiber) zugreift. Diese Protokolldateien enthalten detaillierte Informationen zu den folgenden Punkten:

- Reihenfolge, in der CLI- oder JDBC-Funktionen von der Anwendung aufgerufen werden.
- Inhalt der Ein- und Ausgabeparameter, die an die CLI- oder JDBC-Funktionen übertragen bzw. von diesen Funktionen empfangen werden.
- Rückkehrcodes und Fehlernachrichten oder Warnungen, die von den CLI- oder JDBC-Funktionen generiert werden.

Anmerkung: Diese Tracefunktion gilt nicht für den DB2 Universal JDBC-Treiber.

Die Analyse der CLI- und DB2-JDBC-Tracedateien bietet eine Reihe von Vorteilen für Anwendungsentwickler. Schleichende Fehler innerhalb der Programmlogik und bei der Parameterinitialisierung können häufig anhand dieser Traces identifiziert werden. Außerdem geben die CLI- und DB2-JDBC-Traces Anhaltspunkte zur Erreichung einer verbesserten Optimierung einer Anwendung oder der Datenbanken, auf die zugegriffen wird. Beispiel: Wenn ein CLI-Trace feststellt, dass eine Tabelle sehr häufig auf eine bestimmte Attributgruppe hin abgefragt wird, wird ein Tabellenindex für diese Attribute erstellt, um die Anwendungsleistung zu verbessern. Abschließend kann die Analyse der CLI- und DB2-JDBC-Tracedateien Anwendungsentwickler dabei unterstützen, sich in das Verhalten einer Anwendung oder Schnittstelle eines Fremdanbieters einzuarbeiten.

Konfiguration des CLI- und DB2-JDBC-Trace

Die Konfigurationsparameter für die DB2-CLI- und die DB2-JDBC-Tracefunktionen werden aus der DB2-CLI-Konfigurationsdatei `db2cli.ini` gelesen. Die Position der Datei `db2cli.ini` kann sich ändern, je nachdem, ob der Microsoft ODBC Driver Manager verwendet wird, welcher Typ von Datenquellennamen (DSN) verwendet wird, welcher Typ von Client oder Treiber installiert ist und ob die Registrierdatenbankvariable **DB2CLIINIPATH** definiert ist. Weitere Informationen hierzu finden Sie im Abschnitt zur Initialisierungsdatei 'db2cli.ini'.

Um die aktuellen Tracekonfigurationsparameter in `db2cli.ini` über den Befehlszeilenprozessor anzuzeigen, müssen Sie den folgenden Befehl absetzen:

```
db2 GET CLI CFG FOR SECTION COMMON
```

Die Datei `db2cli.ini` kann auf drei verschiedene Arten geändert werden, um die DB2-CLI- und die DB2-JDBC-Tracefunktionen zu konfigurieren:

- Verwendung des DB2-Konfigurationsassistenten, sofern dieser verfügbar ist.
- Manuelle Bearbeitung der Datei `db2cli.ini` mit einem Texteditor.
- Eingabe des Befehls **UPDATE CLI CFG** über den Befehlszeilenprozessor.

Der folgende Befehl, der über den Befehlszeilenprozessor eingegeben wird, dient z. B. zur Aktualisierung der Datei `db2cli.ini` und aktiviert die JDBC-Tracefunktion.

```
db2 UPDATE CLI CFG FOR SECTION COMMON USING jdbctrace 1
```

Anmerkung:

1. Normalerweise werden die Konfigurationsoptionen für den DB2-CLI- und den DB2-JDBC-Trace nur aus der Konfigurationsdatei `db2cli.ini` gelesen, wenn eine Anwendung initialisiert wird. Allerdings kann eine spezielle Traceoption von `db2cli.ini` (**TraceRefreshInterval**) verwendet werden, um ein Intervall anzugeben, innerhalb dessen bestimmte DB2-CLI-Traceoptionen erneut aus der Datei `db2cli.ini` gelesen werden sollen.

2. Die DB2-CLI-Tracefunktion kann auch über das Programm konfiguriert werden, indem das Umgebungsattribut SQL_ATTR_TRACE gesetzt wird. Diese Einstellung überschreibt die Einstellungen, die in der Datei db2cli.ini enthalten sind.

Wichtig: Inaktivieren Sie die DB2-CLI- und die DB2-JDBC-Tracefunktionen, wenn sie nicht benötigt werden. Die unnötige Verwendung der Tracefunktion kann zur Reduzierung der Anwendungsleistung und dazu führen, dass nicht erwünschte Traceprotokolldateien generiert werden. DB2 löscht keine der generierten Tracedateien und fügt neue Traceinformationen an die bereits vorhandenen Tracedateien an.

CLI-Traceoptionen und die Datei 'db2cli.ini'

Wenn eine Anwendung, die den DB2-CLI-Treiber verwendet, mit der Ausführung beginnt, dann überprüft der Treiber im Abschnitt [COMMON] der Datei db2cli.ini, ob Optionen der Tracefunktion vorhanden sind. Diese Traceoptionen sind spezielle Traceschlüsselwörter, die in der Datei db2cli.ini im Abschnitt [COMMON] auf bestimmte Werte gesetzt sind.

Anmerkung: Da die DB2-CLI-Traceschlüsselwörter im Abschnitt [COMMON] der Datei db2cli.ini angezeigt werden, gelten ihre Werte für alle Datenbankverbindungen über den DB2-CLI-Treiber.

Die folgenden CLI-Traceschlüsselwörter können definiert werden:

- com.ibm.db2.luw.admin.trb.doc/com.ibm.db2.luw.apdv.cli.doc/doc/r0008821.dita
- com.ibm.db2.luw.admin.trb.doc/com.ibm.db2.luw.apdv.cli.doc/doc/r0008822.dita
- com.ibm.db2.luw.admin.trb.doc/com.ibm.db2.luw.apdv.cli.doc/doc/r0011527.dita
- com.ibm.db2.luw.admin.trb.doc/com.ibm.db2.luw.apdv.cli.doc/doc/r0008823.dita
- com.ibm.db2.luw.admin.trb.doc/com.ibm.db2.luw.apdv.cli.doc/doc/r0008824.dita
- com.ibm.db2.luw.admin.trb.doc/com.ibm.db2.luw.apdv.cli.doc/doc/r0011528.dita
- com.ibm.db2.luw.admin.trb.doc/com.ibm.db2.luw.apdv.cli.doc/doc/r0008825.dita
- com.ibm.db2.luw.admin.trb.doc/com.ibm.db2.luw.apdv.cli.doc/doc/r0008826.dita
- com.ibm.db2.luw.admin.trb.doc/com.ibm.db2.luw.apdv.cli.doc/doc/r0008827.dita
- com.ibm.db2.luw.admin.trb.doc/com.ibm.db2.luw.apdv.cli.doc/doc/r0008839.dita
- com.ibm.db2.luw.admin.trb.doc/com.ibm.db2.luw.apdv.cli.doc/doc/r0008828.dita
- com.ibm.db2.luw.admin.trb.doc/com.ibm.db2.luw.apdv.cli.doc/doc/r0008829.dita
- com.ibm.db2.luw.admin.trb.doc/com.ibm.db2.luw.apdv.cli.doc/doc/r0008840.dita
- com.ibm.db2.luw.admin.trb.doc/com.ibm.db2.luw.apdv.cli.doc/doc/r0008831.dita

Anmerkung: DB2-CLI-Traceschlüsselwörter werden nur einmal während der Anwendungsinitialisierung aus der Datei db2cli.ini gelesen, es sei denn, das Schlüsselwort **TraceRefreshInterval** wurde gesetzt. Wenn dieses Schlüsselwort gesetzt wurde, dann werden die Schlüsselwörter **Trace** und **TracePIDList** in dem angegebenen Intervall erneut aus der Datei db2cli.ini gelesen und entsprechend auf die momentan ausgeführte Anwendung angewendet.

Im Folgenden ist ein Beispiel für die Tracekonfiguration anhand der Datei db2cli.ini mit den folgenden DB2-CLI-Schlüsselwörtern und -Werten dargestellt:

```
[COMMON]
trace=1
TraceFileName=\temp\clitrace.txt
TraceFlush=1
```

Anmerkung:

1. Bei den CLI-Traceschlüsselwörtern muss die Groß-/Kleinschreibung nicht beachtet werden. Allerdings kann es vorkommen, dass bei den Schlüsselwortwerten für Pfad- und Dateinamen unter bestimmten Betriebssystemen (z. B. UNIX) die Groß-/Kleinschreibung beachtet werden muss.
2. Wenn entweder ein DB2-CLI-Traceschlüsselwort oder der zugehörige Wert in der Datei `db2cli.ini` ungültig ist, dann ignoriert die DB2-CLI-Tracefunktion diesen Wert und verwendet stattdessen für dieses Traceschlüsselwort den Standardwert.

DB2-JDBC-Traceoptionen und die Datei 'db2cli.ini'

Wenn eine Anwendung, die den DB2 JDBC Type 2-Treiber verwendet, mit der Ausführung beginnt, dann überprüft der Treiber auch die Datei `db2cli.ini` auf Optionen der Tracefunktion. Wie bei den DB2-CLI-Traceoptionen werden auch die DB2-JDBC-Traceoptionen als Paare aus Schlüsselwort und Wert angegeben, die im Abschnitt [COMMON] der Datei `db2cli.ini` definiert sind.

Anmerkung: Da die DB2-JDBC-Traceschlüsselwörter im Abschnitt [COMMON] der Datei `db2cli.ini` angezeigt werden, gelten ihre Werte für alle Datenbankverbindungen über den DB2 JDBC Type 2-Treiber.

Die folgenden DB2-JDBC-Traceschlüsselwörter können definiert werden:

- **JDBCTrace**
- **JDBCTracePathName**
- **JDBCTraceFlush**

JDBCTrace = 0 | 1
: Das Schlüsselwort **JDBCTrace** steuert, ob andere DB2-JDBC-Traceschlüsselwörter sich auf die Ausführung des Programms auswirken. Wenn Sie für **JDBCTrace** den Standardwert `0` angeben, dann wird die DB2-JDBC-Tracefunktion inaktiviert. Wenn Sie für **JDBCTrace** den Wert 1 angeben, dann wird die Funktion aktiviert.

  Alleine genommen hat das Schlüsselwort **JDBCTrace** nur geringfügige Auswirkungen und generiert keine Traceausgabe, es sei denn, das Schlüsselwort **JDBCTracePathName** wird ebenfalls angegeben.

JDBCTracePathName = *vollständig_qualifizierter_tracepfadname*
: Der Wert von **JDBCTracePathName** ist der vollständig qualifizierte Pfad des Verzeichnisses, in das sämtliche DB2-JDBC-Traceinformationen geschrieben werden. Die DB2-JDBC-Tracefunktion versucht, eine neue Traceprotokolldatei zu generieren, sobald eine JDBC-Anwendung mit dem DB2 JDBC Type 2-Treiber ausgeführt wird. Wenn die Anwendung im Multithreadmodus arbeitet, wird eine separate Traceprotokolldatei für jeden Thread erstellt. Eine Verknüpfung der Anwendungsprozess-ID, der Threadfolgenummer und einer Zeichenfolge zur Identifikation des Threads wird automatisch verwendet, um Traceprotokolldateien zu benennen. Ein Name eines Standardpfads, in den die Ausgabeprotokolldateien für den DB2-JDBC-Trace geschrieben werden, existiert nicht.

JDBCTraceFlush = 0 | 1
: Das Schlüsselwort **JDBCTraceFlush** gibt an, wie oft Traceinformationen in die DB2-JDBC-Traceprotokolldatei geschrieben werden. Standardmäßig ist **JDBCTraceFlush** auf den Wert `0` gesetzt und jede DB2-JDBC-Traceprotokolldatei bleibt geöffnet, bis die mit dem Trace protokollierte Anwendung bzw. der entsprechende Thread normal beendet wird. Wenn die Anwendung ab-

normal beendet wird, können bestimmte Traceinformationen, die noch nicht in die Traceprotokolldatei geschrieben wurden, möglicherweise verloren gehen.

Um die Integrität und Vollständigkeit der Traceinformationen, die in die DB2-JDBC-Traceprotokolldatei geschrieben werden, sicherzustellen, kann das Schlüsselwort **JDBCTraceFlush** auf den Wert 1 gesetzt werden. Nachdem jeder Traceeintrag in die Traceprotokolldatei geschrieben wurde, schließt der DB2-JDBC-Treiber die Datei und öffnet sie anschließend erneut, wobei neue Traceeinträge am Ende der Datei angefügt werden. Dadurch wird sichergestellt, dass keine Traceinformationen verloren gehen.

Anmerkung: *Jede Operation zum Schließen und erneuten Öffnen der DB2-JDBC-Protokolldatei verursacht einen erheblichen Ein-/Ausgabeaufwand und kann die Anwendungsleistung beträchtlich reduzieren.*

Im Folgenden ist ein Beispiel für eine Tracekonfiguration in der Datei `db2cli.ini` aufgeführt, die diese DB2-JDBC-Schlüsselwörter und -Werte verwendet:

```
[COMMON]
jdbctrace=1
JdbcTracePathName=\temp\jdbctrace\
                    JDBCTraceFlush=1
```

Anmerkung:

1. Bei JDBC-Traceschlüsselwörtern muss die Groß-/Kleinschreibung nicht beachtet werden. Allerdings kann es vorkommen, dass bei den Schlüsselwortwerten für Pfad- und Dateinamen unter bestimmten Betriebssystemen (z. B. UNIX) die Groß-/Kleinschreibung beachtet werden muss.
2. Wenn entweder ein DB2-JDBC-Traceschlüsselwort oder der zugehörige Wert in der Datei `db2cli.ini` ungültig ist, dann ignoriert die DB2-JDBC-Tracefunktion diesen Wert und verwendet stattdessen für dieses Traceschlüsselwort den Standardwert.
3. Wenn die DB2-JDBC-Tracefunktion aktiviert wird, dann wird nicht auch die DB2-CLI-Tracefunktion aktiviert. Der DB2 JDBC Type 2-Treiber benötigt für den Zugriff auf die Datenbank den DB2-CLI-Treiber. Folglich müssen Java-Entwickler auch die DB2-CLI-Tracefunktion aktivieren, um zusätzliche Informationen darüber zu erhalten, wie ihre Anwendungen mit der Datenbank über die verschiedenen Softwareschichten interagieren. Die DB2-JDBC- und die DB2-CLI-Traceoptionen sind unabhängig von einander und können im Abschnitt [COMMON] der Datei `db2cli.ini` in beliebiger Reihenfolge gemeinsam angegeben werden.

CLI-Treibertrace versus Trace des ODBC-Treibermanagers

Es ist wichtig, den Unterschied zwischen einem Trace des ODBC-Treibermanagers und einem Trace des DB2-CLI-Treibers zu berücksichtigen. Ein Trace des ODBC-Treibermanagers zeigt die ODBC-Funktionsaufrufe an, die von einer ODBC-Anwendung an den ODBC-Treibermanager abgesetzt wurden. Im Gegensatz hierzu zeigt ein Trace des DB2-CLI-Treibers die Funktionsaufrufe an, die vom ODBC-Treibermanager *für die Anwendung* an den DB2-CLI-Treiber abgesetzt wurden.

Ein ODBC-Treibermanager kann bestimmte Funktionsaufrufe direkt von der Anwendung an den DB2-CLI-Treiber weiterleiten. Der ODBC-Treibermanager kann allerdings die Weiterleitung bestimmter Funktionsaufrufe an den Treiber auch verzögern oder verhindern. Der ODBC-Treibermanager kann darüber hinaus

Anwendungsfunktionsargumente ändern oder Anwendungsfunktionen anderen Funktionen zuordnen, bevor der Aufruf an den DB2-CLI-Treiber weitergeleitet wird.

Die Intervention durch den ODBC-Treibermanager beim Anwendungsfunktionsaufruf kann die folgenden Ursachen haben:

- In Anwendungen, die mit ODBC 2.0-Funktionen geschrieben wurden, die unter ODBC 3.0 nicht mehr unterstützt werden, wurden die alten Funktionen neuen Funktionen zugeordnet.
- ODBC 2.0-Funktionsargumente, die in ODBC 3.0 nicht mehr unterstützt werden, sind gleichwertigen ODBC 3.0-Argumenten zugeordnet.
- Die Microsoft-Cursorbibliothek kann Aufrufe wie z. B. `SQLExtendedFetch()` mehreren Aufrufen für `SQLFetch()` und anderen unterstützenden Funktionen zuordnen, um das gleiche Endergebnis zu erzielen.
- Das Verbindungspooling des ODBC-Treibermanagers verzögert normalerweise `SQLDisconnect()`-Anforderungen (oder verhindert sie, wenn die Verbindung erneut verwendet wird).

Aus diesem und anderen Gründen schätzen Anwendungsentwickler den Trace eines ODBC-Treibermanagers als eine wertvolle Ergänzung zum Trace des DB2-CLI-Treibers.

Weitere Informationen zur Erfassung und Interpretation von Traces des ODBC-Treibermanagers finden Sie in der Dokumentation zum ODBC-Treibermanager. Unter den Windows-Betriebssystemen sollten Sie die Informationen in der Veröffentlichung "Microsoft ODBC 3.0 Software Development Kit and Programmer's Reference" lesen, die auch online unter der Adresse http://www.msdn.microsoft.com/ zur Verfügung steht.

CLI-Treiber, DB2 JDBC Type 2-Treiber und DB2-Traces

Intern verwendet der DB2 JDBC Type 2-Treiber für den Datenbankzugriff den DB2-CLI-Treiber. Die Java-Methode getConnection() wird z. B. intern vom DB2 JDBC Type 2-Treiber der DB2-CLI-Funktion `SQLConnect()` zugeordnet. Daher kann für Java-Entwickler zusätzlich zum DB2-JDBC-Trace auch ein DB2-CLI-Trace von Nutzen sein.

Der DB2-CLI-Treiber verwendet zahlreiche interne und DB2-spezifische Funktionen, um seine Verarbeitungsoperationen auszuführen. Diese internen und DB2-spezifischen Funktionsaufrufe werden im DB2-Trace protokolliert. Für Anwendungsentwickler haben DB2-Traces keinen Nutzen, da sie lediglich für den IBM Service bestimmt sind und zur Ermittlung und Behebung von Problemen dienen.

CLI- und DB2-JDBC-Traces und gespeicherte CLI- oder Java-Prozeduren

Auf allen Workstationplattformen können die DB2-CLI- und DB2-JDBC-Tracefunktionen verwendet werden, um gespeicherte DB2-CLI- und DB2-JDBC-Prozeduren zu protokollieren.

Die meisten der DB2-CLI- und DB2-JDBC-Traceinformationen und -Anweisungen, die in den vorherigen Abschnitten erläutert wurden, sind generisch und gelten sowohl für Anwendungen als auch für gespeicherte Prozeduren. Anders als Anwendungen, bei denen es sich um Clients eines Datenbankservers handelt (und die normalerweise auf einer anderen Maschine als der Datenbankserver ausgeführt

werden), werden gespeicherte Prozeduren auf dem System des Datenbankservers ausgeführt. Aus diesem Grund müssen die folgenden zusätzlichen Schritte ausgeführt werden, wenn gespeicherte DB2-CLI- oder DB2-JDBC-Prozeduren protokolliert werden:

- Vergewissern Sie sich, dass die Traceschlüsselwortoptionen in der Datei `db2cli.ini` angegeben werden, die sich auf dem DB2-Server befindet.
- Wenn das Schlüsselwort **`TraceRefreshInterval`** nicht auf einen positiven Wert ungleich null gesetzt ist, dann müssen Sie sicherstellen, dass alle Schlüsselwörter korrekt konfiguriert sind. Dieser Arbeitsschritt muss vor dem Start der Datenbank (d. h. beim Absetzen des Befehls **`db2start`**) ausgeführt werden. Die Änderung von Traceeinstellungen während der Ausführung des Datenbankservers kann zu unvorhersehbaren Ergebnissen führen. Wenn z. B. **`TracePathName`** während der Ausführung des Servers geändert wird, werden bei der nächsten Ausführung einer gespeicherten Prozedur bestimmte Tracedateien in einen neuen Pfad geschrieben, während andere in den Originalpfad geschrieben werden. Um die Konsistenz sicherzustellen, müssen Sie den Server immer dann neu starten, wenn ein anderes Traceschlüsselwort außer **`Trace`** oder **`TracePIDList`** geändert wird.

Abrufen von CLI-Traces

Zum Aktivieren eines CLI-Traces müssen Sie eine Reihe von CLI-Konfigurationsschlüsselwörtern aktivieren.

Vorbereitende Schritte

Anmerkung: Es gibt viele Schlüsselwörter, die der Datei `db2cli.ini` hinzugefügt werden können und die sich auf das Anwendungsverhalten auswirken können. Diese Schlüsselwörter können Anwendungsfehler entweder beheben oder auch verursachen. Einige Schlüsselwörter werden in der CLI-Dokumentation nicht behandelt. Sie sind nur über IBM Software Support erhältlich. Befinden sich in Ihrer Datei `db2cli.ini` Schlüsselwörter, die nicht dokumentiert sind, wurden sie wahrscheinlich vom IBM Software Support-Team empfohlen.

Informationen zu diesem Vorgang

Wenn Sie die Tracefunktion zum Diagnostizieren von Anwendungsproblemen verwenden, ist zu bedenken, dass sich die Funktion auf die Anwendungsleistung auswirkt und dass dies alle Anwendungen betrifft, nicht nur die Testanwendung. Daher sollte darauf geachtet werden, die Tracefunktion wieder zu inaktivieren, nachdem der Fehler ermittelt wurde.

Vorgehensweise

Gehen Sie wie folgt vor, um einen CLI-Trace abzurufen:

1. Erstellen Sie einen Pfad für die Tracedateien.

   Es muss ein Pfad erstellt werden, auf den alle Benutzer Schreibzugriff haben. Unter einem Windows-Betriebssystem ist dies z. B. der folgende Pfad:

   ```
   mkdir c:\temp\trace
   ```

   Unter Linux- und UNIX-Betriebssystemen:

   ```
   mkdir /tmp/trace
   chmod 777 /tmp/trace
   ```

2. Aktualisieren Sie die Schlüsselwörter der CLI-Konfiguration.

Dies kann entweder (A) durch manuelles Bearbeiten der Datei db2cli.ini oder (B) durch Verwenden des Befehls **UPDATE CLI CFG** erfolgen.

- Gehen Sie wie folgt vor, um die Datei db2cli.ini manuell zu bearbeiten:
  a. Öffnen Sie die Datei db2cli.ini in einem einfachen Texteditor. Die Position der Datei db2cli.ini kann sich ändern, je nachdem, ob der Microsoft ODBC Driver Manager verwendet wird, welcher Typ von Datenquellennamen (DSN) verwendet wird, welcher Typ von Client oder Treiber installiert ist und ob die Registrierdatenbankvariable **DB2CLIINIPATH** definiert ist. Weitere Informationen finden Sie im Abschnitt zur „Initialisierungsdatei 'db2cli.ini'" in der Veröffentlichung *Call Level Interface Guide and Reference, Volume 1*.
  b. Fügen Sie der Datei den folgenden Abschnitt hinzu (bzw. hängen Sie einfach die Variablen an, falls der Abschnitt COMMON bereits vorhanden ist):

```
[COMMON]
Trace=1
TracePathName=pfad
TraceComm=1
TraceFlush=1
TraceTimeStamp=1
```

     Hierbei steht *pfad* beispielsweise für C:\temp\trace unter Windows bzw. für /tmp/trace unter Linux und UNIX.
  c. Speichern Sie die Datei mit mindestens einer Leerzeile am Dateiende. (Hierdurch werden einige Parsingfehler vermieden.)
- Gehen Sie wie folgt vor, um die CLI-Konfigurationsschlüsselwörter mit dem Befehl **UPDATE CLI CFG** zu aktualisieren. Setzen Sie dazu die folgenden Befehle ab:

```
db2 UPDATE CLI CFG FOR SECTION COMMON USING Trace 1
db2 UPDATE CLI CFG FOR SECTION COMMON USING TracePathName pfad
db2 UPDATE CLI CFG FOR SECTION COMMON USING TraceComm 1
db2 UPDATE CLI CFG FOR SECTION COMMON USING TraceFlush 1
db2 UPDATE CLI CFG FOR SECTION COMMON USING TraceTimeStamp 3
```

  Hierbei steht *pfad* beispielsweise für C:\temp\trace unter Windows bzw. für /tmp/trace unter Linux und UNIX.

3. Überprüfen Sie die Konfiguration der Datei db2cli.ini.

   Geben Sie den folgenden Befehl aus, um zu prüfen, ob die korrekten Schlüsselwörter eingerichtet wurden und berücksichtigt werden:

```
db2 GET CLI CFG FOR SECTION COMMON
```

4. Starten Sie die Anwendung erneut.

   Die Datei db2cli.ini wird nur beim Start der Anwendung gelesen. Daher muss die Anwendung erneut gestartet werden, damit die Änderungen wirksam werden.

   Wird ein Trace für eine gespeicherte CLI-Prozedur erstellt, bedeutet dies den Neustart der DB2-Instanz.

5. Erfassen Sie den Fehler.

   Führen Sie die Anwendung so lange aus, bis der Fehler generiert wird, und schließen Sie die Anwendung danach wieder. Falls möglich, grenzen Sie das Fehlerumfeld ein, sodass zum Zeitpunkt der Traceerstellung nur diejenigen Anwendungen aktiv sind, die mit der erneuten Generierung des Fehlers in Zusammenhang stehen. Dadurch wird die Analyse des Traces deutlich vereinfacht.

6. Inaktivieren Sie die CLI-Tracefunktion.

Setzen Sie im Abschnitt [COMMON] der Datei db2cli.ini das Schlüsselwort **Trace** manuell auf den Wert '0' oder setzen Sie den folgenden Befehl ab:

```
db2 UPDATE CLI CFG FOR SECTION COMMON USING Trace 0
```

7. (Optional) Starten Sie alle Anwendungen erneut, die möglicherweise aktiv sind und Traces erstellen.

Ergebnisse

Die CLI-Tracedateien werden in den Pfad geschrieben, der im Schlüsselwort **TracePathName** angegeben ist. Die Dateinamen weisen das Format p*pid*t*tid*.cli auf. Hierbei steht *pid* für die vom Betriebssystem zugeordnete Prozess-ID und *tid* für den numerischen Zähler, der bei 0 beginnt und jeweils für jeden Thread vom Anwendungsprozess generiert wird. Beispiel: p1234t1.cli. Wenn Sie sich zur Fehlerdiagnose an IBM Software Support wenden, müssen Sie möglicherweise alle Dateien einreichen, die im Tracepfad generiert wurden.

Interpretieren von Eingabe- und Ausgabeparametern in CLI-Tracedateien

Wie alle anderen regulären Funktionen haben auch die DB2 Call Level Interface-Funktionen (CLI-Funktionen) Eingabe- und Ausgabeparameter. In einem CLI-Trace werden diese Eingabe- und Ausgabeparameter angezeigt und liefern detaillierte Informationen darüber, wie die einzelnen Anwendungen jeweils eine bestimmte CLI-API aufrufen. Die Eingabe- und Ausgabeparameter der im CLI-Trace angezeigten CLI-Funktionen können jeweils mit der Definition der entsprechenden CLI-Funktion in den Abschnitten mit der CLI-Referenz der betreffenden Dokumentation verglichen werden.

Das folgende Beispiel zeigt ein Snippet einer CLI-Tracedatei:

```
SQLConnect( hDbc=0:1, szDSN="sample", cbDSN=-3, szUID="",
                      cbUID=-3, szAuthStr="", cbAuthStr=-3 )
        ---> Time elapsed - +6.960000E-004 seconds

SQLRETURN   SQLConnect       (
                      SQLHDBC           ConnectionHandle,  /* hdbc */
                      SQLCHAR      *FAR ServerName,        /* szDSN */
                      SQLSMALLINT       NameLength1,       /* cbDSN */
                      SQLCHAR      *FAR UserName,          /* szUID */
                      SQLSMALLINT       NameLength2,       /* cbUID */
                      SQLCHAR      *FAR Authentication,    /* szAuthStr */
                      SQLSMALLINT       NameLength3);      /* cbAuthStr */
```

Der erste Aufruf an die CLI-Funktion zeigt die Eingabeparameter und die ihnen zugeordneten Werte (wie zutreffend).

Wenn CLI-Funktionen eine Rückgabe liefern, zeigen sie die resultierenden Ausgabeparameter. Beispiel:

```
SQLAllocStmt( phStmt=1:1 )
        <--- SQL_SUCCESS   Time elapsed - +4.444000E-003 seconds
```

In diesem Fall gibt die CLI-Funktion 'SQLAllocStmt()' den Ausgabeparameter 'phStmt' mit dem Wert '1:1' (Verbindungskennung 1, Anweisungskennung 1) zurück.

Analysieren von dynamischem SQL in CLI-Traces

Die CLI-Traces zeigen ebenfalls, wie dynamisches SQL anhand der Deklaration und Verwendung von Parametermarken in den Anweisungen 'SQLPrepare()' und

'SQLBindParameter()' ausgeführt wird. Dadurch haben Sie die Möglichkeit, während der Laufzeit zu ermitteln, welche SQL-Anweisungen ausgeführt werden.

Der folgende Traceeintrag zeigt die Vorbereitung der SQL-Anweisung (ein Fragezeichen oder ein Doppelpunkt gefolgt von einem Namen (*name*) gibt eine Parametermarke an):

```
  SQLPrepare( hStmt=1:1, pszSqlStr=
      "select * from employee where empno = ?",
      cbSqlStr=-3 )
        ---> Time elapsed - +1.648000E-003 seconds
( StmtOut="select * from employee where empno = ?" )
SQLPrepare( )
    <--- SQL_SUCCESS   Time elapsed - +5.929000E-003 seconds
```

Der folgende Traceeintrag zeigt das Binden der Parametermarke als Zeichensatz (CHAR) mit einer Maximallänge von 7:

```
SQLBindParameter( hStmt=1:1, iPar=1, fParamType=SQL_PARAM_INPUT,
fCType=SQL_C_CHAR, fSQLType=SQL_CHAR, cbColDef=7, ibScale=0,
  rgbValue=&00854f28, cbValueMax=7, pcbValue=&00858534 )
        ---> Time elapsed - +1.348000E-003 seconds
SQLBindParameter( )
        <--- SQL_SUCCESS   Time elapsed - +7.607000E-003 seconds
```

Die dynamische SQL-Anweisung wird nun ausgeführt. Mit rbgValue="000010" wird der Wert angegeben, durch den die Parametermarke während der Laufzeit von der Anwendung ersetzt wurde:

```
SQLExecute( hStmt=1:1 )
        ---> Time elapsed - +1.317000E-003 seconds
( iPar=1, fCType=SQL_C_CHAR, rgbValue="000010" - X"303030303130",
  pcbValue=6, piIndicatorPtr=6 )
        sqlccsend( ulBytes - 384 )
        sqlccsend( Handle - 14437216 )
        sqlccsend( ) - rc - 0, time elapsed - +1.915000E-003
        sqlccrecv( )
        sqlccrecv( ulBytes - 1053 ) - rc - 0, time elapsed - +8.808000E-003
SQLExecute( )
        <--- SQL_SUCCESS   Time elapsed - +2.213300E-002 seconds
```

Interpretieren von Zeitdaten in CLI-Traces

Es gibt verschiedene Möglichkeiten, um Zeitdaten aus einem CLI-Trace zusammenzustellen. Ein CLI-Trace erfasst standardmäßig die Zeit, die in der Anwendung verbracht wurde, seit die CLI-API in einem Thread das letzte Mal aufgerufen wurde.

Neben der in DB2 verbrachten Zeit enthalten die Daten auch die Netzübertragungszeit für Übertragungen zwischen dem Client und dem Server. Beispiel:

```
SQLAllocStmt( hDbc=0:1, phStmt=&0012ee48 )
        ---> Time elapsed - +3.964187E+000 seconds
```

(Dieser Zeitwert gibt die Zeit an, die seit dem letzten CLI-API-Aufruf in der Anwendung verbracht wurde.)

```
SQLAllocStmt( phStmt=1:1 )
        <--- SQL_SUCCESS   Time elapsed - +4.444000E-003 seconds
```

(Seit Abschluss der Funktion gibt dieser Zeitwert die in DB2 verbrachte Zeit an, einschließlich der Netzübertragungszeit.)

Eine andere Möglichkeit zur Erfassung von Zeitdaten ist die Verwendung des CLI-Schlüsselworts 'TraceTimeStamp'. Mithilfe dieses Schlüsselworts wird für jeden Aufruf und jedes Ergebnis der CLI-API eine Zeitmarke erstellt. Es gibt vier Anzei-

geoptionen für das Schlüsselwort: keine Zeitmarkeninformationen, Prozessortaktimpuls und ISO-Zeitmarke, Prozessortaktimpuls oder ISO-Zeitmarke.

Dies kann von großem Nutzen sein, wenn Sie mit zeitbezogenen Problemen wie beispielsweise Funktionsfolgefehlern (CLI0125E) zu tun haben. Dies kann ebenfalls nützlich sein, wenn Sie mit Multithreading-Anwendungen arbeiten und versuchen zu ermitteln, welches Ereignis zuerst auftrat.

Interpretieren von unbekannten Werten in CLI-Traces

Es ist möglich, dass eine CLI-Funktion einen 'unbekannten Wert' als Wert für einen Eingabeparameter in einem CLI-Trace zurückgibt.

Dies kann vorkommen, wenn der CLI-Treiber für den betreffenden Eingabeparameter nach einem bestimmten Wert sucht, die Anwendung aber einen anderen Wert bereitstellt. Dies kann beispielsweise der Fall sein, wenn Sie veraltete Definitionen von CLI-Funktionen befolgen oder CLI-Funktionen verwenden, die nicht weiter unterstützt werden.

Es ist außerdem möglich, dass eine CLI-Funktion einen geänderten Optionswert ('Option value changed') oder einen Rückkehrcode für den Schlüsselsatzparser ('Keyset Parser Return Code') zurückgibt. Dies geschieht, wenn der Schlüsselsatzcursor eine Nachricht anzeigt, beispielsweise dann, wenn für den Cursor aus einem bestimmten Grund ein Downgrade auf einen statischen Cursor durchführt wird.

```
SQLExecDirect( hStmt=1:1, pszSqlStr="select * from org", cbSqlStr=-3 )
        ---> Time elapsed - +5.000000E-002 seconds
( StmtOut="select * from org" )
( COMMIT=0 )
( StmtOut=" SELECT A.TABSCHEMA, ..... )
( StmtOut=" SELECT A.TABSCHEMA, ..... )
( Keyset Parser Return Code=1100 )

SQLExecDirect( )
        <--- SQL_SUCCESS_WITH_INFO   Time elapsed - +1.06E+001 seconds
```

Im obigen CLI-Trace gibt der Schlüsselsatzparser den Rückkehrcode 1100 an. Dieser Code bedeutet, dass für die Tabelle kein eindeutiger Index oder Primärschlüssel vorhanden ist und daher kein Schlüsselsatzcursor erstellt werden kann. Diese Rückkehrcodes werden nicht extern bereitgestellt, sodass Sie an dieser Stelle IBM Software Support kontaktieren müssten, um weitere Informationen zur Bedeutung des Rückkehrcodes zu erhalten.

Das Aufrufen von 'SQLError' oder 'SQLDiagRec' gibt an, dass der Cursortyp geändert wurde. Die Anwendung sollte in diesem Fall den Cursortyp und den gemeinsamen Zugriff abfragen, um zu ermitteln, welches Attribut geändert wurde.

Interpretieren der CLI-Traceausgabe für Multithread-Anwendungen

Mit der CLI-Tracefunktion können Traces für Multithread-Anwendungen durchgeführt werden. Hierfür wird am besten das CLI-Schlüsselwort 'TracePathName' verwendet. Auf diese Weise werden Tracedateien mit dem Namen p<pid>t<tid>.cli erstellt, wobei <tid> die Thread-ID der Anwendung ist.

Die entsprechende Thread-ID finden Sie im CLI-Trace-Header:

```
[ Process: 3500, Thread: 728 ]
[ Date & Time:          02/17/2006 04:28:02.238015 ]
[ Product:              QDB2/NT DB2 v9.1.0.190 ]
...
```

Sie können die Traceausgabe für eine Multithreading-Anwendung auch in eine Datei schreiben lassen. Verwenden Sie hierzu das CLI-Schlüsselwort 'TraceFileName'. Auf diese Weise wird eine Datei Ihrer Wahl generiert, die unter Umständen jedoch mühsam zu lesen ist, da bestimmte APIs in einem Thread gleichzeitig mit einer API in einem anderen Thread ausgeführt werden können, was beim Prüfen der Traceausgabe verwirrend sein kann.

Es wird generell empfohlen, 'TraceTimeStamp' einzuschalten, um die tatsächliche Reihenfolge der Ereignisse ermitteln zu können, indem die Uhrzeit geprüft wird, zu der eine betreffende API ausgeführt wurde. Dies kann besonders bei der Untersuchung von Problemen nützlich sein, die darauf zurückzuführen sind, dass ein Thread einen Fehler in einem anderen Thread ausgelöst hat (beispielsweise CLI0125E - Funktionsfolgefehler).

CLI- und JDBC-Tracedateien

Anwendungen, die auf die CLI- und DB2-JDBC-Treiber zugreifen, können die CLI- und DB2-JDBC-Tracefunktionen verwenden. Diese Dienstprogramme zeichnen alle Funktionsaufrufe, die von den CLI- oder DB2-JDBC-Treibern ausgeführt wurden, in einer Protokolldatei auf, die zur Problembestimmung herangezogen werden kann. In diesem Abschnitt wird beschrieben, wie auf diese Protokolldateien, die von den Tracefunktionen generiert werden, zugegriffen werden kann und wie die aufgezeichneten Daten interpretiert werden können.

- Position der CLI- und JDBC-Tracedatei
- Interpretation der CLI-Tracedatei
- Interpretation der JDBC-Tracedatei

Position der CLI- und JDBC-Tracedatei

Wenn das Schlüsselwort TraceFileName in der Datei `db2cli.ini` verwendet wurde, um einen vollständig qualifizierten Dateinamen anzugeben, dann befindet sich die CLI-Tracedatei unter der angegebenen Position. Wenn ein relativer Dateiname für die CLI-Tracedatei angegeben wurde, dann hängt die Position dieser Datei davon ab, welcher Pfad vom Betriebssystem als der aktuelle Pfad der Anwendung angenommen wird.

Anmerkung: Wenn der Benutzer, der die Anwendung ausführt, nicht über die entsprechende Berechtigung zum Schreiben in der Tracedatei im angegebenen Pfad verfügt, dann wird keine Datei generiert und keine Warnung oder Fehlernachricht ausgegeben.

Wenn das Schlüsselwort TracePathName und/oder das Schlüsselwort JDBCTracePathName in der Datei `db2cli.ini` verwendet wurden, um die vollständig qualifizierten Verzeichnisse anzugeben, dann befinden sich die CLI- und DB2-JDBC-Tracedateien unter der angegebenen Position. Wenn ein relativer Verzeichnisname für mindestens eines der beiden Traceverzeichnisse angegeben wurde, dann ermittelt das Betriebssystem die Position anhand des aktuellen Pfads der Anwendung.

Anmerkung: Wenn der Benutzer, der die Anwendung ausführt, nicht über die entsprechende Berechtigung zum Schreiben von Tracedateien im angegebenen Pfad

verfügt, dann wird keine Datei generiert und keine Warnung oder Fehlernachricht ausgegeben. Wenn der angegebene Tracepfad nicht vorhanden ist, dann wird er auch nicht erstellt.

Die CLI- und DB2-JDBC-Tracefunktionen verwenden automatisch die Prozess-ID der Anwendung und die Threadfolgenummer, um die Tracedateien zu benennen, wenn die Schlüsselwörter TracePathName und JDBCTracePathName definiert wurden. Ein CLI-Trace einer Anwendung mit drei Threads kann z. B. die folgenden CLI-Tracedateien generieren: `100390.0`, `100390.1`, `100390.2`.

Ähnlich kann ein DB2-JDBC-Trace einer Java-Anwendung mit zwei Threads die folgenden JDBC-Tracedateien generieren: `7960main.trc`, `7960Thread-1.trc`.

Anmerkung: Wenn das Traceverzeichnis sowohl alte als auch neue Tracedateien enthält, können die Dateidatums- und Zeitmarkeninformationen verwendet werden, um die aktuellsten Tracedateien zu lokalisieren.

Wenn Sie der Ansicht sind, dass keine CLI- oder DB2-JDBC-Traceausgabedateien erstellt wurden, dann gehen Sie wie folgt vor:

- Überprüfen Sie, ob die Tracekonfigurationsschlüsselwörter in der Datei `db2cli.ini` korrekt definiert wurden. Eine schnelle Möglichkeit zur Ausführung dieses Arbeitsschrittes besteht in der Verwendung des Befehls `db2 GET CLI CFG FOR SECTION COMMON` über den Befehlszeilenprozessor.
- Vergewissern Sie sich, dass die Anwendung nach der Aktualisierung der Datei `db2cli.ini` erneut gestartet wird. Während des Anwendungsstarts werden speziell die CLI- und DB2-JDBC-Tracefunktionen initialisiert. Nach der Initialisierung kann die DB2-JDBC-Tracefunktion nicht rekonfiguriert werden. Die CLI-Tracefunktion kann während der Laufzeit rekonfiguriert werden. Dies ist jedoch nur dann der Fall, wenn das Schlüsselwort TraceRefreshInterval vor dem Anwendungsstart entsprechend definiert wurde.

  Anmerkung: Nur die CLI-Schlüsselwörter Trace und TracePIDList können während der Laufzeit rekonfiguriert werden. *Änderungen an anderen CLI-Schlüsselwörtern (einschließlich TraceRefreshInterval) werden nur wirksam, wenn für die Anwendung ein Neustart durchgeführt wird*.
- Wenn das Schlüsselwort TraceRefreshInterval vor dem Anwendungsstart angegeben wurde und wenn das Schlüsselwort Trace zu Beginn auf den Wert 0 gesetzt war, dann müssen Sie überprüfen, ob genügend Zeit verstrichen ist, um der CLI-Tracefunktion das erneute Lesen des Schlüsselwortwerts für Trace zu ermöglichen.
- Bei Verwendung des Schlüsselworts TracePathName und/oder des Schlüsselworts JDBCTracePathName zur Angabe der Traceverzeichnisse müssen Sie sicherstellen, dass die entsprechenden Verzeichnisse vor dem Starten der Anwendung vorhanden sind.
- Vergewissern Sie sich, dass die Anwendung über Schreibzugriff auf die angegebene Tracedatei oder das angegebene Traceverzeichnis verfügt.
- Überprüfen Sie die Umgebungsvariable DB2CLIINIPATH. Wenn sie gesetzt wurde, dann gehen die CLI- und DB2-JDBC-Tracefunktionen davon aus, dass die Datei `db2cli.ini` sich an der in dieser Variablen definierten Position befindet.
- Wenn die Anwendung zur Herstellung einer Verbindung zum CLI-Treiber ODBC verwendet, dann müssen Sie überprüfen, ob eine der Funktionen `SQLConnect()`, `SQLDriverConnect()` oder `SQLBrowseConnect()` erfolgreich aufgerufen werden konnte. Einträge in die CLI-Tracedateien können erst nach der erfolgreichen Herstellung einer Datenbankverbindung geschrieben werden.

Interpretation der CLI-Tracedatei

CLI-Traces beginnen immer mit einem Header, in dem die Prozess-ID und Thread-ID der Anwendung angegeben sind, die den Trace erstellt hat. Außerdem enthält der Header die Uhrzeit, zu der der Trace gestartet wurde, sowie die produktspezifischen Informationen wie z. B. die lokale DB2-Buildstufe und die Version des CLI-Treibers. Beispiel:

```
1   [ Process: 1227, Thread: 1024 ]
2   [ Date, Time:           01-27-2002 13:46:07.535211 ]
3   [ Product:              QDB2/LINUX 7.1.0 ]
4   [ Level Identifier:     02010105 ]
5   [ CLI Driver Version:   07.01.0000 ]
6   [ Informational Tokens: "DB2 v7.1.0","n000510","" ]
```

Anmerkung: Die Tracebeispiele, die in diesem Abschnitt verwendet werden, sind auf der linken Seite des Trace mit Zeilennummern versehen. Diese Zeilennummern wurden hinzugefügt, um die Erläuterungen zu vereinfachen und sind im tatsächlichen CLI-Trache *nicht* enthalten.

Direkt auf den Trace-Header folgen im Allgemeinen einige Traceeinträge, die sich auf die Umgebung und die Reservierung der Verbindungshandle sowie die Initialisierung beziehen. Beispiel:

```
7   SQLAllocEnv( phEnv=&bffff684 )
8       ---> Time elapsed - +9.200000E-004 seconds

9   SQLAllocEnv( phEnv=0:1 )
10      <--- SQL_SUCCESS   Time elapsed - +7.500000E-004 seconds

11  SQLAllocConnect( hEnv=0:1, phDbc=&bffff680 )
12      ---> Time elapsed - +2.334000E-003 seconds

13  SQLAllocConnect( phDbc=0:1 )
14      <--- SQL_SUCCESS   Time elapsed - +5.280000E-004 seconds

15  SQLSetConnectOption( hDbc=0:1, fOption=SQL_ATTR_AUTOCOMMIT, vParam=0 )
16      ---> Time elapsed - +2.301000E-003 seconds

17  SQLSetConnectOption( )
18      <--- SQL_SUCCESS   Time elapsed - +3.150000E-004 seconds

19  SQLConnect( hDbc=0:1, szDSN="SAMPLE", cbDSN=-3, szUID="", cbUID=-3,
                          szAuthStr="", cbAuthStr=-3 )
20      ---> Time elapsed - +7.000000E-005 seconds
21  ( DBMS NAME="DB2/LINUX", Version="07.01.0000", Fixpack="0x22010105" )

22  SQLConnect( )
23      <--- SQL_SUCCESS   Time elapsed - +5.209880E-001 seconds
24  ( DSN=""SAMPLE"" )

25  ( UID=" " )

26  ( PWD="*" )
```

Im oben aufgeführten Tracebeispiel sollten Sie beachten, dass zwei Einträge für jeden CLI-Funktionsaufruf (z. B. Zeilen 19 - 21 und 22 - 26 für den `SQLConnect()`-Funktionsaufruf) vorhanden sind. Dies ist bei CLI-Traces immer der Fall. Der erste Eintrag zeigt die Eingabeparameterwerte für den Funktionsaufruf, während der zweite Eintrag die Ausgabeparameterwerte der Funktion und den an die Anwendung zurückgegebenen Rückkehrcode zeigt.

Das oben aufgeführte Tracebeispiel zeigt, dass die SQLAllocEnv()-Funktion erfolgreich eine Umgebungskennung (phEnv=0:1) zuordnen konnte (Zeile 9). Diese Kennung wurde dann an die SQLAllocConnect()-Funktion übergeben, die erfolgreich eine Datenbankverbindungskennung (phDbc=0:1) zuordnen konnte (Zeile 13). Als Nächstes wurde die SQLSetConnectOption()-Funktion verwendet, um das Attribut SQL_ATTR_AUTOCOMMIT der Verbindung phDbc=0:1 auf SQL_AUTOCOMMIT_OFF (vParam=0) zu setzen (Zeile 15). Abschließend wurde SQLConnect() aufgerufen, um eine Verbindung zur Zieldatenbank (SAMPLE) herzustellen (Zeile 19).

Der Eingabetraceeintrag der SQLConnect()-Funktion (Zeile 21) umfasst den Build und die Fixpackstufe des Zieldatenbankservers. Weitere Informationen, die ebenfalls in diesem Traceeintrag erscheinen können, umfassen die Schlüsselwörter für die Eingabeverbindungszeichenfolge und die Codepages des Clients und des Servers. Beispiel: Die folgenden Informationen wurden ebenfalls im SQLConnect()-Traceeintrag aufgelistet:

```
( Application Codepage=819, Database  Codepage=819,
  Char Send/Recv Codepage=819, Graphic Send/Recv Codepage=819,
  Application Char Codepage=819, Application Graphic Codepage=819 )
```

Dies bedeutet, dass die Anwendung und der Datenbankserver die gleiche Codepage (819) verwenden.

Der zurückgegebene Traceeintrag der SQLConnect()-Funktion enthält außerdem wichtige Verbindungsinformationen (Zeilen 24 - 26 im oben aufgeführten Beispieltrace). Zusätzliche Informationen, die im zurückgegebenen Eintrag möglicherweise angezeigt werden, umfassen alle Werte für das Schlüsselwort PATCH1 oder PATCH2, die für die Verbindung gelten. Wenn z. B. PATCH2=27,28 in der Datei 'db2cli.ini' im Abschnitt COMMON angegeben wurde, dann enthält der zurückgegebene SQLConnect()-Eintrag auch die folgende Zeile:

```
( PATCH2="27,28" )
```

Im Anschluss an die Traceeinträge für die Umgebung und die Verbindung folgen die Traceeinträge für die Anweisung. Beispiel:

```
27  SQLAllocStmt( hDbc=0:1, phStmt=&bffff684 )
28      ---> Time elapsed - +1.868000E-003 seconds

29  SQLAllocStmt( phStmt=1:1 )
30      <--- SQL_SUCCESS   Time elapsed - +6.890000E-004 seconds

31  SQLExecDirect( hStmt=1:1, pszSqlStr="CREATE TABLE GREETING (MSG
                                  VARCHAR(10))", cbSqlStr=-3 )
32      ---> Time elapsed - +2.863000E-003 seconds
33  ( StmtOut="CREATE TABLE GREETING (MSG VARCHAR(10))" )

34  SQLExecDirect( )
35      <--- SQL_SUCCESS   Time elapsed - +2.387800E-002 seconds
```

Im oben aufgeführten Tracebeispiel wurde die Datenbankverbindungskennung (phDbc=0:1) verwendet, um eine Anweisungskennung (phStmt=1:1) zuzuordnen (Zeile 29). Für die Anweisungskennung wurde dann eine unvorbereitete SQL-Anweisung ausgeführt (Zeile 31). Wenn das Schlüsselwort TraceComm=1 in der Datei db2cli.ini gesetzt wurde, dann werden in den Traceeinträgen für den SQLExecDirect()-Funktionsaufruf zusätzliche Client/Server-Kommunikationsinformationen wie folgt angezeigt:

```
SQLExecDirect( hStmt=1:1, pszSqlStr="CREATE TABLE GREETING (MSG
                           VARCHAR(10))", cbSqlStr=-3 )
    ---> Time elapsed - +2.876000E-003 seconds
( StmtOut="CREATE TABLE GREETING (MSG VARCHAR(10))" )
```

```
    sqlccsend( ulBytes - 232 )
    sqlccsend( Handle - 1084869448 )
    sqlccsend( ) - rc - 0, time elapsed - +1.150000E-004
    sqlccrecv( )
sqlccrecv( ulBytes - 163 ) - rc - 0, time elapsed - +2.243800E-002

SQLExecDirect( )
<--- SQL_SUCCESS   Time elapsed - +2.384900E-002 seconds
```

Beachten Sie hierbei die zusätzlichen Informationen zu den sqlccsend()- und sqlccrecv()-Funktionsaufrufen in diesem Traceeintrag. Die Informationen zum sqlccsend()-Aufruf zeigen, wie viele Daten vom Client an den Server gesendet wurden, wie lange die Übertragung dauerte und ob diese Übertragung erfolgreich verlaufen ist (0 = SQL_SUCCESS). Die Informationen zum sqlccrecv()-Aufruf zeigen, wie lange der Client auf eine Antwort vom Server gewartet hat und wieviele Daten in der Antwort enthalten waren.

Häufig werden im CLI-Trace mehrere Anweisungskennungen aufgelistet. Durch eine genaue Auswertung der Anweisungskennungs-ID können Sie den Ausführungspfad einer Anweisungskennung unabhängig von allen anderen Anweisungskennungen innerhalb des Trace einfach verfolgen.

Anweisungsausführungspfade, die im CLI-Trace aufgelistet werden, sind normalerweise komplexer als in dem oben dargestellten Beispiel. Beispiel:

```
36  SQLAllocStmt( hDbc=0:1, phStmt=&bffff684 )
37      ---> Time elapsed - +1.532000E-003 seconds

38  SQLAllocStmt( phStmt=1:2 )
39      <--- SQL_SUCCESS   Time elapsed - +6.820000E-004 seconds

40  SQLPrepare( hStmt=1:2, pszSqlStr="INSERT INTO GREETING VALUES ( ? )",
                  cbSqlStr=-3 )
41      ---> Time elapsed - +2.733000E-003 seconds
42  ( StmtOut="INSERT INTO GREETING VALUES ( ? )" )

43  SQLPrepare( )
44      <--- SQL_SUCCESS   Time elapsed - +9.150000E-004 seconds

45  SQLBindParameter( hStmt=1:2, iPar=1, fParamType=SQL_PARAM_INPUT,
                      fCType=SQL_C_CHAR, fSQLType=SQL_CHAR, cbColDef=14,
                      ibScale=0, rgbValue=&080eca70, cbValueMax=15,
                      pcbValue=&080eca4c )
46      ---> Time elapsed - +4.091000E-003 seconds

47  SQLBindParameter( )
48      <--- SQL_SUCCESS   Time elapsed - +6.780000E-004 seconds

49  SQLExecute( hStmt=1:2 )
50      ---> Time elapsed - +1.337000E-003 seconds
51  ( iPar=1, fCType=SQL_C_CHAR, rgbValue="Hello World!!!", pcbValue=14,
      piIndicatorPtr=14 )

52  SQLExecute( )
53      <--- SQL_ERROR   Time elapsed - +5.951000E-003 seconds
```

Im oben aufgeführten Tracebeispiel wurde die Datenbankverbindungskennung (phDbc=0:1) verwendet, um eine zweite Anweisungskennung (phStmt=1:2) zuzuordnen (Zeile 38). Für diese Anweisungskennung wurde dann eine SQL-Anweisung mit einer Parametermarke vorbereitet (Zeile 40). Als Nächstes wurde ein Eingabeparameter (iPar=1) des korrekten SQL-Typs (SQL_CHAR) an die Parametermarke gebunden (Zeile 45). Abschließend wurde die Anweisung ausge-

führt (Zeile 49). Beachten Sie hierbei, dass sowohl der Inhalt als auch die Länge des Eingabeparameters (rgbValue="Hello World!!!", pcbValue=14) im Trace angezeigt werden (Zeile 51).

Die Ausführung der SQLExecute()-Funktion schlägt fehl (Zeile 52). Wenn die Anwendung eine CLI-Diagnosefunktion wie z. B. SQLError() aufruft, um die Fehlerursache zu untersuchen, dann wird diese Ursache im Trace angezeigt. Beispiel:

```
54  SQLError( hEnv=0:1, hDbc=0:1, hStmt=1:2, pszSqlState=&bffff680,
              pfNativeError=&bfffee78, pszErrorMsg=&bffff280,
              cbErrorMsgMax=1024, pcbErrorMsg=&bfffee76 )
55      ---> Time elapsed - +1.512000E-003 seconds

56  SQLError( pszSqlState="22001", pfNativeError=-302, pszErrorMsg="[IBM][CLI
       Driver][DB2/LINUX] SQL0302N  The value of a host variable in the EXECUTE
       or OPEN statement is too large for its corresponding use.
       SQLSTATE=22001", pcbErrorMsg=157 )
57      <--- SQL_SUCCESS   Time elapsed - +8.060000E-004 seconds
```

Die Fehlernachricht, die zurückgegeben wird (Zeile 56), enthält den nativen DB2-Fehlercode, der generiert wurde (SQL0302N), den SQLSTATE-Wert, der diesem Code entspricht (SQLSTATE=22001), und eine kurze Beschreibung des Fehlers. In diesem Beispiel ist die Fehlerursache offensichtlich. In Zeile 49 versucht die Anwendung, eine Zeichenfolge mit 14 Zeichen in eine Spalte einzufügen, die in Zeile 31 als VARCHAR(10) definiert wurde.

Wenn die Anwendung nicht auf eine Warnung oder einen Fehlercode für die CLI-Funktion reagiert, indem eine Diagnosefunktion wie z. B. SQLError() aufgerufen wird, wird die Warnung bzw. die Fehlernachricht dennoch in den CLI-Trace geschrieben. Allerdings ist es möglich, dass die Position der Nachricht innerhalb des Trace nicht in der Nähe der Position liegt, bei der der Fehler tatsächlich aufgetreten ist. Darüber hinaus gibt der Trace an, dass die Fehlernachricht oder die Warnung nicht von der Anwendung abgerufen wurde. Beispiel: Falls kein Abruf erfolgt, erscheint die Fehlernachricht im oben aufgeführten Beispiel möglicherweise erst später und scheinbar ohne Zusammenhang mit dem CLI-Funktionsaufruf wie folgt:

```
SQLDisconnect( hDbc=0:1 )
    ---> Time elapsed - +1.501000E-003 seconds
    sqlccsend( ulBytes - 72 )
    sqlccsend( Handle - 1084869448 )
    sqlccsend( ) - rc - 0, time elapsed - +1.080000E-004
    sqlccrecv( )
sqlccrecv( ulBytes - 27 ) - rc - 0, time elapsed - +1.717950E-001
( Unretrieved error message="SQL0302N  The value of a host variable in the
  EXECUTE or OPEN statement is too large for its corresponding use.
  SQLSTATE=22001" )

SQLDisconnect( )
    <--- SQL_SUCCESS   Time elapsed - +1.734130E-001 seconds
```

Im letzten Teil eines CLI-Trace sollte die Anwendung angezeigt werden, die die Datenbankverbindung und die Umgebungskennungen freigibt, die zuvor im Trace zugeordnet wurden. Beispiel:

```
58  SQLTransact( hEnv=0:1, hDbc=0:1, fType=SQL_ROLLBACK )
59      ---> Time elapsed - +6.085000E-003 seconds
60  ( ROLLBACK=0 )

61  SQLTransact( )
        <--- SQL_SUCCESS   Time elapsed - +2.220750E-001 seconds

62  SQLDisconnect( hDbc=0:1 )
63      ---> Time elapsed - +1.511000E-003 seconds
```

```
64  SQLDisconnect( )
65      <---- SQL_SUCCESS   Time elapsed - +1.531340E-001 seconds

66  SQLFreeConnect( hDbc=0:1 )
67      ----> Time elapsed - +2.389000E-003 seconds

68  SQLFreeConnect( )
69      <---- SQL_SUCCESS   Time elapsed - +3.140000E-004 seconds

70  SQLFreeEnv( hEnv=0:1 )
71      ----> Time elapsed - +1.129000E-003 seconds

72  SQLFreeEnv( )
73      <---- SQL_SUCCESS   Time elapsed - +2.870000E-004 seconds
```

Interpretation der JDBC-Tracedatei

DB2-JDBC-Traces beginnen immer mit einem Header, in dem wichtige Systeminformationen wie z. B. die Schlüsselumgebung, Variableneinstellungen, das SDK für Java oder die JRE-Version, die Version des DB2-JDBC-Treibers und die Version des DB2-Builds angegeben sind. Beispiel:

```
1   ========================================================
2   |    Trace beginning on 2002-1-28 7:21:0.19
3   ========================================================

4   System Properties:
5   ------------------
6   user.language = en
7   java.home = c:\Program Files\SQLLIB\java\jdk\bin\..
8   java.vendor.url.bug =
9   awt.toolkit = sun.awt.windows.WToolkit
10  file.encoding.pkg = sun.io
11  java.version = 1.1.8
12  file.separator = \
13  line.separator =
14  user.region = US
15  file.encoding = Cp1252
16  java.compiler = ibmjitc
17  java.vendor = IBM Corporation
18  user.timezone = EST
19  user.name = db2user
20  os.arch = x86
21  java.fullversion = JDK 1.1.8 IBM build n118p-19991124 (JIT ibmjitc
                       V3.5-IBMJDK1.1-19991124)
22  os.name = Windows NT
23  java.vendor.url = http://www.ibm.com/
24  user.dir = c:\Program Files\SQLLIB\samples\java
25  java.class.path =
         .:C:\Program Files\SQLLIB\lib;C:\Program Files\SQLLIB\java;
         C:\Program Files\SQLLIB\java\jdk\bin\
26  java.class.version = 45.3
27  os.version = 5.0
28  path.separator = ;
29  user.home = C:\home\db2user
30  ----------------------------------------
```

Anmerkung: Die Tracebeispiele, die in diesem Abschnitt verwendet werden, sind auf der linken Seite des Trace mit Zeilennummern versehen. Diese Zeilennummern wurden hinzugefügt, um die Erläuterungen zu vereinfachen und sind im tatsächlichen DB2-JDBC-Trace *nicht* enthalten.

Direkt im Anschluss an den Trace-Header befindet sich normalerweise eine Reihe von Traceeinträgen, die in Zusammenhang mit der Initialisierung der JDBC-Umgebung und der Herstellung der Datenbankverbindung stehen. Beispiel:

```
31  jdbc.app.DB2Driver -> DB2Driver() (2002-1-28 7:21:0.29)
32  | Loaded db2jdbc from java.library.path
33  jdbc.app.DB2Driver <- DB2Driver() [Time Elapsed = 0.01]

34  DB2Driver - connect(jdbc:db2:sample)

35  jdbc.app.DB2ConnectionTrace -> connect( sample, info, db2driver, 0, false )
                 (2002-1-28 7:21:0.59)
36  | 10: connectionHandle = 1
37  jdbc.app.DB2ConnectionTrace <- connect() [Time Elapsed = 0.16]

38  jdbc.app.DB2ConnectionTrace -> DB2Connection (2002-1-28 7:21:0.219)
39  | source = sample
40  | Connection handle = 1
41  jdbc.app.DB2ConnectionTrace <- DB2Connection
```

Im oben aufgeführten Tracebeispiel wurde in Zeile 31 eine Anforderung zum Laden des DB2-JDBC-Treibers abgesetzt. Diese Anforderung wurde erfolgreich ausgeführt. Ein entsprechender Rückgabewert befindet sich in Zeile 33.

Die DB2-JDBC-Tracefunktion verwendet spezielle Java-Klassen, um die Traceinformationen zu erfassen. Im oben aufgeführten Tracebeispiel hat eine dieser Traceklassen (DB2ConnectionTrace) zwei Traceeinträge mit den Nummern 35 - 37 und 38 - 41 generiert.

In Zeile 35 wird die Methode connect() dargestellt, die aufgerufen wird. Außerdem werden hier die Eingabeparameter für diesen Methodenaufruf angegeben. In Zeile 37 wird dargestellt, dass die Methode connect() erfolgreich ausgeführt wurde, während in Zeile 36 der Ausgabeparameter dieses Aufrufs (Connection handle = 1) angegeben ist.

Im Anschluss an die Einträge zur Verbindung befinden sich im JDBC-Trace normalerweise Einträge zur Anweisung. Beispiel:

```
42  jdbc.app.DB2ConnectionTrace -> createStatement() (2002-1-28 7:21:0.219)
43  | Connection handle = 1
44  | jdbc.app.DB2StatementTrace -> DB2Statement( con, 1003, 1007 )
          (2002-1-28 7:21:0.229)
45  | jdbc.app.DB2StatementTrace <- DB2Statement() [Time Elapsed = 0.0]
46  | jdbc.app.DB2StatementTrace -> DB2Statement (2002-1-28 7:21:0.229)
47  | | Statement handle = 1:1
48  | jdbc.app.DB2StatementTrace <- DB2Statement
49  jdbc.app.DB2ConnectionTrace <- createStatement - Time Elapsed = 0.01

50  jdbc.app.DB2StatementTrace -> executeQuery(SELECT * FROM EMPLOYEE WHERE
                                  empno = 000010) (2002-1-28 7:21:0.269)
51  | Statement handle = 1:1
52  | jdbc.app.DB2StatementTrace -> execute2( SELECT * FROM EMPLOYEE WHERE
                                     empno = 000010 ) (2002-1-28 7:21:0.269)
52  | | jdbc.DB2Exception -> DB2Exception() (2002-1-28 7:21:0.729)
53  | | | 10: SQLError = [IBM][CLI Driver][DB2/NT] SQL0401N  The data types of
              the operands for the operation "=" are not compatible.
              SQLSTATE=42818
54  | | |     SQLState = 42818
55  | | |     SQLNativeCode = -401
56  | | |     LineNumber = 0
57  | | |     SQLerrmc = =
58  | | jdbc.DB2Exception <- DB2Exception() [Time Elapsed = 0.0]
59  | jdbc.app.DB2StatementTrace <- executeQuery - Time Elapsed = 0.0
```

In Zeile 42 und Zeile 43 hat die Klasse DB2ConnectionTrace gemeldet, dass die JDBC-Methode createStatement() mit der Verbindungskennung 1 aufgerufen wurde. In dieser Methode wurde die interne Methode DB2Statement() aufgerufen. Dieser Vorgang wurde von einer anderen Klasse der DB2-JDBC-Tracefunktion (DB2StatementTrace) zurückgemeldet. Beachten Sie hierbei, dass dieser interne Methodenaufruf innerhalb des Traceeintrags 'verschachtelt' erscheint. In den Zeilen 47 - 49 wird gezeigt, dass die Methoden erfolgreich ausgeführt wurden und dass die Anweisungskennung 1:1 zugeordnet wurde.

In Zeile 50 wird eine SQL-Abfragemethode für die Anweisung 1:1 aufgerufen, der Aufruf generiert jedoch eine Ausnahmebedingung (Zeile 52). Die Fehlernachricht wird in Zeile 53 zurückgemeldet und enthält den nativen DB2-Fehlercode, der generiert wurde (SQL0401N), den SQLSTATE-Wert, der diesem Code entspricht (SQLSTATE=42818), und eine kurze Beschreibung des Fehlers. In diesem Beispiel ist der Fehler darauf zurückzuführen, dass die Spalte EMPLOYEE.EMPNO als CHAR(6) definiert ist und nicht (wie in der Abfrage angenommen) als ganzzahliger Wert.

Plattformspezifische Tools

Diagnosetools (Windows)

In diesem Abschnitt werden drei nützliche Diagnosetools für Windows-Systeme beschrieben.

Die folgenden Diagnosetools stehen unter den Windows-Betriebssystemen zur Verfügung:

Ereignisanzeige, Systemmonitor und andere Verwaltungstools
Der Ordner **Verwaltung** stellt eine Reihe von Diagnosemöglichkeiten bereit, zu denen der Zugriff auf das Ereignisprotokoll und der Zugriff auf Systemleistungsinformationen gehören.

Task-Manager
Der Task-Manager zeigt alle auf dem Windows-Server aktiven Prozesse zusammen mit Informationen zur Speicherbelegung an. Verwenden Sie dieses Tool, um zu ermitteln, welche DB2-Prozesse aktiv sind, und um Leistungsprobleme zu diagnostizieren. Mithilfe dieses Tools können Sie die Speichernutzung, die Speichergrenzen, den verwendeten Auslagerungsspeicher und den Speicherverlust für ein Prozess feststellen.

Zum Öffnen des Task-Managers drücken Sie die Tasten Strg + Alt + Entf, und klicken Sie in den verfügbaren Optionen **Task-Manager** an.

Dr. Watson
Das Dienstprogramm Dr. Watson wird aufgerufen, wenn ein allgemeiner Schutzfehler (GPF - General Protection Fault) auftritt. Es protokolliert Daten, die bei der Diagnose eines Problems behilflich sein können, und speichert diese Informationen in einer Datei. Sie müssen dieses Dienstprogramm durch Eingabe des Befehls `drwatson` in die Eingabeaufforderung starten.

Diagnosetools (Linux und UNIX)

In diesem Abschnitt werden einige wesentliche Befehle zur Fehlerbehebung und zur Leistungsüberwachung auf Linux- und UNIX-Plattformen beschrieben.

Zum Anzeigen von Details zu diesen Befehlen setzen Sie dem Befehlsnamen den Befehl 'man' in der Befehlszeile voran. Verwenden Sie diese Befehle zum Erfassen und Verarbeiten von Daten, die bei der Ermittlung der Ursache eines Fehlers auf Ihrem System helfen können. Sobald die Daten erfasst sind, können sie von einer Person untersucht werden, die mit dem Problem vertraut ist, oder auf Anfrage IBM Software Support zur Verfügung gestellt werden.

Befehle zur Fehlerbehebung (AIX)

Die folgenden AIX-Systembefehle sind zur DB2-Fehlerbehebung nützlich:

errpt Der Befehl **errpt** meldet Systemfehler, wie zum Beispiel Hardwarefehler und Netzausfälle.

- Verwenden Sie den Befehl **errpt**, um eine Übersicht anzuzeigen, die eine Zeile pro Fehler enthält.
- Verwenden Sie den Befehl **errpt -a**, um eine detailliertere Anzeige aufzurufen, die eine Seite pro Fehler bereitstellt.
- Verwenden Sie den Befehl **errpt -a -j 1581762B**, um Fehler mit der Fehlernummer "1581762B" anzuzeigen.
- Geben Sie den Befehl **errpt | grep SYSVMM** ein, um zu ermitteln, ob es in der Vergangenheit Situationen gab, in denen kein Paging-Bereich mehr zur Verfügung stand.
- Wenn Sie herausfinden wollen, ob es Probleme mit der Token-Ring-Karte oder der Festplatte gibt, prüfen Sie die Ausgabe des Befehls errpt auf die Zeichenfolgen "disk" und "tr0".

lsps Der Befehl **lsps -a** überwacht die Verwendung von Paging-Bereich und zeigt Informationen dazu an.

lsattr Dieser Befehl zeigt verschiedene Parameter des Betriebssystems an. Verwenden Sie zum Beispiel den folgenden Befehl, um die Größe des Realspeichers der jeweiligen Datenbankpartition zu ermitteln:

```
lsattr -l sys0 -E
```

xmperf

Für AIX-Systeme mit Motif startet dieser Befehl einen grafischen Monitor, der Leistungsdaten zum jeweiligen System erfasst und anzeigt. Der Monitor zeigt dreidimensionale Diagramme für jede Datenbankpartition in einem Fenster an und eignet sich gut zur Überwachung auf hoher Ebene. Wenn jedoch das Aktivitätsvolumen niedrig ist, hat die Ausgabe dieses Monitors nur begrenzten Wert.

spmon

Wird die Systempartitionierung als Teil von PSSP (Parallel System Support Program) verwendet, müssen Sie unter Umständen prüfen, ob der SP-Switch auf allen Workstations aktiv ist. Zum Anzeigen des Status aller Datenbankpartitionen verwenden Sie von der Steuerworkstation aus einen der folgenden Befehle:

- **spmon -d** für ASCII-Ausgabe
- **spmon -g** für eine grafische Benutzerschnittstelle

Alternativ können Sie den Befehl **netstat -i** auf einer Datenbankpartitionsworkstation verwenden, um zu prüfen, ob der Switch inaktiv ist. Wenn der Switch inaktiv ist, steht neben der Datenbankpartition ein Stern (*). Beispiel:

```
css0* 65520 <Link>0.0.0.0.0.0
```

Der Stern wird nicht angezeigt, wenn der Switch aktiv ist.

Befehle zur Fehlerbehebung (Linux und UNIX)

Die folgenden Befehle gelten für alle Linux- und UNIX-Systeme, einschließlich AIX, sofern nicht anders angegeben.

df Mit dem Befehl **df** können Sie prüfen, ob Dateisysteme voll sind.

- Geben Sie den Befehl **df** ein, um anzuzeigen, wie viel freier Speicher in allen Dateisystemen (einschließlich angehängter Dateisysteme) verfügbar ist.
- Geben Sie den Befehl **df | grep dev** ein, um zu prüfen, wie viel freier Speicherbereich in allen Dateisystemen vorhanden ist, deren Namen die Zeichenfolge "dev" enthalten.
- Geben Sie den Befehl **df /home** ein, um zu prüfen, wie viel Speicherbereich in Ihrem Ausgangsdateisystem verfügbar ist.
- Geben Sie den Befehl **df /tmp** ein, um zu prüfen, wie viel freier Speicherbereich im Dateisystem "tmp" verfügbar ist.
- Um zu ermitteln, ob auf der Maschine ausreichend freier Speicherplatz verfügbar ist, prüfen Sie die Ausgabe der folgenden Befehle: **df /usr , df /var , df /tmp und df /home**

truss Dieser Befehl dient zur Traceerstellung für Systemaufrufe in mindestens einem Prozess.

pstack Der Befehl **/usr/proc/bin/pstack** steht unter Solaris 2.5.1 oder einer späteren Version zur Verfügung und zeigt Stack-Traceback-Informationen an. Das Verzeichnis '**/usr/proc/bin**' enthält weitere Tools zur Behebung von Fehlern in Prozessen, die blockiert zu sein scheinen.

Tools zur Leistungsüberwachung

Die folgenden Tools stehen zur Überwachung der Leistung Ihres Systems zur Verfügung:

vmstat

Dieser Befehl hilft bei der Feststellung, ob ein Prozess blockiert ist oder lediglich viel Zeit benötigt.Sie können die Paging-Rate überwachen, die sich in den Spalten "pi" (page in) und "po" (page out) ablesen lässt. Andere wichtige Spalten sind die Größe des zugeordneten virtuellen Speichers (avm - allocated virtual storage) und die Größe des freien virtuellen Speichers (fre - free virtual storage).

iostat Dieser Befehl dient zur Überwachung von E/A-Aktivitäten. Sie können anhand der Lese- und Schreibgeschwindigkeit die Zeit abschätzen, die für bestimmte SQL-Operationen benötigt wird (falls diese die einzige Aktivität auf dem System sind).

netstat

Mithilfe dieses Befehls können Sie den Netzverkehr auf jeder Datenbankpartition und die Anzahl angetroffener Fehlerpakete ermitteln. Er leistet nützliche Dienste bei der Isolierung von Netzproblemen.

Datei 'system'

Die Datei `/etc/system` ist in der Solaris-Betriebsumgebung verfügbar und enthält Definitionen für Kernelkonfigurationsgrenzwerte, wie zum Beispiel die maximale Anzahl von Benutzern, die gleichzeitig auf dem System zuzulassen sind, die maximale Anzahl von Prozessen pro Benutzer sowie die

Grenzwerte der Interprozesskommunikation (Inter-Process Communication, IPC) für Größe und Anzahl von Ressourcen. Diese Begrenzungen sind wichtig, weil sie sich auf die DB2-Leistung auf einer Maschine mit dem Solaris-Betriebssystem auswirken.

Kapitel 6. Fehlerbehebung für DB2-Datenbank

Bei der Fehlerbehebung geht es im Wesentlichen darum, einen Fehler einzugrenzen und zu bestimmen, um anschließend nach einer Lösung suchen zu können. In diesem Abschnitt wird die Fehlerbehebung für bestimmte Funktionen von DB2-Produkten erläutert.

Wenn allgemeine Fehler bekannt werden sollten, werden sie in diesem Abschnitt in Form von Checklisten hinzugefügt, sobald nähere Informationen verfügbar sind. Sollten Sie den bei Ihnen aufgetretenen Fehler nicht mithilfe der Checkliste beheben können, empfiehlt es sich, weitere Diagnosedaten zu sammeln, die Sie dann selbst analysieren oder zur Analyse bei IBM Software Support einreichen können.

Mithilfe der folgenden Fragen ermitteln Sie die entsprechenden Fehlerbehebungstasks:

1. Haben Sie alle bekannten Fixpacks angewendet? Falls nicht, sollten Sie die im Abschnitt „Fixpacks anwenden" im Handbuch *DB2-Server - Installation* beschriebenen Schritte ausführen.
2. Tritt das Problem in den folgenden Fällen auf?
   - Bei der Installation von DB2-Datenbankservern oder -Clients? Ist dies der Fall, siehe den Abschnitt „Erfassen Sie Daten für Installationsprobleme" an anderer Stelle in diesem Handbuch.
   - Beim Erstellen, Löschen, Aktualisieren oder Aufrüsten einer Instanz oder des DB2-Verwaltungsservers (DAS)? Ist dies der Fall, fahren Sie mit dem Abschnitt „Erfassen von Daten für DAS- und Instanzverwaltungsprobleme" an anderer Stelle in diesem Handbuch fort.
   - Beim Versetzen von Daten mit den Befehlen **EXPORT**, **IMPORT**, **LOAD** oder **db2move**? Ist dies der Fall, fahren Sie mit dem Abschnitt „Erfassen von Daten für Probleme beim Versetzen von Daten" an anderer Stelle in diesem Handbuch fort.

Passt das Problem zu keiner dieser Kategorien, sollten Sie dennoch allgemeine Diagnosedaten bereithalten, wenn Sie Kontakt mit IBM Software Support aufnehmen. Es ist wichtig, dass Sie .

Erfassen von Daten für DB2

In manchen Fällen kann ein Problem nicht durch die Behebung der Symptome gelöst werden. In diesen Situationen ist die Erfassung von Diagnosedaten erforderlich. Die Diagnosedaten, die erfasst werden müssen, sowie die Quellen dieser Diagnosedaten sind abhängig von der Art des untersuchten Problems. Diese Schritte beschreiben die Erfassung der grundlegenden Informationen, die Sie normalerweise bereitstellen müssen, wenn Sie ein Problem an IBM Software Support weitermelden.

Vorbereitende Schritte

Um eine möglichst umfassende Ausgabe zu erhalten, sollte das Dienstprogramm **db2support** vom Instanzeigner aufgerufen werden.

Vorgehensweise

Für die Erfassung der grundlegenden Diagnoseinformationen in einem komprimierten Dateiarchiv geben Sie den Befehl **db2support** wie folgt ein:

```
db2support output_directory -s -d database_name -c
```

Die Verwendung der Option **-s** bewirkt die Angabe einiger Systemdetails zur verwendeten Hardware und zum Betriebssystem. Die Verwendung der Option **-d** bewirkt die Angabe von Details zur angegebenen Datenbank. Die Verwendung der Option **-c** ermöglicht einen Verbindungsversuch zur angegebenen Datenbank. Die so erfasste Ausgabe wird benutzerfreundlich in einem komprimierten Archiv (ZIP), `db2support.zip`, gespeichert und kann so an ein beliebiges System übertragen und dort extrahiert werden.

Nächste Schritte

Für bestimmte Symptome oder für Probleme bei einer bestimmten Komponente des Produkts müssen möglicherweise zusätzliche Daten gesammelt werden. Informationen hierzu finden Sie in den Dokumenten zur Datenerfassung für den jeweiligen Problemtyp.

Als nächstes können Sie eine der folgenden Tasks ausführen:

- Die Daten analysieren.
- Die Daten an IBM Software Support übergeben.

Erfassen von Daten für Probleme beim Versetzen von Daten

Wenn bei der Ausführung von Befehlen für das Versetzen von Daten Fehler auftreten und die Ursache des Problems nicht festgestellt werden kann, können Sie Diagnosedaten erfassen, die Sie selbst oder die Mitarbeiter von IBM Software Support dazu verwenden können, das Problem zu diagnostizieren und zu beheben.

Führen Sie für Ihre Situation geeignete Anweisungen der folgenden Liste zum Erfassen von Daten aus:

- Zum Erfassen von Daten für Probleme beim Befehl **db2move** wechseln Sie in das Verzeichnis, in dem Sie den Befehl abgesetzt haben. Suchen Sie die folgende(n) Datei(en), abhängig von der im Befehl angegebenen Aktion:
  - Suchen Sie für die Aktion COPY nach Dateien mit dem Namen `COPY.zeitmarke.ERR` und `COPYSCHEMA.zeitmarke.MSG`. Wenn Sie darüber hinaus den Modus LOAD_ONLY oder DDL_AND_LOAD angegeben haben, suchen Sie außerdem nach einer Datei mit dem Namen `LOADTABLE.zeitmarke.MSG`.
  - Suchen Sie für die Aktion EXPORT nach der Datei EXPORT.out.
  - Suchen Sie für die Aktion IMPORT nach der Datei IMPORT.out.
  - Suchen Sie für die Aktion LOAD nach der Datei LOAD.out.
- Zum Erfassen von Daten für Probleme bei den Befehlen **EXPORT**, **IMPORT** oder **LOAD** stellen Sie fest, ob der Befehl den Parameter MESSAGES enthielt. War dies der Fall, erfassen Sie die Daten der entsprechenden Ausgabedatei. Diese Dienstprogramme verwenden das aktuelle Verzeichnis und das Standardlaufwerk als Ziel, falls keine anderen Angaben gemacht werden.
- Suchen Sie zum Erfassen von Daten für Probleme beim Befehl **REDISTRIBUTE** nach der Datei "*datenbankname.name_der_datenbankpartitionsgruppe.uhrzeit*" unter Linux und UNIX bzw. "*datenbankname.name_der_datenbankpartitions-*

gruppe.datum.uhrzeit" unter Windows. Sie befindet sich im Verzeichnis *$HOME*/sqllib/db2dump bzw. $DB2PATH\sqllib\redist, wobei *$HOME* das Ausgangsverzeichnis des Instanzeigners ist.

Erfassen von Daten für DAS- und Instanzverwaltungsprobleme

Wenn bei der Ausführung des DB2-Verwaltungsservers (DAS) oder der Instanzverwaltung Fehler auftreten und die Ursache des Fehlers nicht festgestellt werden kann, können Sie Diagnosedaten erfassen, die Sie selbst oder die Mitarbeiter von IBM Software Support dazu verwenden können, den Fehler zu diagnostizieren und zu beheben.

Diese Schritte beziehen sich ausschließlich auf die Situationen, in denen Sie den Fehler reproduzieren können und in denen DB2 unter Linux oder UNIX ausgeführt wird.

Gehen Sie wie folgt vor, um Diagnosedaten für Probleme beim DAS oder bei der Instanzverwaltung zu erfassen:

1. Wiederholen Sie den fehlschlagenden Befehl mit aktivierter Tracefunktion bzw. aktiviertem Debugmodus. Beispielbefehle:

   ```
   db2setup -t trace.out
   dascrt -u DASUSER -d
   dasdrop -d
   dasmigr -d
   dasupdt -d
   db2icrt -d INSTNAME
   db2idrop INSTNAME -d
   db2iupgrade -d INSTNAME
   db2iupdt -d INSTNAME
   ```

2. Suchen Sie die Diagnosedatei. Möglicherweise sind mehrere Dateien vorhanden; vergleichen Sie daher die Zeitmarken, um sicherzustellen, dass Sie alle relevanten Dateien verwenden.

   Die Ausgabe wird standardmäßig in das Verzeichnis /tmp gestellt.

   Beispielnamen sind dascrt.log, dasdrop.log , dasupdt.log , db2icrt.log.PID, db2idrop.log.PID, db2iupgrade.log.PID und db2iupdt.log.PID. Dabei ist PID die Prozess-ID.

3. Stellen Sie die Diagnosedatei(en) IBM Software Support zur Verfügung.

Besteht das Problem im Fehlschlagen des Befehls **db2start** oder START DATABASE MANAGER, suchen Sie nach der Datei db2start.*zeitmarke*.log im Verzeichnis insthome/sqllib/log, wobei insthome das Ausgangsverzeichnis des Instanzeigners ist. Besteht das Problem im Fehlschlagen des Befehls **db2stop** oder STOP DATABASE MANAGER, suchen Sie nach der Datei db2stop.*zeitmarke*.log. Diese Dateien sind nur dann vorhanden, wenn der Datenbankmanager nicht innerhalb der im Konfigurationsparameter **start_stop_time** des Datenbankmanagers angegebenen Zeit auf den Befehl geantwortet hat.

Erfassen von Diagnosedaten zu bestimmten Leistungsproblemen

Die Erfassung von Diagnosedaten zu Leistungsproblemen, die möglichst genau auf das jeweilige Problem ausgerichtet sind, unterstützt Sie dabei, die Auswirkungen der Erfassung des gesamten Spektrums leistungsbezogener Diagnosedaten zu vermeiden. Dies ist insbesondere auf einem System von Bedeutung, auf dem die verfügbaren Ressourcen bereits knapp sind. Die Leistungsprobleme, für die spezielle

Diagnosedaten erfasst werden können, müssen in Zusammenhang mit der Prozessorauslastung sowie mit den Speichereinheiten oder Datenbankverbindungen stehen.

Die Erfassung von Diagnosedaten für diese Probleme kann gestartet werden, sobald Sie feststellen, dass mindestens ein Fehlersymptom vorhanden ist. Um die erforderlichen Daten für die Fehlerdiagnose zu erfassen, müssen Sie den Befehl **db2fodc** zusammen mit einem der Parameter **-connections**, **-cpu** oder **-memory** verwenden.

Für nur gelegentlich auftretende Probleme können Sie auch mit dem Befehl **db2fodc -detect** eine Schwellenwertregel erstellen, mit der eine bestimmte Fehlerbedingung festgestellt und die Erfassung von Diagnosedaten gestartet werden kann, wenn die von Ihnen angegebenen Triggerbedingungen überschritten werden.

Symptome

Abhängig von dem jeweils vorhandenen Problem können Sie möglicherweise einige der folgenden Symptome auf Ihrem Datenserver beobachten:

- Bei Leistungsproblemen in Zusammenhang mit Datenbankverbindungen kann es zu plötzlichen Spitzenwerten bei der Anzahl der Anwendungen kommen, die sich im Ausführungs- oder Kompilierungsstatus befinden, oder es kann vorkommen, dass neue Datenbankverbindungen zurückgewiesen werden.
- Bei Leistungsproblemen in Zusammenhang mit der Prozessorauslastung kann es zu hohen Prozessorauslastungsraten, zu einer hohen Anzahl ausgeführter Prozesse oder zu langen Prozessorwartezeiten kommen.
- Bei Leistungsproblemen in Zusammenhang mit der Speicherbelegung kann es möglich sein, dass kein freier Speicherbereich verfügbar ist, dass der Auslagerungsspeicher zu einem hohen Anteil belegt ist, dass eine übermäßig hohe Pagingaktivität zu verzeichnen ist oder dass möglicherweise ein Speicherleck aufgetreten ist.

Fehlerdiagnose

Um die Anwendungsverbindungen zur Datenbank zu überprüfen, können Sie den Befehl **db2pd -applications** ausgeben. Abgesehen davon, dass das Feld `Status` Aufschluss über die Anzahl der zur Datenbank existierenden Verbindungen gibt, können Sie anhand dieses Feldes feststellen, ob sich diese Anwendungen im Ausführungs- oder Kompilierungsstatus befinden. Wenn die Anzahl der Anwendungen, die über eine Datenbankverbindung verfügen, im Zeitverlauf weiter steigt, ohne dass jemals ein Rückgang zu verzeichnen ist, oder falls die Anzahl der Anwendungsverbindungen plötzliche Spitzenwerte aufweist, dann kann möglicherweise ein Problem bei den Verbindungen vorliegen, das einer eingehenderen Diagnose bedarf.

Unter UNIX- und Linux-Betriebssystemen können Sie überprüfen, ob ein potenzielles Problem in Bezug auf die Prozessorauslastung vorliegt, indem Sie den Befehl **vmstat** mit den geeigneten Parametern für das jeweilige Betriebssystem ausgeben, um den Wert für die Prozessorauslastung des Benutzers (`us`) zurückzugeben. Sie können außerdem die Werte überprüfen, die für `sy` zurückgegeben werden, um die Informationen zur Systemprozessorauslastung und zur kombinierten Benutzer- und Systemprozessorauslastung abzurufen. Sie können den Befehl **vmstat** mehrmals ausgeben, um präzisere Rückschlüsse darauf zu ermöglichen, ob die Fehlersymptome im Zeitverlauf permanent auftreten. Wenn Sie feststellen, dass der Wert für `us` häufig über 90 % liegt, dann ist dies ein Hinweis darauf, dass möglicherwei-

se ein Problem in Bezug auf die Prozessorauslastung vorliegt, das genauer diagnostiziert werden muss. Unter Windows-Betriebssystemen können Sie den Befehl **db2pd -vmstat** verwenden, um vergleichbare Informationen abzurufen.

Um zu überprüfen, ob möglicherweise ein speicherbezogenes Problem vorliegt, können Sie den entsprechenden Befehl auf Betriebssystemebene absetzen, der den Wert für den belegten Auslagerungsspeicher zurückgibt. (Unter Linux-Betriebssystemen können Sie z. B. den Befehl **free** verwenden.) Wenn der Wert für den belegten Auslagerungsspeicher durchgehend höher als 90 % ist, dann liegt möglicherweise ein speicherbezogenes Leistungsproblem vor.

Fehlerbehebung

In den folgenden Beispielen wird der Befehl **db2fodc** zusammen mit den Parametern **-connection**, **-cpu**, **-memory** und **-detect** verwendet. Sie können diese Befehle selbst ausführen. Dabei müssen Sie jedoch die möglichen Auswirkungen berücksichtigen. Auf einem System, auf dem erhebliche Ressourcenknappheit herrscht, kann sich die Systemsituation selbst durch die Erfassung von stark problemspezifischen Diagnosedaten in einem Maß weiter verschlechtern, die nicht akzeptabel ist. In diesem Fall sollten Sie diese Befehle unter der Anleitung des IBM Support ausführen. Normalerweise werden die erfassten Daten zur Analyse an den IBM Technical Support gesendet.

- Zur Erfassung von Diagnosedaten für Probleme im Zusammenhang mit Datenbankverbindungen geben Sie den folgenden Befehl ein:

  ```
  db2fodc -connections
  ```

  Mit diesem Befehl können verbindungsbezogene Diagnosedaten erfasst und in einem Verzeichnis mit dem Namen FODC_Connections_*zeitmarke*_*teilkomponente* gespeichert werden, wobei *zeitmarke* die Uhrzeit angibt, zu der der Befehl **db2fodc -connections** ausgeführt wurde, und *teilkomponente* die Teilkomponente(n) darstellt, für die die Erfassung ausgeführt wurde.
- Zur Erfassung von Diagnosedaten für Probleme in Zusammenhang mit der Prozessorauslastung können Sie, wenn bereits entsprechende Problemsymptome überwacht werden, den folgenden Befehl absetzen:

  ```
  db2fodc -cpu
  ```

  Der Befehl **db2fodc -cpu** erfasst prozessorbezogene Diagnosedaten und speichert sie in einem Verzeichnis mit dem Namen FODC_Cpu_*zeitmarke*_*teilkomponente*. Dabei steht *zeitmarke* für den Zeitpunkt, zu dem der Befehl **db2fodc -connections** ausgeführt wurde, und *teilkomponente* für die Teilkomponente(n), für die die Erfassung ausgeführt wurde. Alternativ hierzu können Sie eine Variante des folgenden Befehls absetzen, wenn das Problem nur vorübergehend auftritt oder wenn Sie Ihr System vorab zur Erfassung von Diagnosedaten bei Auftreten bestimmter Fehlerbedingungen konfigurieren wollen:

  ```
  db2fodc -cpu basic -detect us">=90" rqueue"<=1000" condition="and" interval="2"
  sleeptime="500" triggercount="3" iteration="4" duration="500" -member all
  ```

  Mit dem Parameter **-detect** und den angegebenen Schwellenwertregeln wird die Erfassung von prozessorbezogenen Informationen verzögert, bis die Triggerbedingungen, die in der Schwellenwertregel angegeben wurden, festgestellt wurden. Sie können eigene Regeln für den Parameter **-detect** angeben, um zu ermitteln, wann die Erfassung von Diagnosedaten gestartet werden soll. Im oben aufgeführten Beispiel müssen die Bedingungen für die Prozessorauslastung des Benutzers und die Ausführungswarteschlange beide dreimal im Verlauf von drei Iterationen erfüllt werden. Dies bedeutet, dass die Triggerbedingungen für insge-

samt sechs Sekunden vorhanden sein müssen, um die Erfassung von Diagnosedaten für alle Teilkomponenten auszulösen. (Ein Triggerzähler von 3 x 2 Sekundenintervallen = eine Triggerbedingung, die für sechs Sekunden vorhanden sein muss.) Die Option `iteration` gibt an, dass die Erkennung von Triggerbedingungen und die anschließende Erfassung von Diagnosedaten dreimal ausgeführt wird, wobei eine Ruhezeit von 500 Sekunden zwischen den Iterationen eingehalten wird.

- Zur Erfassung von Diagnosedaten für Probleme in Zusammenhang mit der Speicherbelegung können Sie, wenn bereits entsprechende Problemsymptome überwacht werden, den folgenden Befehl absetzen:

  ```
  db2fodc -memory
  ```

  Der Befehl **db2fodc -memory** erfasst speicherbezogene Diagnosedaten und speichert sie in einem Verzeichnis mit dem Namen `FODC_Memory_`*`zeitmarke_teilkomponente`*. Dabei steht *zeitmarke* für den Zeitpunkt, zu dem der Befehl **db2fodc -connections** ausgeführt wurde, und *teilkomponente* für die Teilkomponente(n), für die die Erfassung ausgeführt wurde. Alternativ hierzu können Sie eine Variante des folgenden Befehls absetzen, wenn das Problem nur vorübergehend auftritt oder wenn Sie Ihr System vorab zur Erfassung von Diagnosedaten bei Auftreten bestimmter Fehlerbedingungen konfigurieren wollen:

  ```
  db2fodc -memory basic -detect free"<=10" connections">=1000" sleeptime="1"
  iteration="10" interval="10" triggercount="4" duration="5" -member 3
  ```

  Der Parameter **-detect** mit den angegebenen Regeln verzögert die Erfassung bis zur Feststellung der Regeln. In diesem Beispiel müssen die Triggerbedingung für freien Speicher und die Anzahl der Verbindungen zur Datenbank 40 Sekunden lang vorhanden sein, um die Erfassung von Diagnosedaten für die Teilkomponente 3 auszulösen. (Der Triggerzähler von 4 x 10 Sekundenintervallen = 40 Sekunden insgesamt.) Zehn Iterationen der Erkennung und der Erfassung von Diagnosedaten können ausgeführt werden und sind über eine Zeitdauer von fünf Stunden möglich.

- Um nur Triggerbedingungen festzustellen, die Erfassung von Diagnosedaten jedoch nicht auszuführen, können Sie den Befehl **db2fodc -detect** zusammen mit der Option `nocollect` verwenden. In den db2diag-Protokolldateien wird ein Eintrag aufgezeichnet, wenn festgestellt wird, dass die angegebene Fehlerbedingung den Schwellenwert erreicht hat. Wenn Sie keine Diagnosedaten erfassen möchten, dann müssen Sie zur Erfassung detaillierter Daten zur Erreichung der definierten Grenzwerte den db2diag-Protokolleintrag verwenden, um festzustellen, ob eine Fehlerbedingung aufgetreten ist, für die Sie eine Schwellenwertregel erstellt haben.

  ```
  2011-05-31-15.53.42.639270-240 I2341E790 LEVEL: Event
  PID : 13279 TID : 47188095859472PROC : db2fodc
  INSTANCE: kunxu NODE : 000
  FUNCTION: DB2 UDB, RAS/PD component, pdFodcDetectAndRunCollection, probe:100
  CHANGE :

  Hostname: hotel36 Member(s): 0 Iteration: 0
  Thresholds hit 0: cs(0)>=0 us(0)<=50 rQueue(0)>=0 free(7980936)>=100
  avm(7297432)>=1 swapused(0)>=0 UOW_Executing(4)>=3
  Thresholds hit 1: cs(0)>=0 us(0)<=50 rQueue(0)>=0 free(7981000)>=100
  avm(7297528)>=1 swapused(0)>=0 UOW_Executing(4)>=3
  Thresholds hit 2: cs(0)>=0 us(0)<=50 rQueue(0)>=0 bQueue(1)>=1
  free(7981420)>=100 avm(7297596)>=1 swapused(0)>=0 UOW_Executing(4)>=3
  Thresholds hit 3: cs(0)>=0 us(0)<=50 rQueue(0)>=0 free(7981436)>=100
  avm(7297668)>=1 swapused(0)>=0 UOW_Executing(4)>=3
  ```

In diesem Beispiel wird festgestellt, dass die angegebene Fehlerbedingung für die Teilkomponente 0 den definierten Schwellenwert vier Mal erreicht hat. Beide von Ihnen für jede Schwellenwertregel angegebenen Werte und die tatsächlich erkannten Werte werden protokolliert.

Analysieren von Daten für DB2

Nach dem Erfassen der Daten müssen Sie ermitteln, wie Sie diese Daten zur Lösung des aufgetretenen Problems nutzen können. Die Art der Analyse ist abhängig vom Typ des untersuchten Problems sowie von den erfassten Daten. Diese Schritte beschreiben, wie Sie mit der Untersuchung grundlegender DB2-Diagnosedaten beginnen.

Informationen zu diesem Vorgang

Führen Sie zum Analysieren von Diagnosedaten die folgenden Aktionen aus:

Vorgehensweise

- Verschaffen Sie sich genaue Kenntnisse darüber, wie die verschiedenen Daten zusammenhängen. Wenn die Daten beispielsweise mehrere Systeme umfassen, müssen Sie sie so organisieren, dass die Daten ihren Quellen eindeutig zugeordnet werden können.
- Prüfen Sie anhand der Zeitmarken, ob die einzelnen Diagnosedaten für den Zeitpunkt des Auftretens des Problems relevant sind. Beachten Sie dabei, dass Daten verschiedener Quellen unterschiedliche Zeitmarkenformate aufweisen können; vergewissern Sie sich, dass Sie die Reihenfolge der unterschiedlichen Elemente in den einzelnen Zeitmarkenformaten korrekt interpretieren, sodass eine eindeutige Aussage über den Zeitpunkt des Auftretens der verschiedenen Ereignisse möglich ist.
- Stellen Sie fest, für welche Datenquellen die Wahrscheinlichkeit, dass sie Informationen zu dem Problem enthalten, am höchsten ist, und beginnen Sie dort mit der Analyse. Wenn das Problem beispielsweise mit der Installation zusammenhängt, beginnen Sie mit der Analyse bei den Installationsprotokolldateien (falls vorhanden), nicht mit den allgemeinen Programm- oder Betriebssystemprotokolldateien.
- Die jeweilige Analysemethode ist für jede Datenquelle spezifisch, für die meisten Traces und Protokolldateien gilt jedoch der Tipp, zunächst die Stelle in den Daten zu identifizieren, an der das Problem auftritt. Nachdem Sie diese Stelle identifiziert haben, können Sie die Daten entgegengesetzt zur zeitlichen Reihenfolge durcharbeiten, um die eigentliche Fehlerursache zu ermitteln.
- Wenn Sie ein Problem untersuchen, für das Vergleichsdaten von einer ordnungsgemäß funktionierenden Umgebung und einer nicht funktionierenden Umgebung vorliegen, beginnen Sie mit dem Vergleichen der Betriebssystem- und Produktkonfigurationsdetails für die jeweilige Umgebung.

Recovery bei Traps ohne Instanzstopp

Die DB2-Instanz erstellt das FODC-Paket (FODC = First Occurrence Data Capture) für den von Ihnen festgestellten Trap. Standardmäßig ist die DB2-Instanz so konfiguriert, dass die Verfügbarkeit bei Traps gewährleistet ist. Die DB2-Instanz stellt ebenfalls fest, ob die Instanzausführung bei einem Trap beendet werden muss oder ob die DB2-Instanz weiter ausgeführt werden kann, wenn der Thread der DB2-Steuerkomponente, bei dem der Trap aufgetreten ist, ausgesetzt ist.

Informationen zu diesem Vorgang

Standardmäßig ist die DB2-Instanz so konfiguriert, dass die Verfügbarkeit bei Traps gewährleistet ist. Für diese Einstellung ist die Registrierdatenbankvariable **DB2RESILIENCE** zuständig.

Trap ohne Instanzstopp erkennen

Traps werden ohne Instanzstopp ausgesetzt, um die Auswirkungen beim Auftreten von Traps (DB2-Programmierfehlern) möglichst gering zu halten. Ein Trap ohne Instanzstopp lässt sich folgendermaßen diagnostizieren:

1. In dem vollständig qualifizierten Pfad, der über den Konfigurationsparameter **diagpath** des Datenbankmanagers angegeben ist, wird ein FODC-Verzeichnis erstellt.
2. Im Protokoll mit Benachrichtigungen für die Systemverwaltung und in den db2diag-Protokolldateien wird die Fehlernachricht ADM14013C aufgezeichnet.

   Anmerkung: Die Fehlernachricht ADM14011C wird aufgezeichnet, wenn der Trap nicht aufgefangen werden konnte, sondern zu einem Instanzstopp geführt hat.
3. An die Anwendung wird der Fehler mit dem SQLCODE-Wert -1224 zurückgegeben.
4. Der EDU-Thread (Engine-Dispatchable-Unit) wird ausgesetzt. Dies wird in der Ausgabe zu **db2pd -edus** angezeigt.

Recovery

Ein Trap, bei dem kein Instanzstopp erfolgt, stört den normalen Instanzbetrieb in der Regel nicht. Da ein ausgesetzter EDU-Thread jedoch einige Ressourcen belegt, empfiehlt es sich, die jeweilige Instanz baldmöglichst zu einem günstigen Zeitpunkt wie folgt zu stoppen und erneut zu starten:

1. Beenden Sie alle aktiven Anwendungen, die innerhalb des Zeitlimits ein COMMIT oder ROLLBACK ausgeben, mit dem folgenden Befehl (dadurch wird das Recovery-Fenster für eine Recovery nach Systemabsturz bei Ausgabe des Befehls **db2start** minimiert):

   ```
   db2 quiesce instance instanzname user benutzername defer with timeout minuten
   ```
2. [Optional] Geben Sie folgenden Befehl aus, um alle Anwendungen, die in dem in Schritt 1 angegebenen Zeitraum kein COMMIT oder ROLLBACK ausgegeben haben, sowie alle neuen Anwendungen zu beenden, die nach Ablauf des Zeitraums auf die Datenbank zugegriffen haben:

   ```
   db2 quiesce instance instanzname user benutzername immediate
   ```
3. Geben Sie ab DB2 V9.7 Fixpack 3 Folgendes ein:

   ```
   db2stop force
   ```

   Erzwingen Sie in DB2 bis Version 9.7 Fixpack 2 einschließlich mit folgendem Befehl eine Beendigung der Instanz und der ausgesetzten EDUs:

   ```
   db2stop -kill
   ```
4. Starten Sie die DB2-Instanz mit einem der folgenden Befehle:

   ```
   db2start
   ```

   oder

   ```
   START DATABASE MANAGER
   ```

Diagnose

Suchen Sie das FODC-Verzeichnis, das sich in dem über den Konfigurati-

onsparameter **diagpath** des Datenbankmanagers angegebenen Verzeichnis befindet. Die Position des FODC-Verzeichnisses können Sie ebenfalls anhand der Benachrichtigungen für die Systemverwaltung oder der db2diag-Protokolldateien ermitteln. Leiten Sie die FODC-Angaben an IBM Software Support weiter.

Identifizieren von db2diag-Protokolleinträgen für eine Ladeoperation

Das Identifizieren von Diagnoseinformationen für eine Ladeoperation und das Ermitteln der **db2diag**-Befehlsprotokollnachrichten zur jeweiligen Ladeoperation stellen wichtige Schritte bei der Fehlerbehebung für problematische Ladeoperationen dar.

Symptome

Beim Absetzen mehrerer Ladeoperationen stellen Sie möglicherweise fest, dass die Ausführung einer Ladeoperation mehr Zeit in Anspruch nimmt als gewohnt oder blockiert zu sein scheint.

Fehlerbehebung

Gehen Sie wie folgt vor, um zu ermitteln, welche der Ladeoperationen problematisch sind:

1. Geben Sie den Befehl **db2pd -utilities** ein, um die IDs aller Ladeoperationen anzuzeigen. In der folgenden Beispielausgabe ist nur eine einzige Ladeoperation mit der ID `LOADID: 16.2011-05-13-12.44.34.638811.0 (3;5)` enthalten:

```
Database Partition 0 -- Active -- Up 0 days 00:16:15 -- Date 05/13/2011 13:00:33

Utilities:
Address             ID   Type   State   Invoker   Priority   StartTime            DBName   NumPhases   CurPhase   Description
0x000000020120E2A0  1    LOAD   0       0         0          Fri May 13 12:44:34  SAMPLE   2           2          [LOADID: 16.2011-05-13-12.44.34.638811.0 (3;5)]...

Progress:
Address             ID   PhaseNum   CompletedWork   TotalWork   StartTime            Description
0x000000020120E600  1    1          0 bytes         0 bytes     Fri May 13 12:44:34  SETUP
0x000000020120E7E0  1    2          0 rows          0 rows      Fri May 13 12:44:34  LOAD
```

2. Geben Sie den Befehl **db2diag -g 'dataobj:=***Lade-ID***'** ein, wobei *Lade-ID* die ID der Ladeoperation ist, die im vorhergehenden Schritt ermittelt wurde. Mit diesem Befehl werden alle Diagnoseprotokollnachrichten im **db2diag**-Befehlsprotokoll angezeigt, die sich auf die angegebene Ladeoperation beziehen. Das nachfolgend dargestellte Beispiel zeigt die Ausgabe des Befehls mit der zuvor ermittelten ID der Ladeoperation:

```
$ db2diag -g 'dataobj:= LOADID: 16.2011-05-13-12.44.34.638811.0 (3;5)'


2011-05-13-14.27.03.134598-240 I1700E525          LEVEL: Warning

PID     : 29615                      TID  : 47039995963712PROC : db2sysc

INSTANCE: vivmak                     NODE : 000          DB   : SAMPLE

APPHDL  : 0-7                        APPID: *LOCAL.vivmak.110513164421

AUTHID  : VIVMAK

EDUID   : 44                         EDUNAME: db2lrid

FUNCTION: DB2 UDB, database utilities, sqlulPrintPhaseMsg, probe:314

DATA #1 : String, 99 bytes
```

```
LOADID: 16.2011-05-13-12.44.34.638811.0 (3;5)

Completed LOAD phase at 05/13/2011 14:27:03.134463.
```

Nach der Ausführung dieser Schritte liegen ausreichend Informationen zur Identifizierung der problematischen Ladeoperation vor. Wenn Sie jedoch weitere Informationen zur Ladeoperation benötigen, können Sie den Befehl **db2pd -db** *<datenbankname>* **-load loadID=***"LOADID"* **stacks** eingeben, um einen Stapeltrace abzurufen. Die Option **stacks** ist nur unter den Betriebssystemen UNIX und Linux verfügbar. Das nachfolgend dargestellte Beispiel zeigt die Ausgabe des Befehls, wenn dieser für eine Beispieldatenbank und mit der zuvor ermittelten ID der Ladeoperation eingegeben wird:

```
$ db2pd -db sample –load loadID="LOADID: 16.2011-05-13-12.34.34.638811.0 (3;5)"
stacks

Attempting to produce stack traces for LOAD edu 40

Attempting to produce stack traces for LOAD edu 41

Attempting to produce stack traces for LOAD edu 42

Attempting to produce stack traces for LOAD edu 44

Attempting to produce stack traces for LOAD edu 45


Database Partition 0 -- Database SAMPLE -- Active -- Up 0 days 00:00:27 --
Date 05/13/2011 14:28:32

Node Number : 0

Database Name: SAMPLE


LoadID                                          EDUID      EDUNAME          TableName    SchemaName   AppHandl [nod-index]    Application ID               StartTime
LOADID: 16.2011-05-13-12.44.34.638811.0 (3;5)   40         db2lfrm0         T1           VIVMAK       7        [000-00007]    *LOCAL.vivmak.110513164421   2011-05-13-12.44.34.638811
LOADID: 16.2011-05-13-12.44.34.638811.0 (3;5)   41         db2lfrm1         T1           VIVMAK       7        [000-00007]    *LOCAL.vivmak.110513164421   2011-05-13-12.44.34.638811
LOADID: 16.2011-05-13-12.44.34.638811.0 (3;5)   42         db2lfrm2         T1           VIVMAK       7        [000-00007]    *LOCAL.vivmak.110513164421   2011-05-13-12.44.34.638811
LOADID: 16.2011-05-13-12.44.34.638811.0 (3;5)   44         db2lrid          T1           VIVMAK       7        [000-00007]    *LOCAL.vivmak.110513164421   2011-05-13-12.44.34.638811
LOADID: 16.2011-05-13-12.44.34.638811.0 (3;5)   45         db2lbm0          T1           VIVMAK       7        [000-00007]    *LOCAL.vivmak.110513164421   2011-05-13-12.44.34.638811
```

Der Befehl **db2pd -db** *<datenbankname>* **-load loadID=***"LOADID"* **stacks** zeigt alle EDU-Informationen an, die sich auf die angegebene Ladeoperation beziehen, und generiert Stack-Trace-Dateien im Verzeichnis `diagpath`.

Sie können die gesamten abgerufenen Informationen nutzen, um weitere Fehlerbehebungsverfahren anzuwenden, wie zum Beispiel das Überwachen oder das Beenden der Ladeoperation. Darüber hinaus werden die erfassten Informationen möglicherweise von der technischen Unterstützung von IBM angefordert, um die Fehlerbehebung für die problematischen Ladeoperationen durchzuführen.

Sie können die erfassten Informationen auch zur Ausführung des Befehls **db2trc** zur Durchführung weiterer Fehlerbehebungsschritte benutzen. Gehen Sie wie folgt vor, um den Befehl **db2trc** für eine bestimmte Ladeoperation auszuführen und dazu die abgerufenen Informationen zu verwenden:

1. Führen Sie den Befehl **db2pd -load** aus, um die Anwendungs-ID der speziellen Ladeoperation abzurufen, die für Sie relevant ist.
2. Führen Sie anschließend den Befehl **db2trc -appid** oder **db2trc -apphdl** aus, um weitere Informationen zur Ladeoperation aufzuzeichnen.

Im folgenden Beispiel wird die Anwendungs-ID `*LOCAL.vivmak.110513164421` der Lade-ID `LOADID: 16.2011-05-13-12.44.34.638811.0 (3;5)` des vorhergehenden Beispiels im vorliegenden Thema bei der Ausführung des Befehls **db2trc** verwendet:

```
$ db2trc  on -appid  *LOCAL.vivmak.110513164421
 Trace is turned on
 $ db2trc info
 Marker                 :  @TRACE@
 Trace version          :      7.0
 Platform               :  Linux/X
 Build level            :  n110612
 maxBufferSize          : 33554432 bytes (32 MB)
 auxBufferSize          : 524288 bytes (0 MB)
 allocationCount        : 2
 DB2TRCD pid            : 0
 Trace destination      : <shared memory buffer>
 numSuspended           : 0
 Trace starting time    : 2011-06-16-10.07.52.209999-240

 Buffer size            : 33554432 bytes (32 MB)
 Allow buffer to wrap   : yes
 Mask                   : *.*.*.*.*
 Timestamps             : disabled
 PID.TID mask           : all
 Fixed data mask #1     : all
 Fixed data mask #2     : all
 Max system errors      : infinite
 Treat this rc as sys err: none
 Member mask            : none
 Application handle mask : none
 Application ID mask    : *LOCAL.vivmak.110513164421
```

Im nächsten Beispiel wird die Anwendungskennung, die der Ausgabe des Befehls **db2pd -load** entnommen wird, dazu verwendet, die Traceoptionen zu ändern, die für den Befehl **db2trc** gelten:

```
$ db2trc chg -apphdl 7
 Trace has been changed
 $ db2trc info
 Marker                 :  @TRACE@
 Trace version          :      7.0
```

```
Platform                :  Linux/X

Build level             :  n110612

maxBufferSize           : 33554432 bytes (32 MB)

auxBufferSize           : 524288 bytes (0 MB)

allocationCount         : 2

DB2TRCD pid             : 0

Trace destination       : <shared memory buffer>

numSuspended            : 0

Trace starting time     : 2011-06-16-10.10.11.816264-240


Buffer size             : 33554432 bytes (32 MB)

Allow buffer to wrap    : yes

Mask                    : *.*.*.*.*

Timestamps              : disabled

PID.TID mask            : all

Fixed data mask #1      : all

Fixed data mask #2      : all

Max system errors       : infinite

Treat this rc as sys err: none

Member mask             : none

Application handle mask : 7

Application ID mask     : none
```

Fehlerbehebung beim Scheduler für Verwaltungstasks

Anhand dieser Checkliste können Sie Fehler beheben, die während der Ausführung von Tasks im Scheduler für Verwaltungstasks auftreten.

Vorgehensweise

1. Wenn Ihre Task nicht wie erwartet ausgeführt wird, sollten Sie zuerst nach einem Ausführungsstatuseintrag in der Verwaltungssicht ADMIN_TASK_STATUS suchen.
   - Ist ein Eintrag vorhanden, prüfen Sie die verschiedenen Werte. Achten Sie dabei besonders auf die Spalten STATUS, INVOCATION, SQLCODE, SQLSTATE, SQLERRMC und RC. Häufig kann anhand der Werte die zugrunde liegende Fehlerursache identifiziert werden.
   - Wenn die Sicht keinen Ausführungsstatuseintrag enthält, wurde die Task nicht ausgeführt. Hierfür gibt es eine Reihe möglicher Erklärungen:
     – Der Scheduler für Verwaltungstasks ist inaktiviert. Wenn der Scheduler für Verwaltungstasks inaktiviert ist, werden keine Tasks ausgeführt. Definie-

ren Sie zum Aktivieren des Schedulers für Verwaltungstasks die Registrierdatenbankvariable **DB2_ATS_ENABLE**.

- Die Task wurde entfernt. Möglicherweise wurde die Task von einem anderen Benutzer entfernt. Stellen Sie sicher, dass die Task vorhanden ist, indem Sie die Verwaltungssicht ADMIN_TASK_LIST abfragen.
- Der Scheduler hat die Task nicht erkannt. Der Scheduler für Verwaltungstasks sucht nach neuen und aktualisierten Tasks, indem er in Intervallen von fünf Minuten eine Verbindung zu jeder aktiven Datenbank herstellt. Vor Ablauf dieses Intervalls erkennt der Scheduler Ihre Task nicht. Warten Sie mindestens fünf Minuten.
- Die Datenbank ist inaktiv. Der Scheduler für Verwaltungstasks kann keine Tasks abrufen oder ausführen, wenn die Datenbank inaktiv ist. Aktivieren Sie die Datenbank.
- Die Transaktion ist nicht festgeschrieben. Der Scheduler für Verwaltungstasks ignoriert nicht festgeschriebene Tasks. Führen Sie nach dem Hinzufügen, Aktualisieren oder Entfernen einer Tasks ein Commit durch.
- Der Zeitplan ist ungültig. Möglicherweise verhindert der Zeitplan der Task deren Ausführung. So kann beispielsweise die maximal zulässige Anzahl von Aufrufen für die Task bereits erreicht sein. Überprüfen Sie den Zeitplan der Task in der Sicht ADMIN_TASK_LIST und aktualisieren Sie ihn, falls erforderlich.

2. Wenn Sie die Fehlerursache nicht anhand der Verwaltungssicht ADMIN_TASK_STATUS ermitteln können, lesen Sie die Informationen im DB2-Diagnoseprotokoll. Alle kritischen Fehler werden in den **db2diag**-Protokolldateien protokolliert. Informationsereignisnachrichten werden während der Taskausführung ebenfalls vom Dämon des Schedulers für Verwaltungstasks protokolliert. Diese Fehler und Nachrichten werden durch die Komponente 'Scheduler für Verwaltungstasks' identifiziert.

Nächste Schritte

Wenn Sie die oben angeführten Schritte ausgeführt haben und die Ursache des Problems dennoch nicht feststellen können, sollten Sie gegebenenfalls einen PMR (Problem Management Record) bei IBM Software Support öffnen. Teilen Sie den Mitarbeitern von IBM Software Support mit, dass Sie die hier beschriebenen Anweisungen ausgeführt haben, und leiten Sie die erfassten Diagnosedaten an sie weiter.

Fehlerbehebung bei der Komprimierung

Wörterverzeichnis der Datenkomprimierung wird nicht automatisch erstellt

Sie verfügen über eine umfangreiche Tabelle oder ein großes XML-Speicherobjekt für die Tabelle, das Wörterverzeichnis der Datenkomprimierung wurde jedoch nicht erstellt. Sie möchten wissen, warum das Wörterverzeichnis der Datenkomprimierung nicht wie erwartet erstellt wurde. Die Informationen in diesem Abschnitt beziehen sich sowohl auf das Komprimierungswörterverzeichnis (Compression Dictionary) für das Tabellenobjekt als auch auf das Komprimierungswörterverzeichnis für das XML-Speicherobjekt.

Sie befinden sich in folgender Situation:

- Sie verfügen über eine Tabelle, in der das Attribut COMPRESS auf YES gesetzt wurde.

- Die Tabelle bestand schon einige Zeit und Daten wurde hinzugefügt und entfernt.
- Die Größe der Tabelle entspricht ungefähr der als Schwellenwert definierten Größe. Sie erwarten, dass das Wörterverzeichnis der Datenkomprimierung automatisch erstellt wird.
- Sie führen eine Operation zum Füllen der Tabelle aus (zum Beispiel INSERT, LOAD INSERT oder REDISTRIBUTE), wobei Sie erwarten, dass die Größe der Tabelle über den Schwellenwert hinaus zunimmt.
- Das Wörterverzeichnis der Datenkomprimierung wird nicht automatisch erstellt. Das Wörterverzeichnis der Datenkomprimierung wird nicht erstellt und nicht in die Tabelle gestellt. Sie erwarten, dass die Komprimierung für die der Tabelle hinzugefügten Daten nach diesem Punkt erfolgt, aber die Daten bleiben dekomprimiert.
- XML-Daten liegen im Speicherformat von DB2 Version 9.7 vor.

  Eine Datenkomprimierung in dem XML-Speicherobjekt einer Tabelle wird nicht unterstützt, wenn die Tabelle XML-Spalten enthält, die mit DB2 Version 9.5 oder einer früheren Version erstellt wurden. Wenn Sie eine derartige Tabelle für die Komprimierung von Datenzeilen aktivieren, werden nur die Zeilendaten der Tabelle im Tabellenobjekt komprimiert. Kann das XML-Speicherobjekt bei einer Einfüge-, Lade- oder Reorganisationsoperation nicht komprimiert werden, wird nur dann eine Nachricht in eine db2diag-Protokolldatei geschrieben, wenn die XML-Spalten mit DB2 Version 9 oder DB2 Version 9.5 erstellt wurden.

Warum werden die Daten nicht komprimiert?

Der Datenumfang liegt zwar über dem Schwellenwert, der für eine Aktivierung der automatischen Erstellung des Komprimierungswörterverzeichnis festgelegt ist, es wird jedoch eine andere Bedingung geprüft. Diese Bedingung besagt, dass die Daten im Objekt für das Erstellen des Verzeichnisses ausreichen müssen. Über diese Anforderung werden Sie in der Nachricht ADM5591W informiert. Vorherige Aktivitäten mit den Daten haben möglicherweise auch das Löschen oder Entfernen von Daten umfasst. Es gibt möglicherweise große Bereiche im Objekt, die keine Daten enthalten. Sie verfügen somit über ein großes Objekt, das den Schwellenwert für die Objektgröße erreicht oder übersteigt, in dem jedoch nicht genügend Daten für die Erstellung eines Wörterverzeichnisses vorhanden sind.

Wenn zahlreiche Aktivitäten mit dem Objekt durchgeführt werden, müssen Sie das Objekt regelmäßig reorganisieren. Bei XML-Daten müssen Sie die Tabelle mit der Option `longlobdata` reorganisieren. Wenn Sie dies nicht tun, kann das Objekt sehr groß, jedoch mit nur wenigen Daten gefüllt sein. Durch das Reorganisieren des Objekts werden fragmentierte Daten gelöscht und die Daten im Objekt werden kompakt organisiert. Nach der Reorganisation ist das Objekt kleiner und dichter gefüllt. Das reorganisierte Objekt stellt die Datenmenge im Objekt präziser dar und kann kleiner als der Schwellenwert sein, der für die Aktivierung der automatischen Erstellung des Wörterverzeichnisses der Datenkomprimierung festgelegt ist.

Ist das Objekt mit nur wenigen Daten gefüllt, kann die Tabelle mit dem Befehl **REORG TABLE** (mit der Option `LONGLOBDATA` für XDA) reorganisiert werden, um das Verzeichnis zu erstellen. Die Standardoption lautet `KEEPDICTIONARY`. Mit der Option `RESETDICTIONARY` können Sie eine Verzeichniserstellung erzwingen.

Verwenden Sie den Befehl **REORGCHK**, um festzustellen, ob eine Tabelle reorganisiert werden muss.

Die automatische Wörterverzeichniserstellung wird für die Tabelle nicht erfolgen, wenn die Tabelle nicht für eine Komprimierung von Datenzeilen aktiviert ist. Wenn die automatische Wörterverzeichniserstellung für die Datenbank inaktiviert ist, wird die Nachricht ADM5594I zurückgegeben, die den jeweiligen Grund für die Inaktivierung erläutert.

Enthält die Tabelle XML-Spalten, die mit DB2 Version 9.5 oder einer früheren Version erstellt wurden, können Sie ein Upgrade für die Tabelle mit der gespeicherten Prozedur ADMIN_MOVE_TABLE durchführen und die Komprimierung von Datenzeilen anschließend aktivieren.

Keine Verringerung des Plattenspeicherplatzes für temporäre Tabellen durch Zeilenkomprimierung

Es gibt Situationen, bei denen Plattenspeicherplatz für temporäre Tabellen nicht im erwarteten Maß eingespart werden kann, auch wenn eine Lizenz für Storage Optimization Feature vorliegt.

Symptome

Die Einsparungen beim Plattenspeicherplatz durch die Aktivierung der Zeilenkomprimierung für temporäre Tabellen entsprechen nicht den Erwartungen.

Ursachen

- Diese Situation tritt im Allgemeinen auf, wenn eine große Anzahl von Anwendungen gleichzeitig ausgeführt werden und temporäre Tabellen erstellen, die jeweils einen Teil des Datenbankmanagerspeichers belegen. Dies führt dazu, dass der Speicherplatz nicht zum Erstellen des Komprimierungswörterverzeichnisses (Compression Dictionary) ausreicht. Tritt diese Situation ein, erfolgt keine Benachrichtigung.
- Zeilen werden anhand einer Methode komprimiert, die auf einem Wörterverzeichnis basiert und an einem Algorithmus ausgerichtet ist. Ist eine Zeile einer temporären Tabelle lang genug, um in nennenswertem Umfang zu einer Einsparung von Plattenspeicherplatz beitragen zu können, wird die betreffende Zeile komprimiert. Kleine Zeilen in temporären Tabellen werden nicht komprimiert. Dieser Umstand führt dazu, dass die Einsparungen an Plattenspeicherplatz nicht wie erwartet ausfallen. Tritt diese Situation ein, erfolgt keine Benachrichtigung.

Risiko

Der Systembetrieb ist in keiner Weise gefährdet, es erfolgt lediglich keine Zeilenkomprimierung bei temporären Tabellen, deren Zeilenlänge unter dem Schwellenwert liegt. Dies könnte andere nachteilige Auswirkungen für den Datenbankmanager haben, wenn der verfügbare Speicher dadurch extrem begrenzt bleibt.

Dekomprimierung von komprimiertem Zeilenimage schlägt bei Datenreplikation fehl

Es können Situationen vorkommen, die dazu führen, dass bei einer Datenreplikationslösung die Dekomprimierung eines Protokollsatzes mit einem komprimierten Zeilenimage fehlschlägt. Bei vorübergehenden Fehlern wird ein SQL-Code ausgegeben, der der Fehlerursache entspricht. Permanente Fehler werden im Allgemeinen durch die Benachrichtigung SQL0204N angezeigt. Lediglich bei vorübergehenden Fehlersituationen ist es möglich, dass die Dekomprimierung eines Zeilenimages in einem Protokollsatz anschließend erfolgreich ausgeführt wird. Die

API db2ReadLog fährt mit der Verarbeitung anderer Protokollsätze fort, wenn ein Protokollsatz nicht dekomprimiert werden kann.

Symptome

Die Protokolllesefunktion kann beim Lesen von Protokollsätzen, die komprimierte Benutzerdaten enthalten, auf vorübergehende oder permanente Fehler stoßen. Als Fehler können beim Lesen von Protokollsätzen mit komprimierten Daten (Zeilenimages) beispielsweise die im Folgenden aufgeführten vorübergehenden oder permanenten Fehler auftreten. Bei der Aufzählung handelt es sich um eine Beispielsammlung, die keinen Anspruch auf Vollständigkeit erhebt.

Vorübergehende Fehler:

- Tabellenbereichszugriff wird nicht zugelassen.
- Auf die Tabelle kann nicht zugegriffen werden (Überschreitung der Sperrzeit)
- Der Speicher reicht zum Laden und Speichern des erforderlichen Wörterverzeichnisses nicht aus.

Permanente Fehler:

- Der Tabellenbereich, in dem sich die Tabelle befindet, ist nicht vorhanden.
- Die Tabelle bzw. die Tabellenpartition, zu der der Protokollsatz gehört, ist nicht vorhanden.
- Ein Wörterverzeichnis für die Tabelle bzw. die Tabellenpartition ist nicht vorhanden.
- Der Protokollsatz enthält Zeilenimages, die mit einem Wörterverzeichnis komprimiert sind, das älter ist als die Wörterverzeichnisse in der Tabelle.

Ursachen

Eine Replikationslösung oder eine sonstige Protokolllesefunktion kann bei Datenbankaktivitäten in Verzug sein und dadurch beim Lesen eines Protokollsatzes, der komprimierte Benutzerdaten enthält, einen Fehler auslösen (siehe Szenario 1). Ein solcher Fall kann eintreten, wenn der zu lesende Protokollsatz komprimierte Benutzerdaten enthält, die mit einem Komprimierungswörterverzeichnis komprimiert wurden, das älter ist als das in der Tabelle zum Zeitpunkt des Lesens verfügbare Verzeichnis.

Gleichermaßen gilt, dass beim Löschen einer Tabelle auch die der Tabelle zugeordneten Wörterverzeichnisse gelöscht werden. Komprimierte Zeilenimages für die Tabelle können in diesem Fall nicht dekomprimiert werden (siehe Szenario 2). Dabei ist zu beachten, dass diese Einschränkung nicht für Zeilenimages gilt, die keinen Komprimierungsstatus aufweisen. Diese Zeilenimages können weiterhin gelesen und repliziert werden, auch wenn die Tabelle gelöscht wird.

Für eine einzelne Tabelle kann es nur ein einziges aktives Datenkomprimierungswörterverzeichnis und ein einziges Protokollwörterverzeichnis geben.

Szenario 1:

Für Tabelle t6 ist die Komprimierung aktiviert. Das Attribut DATA CAPTURE CHANGES ist für die Tabelle für die Replikation aktiviert. Die Tabelle wird über eine Datenreplikationsanwendung repliziert und die Protokolllesefunktion liest Protokollsätze, die komprimierte Daten (Zeilenimages) enthalten. Eine Protokollle-

sefunktion des Clients liest mithilfe der API db2ReadLog den ersten Protokollsatz für die erste Anweisung INSERT, da eine Operation **LOAD** für Tabelle t6 ausgeführt wird, nachdem der Befehl REORG TABLE (der zu einer erneuten Erstellung des Wörterverzeichnisses der Tabelle führt) ausgegeben wurde.

Die folgenden Anweisungen werden für Tabelle t6 ausgeführt, die bereits ein Komprimierungswörterverzeichnis enthält und für die das Attribut DATA CAPTURE CHANGES aktiviert ist:

```
-> db2 alter table t6 data capture changes
-> db2 insert into t6 values (...)
-> db2 insert into t6 values (...)
```

Da bereits ein Datenkomprimierungswörterverzeichnis für Tabelle t6 vorhanden ist, erfolgt bei beiden INSERT-Anweisungen, die auf die Anweisung ALTER folgen, eine Komprimierung (mit dem Komprimierungswörterverzeichnis von Tabelle t6). Zu diesem Zeitpunkt hat die Protokolllesefunktion die erste Anweisung INSERT noch nicht erreicht.

Mit dem folgenden Befehl **REORG TABLE** wird ein neues Komprimierungswörterverzeichnis für Tabelle t6 erstellt und das aktuelle Komprimierungswörterverzeichnis wird als Protokollverzeichnis beibehalten, sodass die Protokolllesefunktion ein Protokoll vor dem aktuellen Komprimierungswörterverzeichnis verwendet (dieses Protokollwörterverzeichnis wird jedoch nach der Ausführung von REORG nicht mehr in den Speicher geladen):

```
-> db2 reorg table t6 resetdictionary
```

Da die Protokolllesefunktion das INSERT-Protokoll für die INSERT-Anweisungen liest und das Protokollverzeichnis demzufolge jetzt in den Speicher gelesen werden muss, wird für Tabelle t6 eine Operation **LOAD** ausgeführt:

```
-> db2 load from data.del of del insert into table t6 allow no access
```

Nach dem Ausführen der Operation **LOAD** für die Quellentabelle wird für Tabelle t6 aufgrund der Option `ALLOW NO ACCESS` eine Sperre vom Typ "Z" aktiviert. Die Protokolllesefunktion muss nun das Protokollwörterverzeichnis in den Speicher laden, um die in den INSERT-Protokollsätzen gefundenen Zeilenimages zu dekomprimieren. Das Abrufen des Wörterverzeichnisses erfordert jedoch eine IN-Tabellensperre. In diesem Fall schlägt der Versuch der Protokolllesefunktion, diese Sperre zu aktivieren, fehl. Dies führt dazu, dass der SQLCODE-Eintrag der Struktur 'db2ReadLogFilterData' den SQLCODE-Wert SQL2048N zurückgibt. Dieser Code bezieht sich auf einen vorübergehenden Fehler (d. h. es besteht die *Möglichkeit*, dass der Protokollsatz bei einem erneuten Aufruf der API dekomprimiert werden kann). Die Protokolllesefunktion gibt das komprimierte Zeilenimage im Protokollsatz zurück und fährt mit dem nächsten Protokollsatz fort.

Szenario 2:

Für Tabelle t7 ist das Attribut DATA CAPTURE CHANGES aktiviert. Die Komprimierung ist für die Tabelle aktiviert, um den Speicheraufwand zu reduzieren. Die Tabelle wird über eine Datenreplikationsanwendung repliziert, die Protokolllesefunktion ist jedoch in Bezug auf die Quellentabellenaktivität nicht auf dem neuesten Stand und das Wörterverzeichnis der Datenkomprimierung wurde bereits zweimal erneut erstellt, bevor die Protokolllesefunktion erneut in den Protokollsätzen liest.

Die folgenden Anweisungen werden für Tabelle t7 ausgeführt, wobei das Attribut DATA CAPTURE CHANGES bereits aktiviert ist, die Tabellenkomprimierung ebenfalls aktiviert ist und ein neues Wörterverzeichnis erstellt wird:

```
-> db2 alter table t7 compress yes
-> db2 reorg table t7 resetdictionary
-> db2 insert into t7 values (...)
```

Eine Protokolllesefunktion des Clients wird mithilfe der API db2ReadLog versuchen, das nächste Protokoll entsprechend der ersten Anweisung INSERT (siehe unten) zu lesen:

```
-> db2 insert into t7 values (...)
...
-> db2 reorg table t7 resetdictionary
-> db2 insert into t7 values (...)
...
-> db2 reorg table t7 resetdictionary
```

Die API db2ReadLog kann den Inhalt des Protokollsatzes in diesem Fall jedoch nicht dekomprimieren, da die Protokolllesefunktion bereits mindestens zwei Operationen **REORG RESETDICTIONARY** nachzuholen hat. Das Wörterverzeichnis, das zum Dekomprimieren des Zeilenimages im Protokollsatz erforderlich ist, kann in der Tabelle nicht gefunden werden, da mit der Tabelle lediglich das Komprimierungswörterverzeichnis der zweiten Anweisung REORG und das Komprimierungswörterverzeichnis der letzten Anweisung REORG gespeichert wurden. Die Ausführung der API db2ReadLog wird jedoch nicht mit einem Fehler beendet. Stattdessen wird das nicht komprimierte Zeilenimage an den Benutzerpuffer zurückgegeben und der SQLCODE-Eintrag gibt in der Struktur 'db2ReadLogFilterData', die dem Protokollsatz vorausgeht, den SQLCODE-Wert SQL0204N zurück. Dieser Code bezieht sich auf einen permanenten Fehler (d. h. der Protokollsatz kann nicht mehr dekomprimiert werden).

Umgebung

Dieser Fehlschlag beim Dekomprimieren eines komprimierten Protokollsatzes, der durch ein fehlendes älteres Komprimierungsverzeichnis bedingt ist, kann auf allen Plattformen entstehen, bei denen eine Datenreplikationslösung die API db2ReadLog verwendet und das Attribut DATA CAPTURE CHANGES für die betreffende Tabelle definiert ist.

Problemlösung

Benutzeraktion:

Bei vorübergehenden Fehlern kann das Protokoll möglicherweise nach einer erneuten Ausgabe der Leseanforderung erfolgreich gelesen werden. Gehört der Protokollsatz z. B. zu einer Tabelle in einem Tabellenbereich und ist der Zugriff auf die Tabelle nicht gestattet, ist das Wörterverzeichnis möglicherweise nicht zum Dekomprimieren des Protokollsatzes verfügbar (siehe Szenario 1). Der Tabellenbereich steht möglicherweise zu einem späteren Zeitpunkt zur Verfügung und die erneute Ausgabe der Protokollleseanforderung zu diesem Zeitpunkt kann so zu einer erfolgreichen Dekomprimierung des Protokollsatzes führen.

- Informieren Sie sich anhand der Fehlerinformationen über geeignete Maßnahmen, wenn ein vorübergehender Fehler zurückgegeben wird (siehe Szenario 1). Dies kann z. B. das Warten auf den Abschluss einer Tabellenoperation beinhalten, das anschließend das erneute Lesen des Protokollsatzes und ein erfolgreiches Dekomprimieren ermöglichen könnte.

- Tritt ein permanenter Fehler auf (Szenario 2), kann das Zeilenimage im Protokollsatz nicht dekomprimiert werden, da das Komprimierungswörterverzeichnis, das zum Komprimieren des Zeilenimages verwendet wurde, nicht mehr verfügbar ist. In diesem Fall muss die betroffene Tabelle (Zieltabelle) möglicherweise über Replikationslösungen erneut initialisiert werden.

Fehlerbehebung bei globalen Variablenfehlern

Die Fehlerbehebung bei globalen Variablen in Anwendungen ist kein Problem, wenn der Benutzer, bei dem der Fehler auftritt, über die Berechtigung zum Lesen (READ) der globalen Variablen verfügt. Sie benötigen lediglich die Leseberechtigung READ, um den Wert einer globalen Variablen zu erfahren. Geben Sie dazu die Anweisung VALUES(Globaler Variablenname) aus. Es gibt Fälle, in denen der Benutzer, der die Anwendung ausführt, nicht über die Leseberechtigung READ für globale Variablen verfügt.

Das erste Szenario verdeutlicht ein mögliches Problem beim Verweis auf globale Variablen, für das es eine einfache Lösung gibt. Das zweite Szenario stellte eine wahrscheinlichere Situation dar, in der die Berechtigung zum Lesen der globalen Variable dem entsprechenden Benutzer noch erteilt werden muss.

Szenario 1

Verweise auf globale Variablen müssen ordnungsgemäß qualifiziert sein. Es ist möglich, dass eine Variable mit demselben Namen und einem anderen Schema vorhanden ist, wobei das falsche Schema früher im Registerwert PATH vorkommt. Eine Lösung besteht darin, sicherzustellen, dass die Verweise auf globale Variablen vollständig qualifiziert sind.

Szenario 2

Ein Anwendungsentwickler (developerUser) erstellt eine hochkomplexe Reihe von Prozeduren, Sichten, Triggern usw. auf Grundlage einiger globaler Variablen, auf die nur er Lesezugriff hat. Ein Endbenutzer der Anwendung (finalUser) meldet sich an und beginnt mit der Ausgabe von SQL-Anweisungen, wobei er die vom developerUser erstellte Umgebung nutzt. Der finalUser meldet dem developerUser, dass er Daten nicht anzeigen kann, für die er eine Leseberechtigung haben sollte. Als Teil der Fehlerbehebung bei diesem Problem ändert der developerUser seine Berechtigungs-ID in die Berechtigungs-ID von finalUser, er meldet sich als finalUser an und gibt dieselben SQL-Anweisungen wie finalUser aus. Der developerUser stellt fest, dass der finalUser das Problem richtig beschrieben hat.

Der developerUser muss prüfen, ob dem finalUser dieselben Werte der globalen Variablen angezeigt werden wie ihm. Der developerUser führt SET SESSION USER aus, um die Werte der globalen Variablen anzuzeigen, die dem finalUser angezeigt werden. Nun folgt eine empfohlene Methode zur Feststellung und Behebung des Problems.

Der developerUser bittet den Sicherheitsadministrator (secadmUser), ihm die Berechtigung zur Verwendung von SET SESSION USER als finalUser zu erteilen. Anschließend meldet sich der developerUser selbst an und verwendet die Anweisung SET SESSION AUTHORIZATION, um das Sonderregister SESSION_USER auf das Sonderregister von finalUser festzulegen. Nach Ausführung der fraglichen SQL-Anweisungen, schaltet er mithilfe einer anderen Anweisung SET SESSION AU-

THORIZATION zurück zum developerUser. Der developerUser kann jetzt eine VALUES-Anweisung absetzen und den tatsächlichen Wert der globalen Variablen anzeigen.

Es folgt ein SQL-Beispiel, wobei die Aktionen dargestellt werden, die in der Datenbank von developerUser ausgeführt werden.

```
########################################################################
# developerUser connects to database and creates needed objects
########################################################################

db2 "connect to sample user developerUser using xxxxxxxx"

db2 "create table security.users \
(userid varchar(10) not null primary key, \
firstname varchar(10), \
lastname varchar(10), \
authlevel int)"

db2 "insert into security.users values ('ZUBIRI', 'Adriana', 'Zubiri', 1)"
db2 "insert into security.users values ('SMITH', 'Mary', 'Smith', 2)"
db2 "insert into security.users values ('NEWTON', 'John', 'Newton', 3)"

db2 "create variable security.gv_user varchar(10) default (SESSION_USER)"
db2 "create variable security.authorization int default 0"

# Create a procedure that depends on a global variable
db2 "CREATE PROCEDURE SECURITY.GET_AUTHORIZATION() \
SPECIFIC GET_AUTHORIZATION \
RESULT SETS 1 \
LANGUAGE SQL \
   SELECT authlevel INTO security.authorization \
   FROM security.users \
   WHERE userid = security.gv_user"

db2 "grant all on variable security.authorization to public"
db2 "grant execute on procedure security.get_authorization to public"
db2 "terminate"

########################################################################
# secadmUser grants setsessionuser
########################################################################
db2 "connect to sample user secadmUser using xxxxxxxx"
db2 "grant setsessionuser on user finalUser to user developerUser"
db2 "terminate"

########################################################################
# developerUser will debug the problem now
########################################################################

echo "------------------------------------------------------------"
echo " Connect as developerUser "
echo "------------------------------------------------------------"
db2 "connect to sample user developerUser using xxxxxxxx"

echo "------------------------------------------------------------"
echo " SET SESSION AUTHORIZATION = finalUser "
echo "------------------------------------------------------------"
db2 "set session authorization = finalUser"

echo "--- TRY to get the value of gv_user as finalUser (we must not be able to)"
db2 "values(security.gv_user)"

echo "--- Now call the procedure---"
db2 "call security.get_authorization()"

echo "--- if it works it must return 3 ---"
```

```
db2 "values(security.authorization)"

echo "------------------------------------------------------------"
echo " SET SESSION AUTHORIZATION = developerUser "
echo "------------------------------------------------------------"

db2 "set session authorization = developerUser"

echo "--- See what the variable looks like ----"
db2 "values(security.gv_user)"

db2 "terminate"
```

Fehlerbehebung bei Inkonsistenzen

Behebung von Dateninkonsistenzen

Es ist sehr wichtig, das Vorhandensein von Dateninkonsistenzen innerhalb der Datenbank genau zu diagnostizieren. Eine Möglichkeit, um Dateninkonsistenzen zu ermitteln besteht darin, die Ausgabe des Befehls **INSPECT** zu verwenden, um herauszufinden, wo ein Problem besteht. Wenn Inkonsistenzen gefunden werden, müssen Sie entscheiden, wie Sie mit dem Problem umgehen.

Wenn Sie eine Dateninkonsistenz festgestellt haben, haben Sie zwei Möglichkeiten:

- Sie können sich an IBM Software Support wenden und um Unterstützung bei der Behebung der Dateninkonsistenz bitten.
- Sie können das Datenbankobjekt, bei dem die Dateninkonsistenz besteht, löschen und erneut erstellen.

Sie verwenden die Variante **INSPECT CHECK** des Befehls **INSPECT**, um die Datenbank, den Tabellenbereich oder die Tabelle, bei der bzw. dem Anzeichen einer Dateninkonsistenz bestehen, zu überprüfen. Sobald die Ergebnisse des Befehls **INSPECT CHECK** erzeugt wurden, sollten Sie die Überprüfungsergebnisse mithilfe des Befehls **db2inspf** formatieren.

Wenn der Befehl **INSPECT** nicht vollständig ausgeführt wird, wenden Sie sich an IBM Software Support.

Behebung von Inkonsistenzen bei der Index-Datenzuordnung

Indizes müssen präzise sein, um einen schnellen Zugriff auf die richtigen Daten in Tabellen zu ermöglichen. Andernfalls ist die Datenbank beschädigt.

Sie können den Befehl **INSPECT** verwenden, um eine Onlineprüfung der Indexdatenkonsistenz mithilfe der Option INDEXDATA in der Klausel für die Prüfung über Objektgrenzen hinweg auszuführen. Die Indexdatenprüfung wird nicht standardmäßig ausgeführt, wenn Sie den Befehl **INSPECT** verwenden. Sie muss explizit angefordert werden.

Wenn ein Fehler wegen Indexdateninkonsistenz festgestellt wird, während **INSPECT** eine INDEXDATA-Prüfung ausführt, wird die Fehlernachricht SQL1141N zurückgegeben. Gleichzeitig mit der Rückgabe dieser Fehlernachricht werden Datendiagnoseinformationen erfasst und in die **db2diag**-Protokolldatei ausgegeben. Eine dringende Nachricht wird ferner im Protokoll mit Benachrichtigungen für die Systemverwaltung protokolliert. Verwenden Sie das Analysetool **db2diag** für die **db2diag**-Protokolldateien, um den Inhalt der **db2diag**-Protokolldatei zu filtern und zu formatieren.

Auswirkungen der Verriegelung

Bei der Prüfung des Index auf Dateninkonsistenz mithilfe des Befehls **INSPECT** mit der Option INDEXDATA, werden die geprüften Dateien im IS-Modus verriegelt.

Wenn die Option INDEXDATA angegeben wird, werden standardmäßig nur die Werte von Optionen aus explizit angegebenen Ebenen der Klausel verwendet. Für Optionen beliebiger Ebenen der Klausel, die nicht explizit angegeben wurden, werden die Standardebenen (INDEX NORMAL und DATA NORMAL) überschrieben und dabei NORMAL in NONE geändert.

Fehlerbehebung für die Installation von DB2-Datenbanksystemen

Falls bei der Installation von DB2-Datenbankprodukten Probleme auftreten, müssen Sie sicherstellen, dass Ihr System die Installationsvoraussetzungen erfüllt, und die Liste der allgemeinen Installationsprobleme zurate ziehen.

Vorgehensweise

Gehen Sie wie folgt vor, um Probleme bei der Installation von DB2-Datenbanksystemen zu beheben:

- Stellen Sie sicher, dass das verwendete System alle Installationsvoraussetzungen erfüllt.
- Wenn Sie auf Lizenzierungsfehler stoßen, müssen Sie sicherstellen, dass Sie die entsprechenden Lizenzen angewendet haben.

  Informieren Sie sich über die Antworten zu häufig gestellten Fragen in den technischen Hinweisen zu DB2-Lizenzfragen unter folgender Adresse: http://www.ibm.com/support/docview.wss?rs=71&uid=swg21322757
- Ziehen Sie die Liste der Installationsprobleme in der Dokumentation und auf der Website der technischen DB2-Unterstützung zurate: www.ibm.com/software/data/db2/support/db2_9/troubleshoot.html.

Nächste Schritte

Wenn Sie diese Schritte ausgeführt haben, aber die Ursache des Problems noch nicht identifizieren können, beginnen Sie mit der Erfassung von Diagnosedaten, um weitere Informationen zu erhalten.

Erfassen von Daten für Installationsprobleme

Wenn bei der Installation Probleme auftreten, deren Ursache Sie nicht feststellen können, können Sie Diagnosedaten erfassen, die Sie selbst oder die Mitarbeiter von IBM Software Support dazu verwenden können, um das Problem zu diagnostizieren und zu beheben.

Gehen Sie wie folgt vor, um Diagnosedaten für Installationsprobleme zu erfassen:

1. Optional: Wiederholen Sie den Installationsversuch mit aktivierter Tracefunktion. Beispiel:

   Unter Linux- und UNIX-Betriebssystemen:

   ```
   db2setup -t /dateipfad/trace.out
   ```

   Unter Windows-Betriebssystemen:

   ```
   setup -t \dateipfad\trace.out
   ```
2. Suchen Sie die Installationsprotokolldateien.

- Unter Windows lautet der Standarddateiname "DB2-*Produktabkürzung-DatumZeit*.log". Beispiel: DB2-ESE-Wed Jun 21 11_59_37 2006.log. Standardmäßig befindet sich das Installationsprotokoll im Verzeichnis "Eigene Dateien"\DB2LOG\ .

  Anmerkung: Wenn DB2 über den Benutzer SYSTEM installiert wird, z. B. mit Microsoft Systems Center Configuration Manager (SCCM) oder mit Tivoli, wird kein Installationsprotokoll erstellt. Der Benutzer SYSTEM verfügt nicht über den Ordner Eigene Dokumente für die Erstellung des Installationsprotokolls. Wenn Sie das Installationsprotokoll anzeigen möchten, verwenden Sie die Option **-l** des Befehls **setup**, um die Protokolldatei an einer anderen Position zu erstellen.
- Unter Linux und UNIX lauten die Standarddateinamen db2setup.log, db2setup.his und db2setup.err.

  Wenn Sie das Problem mit aktivierter Tracefunktion (bzw. aktiviertem Debugmodus) reproduziert haben, werden möglicherweise zusätzliche Dateien erstellt, z. B. dascrt.log, dasdrop.log, dasupdt.log, db2icrt.log.*PID*, db2idrop.log.*PID*, db2iupgrade.log.*PID* und db2iupdt.log.*PID*, wobei *PID* die Prozess-ID angibt.

  Das Standardverzeichnis für alle diese Dateien lautet /tmp. Für die Tracedatei (trace.out) ist kein Standardverzeichnis vorgesehen, wenn keine Verzeichnisangabe vorliegt. Geben Sie deshalb den Dateipfad zu dem Ordner an, in dem die Datei mit der Traceausgabe erstellt wurde.

3. Optional: Wenn Sie die Daten an IBM Software Support weiterleiten möchten, sollten Sie auch Daten für DB2 erfassen. Weitere Informationen hierzu finden Sie in dem Abschnitt "Erfassen von Daten für DB2".

Analysieren von Daten zu Installationsproblemen

Nach dem Erfassen von Diagnosedaten zu Installationsproblemen können Sie die Daten analysieren, um die Ursache des Problems zu ermitteln. Diese Schritte sind optional. Wenn die Ursache des Problems nur mit großem Aufwand ermittelt werden kann, übergeben Sie die Daten an IBM Software Support.

Vorbereitende Schritte

Bei diesen Schritten wird davon ausgegangen, dass Sie die unter Erfassen von Daten für Installationsprobleme beschriebenen Dateien erstellt haben.

Vorgehensweise

1. Stellen Sie sicher, dass Sie die jeweils relevante Installationsprotokolldatei vorliegen haben. Überprüfen Sie das Erstellungsdatum der Datei oder die im Dateinamen enthaltene Zeitmarke (unter Windows-Betriebssystemen).
2. Stellen Sie fest, ob die Installation erfolgreich abgeschlossen wurde.
   - Unter Windows-Betriebssystemen wird die erfolgreiche Installation durch eine Nachricht ähnlich der folgenden am Ende der Installationsprotokolldatei angegeben:

     ```
     Property(C): INSTALL_RESULT = Setup erfolgreich abgeschlossen
     === Protokollierung gestoppt: 6/21/2006  16:03:09 ===
     MSI (c) (34:38) [16:03:09:109]:
     Produkt: DB2 Enterprise Server Edition - DB2COPY1 -- Installationsoperation
     erfolgreich abgeschlossen.
     ```
   - Unter Linux- und UNIX-Betriebssystemen wird die erfolgreiche Installation durch eine Nachricht am Ende der Installationsprotokolldatei (mit dem Standarddateinamen db2setup.log) angegeben.

3. Optional: Stellen Sie fest, ob Fehler aufgetreten sind. Wenn die Installation erfolgreich abgeschlossen wurde, Sie jedoch während des Installationsprozesses eine Fehlernachricht erhalten haben, suchen Sie diese Fehler in der Installationsprotokolldatei.
   - Unter Windows-Betriebssystemen ist den meisten Fehlern "FEHLER:" oder "WARNUNG:" vorangestellt. Beispiel:

     ```
     1: ERROR: Beim Ausführen des Befehls "D:\IBM\SQLLIB\bin\db2.exe CREATE TOOLS CATALOG
     SYSTOOLS USE EXISTING DATABASE TOOLSDB FORCE" zum Initialisieren und/oder Migrieren der
     DB2-Toolskatalogdatenbank ist ein Fehler aufgetreten. Rückgabewert: "4".

     1: WARNUNG: Bei der Installation von "DB2 Enterprise Server Edition - DB2COPY1"
     auf diesem Computer ist ein geringfügiger Fehler aufgetreten. Einige Komponenten
     funktionieren möglicherweise nicht korrekt.
     ```

   - Unter Linux- und UNIX-Betriebssystemen wird eine Datei mit dem Standarddateinamen `db2setup.err` erstellt, wenn Fehler von Java zurückgegeben wurden (z. B. Ausnahmebedingungen und Trapinformationen).

   Wenn ein Installationstrace aktiviert war, enthalten die Installationsprotokolldateien eine höhere Anzahl von Einträgen, und die Einträge sind ausführlicher.

Ergebnisse

Wenn Sie mithilfe der Analyse dieser Daten das Problem nicht beheben können und Sie einen Wartungsvertrag mit IBM Software Support haben, können Sie einen Problembericht öffnen. Sie werden von IBM Software Support dazu aufgefordert, alle erfassten Daten einzureichen und gegebenenfalls Informationen zu den von Ihnen durchgeführten Analysen zur Verfügung zu stellen.

Nächste Schritte

Falls Sie durch Ihre Nachforschungen das Problem nicht beheben konnten, übergeben Sie die Daten an IBM Software Support.

Erfassen von Diagnoseinformationen bei Problemen mit der Instanzerstellung

Wenn beim Erstellen oder beim Upgrade einer Instanz Probleme auftreten, kann es erforderlich sein, Diagnoseinformationen zur Unterstützung bei der Fehlerbehebung zu erfassen. Mit dem Befehl **db2fodc -clp** können Sie die Diagnoseinformationen schnell erfassen, die IBM Support für die Analyse und Behebung des Problems benötigt.

Symptome

Beim Erstellen oder beim Upgrade einer Instanz erhalten Sie möglicherweise eine Fehlernachricht, wenn Sie versuchen, DBM CFG als Teil der Instanzerstellung zu aktualisieren. Der Fehlercode, der möglicherweise ausgegeben wird, lautet DBI1281E. Diese Fehlernachricht enthält jedoch möglicherweise nicht die zugrunde liegende Ursache des Fehlers und es sind Diagnoseinformationen erforderlich, um das Instanzerstellungsproblem genauer zu analysieren.

Fehlerbehebung

Führen Sie die folgenden Schritte aus, um unter der Anleitung von IBM Support die zur Fehlerbehebung erforderlichen Informationen ohne großen Zeitaufwand zu erfassen:

1. Geben Sie den Befehl **db2fodc -clp** ein. Mit diesem Befehl werden umgebungs- und konfigurationsbezogene Informationen erfasst, die zur Diagnose eines Instanzerstellungsproblems nützlich sind. Nach dem Abschluss der Erfassung werden die Informationen in einem neu erstellten Verzeichnis mit dem Namen FODC_Clp_<*Zeitmarke*>_<*Member*> gespeichert.

Wenn die Erfassung abgeschlossen ist, können Sie den Inhalt des Verzeichnisses FODC_Clp_<*Zeitmarke*>_<*Member*> zur Fehleranalyse an IBM Support senden.

Bekannte Probleme mit den zugehörigen Lösungen

Fehler bei der Installation eines DB2-Datenbankprodukts als Benutzer ohne Rootberechtigung im Standardpfad einer System-WPAR (AIX)

Wenn Sie DB2-Datenbankprodukte im Standardinstallationspfad (/opt/IBM/db2/V9.7) in einer Systemauslastungspartition (Workload Partition, WPAR) unter AIX 6.1 installieren, kann eine Reihe verschiedener Fehler auftreten. Installieren Sie zur Vermeidung dieser Probleme DB2-Datenbankprodukte in einem Dateisystem, auf das nur die WPAR zugreifen kann.

Symptome

Wenn Sie DB2-Datenbankprodukte im Verzeichnis /usr oder /opt in einer System-WPAR installieren, kann abhängig von der Konfiguration der Verzeichnisse eine Reihe verschiedener Fehler auftreten. System-WPARs können so konfiguriert sein, dass sie die Verzeichnisse /usr und /opt gemeinsam mit der globalen Umgebung verwenden (in diesem Fall verfügt die WPAR über Lesezugriff, nicht jedoch über Schreibzugriff auf die Verzeichnisse /usr und /opt), oder sie können so konfiguriert sein, dass sie über eine lokale Kopie der Verzeichnisse /usr und /opt verfügen.

Im ersten Szenario ist, wenn ein DB2-Datenbankprodukt im Standardpfad in der globalen Umgebung installiert wird, diese Installation in der System-WPAR sichtbar. Hierdurch entsteht der Eindruck, dass DB2 in der WPAR installiert ist; wenn jedoch versucht wird, eine DB2-Instanz zu erstellen, tritt der folgende Fehler auf: DBI1288E Die Ausführung des Programms db2icrt ist fehlgeschlagen. Dieses Programm ist fehlgeschlagen, da Sie keinen Schreibzugriff auf das Verzeichnis bzw. die Datei /opt/IBM/db2/V9.7/profiles.reg,/opt/IBM/db2/V9.7/default.env haben.

Im zweiten Szenario wird, wenn ein DB2-Datenbankprodukt im Standardpfad in der globalen Umgebung installiert ist, beim Erstellen der lokalen Kopie der Verzeichnisse /usr und /opt durch die WPAR auch die DB2Datenbankproduktinstallation kopiert. Dies kann zu unerwarteten Problemen führen, wenn ein Systemadministrator versucht, das Datenbanksystem zu verwenden. Da das DB2-Datenbankprodukt für ein anderes System bestimmt war, werden möglicherweise ungenaue Daten kopiert. So scheinen zum Beispiel ursprünglich in der globalen Umgebung erstellte DB2-Instanzen in der WPAR vorhanden zu sein. Dies kann für den Systemadministrator zu Unklarheiten in Bezug darauf führen, welche Instanzen tatsächlich auf dem System installiert sind.

Ursachen

Diese Probleme treten auf, wenn DB2-Datenbankprodukte im Verzeichnis /usr oder /opt einer System-WPAR installiert werden.

Problemlösung

Installieren Sie keine DB2-Datenbankprodukte im Standardpfad in der globalen Umgebung.

Hängen Sie ein Dateisystem an, auf das nur die WPAR zugreifen kann, und installieren Sie das DB2-Datenbankprodukt in diesem Dateisystem.

Keine Koexistenz von Beta- und Nicht-Betaversionen von DB2-Datenbankprodukten

Eine DB2-Kopie kann eines oder mehrere verschiedene DB2-Datenbankprodukte enthalten; eine Koexistenz von Beta- und Nicht-Betaversionen von Produkten ist jedoch nicht möglich. Installieren Sie keine Beta- und Nicht-Betaversionen von DB2-Datenbankprodukten an ein und derselben Speicherposition.

Diese Einschränkung gilt sowohl für Client- als auch Serverkomponenten von DB2-Datenbankprodukten.

Problemlösung

Deinstallieren Sie die Betaversion von DB2 Version 9.7, bevor Sie die Nicht-Betaversion installieren; wählen Sie andernfalls einen anderen Installationspfad aus.

Beheben von Fehlern für Servicenamen bei der Installation von DB2-Datenbankprodukten

Wenn Sie einen Nicht-Standardservicenamen oder eine Nicht-Standardportnummer für das zu verwendende DB2-Datenbankprodukt oder die zu verwendende DB2-Informationszentrale auswählen, müssen Sie sicherstellen, dass Sie keine bereits verwendeten Werte angeben.

Symptome

Wenn Sie versuchen, ein DB2-Datenbankprodukt oder die *DB2-Informationszentrale* zu installieren, meldet der **DB2-Installationsassistent** den Fehler "Der angegebene Servicename wird bereits verwendet.".

Ursachen

Der **DB2-Installationsassistent** fordert Sie dazu auf, bei der Installation der folgenden Komponenten Portnummern und Servicenamen auszuwählen:

- Die *DB2-Informationszentrale*
- Ein DB2-Datenbankprodukt, das TCP/IP-Kommunikation von Clients akzeptiert
- Ein DB2-Datenbankprodukt, das als Datenbankpartitionsserver agiert

Zu diesem Fehler kann es kommen, wenn Sie einen Servicenamen und eine Portnummer auswählen anstatt die Standardwerte zu akzeptieren. Es kommt zu diesem Fehler, wenn Sie einen Servicenamen auswählen, der bereits in der Datei `services` auf dem System vorhanden ist und Sie nur die Portnummer ändern.

Problemlösung

Führen Sie eine der folgenden Aktionen aus:

- Verwenden Sie die Standardwerte.
- Verwenden Sie einen Servicenamen und eine Portnummer, die sich bereits in der Datei `services` befinden.

- Fügen Sie einen noch nicht verwendeten Servicenamen und eine noch nicht verwendete Portnummer zur Datei `services` hinzu. Geben Sie diese Werte im **DB2-Installationsassistenten** an.

Fehlerbehebung bei Lizenzproblemen

Analysieren von DB2-Lizenzeinhaltungsberichten

Analysieren Sie einen DB2-Lizenzeinhaltungsbericht, um die Lizenzeinhaltung für Ihre DB2-Komponenten zu überprüfen.

Vorbereitende Schritte

Bei den folgenden Schritten wird davon ausgegangen, dass Sie mithilfe der Lizenzzentrale oder des Befehls **db2licm** einen DB2-Lizenzeinhaltungsbericht generiert haben.

Vorgehensweise

1. Öffnen Sie die Datei mit dem DB2-Lizenzeinhaltungsbericht.
2. Überprüfen Sie den Status der einzelnen DB2-Komponenten im Lizenzeinhaltungsbericht. Der Bericht enthält für jede Komponente einen der folgenden Werte:

   Lizenzkonform
   Gibt an, dass keine Verstöße festgestellt wurden. Die Komponente wurde verwendet und ist ordnungsgemäß lizenziert.

   Nicht verwendet
   Gibt an, dass keine Aktivitäten durchgeführt wurden, für die diese Komponente erforderlich ist.

   Verstoß
   Gibt an, dass die Komponente nicht lizenziert ist und verwendet wurde.
3. Wenn Verstöße vorliegen, verwenden Sie die Lizenzzentrale oder den Befehl **db2licm -l**, um die Lizenzinformationen anzuzeigen.

   Wenn für die DB2-Komponente der Status `"Nicht lizenziert"` aufgeführt ist, müssen Sie eine Lizenz für diese Komponente erwerben. Der Lizenzschlüssel sowie die Instruktionen für dessen Registrierung sind auf der Aktivierungs-CD verfügbar, die Sie beim Kauf einer DB2-Komponente erhalten.

   Anmerkung: Bei DB2 Workgroup Server Edition und DB2 Express Edition enthält die Beispieldatenbank SAMPLE MQTs (Materialized Query Table, gespeicherte Abfragetabelle) und MDC-Tabellen (Multidimensional Cluster), wodurch eine Lizenzverstoß entsteht. Dieser Lizenzverstoß kann nur durch ein Upgrade auf DB2 Enterprise Server Edition umgangen werden.
4. Verwenden Sie die folgenden Befehle, um zu ermitteln, welche Objekte oder Einstellungen in Ihrem DB2-Datenbankprodukt zu den Lizenzverstößen führen:
   - Für DB2 Advanced Access Control Feature:

     Überprüfen Sie, ob eine oder mehrere Tabellen die kennsatzbasierte Zugriffssteuerung (LBAC) verwenden. Führen Sie den folgenden Befehl für jede Datenbank in jeder Instanz in der DB2-Kopie aus:

     ```
     SELECT TABSCHEMA, TABNAME
     FROM SYSCAT.TABLES
     WHERE SECPOLICYID>0
     ```

- Für DB2 Performance Optimization Feature:
  - Überprüfen Sie, ob MQTs vorhanden sind. Führen Sie den folgenden Befehl für jede Datenbank in jeder Instanz in der DB2-Kopie aus:

    ```
    SELECT OWNER, TABNAME
    FROM SYSCAT.TABLES WHERE TYPE='S'
    ```

  - Überprüfen Sie, ob MDCs vorhanden sind. Führen Sie den folgenden Befehl für jede Datenbank in jeder Instanz in der DB2-Kopie aus:

    ```
    SELECT A.TABSCHEMA, A.TABNAME, A.INDNAME, A.INDSCHEMA
    FROM SYSCAT.INDEXES A, SYSCAT.TABLES B
    WHERE (A.TABNAME=B.TABNAME AND A.TABSCHEMA=B.TABSCHEMA)
    AND A.INDEXTYPE='BLOK'
    ```

  - Überprüfen Sie, ob eine oder mehrere Instanzen Abfrageparallelität (auch als *abfrageübergreifende Parallelität* bezeichnet) verwenden. Führen Sie den folgenden Befehl einmal in jeder Instanz in der DB2-Kopie aus:

    ```
    SELECT NAME, VALUE
    FROM SYSIBMADM.DBMCFG
    WHERE NAME IN ('intra_parallel')
    ```

  - Überprüfen Sie, ob der Verbindungskonzentrator aktiviert ist. Führen Sie den folgenden Befehl für jede Instanz in der DB2-Kopie aus:

    ```
    db2 get dbm cfg
    ```

    Mit diesem Befehl werden die aktuellen Werte für die Konfigurationsparameter des Datenbankmanagers einschließlich MAX_CONNECTIONS und MAX_COORDAGENTS angezeigt. Wenn der Wert für MAX_CONNECTIONS größer ist als der Wert für MAX_COORDAGENTS, ist der Verbindungskonzentrator aktiviert. Wenn Sie keine DB2 Enterprise Server Edition-, DB2 Advanced Enterprise Server Edition- oder DB2 Connect Server-Produkte verwenden, müssen Sie sicherstellen, dass der Verbindungskonzentrator inaktiviert ist. Der Grund hierfür ist, dass der Verbindungskonzentrator nur für DB2 Enterprise Server Edition-, DB2 Advanced Enterprise Server Edition- oder DB2 Connect Server-Produkte unterstützt wird.
- Für DB2 Storage Optimization Feature:
  - Überprüfen Sie, ob für eine oder mehrere Tabellen die Komprimierung auf Zeilenebene aktiviert ist. Führen Sie den folgenden Befehl für jede Datenbank in jeder Instanz in der DB2-Kopie aus:

    ```
    SELECT TABSCHEMA, TABNAME
    FROM SYSCAT.TABLES
    WHERE COMPRESSION IN ('R', 'B')
    ```

  - Überprüfen Sie, ob noch ein Komprimierungswörterverzeichnis für eine Tabelle vorhanden ist, für die die Komprimierung auf Zeilenebene inaktiviert ist. Führen Sie den folgenden Befehl für jede Datenbank in jeder Instanz in der DB2-Kopie aus:

    ```
    SELECT TABSCHEMA, TABNAME
    FROM SYSIBMADM.ADMINTABINFO
    WHERE DICTIONARY_SIZE <> 0 OR XML_DICTIONARY_SIZE <> 0
    ```

    Anmerkung: Diese Abfrage ist möglicherweise ressourcenintensiv und ihre Ausführung kann viel Zeit in Anspruch nehmen. Führen Sie diese Abfrage nur aus, wenn Storage Optimization-Lizenzverstöße gemeldet werden, obwohl keine Tabellen vorhanden sind, für die die Komprimierung auf Zeilenebene aktiviert ist.

Diagnostizieren und Beheben von Sperrenfehlern

Bei einem Sperrenproblem müssen Sie zunächst den Typ des Sperrereignisses diagnostizieren, das die SQL-Abfrageleistung beeinträchtigt oder die Ausführung einer Abfrage verhindert, sowie die beteiligte(n) SQL-Anweisung(en) ermitteln. In diesem Abschnitt wird erläutert, wie Sie den Typ des Sperrenproblems feststellen und das Problem lösen können.

Einführung

Ein Sperrenproblem liegt vor, wenn Anwendungen ihre Tasks nicht mehr ausführen können oder die Leistung von SQL-Abfragen aufgrund von Sperren abnimmt. Ziel ist es deshalb zu erreichen, dass Überschreitungen der Sperrzeit und Deadlocks durch Sperren, die die Taskausführung von Anwendungen blockieren, in einem Datenbanksystem nicht auftreten.

Wartestatus für Sperren stellen normale, zu erwartende Ereignisse dar. Muss jedoch zu lange auf eine Sperre gewartet werden, können diese Wartestatus sowohl die SQL-Abfrageleistung als auch die Anwendungsausführung verlangsamen. Bei zu langen Wartestatus können Überschreitungen der Sperrzeit auftreten, die schließlich dazu führen, dass die betroffene Anwendung die zugehörigen Tasks nicht ausführen kann.

Sperreneskalationen zählen zu den Sperrenfehlern, wenn sie zu Überschreitungen bei der Sperrzeit beitragen. Sperreneskalationen sollten möglichst nicht auftreten, eine kleine Anzahl von Eskalationen ist jedoch akzeptabel, sofern sie keine negativen Auswirkungen haben.

Es empfiehlt sich, Ereignisse, die Wartestatus für Sperren, Überschreitungen der Sperrzeit oder Deadlocks durch Sperren beinhalten, durchgängig zu überwachen. Dies geschieht im Allgemeinen auf Auslastungsebene (bei den Wartestatus für Sperren) und auf Datenbankebene (bei Überschreitungen der Sperrzeit und Deadlocks durch Sperren).

Die Diagnose eines aufgetretenen Sperrenfehlertyps und das Beheben des Fehlers beginnt mit dem Erfassen von Informationen und der Suche nach Diagnosehinweisen. Die folgenden Abschnitte enthalten hilfreiche Anleitungen für diese Zwecke.

Erfassen von Informationen

Im Allgemeinen setzt die objektive Einstufung eines abnormalen Systemverhaltens (z. B. bei Verarbeitungsverzögerungen und Leistungsbeeinträchtigungen) voraus, dass als Vergleichswert Informationen vorliegen, die das typische Verhalten des Systems beschreiben. Auf diese Weise kann im Bedarfsfall ein Vergleich zwischen dem normalen Verhalten und dem von Ihnen beobachteten Verhalten, das Sie für potenziell abnormal halten, gezogen werden. Das Erfassen von Vergleichsdaten durch das Einplanen regelmäßig auszuführender Tasks für die Überwachung des Systembetriebs ist von elementarer Bedeutung für die Fehlerbehebung. Nähere Informationen zum Einrichten einer Datenbasis für einen normalen Systembetrieb, die als Vergleichsmaßstab dienen kann, finden Sie in „Leistungsbezogene Betriebsüberwachung” auf Seite 13.

Um feststellen zu können, welcher Typ von Sperrenfehler die SQL-Abfrageleistung beeinträchtigt oder die Ausführung von Abfragen verhindert, müssen Sie Informationen erfassen, die eine Bestimmung des Typs des Sperrereignisses ermöglichen, eine Bestimmung der Anwendung, die die Sperre anfordert oder hält, sowie eine

Bestimmung der von der Anwendung während des Ereignisses ausgeführten Aktivitäten und der SQL-Anweisung(en), die verlangsamt ausgeführt wird/werden.

Diese Informationen können über das Erstellen eines Ereignismonitors für Sperrereignisse oder mithilfe einer Tabellenfunktion oder des Befehls **db2pd** erfasst werden. Die vom Ereignismonitor für Sperrereignisse erfassten Daten lassen sich in drei Hauptkategorien unterteilen:

- Informationen zur Sperre
- Informationen zu der Anwendung, die die Sperre anfordert, und den aktuellen Aktivitäten dieser Anwendung. Im Falle eines Deadlocks handelt es sich dabei um die Anweisung, zu deren Lasten der Deadlock geht (auch Victim oder Opfer genannt).
- Informationen zu der Anwendung, die die Sperre hält, und den aktuellen Aktivitäten dieser Anwendung. Im Falle eines Deadlocks handelt es sich dabei um die Anweisung, die als beteiligt angegeben wird.

Anweisungen zum Überwachen von Ereignissen, die Wartestatus für Sperren, Überschreitungen der Sperrzeit oder Deadlocks beinhalten, finden Sie in „Überwachung von Datenbanksperren" in *Datenbanküberwachung - Handbuch und Referenz*.

Suche nach Diagnosehinweisen

Informationen, die Aufschluss über die Art von Sperrenfehler geben, können mit dem Ereignismonitor für Sperrereignisse, einer Tabellenfunktion oder mit dem Befehl **db2pd** erfasst werden. Informationen, die Hinweise zur Diagnose liefern und die Diagnose des bei Ihnen vorliegenden Sperrenfehlertyps sichern können, finden Sie vor allem in den folgenden Abschnitten:

- Wenn Sie lange Wartezeiten beobachten, ohne dass Überschreitungen der Sperrzeit auftreten, liegt vermutlich ein Problem mit Wartestatus für Sperren vor. Nähere Informationen hierzu finden Sie in dem Abschnitt Diagnostizieren von Fehlern durch Wartestatus für Sperren.
- Wenn die Anzahl von Deadlocks höher als normal ist, liegt vermutlich ein durch Deadlocks verursachtes Problem vor. Nähere Informationen hierzu finden Sie in dem Abschnitt Diagnostizieren von Deadlock-Fehlern.
- Wenn die Anzahl der Überschreitungen bei der Sperrzeit erhöht und der Datenbankkonfigurationsparameter **locktimeout** auf einen Wert ungleich null gesetzt ist, liegt vermutlich ein Problem mit Überschreitungen der Sperrzeit vor. Nähere Informationen hierzu finden Sie in dem Abschnitt Diagnostizieren von Fehlern durch Überschreitung der Sperrzeit. Beachten Sie in diesem Fall auch die Informationen, die im Falle eines Fehlers durch Wartestatus für Sperren relevant sind.
- Wenn die Anzahl von Wartestatus für Sperren erhöht ist und der Ereignismonitor für Sperrereignisse angibt, dass Sperreneskalationen auftreten, liegt vermutlich ein Problem bei der Sperreneskalation vor. Nähere Informationen hierzu finden Sie im folgenden Abschnitt: Diagnostizieren von Sperreneskalationsfehlern.

Diagnostizieren von Fehlern durch Wartestatus für Sperren

Wartestatus für Sperren entstehen, wenn eine Transaktion versucht, eine Sperre für eine Ressource abzurufen, die bereits für eine andere Transaktion aktiviert ist. Ein lang andauernder Wartestatus für Sperren führt zu einer Verlangsamung bei der Ausführung von SQL-Abfragen. Ein Fehler durch Wartestatus für Sperren liegt in der Regel vor, wenn lange oder unerwartet viele Wartestatus für Sperren vorliegen, ohne dass Überschreitungen bei der Sperrzeit auftreten.

Vorbereitende Schritte

Im Allgemeinen setzt die objektive Einstufung eines abnormalen Systemverhaltens (z. B. bei Verarbeitungsverzögerungen und Leistungsbeeinträchtigungen) voraus, dass als Vergleichswert Informationen vorliegen, die das typische Verhalten des Systems beschreiben. Auf diese Weise kann im Bedarfsfall ein Vergleich zwischen dem normalen Verhalten und dem von Ihnen beobachteten Verhalten, das Sie für potenziell abnormal halten, gezogen werden. Das Erfassen von Vergleichsdaten durch das Einplanen regelmäßig auszuführender Tasks für die Überwachung des Systembetriebs ist von elementarer Bedeutung für die Fehlerbehebung. Nähere Informationen zum Einrichten einer Datenbasis für einen normalen Systembetrieb, die als Vergleichsmaßstab dienen kann, finden Sie in „Leistungsbezogene Betriebsüberwachung" auf Seite 13.

Anweisungen zum Überwachen von Wartestatusereignissen für Sperren finden Sie in „Überwachung von Datenbanksperren" in *Datenbanküberwachung - Handbuch und Referenz*.

Informationen zu diesem Vorgang

Diagnose

Ein Wartestatus für Sperren tritt ein, wenn eine Transaktion (bestehend aus mindestens einer SQL-Anweisung) versucht, eine Sperre abzurufen, deren Modus zu Konflikten mit einer Sperre führt, die für eine andere Transaktion aktiviert ist. Übermäßige Wartezeiten für Sperren schlagen sich häufig in einer schlechten Antwortzeit nieder, sodass es wichtig ist, die Wartezeit zu überwachen. Der Umfang der Wartestatus für Sperren wird am besten bezogen auf Tausend Transaktionen normalisiert, weil die Wartezeit für Sperren für eine einzelne Transaktion in der Regel recht gering ist und normalisierte Messwerte einfacher zu handhaben sind.

Es gibt verschiedene Merkmale bei den Wartestatus für Sperren, die bei der Diagnose der einzelnen Wartestatus berücksichtigt werden müssen. Im Folgenden werden diese Merkmale bei den Wartestatus für Sperren beschrieben und die jeweils einfachsten Diagnosevarianten erläutert:

- Einzelne lange Wartestatus für Sperren
  - Untersuchen Sie die Spitzenwartezeiten für Sperren je nach Serviceklasse und Auslastung (Workload). Beziehen Sie den Ereignismonitor für Sperren auf die Auslastung, um diesen Wert zu ermitteln.
- Lange Wartezeiten für Sperren bei einzelnen kurzen Wartestatus für Sperren.
  - Hierbei handelt es sich in der Regel um eine Folge von Sperren. Ermitteln Sie mit dem Befehl **db2pd -locks wait**, ob Warteketten vorliegen.
- Typen der Sperren, auf die gewartet wird
  - Der Fehler kann möglicherweise leichter behoben werden, wenn Sie den Sperrentyp kennen. Verschaffen Sie sich Informationen zum Sperrentyp, indem Sie den Agenten ermitteln, der auf die Sperre wartet. Mithilfe der Informationen zum Sperrentyp können Sie feststellen, ob ein offensichtliches Problem vorliegt. Eine Paketsperre kann z. B. darauf hinweisen, dass ein Befehl **BIND**/**REBIND** oder eine DDL-Anweisung vorliegt, der/die zu einem Konflikt mit dem Benutzer des Pakets führt. Eine interne Katalogcache-Sperre (c) kann dadurch verursacht werden, dass eine DDL-Anweisung zu einem Konflikt bei der Kompilierung einer Anweisung führt.

Typische Anzeichen

Achten Sie auf die folgenden Anzeichen für Wartestatus für Sperren:

- Die Anzahl der Wartestatus für Sperren steigt (Anstieg bei dem Wert für das Monitorelement **lock_waits**)
- Ein hoher Prozentsatz aktiver Agenten wartet auf Sperren (z. B. 20 % oder mehr der Agenten insgesamt). In dem nachfolgenden Abschnitt mit dem Titel „Zu überwachende Elemente" wird erläutert, wie Sie sich über diese Werte informieren können.
- Der Wert für die Wartezeit für Sperren, der auf Datenbank- oder Auslastungsebene erfasst wird, steigt (Monitorelement **lock_wait_time**).

Zu überwachende Elemente

Im Unterschied zu anderen Typen von DB2-Überwachungsdaten sind Informationen zu Sperren sehr kurzlebig. Abgesehen vom Monitorelement **lock_wait_time**, das eine laufende Summe angibt, verschwinden die meisten anderen Informationen über Sperren, wenn die Sperren selbst freigegeben werden. Daher haben Ereignisdaten über Sperren und Sperrenwartezeiten den größten Wert, wenn sie in regelmäßigen Abständen über einen gewissen Zeitraum erfasst werden, sodass sich ein aussagekräftigeres Bild ergibt.

Erfassen Sie Informationen zu aktiven Agenten, die auf Sperren warten, mithilfe der Tabellenfunktion WLM_GET_SERVICE_CLASS_AGENTS_V97. Agenten, die auf Sperren warten, sind an den folgenden Attribut-Wert-Paaren zu erkennen:

- EVENT_OBJECT = LOCK
- EVENT_TYPE = ACQUIRE

Informationen zu aktiven Agenten, die auf Sperren warten, können Sie auch mithilfe von Momentaufnahmen zu Anwendungen, mit den Verwaltungssichten für Sperren sowie mithilfe der Option für Sperrenwartestatus des Befehls **db2pd -wlocks** abrufen.

Folgende Monitorelemente sind in diesem Zusammenhang in erster Linie relevant:

- Anstieg beim Wert für **lock_waits**
- Hoher Wert für **lock_wait_time**

Wenn bei Ihnen an dieser Stelle beschriebene Anzeichen auftreten, liegt in aller Regel ein Problem mit Wartestatus für Sperren vor. Nähere Informationen dazu, wie Sie dieses Problem lösen können, können Sie über den Link im Abschnitt „Weitere Schritte" abrufen.

Nächste Schritte

Wenn sich herausgestellt hat, dass der bei Ihnen aufgetretene Fehler vermutlich durch Wartestatus für Sperren ausgelöst wird, können Sie diesen Fehler mit den im folgenden Abschnitt beschriebenen Schritten beheben: „Beheben von Fehlern durch Wartestatus für Sperren".

Beheben von Fehlern durch Wartestatus für Sperren

Wenn ein Problem mit Wartestatus für Sperren vorliegt, müssen Sie zunächst versuchen, das Problem, das durch eine Anwendung entsteht, die zu lange auf eine Sperre wartet, zu lösen. In diesem Abschnitt finden Sie Richtlinien, die Ihnen das

Beheben von Problemen mit Wartestatus für Sperren erleichtern und dazu beitragen, dass derartige Störungen in der Zukunft vermieden werden.

Vorbereitende Schritte

Stellen Sie sicher, dass es sich bei dem aufgetretenen Fehler um einen Fehler durch Wartestatus für Sperren handelt, indem Sie die für die Diagnose von Sperrenfehlern erforderlichen Schritte ausführen (siehe „Diagnostizieren und Beheben von Sperrenfehlern" auf Seite 593).

Informationen zu diesem Vorgang

Die Richtlinien in diesem Abschnitt helfen Ihnen beim Beheben des bei Ihnen aufgetretenen Problems mit Wartestatus für Sperren und tragen dazu bei, dass ähnliche Störungen in Zukunft vermieden werden.

Vorgehensweise

Führen Sie die folgenden Schritte aus, um die Ursache für das inakzeptable Problem, das die Wartestatus für Sperren darstellen, zu ermitteln und Abhilfe zu schaffen:

1. Informieren Sie sich anhand des Protokolls mit Benachrichtigungen für die Systemverwaltung über alle Tabellen, bei denen Agenten übermäßig lange auf Sperren warten.
2. Verwenden Sie die Informationen im Protokoll mit Benachrichtigungen für die Systemverwaltung, um zu entscheiden, wie das Problem mit den Wartestatus für Sperren zu lösen ist. Einige Richtlinien können Ihnen helfen, Sperrenkonflikte und Wartezeiten für Sperren zu verringern. Ziehen Sie die folgenden Optionen in Betracht:
   - Vermeiden Sie nach Möglichkeit lange Transaktionen sowie Cursor, die mit WITH HOLD definiert sind. Je länger Sperren aktiviert bleiben, desto höher ist die Wahrscheinlichkeit, dass sie Konflikte mit anderen Anwendungen verursachen. Dies ist nur bei hohen Werten für die Isolationsstufe relevant.
   - Es empfiehlt sich, für die folgenden Aktionen so bald wie möglich ein Commit durchzuführen:
     – Schreibaktionen wie Löschen, Einfügen und Aktualisieren
     – DDL-Anweisungen wie ALTER, CREATE und DROP
     – Befehle **BIND** und **REBIND**
   - Führen Sie nach der Ausgabe von DDL-Anweisungen ALTER oder DROP die Prozedur SYSPROC.ADMIN_REVALIDATE_DB_OBJECTS aus, um vorhandene Datenobjekte zu reaktivieren. Führen Sie auch den Befehl **db2rbind** aus, um einen Rebind für vorhandene Pakete durchzuführen.
   - Vermeiden Sie das Abrufen von Ergebnismengen, die größer sind als erforderlich, insbesondere unter der Isolationsstufe 'Wiederholbares Lesen' (RR). Je mehr Zeilen gelesen werden, desto mehr Sperren werden aktiviert und desto größer ist die Wahrscheinlichkeit, auf eine Zeile zu treffen, die bereits von einer anderen Anwendung gesperrt ist. In der Praxis bedeutet dies in vielen Fällen, die Zeilenauswahlbedingungen in eine WHERE-Klausel der SELECT-Anweisung zu verschieben, anstatt eine größere Anzahl Zeilen in die Anwendung abzurufen und dort zu filtern. Beispiel:

     ```
     exec sql declare curs for
       select c1,c2 from t
       where c1 not null;
     exec sql open curs;
     ```

```
do {
  exec sql fetch curs
    into :c1, :c2;
} while( P(c1) != someVar );

==>

exec sql declare curs for
  select c1,c2 from t
  where c1 not null
  and myUdfP(c1) = :someVar;
exec sql open curs;
exec sql fetch curs
    into :c1, :c2;
```

- Vermeiden Sie die Verwendung höherer Isolationsstufen als erforderlich. Die Isolationsstufe 'Wiederholbares Lesen' (RR) ist möglicherweise erforderlich, um die Integrität von Ergebnismengen in Ihrer Anwendung sicherzustellen. Allerdings verursacht sie zusätzlichen Aufwand durch aktivierte Sperren und potenzielle Sperrenkonflikte.
- Möglicherweise empfiehlt es sich, das Sperrverhalten über die Registrierdatenbankvariablen **DB2_EVALUNCOMMITTED**, **DB2_SKIPDELETED** und **DB2_SKIPINSERTED** zu ändern, wenn dies für die Geschäftslogik der Anwendung sinnvoll erscheint. Durch diese Registrierdatenbankvariablen kann der DB2-Datenbankmanager das Aktivieren von Sperren unter bestimmten Umständen verzögern oder vermeiden, sodass sich die Konfliktwahrscheinlichkeit verringert und der Durchsatz potenziell erhöht.
- Vermeiden Sie Sperreneskalationen, soweit möglich.

Nächste Schritte

Führen Sie die Anwendung(en) erneut aus, um sicherzustellen, dass der Sperrenfehler beseitigt ist. Überprüfen Sie dazu das Protokoll mit Benachrichtigungen für die Systemverwaltung auf Einträge zu Sperren oder überprüfen Sie die Messdaten für Wartestatus und Wartezeit für Sperren für die entsprechende Workload, Verbindung, Serviceunterklasse und UOW sowie für einen unterschiedlichen Auslastungsgrad.

Diagnose von Deadlock-Fehlern

Ein Deadlock entsteht, 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 ihre Ausführung ohne die Intervention eines Deadlock-Detektors fortsetzen kann. Durch das Deadlock und die damit einhergehende Wartezeit für die Deadlock-Erkennung verlangsamt sich die beteiligte Transaktion. Es werden Systemressourcen durch die Rollback-Operation bei der zur Bereinigung ausgewählten Transaktion verschwendet und es entstehen zusätzliche Systembelastungen und Transaktionsprotokollzugriffe während des gesamten Prozesses. Ein Deadlock-Fehler liegt in der Regel vor, wenn die Anzahl der Deadlocks höher ist als normal und Transaktionen erneut ausgeführt werden.

Vorbereitende Schritte

Im Allgemeinen wird jeder aufgetretene Deadlock als eine abnormale Bedingung betrachtet. Um jedoch objektiv feststellen zu können, ob ein abnormales Systemverhalten (z. B. Verarbeitungsverzögerungen und Leistungsbeeinträchtigungen) vorliegt, müssen als Vergleichswert Informationen vorliegen, die das typische Verhalten des Systems beschreiben. Auf diese Weise kann im Bedarfsfall ein Vergleich zwischen dem normalen Verhalten und dem von Ihnen beobachteten Verhalten,

das Sie für potenziell abnormal halten, gezogen werden. Das Erfassen von Vergleichsdaten durch das Einplanen regelmäßig auszuführender Tasks für die Überwachung des Systembetriebs ist von elementarer Bedeutung für die Fehlerbehebung..

Anweisungen zum Überwachen von Sperrereignissen durch Deadlocks finden Sie in „Überwachung von Datenbanksperren" in *Datenbanküberwachung - Handbuch und Referenz*.

Informationen zu diesem Vorgang

Diagnose

Ein Deadlock entsteht, 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 ihre Ausführung ohne die Intervention eines Deadlock-Detektors fortsetzen kann. Nachdem die durch den Deadlock blockierte Transaktion vom System automatisch rückgängig gemacht wurde, muss die Anwendung, zu deren Lasten der Deadlock geht (das Deadlock-Opfer), die betreffende Transaktion von Anfang an erneut ausführen. Die Überwachung der Rate, mit der solche Fälle auftreten, hilft bei der Vermeidung einer Situation, in der viele Deadlocks eine erhebliche Zusatzbelastung im System verursachen, ohne dass sich der Datenbankadministrator dessen bewusst ist.

Typische Anzeichen

Folgende Anzeichen weisen darauf hin, dass Deadlocks vorliegen:

- Von mindestens einer Anwendung werden gelegentlich Transaktionen erneut ausgeführt.
- Das Protokoll mit Benachrichtigungen für die Systemverwaltung enthält Nachrichteneinträge zu Deadlocks.
- Für das Monitorelement **deadlocks** wird eine erhöhte Anzahl von Deadlocks angezeigt.
- Für das Monitorelement **int_deadlock_rollbacks** wird eine erhöhte Anzahl von Rollback-Operationen angezeigt.
- Die Zeit, in der Agenten auf das Schreiben von Protokolleinträgen auf Platte warten, ist erhöht. Dieser Wert wird für das Monitorelement **log_disk_wait_time** angezeigt.

Zu überwachende Elemente

Der Aufwand für einen Deadlock variiert und ist direkt proportional zur Länge der durch Rollback rückgängig gemachten Transaktion. Trotzdem gilt in der Regel, dass jeder Deadlock auf ein Problem hinweist.

Deadlockereignisse können im Wesentlichen auf folgende Arten ermittelt werden:

1. Definieren eines Ereignismonitors für Sperrereignisse und des Datenbankkonfigurationsparameters **mon_deadlock** zum Erfassen von Details zu allen Deadlockereignissen, die in der gesamten Datenbank auftreten
2. Überwachen des Protokolls mit Benachrichtigungen für die Systemverwaltung auf Nachrichten zu Deadlocks einschließlich allgemeiner Informationen, die sich auf Deadlocks beziehen

   Anmerkung: Zur Aktivierung der Funktion zum Schreiben von Nachrichten zu Deadlocks in die Protokolldatei mit Benachrich-

tigungen für die Systemverwaltung setzen Sie den Datenbankkonfigurationsparameter **mon_lck_msg_lvl** auf den Wert 2.

3. Überwachen der in diesem Zusammenhang relevanten Monitorelemente über eine Tabellenfunktion

Die meisten Benutzer entscheiden sich für die erste Option. Durch das Überwachen der relevanten Monitorelemente zum Ermitteln aufgetretener Deadlocks können Benutzer detaillierte Informationen abrufen, indem sie sich über die vom Ereignismonitor erfassten Daten informieren.

Folgende Monitorelemente sind in diesem Zusammenhang in erster Linie relevant:

- Der Wert für **deadlocks** ist ungleich null.
- Für **int_deadlock_rollbacks** wird eine erhöhte Anzahl von Rollbacks angezeigt, die aufgrund von Deadlockereignissen entstehen.
- Bei **log_disk_wait_time** steigt der Wert für die Zeit an, in der Agenten auf das Speichern von Protokollen auf Platte warten.

Wenn bei Ihnen an dieser Stelle beschriebene Anzeichen auftreten, liegt in aller Regel ein Problem mit Deadlocks vor. Nähere Informationen dazu, wie Sie dieses Problem lösen können, können Sie über den Link im Abschnitt „Weitere Schritte" abrufen.

Nächste Schritte

Wenn sich herausgestellt hat, dass das bei Ihnen aufgetretene Problem vermutlich durch Deadlocks ausgelöst wird, können Sie dieses Problem mit den im folgenden Abschnitt beschriebenen Schritten lösen: „Beheben von Deadlockproblemen".

Beheben von Deadlockproblemen

Nach der Diagnose eines Deadlockproblems müssen Sie zunächst versuchen, die Deadlocksituation zwischen den beiden gleichzeitig ausgeführten Anwendungen zu lösen, die eine Sperre für eine Ressource halten, die von der jeweils anderen Anwendung benötigt wird. Die Richtlinien in diesem Abschnitt helfen Ihnen beim Beheben des bei Ihnen aufgetretenen Deadlockproblems und tragen dazu bei, dass ähnliche Störungen in Zukunft vermieden werden.

Vorbereitende Schritte

Stellen Sie sicher, dass es sich bei dem aufgetretenen Fehler um ein Deadlockproblem handelt, indem Sie die für die Diagnose von Sperrenfehlern erforderlichen Schritte ausführen (siehe „Diagnostizieren und Beheben von Sperrenfehlern" auf Seite 593).

Informationen zu diesem Vorgang

Die Richtlinien in diesem Abschnitt helfen Ihnen beim Beheben des bei Ihnen aufgetretenen Deadlockproblems und tragen dazu bei, dass ähnliche Störungen in Zukunft vermieden werden.

Vorgehensweise

Führen Sie die folgenden Schritte aus, um die Ursache für das inakzeptable Deadlockproblem zu ermitteln und Abhilfe zu schaffen:

1. Verschaffen Sie sich anhand des Ereignismonitors für Sperren oder des Protokolls mit Benachrichtigungen für die Systemverwaltung einen Überblick über alle Tabellen, bei denen Agenten Deadlocks feststellen.
2. Entscheiden Sie anhand der Informationen im Protokoll mit Benachrichtigungen für die Systemverwaltung, wie das Deadlockproblem zu beheben ist. Es gibt einige Richtlinien, mit denen sich Sperrenkonflikte und Wartezeiten für Sperren verringern lassen. Ziehen Sie die folgenden Optionen in Betracht:
   - Jede Anwendungsverbindung sollte eigene Gruppen von Zeilen verarbeiten, um Wartestatus für Sperren zu vermeiden.
   - Die Häufigkeit von Deadlocks lässt sich teilweise dadurch verringern, dass alle Anwendungen auf ihre allgemeinen Daten in der gleichen Reihenfolge zugreifen. Das heißt zum Beispiel, dass sie auf Zeilen in Tabelle A zugreifen (und diese natürlich sperren), dann auf Zeilen in Tabelle B, anschließend auf Zeilen in Tabelle C usw. Wenn zwei Anwendungen inkompatible Sperren für dieselben Objekte in unterschiedlicher Reihenfolge aktivieren, besteht ein wesentlich höheres Risiko, dass es zu einem Deadlock kommt.
   - Eine Überschreitung der Sperrzeit ist nicht unbedingt unproblematischer als ein Deadlockproblem, da in beiden Fällen ein Rollback von Transaktionen eintritt. Wenn Sie die Anzahl von Deadlocks jedoch verringern müssen, können Sie dies erreichen, indem Sie sicherstellen, dass eine Überschreitung der Sperrzeit in der Regel eintritt, bevor ein möglicherweise damit in Verbindung stehender Deadlockfehler festgestellt wird. Wählen Sie dazu für den Datenbankkonfigurationsparameter **locktimeout** (in Sekunden) einen wesentlich niedrigeren Wert als für den Datenbankkonfigurationsparameter **dlchktime** (in Millisekunden). Ist das Intervall für **locktimeout** länger als das Intervall für **dlchktime**, könnte der Deadlock-Detektor sofort nach Eintreten der Deadlocksituation aktiviert werden und den Deadlock erkennen, bevor die Sperrzeit überschritten wird.
   - Vermeiden Sie gleichzeitige DDL-Operationen, soweit möglich. Die Anweisungen DROP TABLE können z. B. zu einer hohen Anzahl von Katalogaktualisierungen führen, da möglicherweise Zeilen für die Tabellenindizes, für die primären Schlüssel, für Prüfungen auf Integritätsbedingungen etc. sowie die Tabelle selbst gelöscht werden müssen. Werden von anderen DDL-Operationen Objekte gelöscht und erstellt, können Sperrenkonflikte und teilweise sogar Deadlocks auftreten.
   - Es empfiehlt sich, für die folgenden Aktionen so bald wie möglich ein Commit durchzuführen:
     – Schreibaktionen wie Löschen, Einfügen und Aktualisieren
     – DDL-Anweisungen wie ALTER, CREATE und DROP
     – Befehle **BIND** und **REBIND**
3. Die folgende Situation kann vom Deadlock-Detektor nicht festgestellt und auch nicht behoben werden. Diese Situation muss im Anwendungsentwurf deshalb von vorneherein vermieden werden. Bei einer Anwendung, insbesondere einer Multithread-Anwendung, kann ein Deadlock auftreten, bei dem ein DB2-Wartestatus für Sperren und ein Wartestatus für eine von DB2 unabhängige Ressource, z. B. ein Semaphor, vorliegen. Verbindung A kann z. B. auf eine Sperre warten, die von Verbindung B gehalten wird, und Verbindung B kann auf ein Semaphor warten, das von Verbindung A gehalten wird.

Nächste Schritte

Führen Sie die Anwendung(en) erneut aus, um sicherzustellen, dass das Sperrenproblem beseitigt ist. Überprüfen Sie dazu das Protokoll mit Benachrichtigungen für die Systemverwaltung auf Einträge, die sich auf Sperren beziehen.

Diagnostizieren von Fehlern durch Überschreitung der Sperrzeit

Eine Überschreitung der Sperrzeit tritt auf, wenn eine Transaktion, die auf eine Ressourcensperre wartet, das Zeitlimit für die Wartezeit überschreitet, das durch den Datenbankkonfigurationsparameter **locktimeout** angegeben wird. Da dadurch ein Zeitverlust entsteht, verlangsamt sich die SQL-Abfrageleistung. Ein Fehler durch eine Überschreitung der Sperrzeit liegt im Allgemeinen vor, wenn die Anzahl der Überschreitungen bei der Sperrzeit erhöht und der Datenbankkonfigurationsparameter **locktimeout** auf einen Wert ungleich null gesetzt ist.

Vorbereitende Schritte

Im Allgemeinen setzt die objektive Einstufung eines abnormalen Systemverhaltens (z. B. bei Verarbeitungsverzögerungen und Leistungsbeeinträchtigungen) voraus, dass als Vergleichswert Informationen vorliegen, die das typische Verhalten des Systems beschreiben. Auf diese Weise kann im Bedarfsfall ein Vergleich zwischen dem normalen Verhalten und dem von Ihnen beobachteten Verhalten, das Sie für potenziell abnormal halten, gezogen werden. Das Erfassen von Vergleichsdaten durch das Einplanen regelmäßig auszuführender Tasks für die Überwachung des Systembetriebs ist von elementarer Bedeutung für die Fehlerbehebung. .

Anweisungen zum Überwachen von Ereignissen aufgrund von Überschreitungen der Sperrzeit finden Sie in „Überwachung von Datenbanksperren" in *Datenbanküberwachung - Handbuch und Referenz*.

Informationen zu diesem Vorgang

Diagnose

Situationen mit einem Wartestatus für Sperren können in einigen Fällen auch zu Überschreitungen der Sperrzeit führen, die wiederum ein Rollback von Transaktionen verursachen. Der Zeitraum, der bei einem Wartestatus für Sperren verstreichen muss, bis eine Überschreitung der Sperrzeit eintritt, wird durch den Datenbankkonfigurationsparameter **locktimeout** bestimmt. Überschreitungen der Sperrzeit können den Systembetrieb ebenso stark beeinträchtigen wie Deadlocks. Während Deadlocks in den meisten Produktionssystemen vergleichsweise selten auftreten, können Überschreitungen der Sperrzeit häufiger vorkommen. Die Anwendung muss sie gewöhnlich auf ähnliche Weise behandeln: Die Ausführung der Transaktion muss von Anfang an wiederholt werden. Durch eine Überwachung der Rate, mit der solche Fälle auftreten, können Situationen vermieden werden, in der Überschreitungen der Sperrzeit eine erhebliche Zusatzbelastung im System verursachen, ohne dass sich der Datenbankadministrator dessen bewusst ist.

Typische Anzeichen

Achten Sie auf die folgenden Anzeichen für Überschreitungen der Sperrzeit:

- Anwendungen, bei denen Transaktionen häufig erneut ausgeführt werden
- Steigende Werte beim Monitorelement **lock_timeouts**
- Nachrichteneinträge zu Überschreitungen der Sperrzeit im Protokoll mit Benachrichtigungen für die Systemverwaltung

Zu überwachende Elemente

Aufgrund der relativ flüchtigen Natur der Sperrereignisse haben Ereignisdaten über Sperren den größten Wert, wenn sie in regelmäßigen Abständen über einen gewissen Zeitraum erfasst werden, sodass sich ein aussagekräftigeres Bild ergibt.

Sie können das Protokoll mit Benachrichtigungen für die Systemverwaltung auf Nachrichten zu Überschreitungen der Sperrzeit überwachen.

Anmerkung: Zur Aktivierung der Funktion zum Schreiben von Nachrichten zu Überschreitungen der Sperrzeit in die Protokolldatei mit Benachrichtigungen für die Systemverwaltung setzen Sie den Datenbankkonfigurationsparameter **mon_lck_msg_lvl** auf den Wert 3.

Erstellen Sie einen Ereignismonitor, mit dem Daten zu Überschreitungen der Sperrzeit für eine Auslastung (Workload) oder Datenbank erfasst werden.

Folgende Monitorelemente sind in diesem Zusammenhang in erster Linie relevant:

- Der Wert für **lock_timeouts** steigt an.
- Der Wert für **int_rollbacks** steigt an.

Wenn bei Ihnen an dieser Stelle beschriebene Anzeichen aufgetreten sind, liegt in aller Regel ein Problem mit Überschreitungen der Sperrzeit vor. Nähere Informationen zu diesem Thema können Sie über den Link im Abschnitt „Weitere Schritte" abrufen.

Nächste Schritte

Wenn sich herausgestellt hat, dass der bei Ihnen aufgetretene Fehler vermutlich durch Überschreitungen der Sperrzeit ausgelöst wird, können Sie diesen Fehler mit den im Abschnitt „Beheben von Fehlern durch Überschreitung der Sperrzeit" beschriebenen Schritten beheben.

Beheben von Fehlern durch Überschreitung der Sperrzeit

Wenn ein Problem mit Überschreitungen der Sperrzeit vorliegt, müssen Sie zunächst versuchen, das Problem, das durch Anwendungen entsteht, die beim Warten auf eine Sperre das Zeitlimit überschreiten, zu lösen. Die Richtlinien in diesem Abschnitt helfen Ihnen beim Beheben des bei Ihnen aufgetretenen Problems mit Sperrzeitüberschreitungen und tragen dazu bei, dass ähnliche Störungen in Zukunft vermieden werden.

Vorbereitende Schritte

Stellen Sie sicher, dass es sich bei dem aufgetretenen Problem um ein Problem mit Sperrzeitüberschreitungen handelt, indem Sie die für die Diagnose von Sperrenfehlern erforderlichen Schritte ausführen (siehe „Diagnostizieren und Beheben von Sperrenfehlern" auf Seite 593).

Informationen zu diesem Vorgang

Die Richtlinien in diesem Abschnitt helfen Ihnen beim Beheben des bei Ihnen aufgetretenen Problems mit Sperrzeitüberschreitungen und tragen dazu bei, dass ähnliche Störungen in Zukunft vermieden werden.

Vorgehensweise

Führen Sie die folgenden Schritte aus, um die Ursache für das inakzeptable Problem mit Sperrzeitüberschreitungen zu ermitteln und Abhilfe zu schaffen:

1. Verschaffen Sie sich anhand des Ereignismonitors für Sperren oder des Protokolls mit Benachrichtigungen für die Systemverwaltung einen Überblick über alle Tabellen, bei denen Agenten Überschreitungen der Sperrzeit feststellen.
2. Verwenden Sie die Informationen im Protokoll mit Benachrichtigungen für die Systemverwaltung, um zu entscheiden, wie das Problem mit den Überschreitungen der Sperrzeit zu lösen ist. Einige Richtlinien können Ihnen helfen, Sperrenkonflikte und Wartezeiten für Sperren und somit auch die Anzahl der Sperrzeitüberschreitungen zu verringern. Ziehen Sie die folgenden Optionen in Betracht:
   - Optimieren Sie den Datenbankkonfigurationsparameter **locktimeout**, indem Sie als Wert eine für Ihre Datenbankumgebung optimal geeignete Anzahl von Sekunden angeben.
   - Vermeiden Sie nach Möglichkeit lange Transaktionen sowie Cursor, die mit WITH HOLD definiert sind. Je länger Sperren aktiviert bleiben, desto höher ist die Wahrscheinlichkeit, dass sie Konflikte mit anderen Anwendungen verursachen.
   - Es empfiehlt sich, für die folgenden Aktionen so bald wie möglich ein Commit durchzuführen:
     – Schreibaktionen wie Löschen, Einfügen und Aktualisieren
     – DDL-Anweisungen wie ALTER, CREATE und DROP
     – Befehle **BIND** und **REBIND**
   - Vermeiden Sie das Abrufen von Ergebnismengen, die größer sind als erforderlich, insbesondere unter der Isolationsstufe 'Wiederholbares Lesen' (RR). Je mehr Zeilen gelesen werden, desto mehr Sperren werden aktiviert und desto größer ist die Wahrscheinlichkeit, auf eine Zeile zu treffen, die bereits von einer anderen Anwendung gesperrt ist. In der Praxis bedeutet dies in vielen Fällen, die Zeilenauswahlbedingungen in eine WHERE-Klausel der SELECT-Anweisung zu verschieben, anstatt eine größere Anzahl Zeilen in die Anwendung abzurufen und dort zu filtern. Beispiel:

     ```
     exec sql declare curs for
       select c1,c2 from t
       where c1 not null;
     exec sql open curs;
     do {
       exec sql fetch curs
         into :c1, :c2;
     } while( P(c1) != someVar );

     ==>

     exec sql declare curs for
       select c1,c2 from t
       where c1 not null
       and myUdfP(c1) = :someVar;
     exec sql open curs;
     exec sql fetch curs
         into :c1, :c2;
     ```

   - Vermeiden Sie die Verwendung höherer Isolationsstufen als erforderlich. Die Isolationsstufe 'Wiederholbares Lesen' (RR) ist möglicherweise erforderlich, um die Integrität von Ergebnismengen in Ihrer Anwendung sicherzustellen. Allerdings verursacht sie zusätzlichen Aufwand durch aktivierte Sperren und potenzielle Sperrenkonflikte.

- Möglicherweise empfiehlt es sich, das Sperrverhalten über die Registrierdatenbankvariablen **DB2_EVALUNCOMMITTED**, **DB2_SKIPDELETED** und **DB2_SKIPINSERTED** zu ändern, wenn dies für die Geschäftslogik der Anwendung sinnvoll erscheint. Durch diese Registrierdatenbankvariablen kann der DB2-Datenbankmanager das Aktivieren von Sperren unter bestimmten Umständen verzögern oder vermeiden, sodass sich die Konfliktwahrscheinlichkeit verringert und der Durchsatz potenziell erhöht.

Nächste Schritte

Führen Sie die Anwendung(en) erneut aus, um sicherzustellen, dass der Sperrenfehler beseitigt ist. Überprüfen Sie dazu das Protokoll mit Benachrichtigungen für die Systemverwaltung auf Einträge zu Sperren oder überprüfen Sie die Messdaten für Wartestatus und Wartezeit für Sperren für die entsprechende Workload, Verbindung, Serviceunterklasse und UOW sowie für einen unterschiedlichen Auslastungsgrad.

Diagnostizieren von Problemen durch Sperreneskalation

Eine Sperreneskalation tritt auf, wenn mehrere Sperren auf Zeilenebene zu einer einzigen speichersparenden Tabellensperre eskaliert werden, um den den Sperren zugeordneten Speicher (Sperrbereich) zu verringern. Diese automatisierte Situation, die dem Einsparen von mit Sperren belegter Hauptspeicherkapazität dienen soll, kann gemeinsame Zugriffe auf ein nicht mehr akzeptables Niveau reduzieren. Ein Problem durch Sperreneskalation liegt in der Regel vor, wenn eine erhöhte Anzahl von Wartestatus von Sperren auftritt und das Protokoll mit Benachrichtigungen für die Systemverwaltung Einträge enthält, die auf Sperreneskalationen hinweisen.

Vorbereitende Schritte

Im Allgemeinen setzt die objektive Einstufung eines abnormalen Systemverhaltens (z. B. bei Verarbeitungsverzögerungen und Leistungsbeeinträchtigungen) voraus, dass als Vergleichswert Informationen vorliegen, die das typische Verhalten des Systems beschreiben. Auf diese Weise kann im Bedarfsfall ein Vergleich zwischen dem normalen Verhalten und dem von Ihnen beobachteten Verhalten, das Sie für potenziell abnormal halten, gezogen werden. Das Erfassen von Vergleichsdaten durch das Einplanen regelmäßig auszuführender Tasks für die Überwachung des Systembetriebs ist von elementarer Bedeutung für die Fehlerbehebung. Nähere Informationen zum Einrichten einer Datenbasis für einen normalen Systembetrieb, die als Vergleichsmaßstab dienen kann, finden Sie in „Leistungsbezogene Betriebsüberwachung” auf Seite 13.

Informationen zu diesem Vorgang

Diagnose

Die Eskalation von mehreren Sperren auf Zeilenebene zu einer einzigen Sperre auf Tabellenebene kann aus folgenden Gründen auftreten:

- Die gesamte Speicherkapazität, die von Sperren auf Zeilenebene belegt wird, die sich auf eine Tabelle beziehen, liegt über dem Prozentsatz der gesamten für das Speichern von Sperren zugeordneten Speicherkapazität.
- Der Speicher für die Sperrenliste reicht nicht aus. Die Anwendung, die dazu geführt hat, dass der Speicher für die Sperrenliste nicht mehr ausreicht, setzt die zugehörigen Sperren über den Sperreneskalationsprozess durch, auch wenn es sich bei dieser Anwendung nicht um die Anwendung handelt, die die höchste Anzahl an Sperren hält.

Der Schwellenwert für den Prozentsatz von für das Speichern von Sperren reserviertem Gesamtspeicher, der überschritten werden muss, damit eine Sperreneskalation eintritt, wird über den Datenbankkonfigurationsparameter **maxlocks** definiert. Der für Sperren reservierte Speicher wird über den Datenbankkonfigurationsparameter **locklist** definiert. In einer sinnvoll konfigurierten Datenbank kommt eine Sperreneskalation nur selten vor. Wenn gemeinsame Zugriffe durch die Sperreneskalation übermäßig eingeschränkt werden, sollten Sie eine Fehleranalyse vornehmen und anschließend entsprechende Maßnahmen ergreifen.

Sperreneskalationen sind in Bezug auf die Hauptspeicherkapazität unproblematischer, wenn ein Speichermanager für automatische Leistungsoptimierung (STMM, Self-Tuning Memory Manager) den Speicher für Sperren verwaltet, der ansonsten nur über den Datenbankkonfigurationsparameter **locklist** zugeordnet wird. STMM passt die Hauptspeicherkapazität für Sperren automatisch an, wenn die freie Hauptspeicherkapazität nicht mehr ausreichen sollte.

Typische Anzeichen

Folgende Anzeichen weisen auf Sperreneskalationen hin:

- Nachrichteneinträge zu Sperreneskalationen im Protokoll mit Benachrichtigungen für die Systemverwaltung

Zu überwachende Elemente

Aufgrund der relativ flüchtigen Natur der Sperrereignisse haben Daten zu Sperrereignissen den größten Wert, wenn sie in regelmäßigen Abständen über einen gewissen Zeitraum erfasst werden, sodass sich ein aussagekräftigeres Bild ergibt.

Überprüfen Sie das folgende Monitorelement auf Werte, die darauf hinweisen könnten, dass Sperreneskalationen zu einer Verlangsamung der SQL-Abfrageausführung beitragen:

- **lock_escals**

Wenn bei Ihnen an dieser Stelle beschriebene Anzeichen auftreten, liegt in aller Regel ein Problem mit Sperreneskalationen vor. Nähere Informationen dazu, wie Sie dieses Problem lösen können, können Sie über den Link im Abschnitt „Weitere Schritte" abrufen.

Nächste Schritte

Wenn sich herausgestellt hat, dass das bei Ihnen aufgetretene Problem vermutlich durch Sperreneskalationen ausgelöst wird, können Sie dieses Problem mit den im folgenden Abschnitt beschriebenen Schritten lösen: „Beheben von Sperreneskalationsproblemen".

Beheben von Sperreneskalationsproblemen

Wenn ein Problem bei der Sperreneskalation vorliegt, müssen Sie zunächst versuchen, das Problem zu lösen, das durch die automatische Sperreneskalation von der Zeilenebene auf die Tabellenebene durch den Datenbankmanager entsteht. Die Richtlinien in diesem Abschnitt helfen Ihnen beim Beheben des bei Ihnen aufgetretenen Sperreneskalationsproblems und tragen dazu bei, dass ähnliche Störungen in Zukunft vermieden werden.

Vorbereitende Schritte

Stellen Sie sicher, dass es sich bei dem aufgetretenen Problem um ein Sperreneskalationsproblem handelt, indem Sie die für die Diagnose von Sperrenfehlern erforderlichen Schritte ausführen (siehe „Diagnostizieren und Beheben von Sperrenfehlern" auf Seite 593).

Informationen zu diesem Vorgang

Die Richtlinien in diesem Abschnitt helfen Ihnen beim Beheben des bei Ihnen aufgetretenen Sperreneskalationsproblems und tragen dazu bei, dass ähnliche Störungen in Zukunft vermieden werden.

Ziel ist es, die Anzahl von Sperreneskalationen zu minimieren oder, wenn möglich, Sperreneskalationen vollständig zu verhindern. Sperreneskalationen können durch eine entsprechende Kombination aus Anwendungsdesign und Datenbankkonfiguration reduziert oder verhindert werden. Sperreneskalationen können gemeinsame Zugriffe reduzieren und möglicherweise auch zu Überschreitungen der Sperrzeit führen und sollten unbedingt vermieden werden. Hinweise zum Ermitteln und Beheben von Sperreneskalationen liefern das Monitorelement **lock_escals** sowie das Protokoll mit Benachrichtigungen für die Systemverwaltung.

Stellen Sie zuerst sicher, dass Informationen zur Sperreneskalation aufgezeichnet werden. Setzen Sie den Datenbankkonfigurationsparameter **mon_lck_msg_lvl** auf den Wert 1. Dies ist die Standardeinstellung. Wenn ein Sperreneskalationsereignis auftritt, werden Informationen in Bezug auf die Sperre, die Auslastung (Workload), die Anwendung, die Tabelle und die SQLCODE-Fehlercodes aufgezeichnet. Die Abfrage wird auch protokolliert, wenn es sich um eine momentan ausgeführte dynamische SQL-Anweisung handelt.

Vorgehensweise

Führen Sie die folgenden Schritte aus, um die Ursache für das inakzeptable Sperreneskalationsproblem zu ermitteln und Abhilfe zu schaffen:

1. Entnehmen Sie dem Protokoll mit Benachrichtigungen für die Systemverwaltung Informationen zu allen Tabellen, deren Sperren eskaliert wurden, sowie zu den beteiligten Anwendungen. Diese Protokolldatei enthält die folgenden Informationen:
   - Die Anzahl der momentan aktiven Sperren.
   - Die Anzahl der bis zum Abschluss einer Sperreneskalation erforderlichen Sperren.
   - Die Tabellenkennung und der Tabellenname 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. In der Regel wird eine S- oder X-Sperre aktiviert.
   - Der interne Rückkehrcode, der mit der Aktivierung der neuen Sperre auf Tabellenebene verbunden ist.
2. Informieren Sie sich anhand des Protokolls mit Benachrichtigungen für die Systemverwaltung über die an den Sperreneskalationen beteiligten Anwendungen und richten Sie die Maßnahmen zum Beheben der Eskalationsprobleme entsprechend aus. Ziehen Sie die folgenden Optionen in Betracht:
   - Überprüfen Sie die Datenbankkonfigurationsparameter **maxlocks** und **locklist** und passen Sie einen oder beide Parameter gegebenenfalls an. In einem partitionierten Datenbanksystem nehmen Sie diese Änderung in allen

Datenbankpartitionen vor. Der Wert für den Konfigurationsparameter **locklist** ist möglicherweise zu niedrig für die aktuelle Auslastung. Treten bei mehreren Anwendungen Sperreneskalationen auf, kann dies darauf hinweisen, dass die Sperrenliste vergrößert werden sollte. Eine Zunahme der Auslastung oder das Hinzufügen neuer Anwendungen kann dazu geführt haben, dass die Sperrenliste nicht mehr ausreicht. Treten nur bei einer einzigen Anwendung Sperreneskalationen auf, können Sie das Problem möglicherweise durch ein Anpassen des Werts für den Konfigurationsparameter **maxlocks** beheben. Möglicherweise empfiehlt es sich jedoch auch, den Parameter **locklist** gleichzeitig mit dem Parameter **maxlocks** zu erhöhen. Wenn einer Anwendung ein größerer Anteil der Sperrliste zugestanden wird, können bei den übrigen Anwendungen Eskalationen auftreten, weil die verbleibenden Sperren in der Sperrenliste für diese Anwendungen nicht mehr ausreichen.

- Möglicherweise ist auch eine Änderung der Isolationsstufe sinnvoll, unter der die Anwendung und die SQL-Anweisungen ausgeführt werden, z. B. RR (Repeatable Read, wiederholtes Lesen), RS (Read Stability, Lesestabilität), CS (Cursor Stability, Cursorstabilität) oder UR (Uncommitted Read, nicht festgeschriebener Lesevorgang). Bei den Isolationsstufen RR und RS treten vermehrt Eskalationen auf, da Sperren gehalten werden, bis ein Commit ausgegeben wird. Da dies bei den Isolationsstufen CS und UR nicht der Fall ist, sind Sperreneskalationen bei diesen Isolationsstufen weniger wahrscheinlich. Wählen Sie die niedrigste Isolationsstufe, die von der Anwendung toleriert wird.
- Erhöhen Sie die Häufigkeit der Commits in der Anwendung, wenn Geschäftsanforderungen und Anwendungsdesign dies zulassen. Durch häufigere Commits wird die Anzahl von Sperren verringert, die zu einem beliebigen Zeitpunkt aktiv sind. Dadurch wird vermieden, dass die Anwendung den Wert für **maxlocks** erreicht, der eine Sperreneskalation auslöst. Dies trägt auch dazu bei, dass die Sperrenliste für alle Anwendungen ausreicht.
- Sie können die Anwendung so ändern, dass Tabellensperren über die Anweisung LOCK TABLE angefordert werden. Dies empfiehlt sich für Tabellen, bei denen der gleichzeitige Zugriff durch viele Anwendungen und Benutzer für den Systembetrieb nicht kritisch ist; z. B., wenn die Anwendung eine permanente Arbeitstabelle (keine deklarierte globale temporäre Tabelle) verwendet, die für die betreffende Anwendungsinstanz eindeutig benannt ist. Das Anfordern von Tabellensperren ist in diesem Fall sinnvoll, da es die Anzahl der von der Anwendung gehaltenen Sperren verringert und die Leistung erhöht, da für die Zeilen, auf die in der Arbeitstabelle zugegriffen werden, keine Zeilensperren angefordert und freigegeben werden müssen.

  Wenn die Anwendung nicht über Arbeitstabellen verfügt und Sie die Werte für die Konfigurationsparameter **locklist** oder **maxlocks** nicht erhöhen können, können Sie die Anwendung so ändern, dass Tabellensperren angefordert werden. Wählen Sie die zu sperrende(n) Tabelle(n) jedoch mit Bedacht aus. Vermeiden Sie Tabellen, auf die viele Anwendungen und Benutzer zugreifen, da ein Sperren dieser Tabellen Probleme beim gemeinsamen Zugriff verursacht, der schlimmstenfalls dazu führen kann, dass Anwendungen die Sperrzeit überschreiten.

Nächste Schritte

Führen Sie die Anwendung(en) erneut aus, um sicherzustellen, dass das Sperrenproblem beseitigt ist. Überprüfen Sie dazu das Protokoll mit Benachrichtigungen für die Systemverwaltung auf Einträge, die sich auf Sperren beziehen.

Fehlerbehebung für SQL-Leistung

Einzelne SQL-Abfragen werden fehlerfrei ausgeführt, die Leistung nimmt jedoch ab, wenn mehrere Abfragen ausgeführt werden

Zur Verbesserung der Leistung von SQL-Abfragen, die bei Ausführung in Gruppen mit einem hohen Zeitaufwand verbunden sind, gibt es eine Reihe von Maßnahmen, die Sie ergreifen können.

Eine Abfrage zeigt möglicherweise eine zufriedenstellende Leistung, wenn sie separat in einer Testumgebung ausgeführt wird. Bei der gemeinsamen Ausführung mit anderen Abfragen in einer Produktionsumgebung wird jedoch viel Zeit benötigt.

Symptome

Eine SQL-Abfrage wird alleine verhältnismäßig gut ausgeführt, bei der Ausführung auf einem Produktionssystem und gleichzeitiger Ausführung anderer Abfragen ist die Leistung jedoch nicht zufriedenstellend. Dieses Problem kann jeden Tag zu einer bestimmten Uhrzeit oder auch zu unterschiedlichen Zeiten auftreten.

Bei der Überprüfung der SQL-Anweisung lassen sich keine offensichtlichen Ursachen für das vorübergehende Leistungsproblem feststellen.

Ursachen

Häufig sind solche Leistungsprobleme auf Probleme bei der Speicherzuordnung des Sortierspeichers zurückzuführen.

Der Sortierspeicher wird von einer Reihe von Abfragen (eine Liste befindet sich in der Dokumentation zu **sortheap**) verwendet, sodass es wichtig ist, die Einstellungen der folgenden Konfigurationsparameter zu überprüfen:

- Die Sortierspeichergröße (**sortheap**), die die Menge des Speichers angibt, die für die einzelnen Sortierungen zugeordnet wird.
- Der Schwellenwert für den Sortierspeicher (**sheapthres**), der die Gesamtgröße des Speichers angibt, der für alle Sortiervorgänge verfügbar ist, die in der Instanz ausgeführt werden.

Wenn die Anzahl der Abfragen, die gleichzeitig ausgeführt werden, steigt, dann kann das System den Schwellenwert für **sheapthres** erreichen, wodurch es zu Leistungsproblemen kommen kann.

Fehlerbehebung

In vielen Fällen kann der Manager für Speicher mit automatischer Leistungsoptimierung von DB2 (STMM = Self-tuning Memory Manager) die Zuordnung des Sortierspeichers anpassen, um die effiziente Ausführung von Abfragen zu gewährleisten. Weitere Einzelheiten finden Sie in der Dokumentation zum Datenbankkonfigurationsparameter **self_tuning_mem**. Wenn auf dem System STMM nicht ausgeführt wird, dann prüfen Sie die folgenden Optionen:

Ändern des SQL-Codes

Es ist möglich, die Menge des verwendeten Sortierspeichers zu reduzieren, indem Sie den entsprechenden SQL-Code ändern:

1. Wurden alle erforderlichen Indizes für die Abfrage erstellt? Wenn eine Spalte z. B. eindeutige Werte enthält, dann kann die Erstellung eines eindeutigen Index den erforderlichen Sortierspeicher reduzieren.
2. Ermitteln Sie, ob eine oder mehrere der SQL-Anweisungen geändert werden können, um die Menge des belegten Sortierspeichers zu reduzieren. Sind alle ORDER BY-Klauseln in der Abfrage erforderlich? Kann anstelle von DISTINCT in einer Unterabfrage auch GROUP BY verwendet werden?
3. Verwenden Sie Spaltengruppenstatistiken oder Statistiksichten, die es dem DB2-Abfrageoptimierungsprogramm in bestimmten Fällen ermöglichen können, einen Zugriffsplan zu ermitteln, der eine geringere Menge an Sortierspeicher belegt.

Erhöhen der Größe für sheapthres

Wenn keine der vorherigen Änderungen am SQL-Code möglich ist, oder wenn die Leistung durch keine dieser Änderungen verbessert werden kann, dann sollten Sie als Nächstes die Möglichkeit der Erhöhung des Wertes für den Parameter **sheapthres** prüfen. Dieser Parameter definiert die Gesamtsumme des Speicherplatzes, der von allen privaten Sortierungen zu einem beliebigen Zeitpunkt belegt wird. Wenn die Gesamtnutzung an privatem Sortierspeicherbereich für eine Instanz diesen Grenzwert erreicht, wird der für weitere eingehende private Sortieranforderungen zugeordnete Speicherbereich beträchtlich verkleinert. Das Monitorelement **post_threshold_sorts** wird verwendet, um Sortieranforderungen zu verfolgen, die mit einer reduzierten Speichermenge ausgeführt werden müssen, nachdem der Schwellenwert von **sheapthres** überschritten wurde.

Detaillierte Informationen zur Erhöhung der Speichermenge, die für alle Sortierungen zugeordnet wurde, und zur Lastverteilung für die SQL-Sortierleistung in Bezug auf die Speicherbelegung finden Sie in der Dokumentation zu **sheapthres**. Sie müssen dabei die Speichermenge berücksichtigen, die für die einzelnen Sortierungen (**sortheap**) reserviert ist. Außerdem müssen Sie die typische Anzahl von Sortierungen berücksichtigen, die auf dem System ausgeführt werden, wenn die Leistung beginnt, sich zu verschlechtern. Um festzustellen, ob eine Erhöhung des Wertes für **sheapthres** wirkungsvoll ist, sollte nach der Änderung eine Reduzierung der Anzahl der in **post_threshold_sorts** definierten Sortierungen zu verzeichnen sein.

Reduzieren der Größe für sortheap

In Situationen, in denen Sie den Gesamtwert für den Sortierspeicher (**sheapthres**) nicht erhöhen können und auf Ihrem System zahlreiche Sortierungen mit geringem Umfang ausgeführt werden, kann geprüft werden, ob die Reduzierung des Wertes für den Parameter für die individuelle Sortierspeichergröße (**sortheap**) Vorteile bietet. Dadurch wird weniger Speicherplatz für die einzelnen Sortierungen zugeordnet, die Anzahl der Sortierungen, die gleichzeitig ausgeführt werden können, wird jedoch erhöht.

Sie sollten die Nutzung dieser Option nur in Erwägung ziehen, wenn Sie überprüfen können, dass zahlreiche Sortierungen mit geringem Umfang vorhanden sind, wobei jede einzelne klein genug ist, um in einem kleineren Sortierspeicher ausgeführt werden zu können. Wenn Sie für **sortheap** einen zu niedrigen Wert angeben, dann kommt es zu einem Überlauf der Sortierungen auf die Platte, wodurch die Leistung aufgrund der Platten-E/A beeinträchtigt wird, die erheblich mehr Zeit benötigt als die speicherinternen Operationen. Sie können den Wert für `NumSpills` verwenden, der zusammen mit den Daten des Parameters **-sort** zurückgegeben wird,

wenn Sie den Befehl **db2pd** ausführen, um festzustellen, ob die individuelle Sortierspeichergröße (**sortheap**) für die Abfragen auf einen zu niedrigen Wert eingestellt wurde.

Leistung - Übersicht

Der Begriff *Leistung* bezieht sich auf die Art und Weise, wie sich ein Computersystem in Bezug auf eine bestimmte Auslastung (Workload) verhält. Die Leistung wird an der Antwortzeit, am Durchsatz und an der Ressourcennutzung gemessen.

Die Leistung wird außerdem 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 mit dem Ziel, das Kosten-Nutzen-Verhältnis zu verbessern. Dabei können die folgenden speziellen Optimierungsziele verfolgt werden:

- Verarbeiten größerer oder anspruchsvollerer Auslastungen ohne steigende Verarbeitungskosten
- Erreichen schnellerer Systemantwortzeiten bzw. eines höheren Durchsatzes ohne steigende Verarbeitungskosten
- Reduzieren von Verarbeitungskosten ohne negative Auswirkungen für Benutzer

Einige Vorteile der Leistungsoptimierung, wie zum Beispiel eine effizientere Nutzung von Ressourcen und die Möglichkeit, dem System weitere Benutzer hinzuzufügen, zeigen sich sehr praktisch. Andere Vorteile, wie größere Zufriedenheit seitens der Benutzer aufgrund schnellerer Antwortzeiten, sind weniger fassbar.

Richtlinien zur Leistungsoptimierung

Beachten Sie bei der Entwicklung eines allgemeinen Ansatzes zur Leistungsoptimierung die folgenden Richtlinien.

- **Behalten Sie das Gesetz der abnehmenden Ertragsgewinne im Hinterkopf:** Die größten Leistungsvorteile werden in der Regel durch die ersten Maßnahmen erzielt.
- **Optimieren Sie nicht nur des Optimierens wegen:** Optimieren Sie, um erkannten Engpässen abzuhelfen. Eine Optimierung von Ressourcen, die nicht die Hauptursache für Leistungsprobleme darstellen, kann die nachfolgende Optimierungsarbeit tatsächlich erschweren.
- **Betrachten Sie das System als Ganzes:** Ein Parameter bzw. eine Ressource lässt sich nicht isoliert optimieren. Bevor Sie eine Anpassung vornehmen, überlegen Sie, wie sich diese Änderung auf das System als Ganzes auswirken wird. Die Leistungsoptimierung erfordert Kompromisslösungen zwischen verschiedenen Systemressourcen. Zum Beispiel könnten Sie die Werte für Pufferpoolgrößen erhöhen, um eine bessere Ein-/Ausgabeleistung zu erzielen, jedoch erfordern größere Pufferpools mehr Speicher und können daher andere Aspekte der Leistung wiederum beeinträchtigen.
- **Ändern Sie jeweils nur einen Parameter gleichzeitig:** Ändern Sie immer nur einen Faktor gleichzeitig. Selbst wenn Sie sich sicher sind, dass alle Änderungen vorteilhaft sind, haben Sie hinterher keine Möglichkeit, den Beitrag jeder einzelnen Änderung zu bewerten.

- **Führen Sie Messungen und Konfigurationen nach Ebenen durch:** Optimieren Sie jeweils nur eine Ebene Ihres Systems gleichzeitig. Systemebenen sind zum Beispiel:
  - Hardware
  - Betriebssystem
  - Anwendungsserver und -requester
  - Datenbankmanager
  - SQL- und XQuery-Anweisungen
  - Anwendungsprogramme
- **Prüfen Sie auf Hardware- und Softwareprobleme:** Einige Leistungsprobleme können 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, bevor Sie eine Hardwarewartung oder eine Softwarekorrektur durchgeführt haben.
- **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.
- **Implementieren Sie Rücksetzprozeduren, bevor Sie mit der Optimierung beginnen:** Wenn Optimierungsmaßnahmen zu einer unerwarteten Leistungsverschlechterung führen, sollten die vorgenommenen Änderungen rückgängig gemacht werden, bevor eine alternative Lösung versucht wird. Speichern Sie Ihre ursprünglichen Einstellungen, sodass Sie Änderungen, die Sie nicht beibehalten möchten, leicht rückgängig machen können.

Entwickeln eines Prozesses zur Leistungsverbesserung

Ein Leistungsverbesserungsprozess ist ein iteratives Verfahren zur Überwachung und Optimierung von Leistungsbereichen. Abhängig von den Ergebnissen dieser Leistungsüberwachung passen Sie 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 Ihnen bekannten Datenzugriffsmustern aus. Verschiedene Arten von Anwendungen haben unterschiedliche Leistungsanforderungen.

Jeder Leistungsverbesserungsprozess enthält die folgenden grundlegenden Schritte:

1. Definieren Sie die 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 Überwachungsergebnisse fortlaufend, um zu ermitteln, welche Ressourcen optimiert werden müssen.
5. Nehmen Sie jeweils nur eine Anpassung vor.

Wenn Sie an einem bestimmten Punkt keine weitere Verbesserung der Leistung durch Optimieren des Datenbankservers und der Anwendungen erzielen können,

ist möglicherweise der Zeitpunkt gekommen, die Hardware aufzurüsten.

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 auseinander setzen, indem Sie Ihren Benutzern zuhören. Beginnen Sie, indem Sie einige einfache Fragen stellen, wie zum Beispiel die folgenden:

- Was meinen Sie mit „langsamer Reaktion"? Heißt dies, um zehn Prozent langsamer, als Sie erwarten, oder um das Zigfache 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 das gleiche Problem hat, sind diese Benutzer mit demselben lokalen Netz (LAN) verbunden?
- Scheint das Problem mit einem bestimmten Typ von Transaktions- oder Anwendungsprogramm zusammenzuhängen?
- Erkennen Sie ein Muster im Auftreten des Problems? Zum Beispiel: Tritt dieses Problem zu einer bestimmten Tageszeit auf oder ist es permanent bemerkbar?

Grenzen der Leistungsoptimierung

Die durch die Leistungsoptimierung realisierbaren Vorteile sind begrenzt. Wenn Sie überlegen, wie viel Zeit und Geld in die Verbesserung der Systemleistung investiert werden sollte, müssen Sie unbedingt eine Beurteilung des möglichen Grads vornehmen, bis zu dem eine zusätzliche Investition von Zeit und Geld den Benutzern des Systems hilft.

Eine Optimierung kann die Leistung häufig verbessern, wenn das System Probleme mit der Antwortzeit oder dem Durchsatz hat. Es gibt allerdings einen Punkt, ab dem eine weitere Optimierung keine Hilfe mehr ist. An diesem Punkt müssen Sie Ihre Ziele und Erwartungen überprüfen. Wenn Sie weitere wesentliche Leistungsverbesserungen erreichen wollen, müssen Sie vielleicht mehr Plattenspeicher, schnellere CPUs, zusätzliche CPUs, mehr Arbeitsspeicher, schnellere Kommunikationsverbindungen oder eine Kombination aus diesen Möglichkeiten hinzufügen.

Optimieren der Sortierleistung

Da Abfragen häufig sortierte oder gruppierte Ergebnisse erfordern, spielt eine geeignete Konfiguration des Sortierspeichers eine wichtige Rolle bei der Realisierung einer guten Abfrageleistung.

Sortieren ist in folgenden Fällen erforderlich:

- Es ist kein Index vorhanden, der eine angeforderte Reihenfolge (z. B. durch eine SELECT-Anweisung mit der Klausel ORDER BY) liefert.
- 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:

- Einstellungen für die folgenden Konfigurationsparameter:
  - Die Sortierspeichergröße (**sortheap**), welche die Kapazität des Speichers angibt, der für jede Sortierung verwendet wird
  - Der Schwellenwert für Sortierspeicher (**sheapthres**) und der Schwellenwert für Sortierspeicher für gemeinsame Sortiervorgänge (**sheapthres_shr**), welche die Gesamtgröße des Speichers steuern, der für das Sortieren in der Instanz 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 die Notwendigkeit von Sortierungen nicht minimiert.
- Paralleles Sortieren, das die Sortierleistung erhöht, aber nur erfolgen kann, wenn die Anweisung partitionsinterne Parallelität verwendet.
- Ob die Sortierung einen *Überlauf* verursacht hat oder nicht. Wenn die sortierten Daten nicht vollständig in den Sortierspeicher passen, bei dem es sich um einen Speicherblock handelt, der jedes Mal zugeordnet wird, wenn eine Sortierung ausgeführt wird, laufen die Daten in eine temporäre Tabellen über, deren Eigner die Datenbank ist.
- Ob die Ergebnisse der Sortierung *über eine Pipe* geleitet werden oder nicht. Wenn sortierte Daten direkt zurückgegeben werden können, ohne dass eine temporäre Tabelle zum Speichern der sortierten Liste erforderlich ist, handelt es sich um einen Sortiervorgang mit Piping (d. h. die Daten werden über eine Pipe zurückgegeben).

  Bei einer Sortierung mit Piping wird der Sortierspeicher nicht freigegeben, bevor die Anwendung den Cursor schließt, der dieser Sortierung zugeordnet ist. Eine Sortierung mit Piping kann weiter Speicher belegen, bis der Cursor geschlossen wird.

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 Sortierspeichers 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 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 Abfragen mit Sortieroperationen zu ermitteln.

3. Verwenden Sie diese Anweisungen als Eingabe für den Designadvisor, der Indizes ermittelt und auch erstellen kann, um den Sortierbedarf zu reduzieren.

Sie können den automatischen Speicheroptimierungsmanager (STMM, Self-Tuning Memory Manager) verwenden, der automatisch und dynamisch für die Sortierung erforderliche Speicherressourcen zuordnet und wieder freigibt. Gehen Sie wie folgt vor, um diese Funktion zu verwenden:

- Aktivieren Sie die automatische Speicheroptimierung für die Datenbank, indem Sie den Konfigurationsparameter **self_tuning_mem** auf den Wert ON setzen.
- Setzen Sie die Konfigurationsparameter **sortheap** und **sheapthres_shr** auf den Wert AUTOMATIC.
- Setzen Sie den Konfigurationsparameter **sheapthres** auf den Wert 0.

Sie können auch den Datenbanksystemmonitor und Vergleichstestverfahren verwenden, um geeignete Werte für die Konfigurationsparameter **sortheap**, **sheapthres_shr** und **sheapthres** zu ermitteln. Gehen Sie für jeden Datenbankmanager und jede Datenbank folgendermaßen vor:

1. Erstellen Sie eine repräsentative Auslastung und führen Sie sie aus.
2. Erfassen Sie für jede betroffene Datenbank Durchschnittswerte für die folgenden Leistungsvariablen über den Auslastungszeitraum der Vergleichstests:
   - Gesamter verwendeter Sortierspeicher (der Wert des Monitorelements **sort_heap_allocated**)
   - Aktive Sortiervorgänge und aktive Hash-Joins (die Werte der Monitorelemente **active_sorts** und **active_hash_joins**)
3. Setzen Sie den Parameter **sortheap** auf den Durchschnittswert für den *gesamten verwendeten Sortierspeicher* für jede Datenbank.

   Anmerkung: 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 mithilfe von Vergleichstests diesen Wert optimieren.

IBM InfoSphere Optim Query Workload Tuner stellt Tools für die Leistungsverbesserung einzelner SQL-Anweisungen und die Leistung von Gruppen von SQL-Anweisungen bereit, die als Abfrageworkloads bezeichnet werden. Weitere Informationen zu diesem Produkt finden Sie auf der Seite mit der Produktübersicht unter http://www.ibm.com/software/data/optim/query-workload-tuner-db2-luw/index.html. In Version 3.1.1 oder späteren Versionen des Produkts können Sie auch den Designadvisor für Workloads verwenden, um zahlreiche Operationen auszuführen, die im Assistenten für den DB2-Designadvisor verfügbar waren. Weitere Informationen finden Sie in der Dokumentation des Designadvisors für Workloads unter http://publib.boulder.ibm.com/infocenter/dstudio/v3r1/topic/com.ibm.datatools.qrytune.workloadtunedb2luw.doc/topics/genrecsdsgn.html.

Minimieren der Leistungsbeeinträchtigung durch das Dienstprogramm RUNSTATS

Die Leistung des Dienstprogramms RUNSTATS kann auf verschiedene Arten verbessert werden.

Gehen Sie wie folgt vor, um die Leistungsbeeinträchtigung durch dieses Dienstprogramm zu minimieren:

- Begrenzen Sie die Spalten, für die Statistiken erfasst werden sollen, indem Sie die Klausel COLUMNS verwenden. Viele Spalten werden nie von Vergleichselementen in der Abfrageauslastung verwendet, sodass keine Statistiken für sie erforderlich sind.
- Begrenzen Sie die Spalten, für die Verteilungsstatistiken erfasst werden, wenn die Daten eher gleichmäßig verteilt sind. Die Erfassung von Verteilungsstatistiken erfordert mehr CPU-Zeit und Hauptspeicher als die Erfassung von Basisspaltenstatistiken. Allerdings ist es zur Ermittlung, ob die Werte einer Spalte gleichmäßig verteilt sind, erforderlich, bereits vorhandene Statistiken zur Hand zu haben oder die Daten abzufragen. Diese Methode geht zudem davon aus, dass die Daten gleichmäßig verteilt bleiben, wenn die Tabelle geändert wird.
- Begrenzen Sie die Anzahl der Seiten und Zeilen, die verarbeitet werden, indem Sie eine Stichprobenentnahme auf Seiten- oder Zeilenebene (durch Angabe der Klausel TABLESAMPLE SYSTEM bzw. BERNOULLI) verwenden. Beginnen Sie mit einer Stichprobenentnahme von 10 % auf Seitenebene, indem Sie `TABLESAMPLE SYSTEM(10)` angeben. Prüfen Sie die Genauigkeit der Statistiken und stellen Sie fest, ob sich die Systemleistung aufgrund von Änderungen im Zugriffsplan verschlechtert hat. Wenn sie sich verschlechtert hat, versuchen Sie eine Stichprobenentnahme von 10 % auf Zeilenebene, indem Sie `TABLESAMPLE BERNOULLI(10)` angeben. Wenn die Genauigkeit der Statistiken nicht ausreicht, erhöhen Sie den Stichprobenbetrag. Wenn Sie die Stichprobenentnahme auf Seiten- oder Zeilenebene mit dem Befehl **RUNSTATS** verwenden, verwenden Sie denselben Stichprobensatz für Tabellen, die durch Join verknüpft werden. Dies ist wichtig, um sicherzustellen, dass die Joinspaltenstatistiken den gleichen Genauigkeitsgrad besitzen.
- Erfassen Sie Indexstatistiken bei der Indexerstellung, indem Sie die Option COLLECT STATISTICS in der Anweisung CREATE INDEX verwenden. Diese Methode ist schneller als die Ausführung einer separaten RUNSTATS-Operation nach der Erstellung des Index. Sie stellt außerdem sicher, dass für den neuen Index sofort nach der Erstellung Statistiken generiert werden, sodass das Optimierungsprogramm den Aufwand bei Verwendung des Index genau abschätzen kann.
- Erfassen Sie Statistiken, wenn Sie den Befehl **LOAD** mit der Option REPLACE ausführen. Diese Methode ist schneller als die Ausführung einer separaten RUNSTATS-Operation nach Abschluss der Ladeoperation. Sie stellt außerdem sicher, dass für die Tabelle die aktuellsten Statistikdaten sofort nach dem Laden der Daten verfügbar sind, sodass das Optimierungsprogramm den Aufwand bei Verwendung der Tabelle genau abschätzen kann.

In einer Umgebung mit partitionierten Datenbanken erfasst das Dienstprogramm RUNSTATS Statistikdaten nur aus einer Datenbankpartition. Wenn der Befehl **RUNSTATS** für eine Datenbankpartition ausgeführt wird, in der sich die Tabelle befindet, werden die Statistiken dort erfasst. Ist dies nicht der Fall, werden die Statistiken in der ersten Datenbankpartition in der Datenbankpartitionsgruppe für die Tabelle erfasst. Stellen Sie zur Gewährleistung konsistenter Statistiken sicher, dass Statistiken für durch einen Join verknüpfte Tabellen in derselben Datenbankpartition erfasst werden.

Datenkomprimierung und Leistung

Die Datenkomprimierung kann verwendet werden, um das Volumen der Daten zu verringern, das von der Platte gelesen oder auf die Platte geschrieben werden muss, sodass sich der Ein-/Ausgabeaufwand reduziert.

Zwei Formen von Datenkomprimierung sind gegenwärtig verfügbar:

- Bei der *Wertkomprimierung* werden doppelte Einträge für einen Wert entfernt, sodass nur eine Kopie gespeichert und die Positionen aller Verweise auf den gespeicherten Wert aufgezeichnet werden.
- Bei der *Zeilenkomprimierung* werden sich wiederholende Muster, die sich über mehrere Spaltenwerte innerhalb einer Zeile erstrecken, durch kürzere Symbolzeichenfolgen ersetzt. Die Logik der Zeilenkompression durchsucht eine zu komprimierende Tabelle nach sich wiederholenden und doppelten Daten. Ein Komprimierungswörterverzeichnis (Compression Dictionary) enthält kurze, numerische Schlüssel für diese Daten. In einer komprimierten Zeile ersetzen diese Schlüssel die tatsächlichen Daten.

Vor DB2 Version 9.1 mussten Sie ein Komprimierungswörterverzeichnis durch die Ausführung einer Offlinetabellenreorganisation manuell erstellen. Ab Version 9.5 wird automatisch ein Komprimierungswörterverzeichnis für Tabellen erstellt, für die die Datenkomprimierung aktiviert wird.

Mit diesem Release wird die autonome Zeilenkomprimierung, die in Version 9.5 für permanente Tabellen eingeführt wurde, erweitert, sodass sie nun auch alle temporären Tabellen mit einschließt. Die Datenkomprimierung für temporäre Tabellen ist durch folgende Merkmale gekennzeichnet:

- Sie verringert den Bedarf an temporären Plattenspeicherplatz für große und komplexe Abfragen.
- Sie erhöht die Abfrageleistung.

Die Datenkomprimierung für temporäre Tabellen wird unter DB2 Storage Optimization Feature automatisch aktiviert.

Für jede temporäre Tabelle, für die die Zeilenkomprimierung infrage kommt, sind 2 - 3 MB an zusätzlichem Speicher zur Erstellung des Komprimierungswörterverzeichnis erforderlich. Dieser Speicher bleibt zugeordnet, bis das Komprimierungswörterverzeichnis erstellt ist.

Indexobjekte und Indizes für komprimierte temporäre Tabellen können ebenfalls komprimiert werden, um Speicherkosten zu reduzieren. Dies ist insbesondere für große OLTP-Umgebungen (OLTP - Onlinetransaktionsverarbeitung) und Data-Warehouse-Umgebungen nützlich, in denen gewöhnlich viele umfangreiche Indizes vorhanden sind. In beiden Fällen kann die Indexkomprimierung zu beträchtlichen Leistungsverbesserungen in Umgebungen, die von der E/A-Leistung abhängig sind, sowie zu geringen bis keinen Leistungseinbußen in Umgebungen, die von der CPU-Leistung abhängig sind, führen.

Ist die Komprimierung für eine Tabelle mit einer XML-Spalte aktiviert, werden die in dem XDA-Objekt gespeicherten XML-Daten ebenfalls komprimiert. Es wird ein separates Komprimierungswörterverzeichnis für die XML-Daten in dem XDA-Objekt gespeichert. Dies gilt auch für Tabellen, deren XML-Spalten in der aktuellen Version des DB2-Produkts hinzugefügt wurden. Die XDA-Komprimierung wird für

Tabellen, deren XML-Spalten mit Versionen vor dieser Version erstellt wurden, nicht unterstützt. Bei derartigen Tabellen wird lediglich das Datenobjekt komprimiert.

Fehlerbehebung für Optimierungsrichtlinien und -profile

Diagnoseunterstützung für Optimierungsrichtlinien (implementiert durch Optimierungsprofile) wird über EXPLAIN-Tabellen bereitgestellt.

Sie erhalten die Warnung SQL0437W mit dem Ursachencode 13, wenn das Optimierungsprogramm keine Optimierungsrichtlinie anwendet. Diagnoseinformationen mit Einzelheiten dazu, warum eine Optimierungsrichtlinie nicht angewendet wurde, wird zu den EXPLAIN-Tabellen hinzugefügt. Es sind zwei EXPLAIN-Tabellen für die Diagnoseausgabe des Optimierungsprogramms vorhanden:

- EXPLAIN_DIAGNOSTIC - Jeder Eintrag in dieser Tabelle stellt eine Diagnosenachricht dar, die sich auf die Optimierung einer bestimmten Anweisung bezieht. Jede Diagnosenachricht wird durch einen numerischen Code dargestellt.
- EXPLAIN_DIAGNOSTIC_DATA - Jeder Eintrag in dieser Tabelle besteht aus Diagnosedaten, die sich auf eine bestimmte Diagnosenachricht in der Tabelle EXPLAIN_DIAGNOSTIC beziehen.

Die zur Erstellung der diagnostischen EXPLAIN-Tabellen verwendeten DDLs sind nachfolgend in Abb. 39 auf Seite 619 dargestellt.

Die folgenden Schritte können Ihnen bei der Behebung von Fehlern helfen, die bei der Verwendung von Optimierungsrichtlinien auftreten können:

1. „Prüfen, ob Optimierungsrichtlinien verwendet wurden" in *Fehlerbehebung und Optimieren der Datenbankleistung*.
2. Lesen der vollständigen Fehlernachricht mithilfe der integrierten „Tabellenfunktion EXPLAIN_GET_MSGS" in *Administrative Routines and Views*.

Wenn Sie diese Schritte ausgeführt haben, aber die Ursache des Problems noch nicht identifizieren können, beginnen Sie mit der Erfassung von Diagnosedaten und wenden Sie sich gegebenenfalls an IBM Software Support.

```
CREATE TABLE EXPLAIN_DIAGNOSTIC
      ( EXPLAIN_REQUESTER VARCHAR(128) NOT NULL,
        EXPLAIN_TIME      TIMESTAMP    NOT NULL,
        SOURCE_NAME       VARCHAR(128) NOT NULL,
        SOURCE_SCHEMA     VARCHAR(128) NOT NULL,
        SOURCE_VERSION    VARCHAR(64)  NOT NULL,
        EXPLAIN_LEVEL     CHAR(1)      NOT NULL,
        STMTNO            INTEGER      NOT NULL,
        SECTNO            INTEGER      NOT NULL,
        DIAGNOSTIC_ID     INTEGER      NOT NULL,
        CODE              INTEGER      NOT NULL,
        PRIMARY KEY (EXPLAIN_REQUESTER,
                     EXPLAIN_TIME,
                     SOURCE_NAME,
                     SOURCE_SCHEMA,
                     SOURCE_VERSION,
                     EXPLAIN_LEVEL,
                     STMTNO,
                     SECTNO,
                     DIAGNOSTIC_ID),
        FOREIGN KEY (EXPLAIN_REQUESTER,
                     EXPLAIN_TIME,
                     SOURCE_NAME,
                     SOURCE_SCHEMA,
                     SOURCE_VERSION,
                     EXPLAIN_LEVEL,
                     STMTNO,
                     SECTNO)
        REFERENCES EXPLAIN_STATEMENT ON DELETE CASCADE);

   CREATE TABLE EXPLAIN_DIAGNOSTIC_DATA
      ( EXPLAIN_REQUESTER VARCHAR(128) NOT NULL,
        EXPLAIN_TIME      TIMESTAMP    NOT NULL,
        SOURCE_NAME       VARCHAR(128) NOT NULL,
        SOURCE_SCHEMA     VARCHAR(128) NOT NULL,
        SOURCE_VERSION    VARCHAR(64)  NOT NULL,
        EXPLAIN_LEVEL     CHAR(1)      NOT NULL,
        STMTNO            INTEGER      NOT NULL,
        SECTNO            INTEGER      NOT NULL,
        DIAGNOSTIC_ID     INTEGER      NOT NULL,
        ORDINAL           INTEGER      NOT NULL,
        TOKEN             VARCHAR(1000),
        TOKEN_LONG        BLOB(3M) NOT LOGGED,
        FOREIGN KEY (EXPLAIN_REQUESTER,
                     EXPLAIN_TIME,
                     SOURCE_NAME,
                     SOURCE_SCHEMA,
                     SOURCE_VERSION,
                     EXPLAIN_LEVEL,
                     STMTNO,
                     SECTNO,
                     DIAGNOSTIC_ID)
        REFERENCES EXPLAIN_DIAGNOSTIC ON DELETE CASCADE);
```

Anmerkung: Die Spalten EXPLAIN_REQUESTOR, EXPLAIN_TIME, SOURCE_NAME, SOURCE_SCHEMA, SOURCE_VERSION, EXPLAIN_LEVEL, STMTNO und SECTNO sind Teil beider Tabellen, um den Fremdschlüssel für die Tabelle EXPLAIN_STATEMENT und die Eltern-Kind-Beziehung zischen EXPLAIN_DIAGNOSTIC und EXPLAIN_DIAGNOSTIC_DATA zu bilden.

Abbildung 39. Zum Erstellen der diagnostischen EXPLAIN-Tabellen verwendete DDLs

Die DDL ist in der Datei EXPLAIN.DDL enthalten, die sich im Unterverzeichnis `misc` des Verzeichnisses `sqllib` befindet.

Fehlerbehebung in Umgebungen mit partitionierten Datenbanken

FCM-Fehler im Zusammenhang mit 127.0.0.2 (Linux und UNIX)

In einer Umgebung mit partitionierten Datenbanken können Fehler beim FCM (Fast Communications Manager) auftreten, wenn sich ein Eintrag für 127.0.0.2 in der Datei /etc/hosts befindet.

Symptome

Verschiedene Fehlernachrichten können - abhängig von den Umständen - auftreten. Es kann zum Beispiel der folgende Fehler auftreten, wenn Sie eine Datenbank erstellen: SQL1229N Die aktuelle Transaktion wurde rückgängig gemacht. SQLSTATE=40504

Ursachen

Der Fehler trat durch das Vorhandensein eines Eintrags für die IP-Adresse 127.0.0.2 in der Datei /etc/hosts auf, wobei 127.0.0.2 dem vollständig qualifizierten Hostnamen des Systems zugeordnet ist. Zum Beispiel:

```
127.0.0.2 ServerA.ibm.com ServerA
```

Dabei ist "ServerA.ibm.com" der vollständig qualifizierte Hostname.

Umgebung

Der Fehler ist beschränkt auf DB2 Enterprise Server Edition mit DB2 Database Partitioning Feature.

Problemlösung

Entfernen Sie den Eintrag aus der Datei /etc/hosts oder wandeln Sie ihn in einen Kommentar um. Beispiel:

```
# 127.0.0.2 ServerA.ibm.com ServerA
```

Erstellen einer Datenbankpartition in einem verschlüsselten Dateisystem (AIX)

AIX 6.1 unterstützt die Möglichkeit, ein JFS2-Dateisystem oder eine Gruppe von JFS2-Dateien zu verschlüsseln. Diese Funktion wird für Umgebungen mit partitionierten Datenbanken in DB2-Datenbankprodukten nicht unterstützt. Der Fehler SQL10004C tritt auf, wenn Sie versuchen, eine Umgebung mit partitionierten Datenbanken mit EFS (Encrypted File Systems, verschlüsselte Dateisysteme) unter AIX zu erstellen.

Symptome

Wenn Sie versuchen, eine Datenbank in einem verschlüsselten Dateisystem in einer Datenbankumgebung mit mehreren Partitionen zu erstellen, wird der folgende Fehler ausgegeben: SQL10004C: E/A-Fehler während des Zugriffs auf das Datenbankverzeichnis. SQLSTATE=58031

Ursachen

Es ist zum gegenwärtigen Zeitpunkt nicht möglich, eine Umgebung mit partitionierten Datenbanken mit EFS (Encrypted File Systems, verschlüsselte Dateisysteme)

unter AIX zu erstellen. Da in Umgebungen mit partitionierten Datenbanken rsh oder ssh verwendet wird, geht der Schlüsselspeicher in EFS verloren, und die Datenbankpartitionen können nicht auf die Datenbankdateien zugreifen, die im verschlüsselten Dateisystem gespeichert sind.

Problemdiagnose

Die DB2-Protokolldateien der Diagnoseprogramme (**db2diag**) enthalten die Fehlernachricht sowie den folgenden Text: `OSERR: ENOATTR (112) "No attribute found"` (Kein Attribut gefunden).

Problemlösung

Zur erfolgreichen Erstellung einer Datenbank in einer Umgebung mit partitionierten Datenbanken benötigen Sie ein Dateisystem, das für alle beteiligen Maschinen verfügbar ist und bei dem es sich nicht um ein verschlüsseltes Dateisystem handelt.

Fehlerbehebung beim Tabellenstatus während der Datenumverteilung

Stellen Sie vor dem Starten einer Umverteilungsoperation sicher, dass sich alle Tabellen in der Datenbankpartitionsgruppe im uneingeschränkten Zugriffsmodus befinden und einen normalen Status aufweisen.

Symptome

Wenn durch den Tabellenstatus die Neuverteilung fehlschlägt, erhalten Sie Fehlernachrichten, in denen Sie darüber informiert werden, dass die Datenbankpartitionsgruppe nicht neu verteilt werden kann oder dass die Operation nicht zulässig ist. Die Ausgabe der Nachrichten SQL02436N, SQL6056N und SQL0668N kann z. B. auf diesen Fehler hinweisen.

Anmerkung: Wenn in der Fehlernachricht ein Tabellenname aufgeführt ist, handelt es sich möglicherweise nicht um die einzige problematische Tabelle in der Datenbankpartitionsgruppe. Führen Sie die Fehlerbehebung für den Tabellenstatus aller Tabellen in der Datenbankpartitionsgruppe durch, um mehrere nicht erfolgreiche Umverteilungsversuche zu vermeiden.

Fehlerdiagnose

Benutzeraktion:

1. Stellen Sie fest, welche Tabellen sich in einem funktionsunfähigen Status befinden (SYSCAT.TABLES.STATUS='X').

   Führen Sie die folgende Abfrage aus:

   ```
   SELECT TABNAME
      FROM SYSCAT.TABLES AS TABLES, SYSCAT.TABLESPACES AS TABLESPACES
      WHERE TABLES.TBSPACE = TABLESPACES.TBSPACE AND TABLES.STATUS = 'X'
      AND TABLESPACES.DBPGNAME = 'IBMSTANDARDGRUPPE'
   ```

   Dabei ist *IBMSTANDARDGRUPPE* der Name Datenbankpartitionsgruppe.
2. Stellen Sie fest, welche Tabellen sich im Status 'Festlegen der Integrität anstehend' befinden (SYSCAT.TABLES.STATUS='C').

   Führen Sie die folgende Abfrage aus:

```
SELECT TABNAME
   FROM SYSCAT.TABLES AS TABLES, SYSCAT.TABLESPACES AS TABLESPACES
   WHERE TABLES.TBSPACE = TABLESPACES.TBSPACE AND TABLES.STATUS = 'C'
   AND TABLESPACES.DBPGNAME = 'IBMSTANDARDGRUPPE'
```

Dabei ist *IBMSTANDARDGRUPPE* der Name Datenbankpartitionsgruppe.

3. Stellen Sie fest, welche Tabellen sich in einem normalen Status, jedoch nicht im uneingeschränkten Zugriffsmodus befinden.

   Führen Sie die folgende Abfrage aus:

```
SELECT DISTINCT TRIM(T.OWNER) || \'.\' || TRIM(T.TABNAME) AS NAME, T.ACCESS_MODE,
A.LOAD_STATUS  FROM SYSCAT.TABLES T, SYSCAT.DBPARTITIONGROUPS N,
SYSIBMADM.ADMINTABINFO A
WHERE T.PMAP_ID = N.PMAP_ID
AND A.TABSCHEMA = T.OWNER
AND A.TABNAME = T.TABNAME
AND N.DBPGNAME = 'IBMSTANDARDGRUPPE'
AND (T.ACCESS_MODE <> 'F' OR A.LOAD_STATUS IS NOT NULL)
AND T.STATUS='N'
```

   Dabei ist *IBMSTANDARDGRUPPE* der Name Datenbankpartitionsgruppe. Falls die Ausführung dieser Abfrage viel Zeit in Anspruch nimmt, brechen Sie die Abfrage ab, geben Sie den Befehl **RUNSTATS** für alle an der Abfrage beteiligten Tabellen aus und starten dann die Abfrage erneut.

Fehlerbehebung

Benutzeraktion:

1. Führen Sie die erforderlichen Fehlerbehebungsmaßnahmen aus:
   a. Verwenden Sie für jede Tabelle, die sich in einem nicht funktionsfähigen Status befindet, den Befehl **LOAD QUERY**, um den genauen Tabellenstatus zu bestimmen.
      - Warten Sie bei Tabellen im Status 'Laden läuft', bis die Ladeoperation abgeschlossen ist. Sie können den Befehl **LOAD QUERY** verwenden, um den Fortschritt der Ladeoperation zu überwachen.
      - Bei Tabellen im Status 'Laden anstehend' starten Sie die zuvor fehlgeschlagene Ladeoperation neu oder beenden Sie sie, indem Sie den Befehl **LOAD** mit dem Befehlsparameter **RESTART** bzw. **TERMINATE** eingeben.
      - Bei Tabellen im Status 'Nur Lesezugriff' verwenden Sie den Befehl **LOAD QUERY**, um zu prüfen, ob der Ladeprozess für die Tabelle momentan läuft. Ist dies der Fall, warten Sie, bis das Ladedienstprogramm abgeschlossen ist, oder starten Sie die zuvor fehlgeschlagene Ladeoperation neu bzw. beenden Sie sie, falls erforderlich. Wenn momentan keine Ladeoperation läuft, geben Sie den Befehl SET INTEGRITY mit der Option IMMEDIATE CHECKED ein, um die Integritätsbedingungen im neu geladenen Teil der Tabelle zu prüfen.
      - Bei Tabellen im Status 'Reorganisation anstehend' führen Sie eine klassische Reorganisationsoperation aus, um den Zugriff auf die Tabelle zu ermöglichen.
      - Bei Tabellen im Status 'Für Laden nicht neu startbar' geben Sie den Befehl **LOAD TERMINATE** oder **LOAD REPLACE** ein.
      - Bei Tabellen im Status 'Nicht verfügbar' löschen Sie die Tabelle oder stellen Sie sie aus einem Backup wieder her.
   b. Bei Tabellen im Status 'Festlegen der Integrität anstehend' führen Sie die Anweisung SET INTEGRITY mit der Option IMMEDIATE CHECKED aus.

c. Bei Tabellen, die sich nicht im uneingeschränkten Zugriffsmodus befinden, führen Sie die Anweisung SET INTEGRITY mit der Option IMMEDIATE CHECKED für die abhängigen IMQTs (Immediate Materialized Query Tables, sofort gespeicherte Abfragetabellen) und die sofort gespeicherten Zwischenspeichertabellen aus.

Anmerkung: Sie können die Behebung des Problems auch auf einen späteren Zeitpunkt verschieben und die betreffenden Tabellen temporär aus der Umverteilungsoperation ausschließen, indem Sie den Parameter **EXCLUDE** im Befehl **REDISTRIBUTE DATABASE PARTITION GROUP** angeben. In diesem Fall wird die Umverteilungsoperation erfolgreich abgeschlossen, die Datenbankpartitionsgruppe ist jedoch nur teilweise umverteilt. Dies kann dazu führen, dass die umverteilten Tabellen in der Datenbankpartitionsgruppe eine andere Partitionierungszuordnung verwenden als die nicht umverteilten Tabellen. Darüber hinaus wird, wenn vor der Umverteilungsoperation eine Kollokation zwischen umverteilten und nicht umverteilten Tabellen vorhanden war, die Eigenschaft für die Kollokation zwischen diesen Tabellen temporär inaktiviert. Die Abfrageleistung ist möglicherweise nicht optimal, bis eine vollständige Umverteilung stattgefunden hat.

2. Wenn eine vorhergehende Umverteilungsoperation fehlgeschlagen ist, wiederholen Sie sie und geben Sie dabei die Option **CONTINUE** oder **ABORT** an. Mit **CONTINUE** wird die zuvor abgebrochene Umverteilungsoperation abgeschlossen, mit **ABORT** werden die Auswirkungen der zuvor abgebrochenen Operation rückgängig gemacht.

Fehlerbehebungsscripts

Sie verfügen möglicherweise über interne Tools oder Scripts, die auf den Prozessen basieren, die in der Datenbanksteuerkomponente ausgeführt werden. Diese Tools oder Scripts funktionieren möglicherweise nicht mehr, da alle Agenten, Vorablesefunktionen und Seitenlöschfunktionen jetzt als Threads in einem einzigen Multithread-Prozess betrachtet werden.

Ihre internen Tools und Scripts müssen so geändert werden, dass sie in einem Threadprozess gültig sind. Sie verfügen zum Beispiel über Scripts, die den Befehl **ps** starten, um die Prozessnamen aufzulisten und anschließend Tasks mit bestimmten Agentenprozessen auszuführen. Ihre Scripts müssen umgeschrieben werden.

Der Datenbankbefehl **db2pd** für die Fehlerbestimmung verfügt über die neue Option **-edu** (kurz für „engine dispatchable unit"), mit der alle Agentennamen mit ihren Thread-IDs aufgelistet werden. Der Befehl **db2pd -stack** kann bei der Thread-Steuerkomponente weiterhin verwendet werden, um Speicherauszüge für einzelne EDU-Stacks zu erstellen oder um Speicherauszüge für alle EDU-Stacks des aktuellen Knotens zu erstellen.

Erneutes Kompilieren des statischen Abschnitts zum Erfassen von Ist-Daten für den Abschnitt nach Anwendung von Fixpack 1

Nach der Anwendung von DB2 Version 9.7 Fixpack 1 können Ist-Daten für den Abschnitt nicht für einen statischen Abschnitt erfasst werden, der vor der Anwendung des Fixpacks kompiliert wurde. Der statische Abschnitt muss erneut kompiliert werden, um Ist-Daten für den Abschnitt nach der Anwendung von Fixpack 1 zu erfassen.

Symptome

Ist-Daten für Abschnitte werden nicht erfasst, wenn die Routine EXPLAIN_FROM_ACTIVITY ausgeführt wird.

Ursachen

Ist-Daten den Abschnitt können nicht für einen statischen Abschnitt erfasst werden, der vor der Anwendung des Fixpack kompiliert wurde.

Problemlösung

Stellen Sie nach der Installation von DB2 Version 9.7 Fixpack 1 sicher, dass der statische Abschnitt nach der Anwendung des Fixpacks mit dem Befehl **REBIND** erneut gebunden wurde. Dazu prüfen Sie die Spalte LAST_BIND_TIME in der Katalogsicht SYSCAT.PACKAGES.

Fehlerbehebung bei der Speicherschlüsselunterstützung

Speicherschutzschlüssel (Hardwareschlüssel auf Threadebene) werden dazu verwendet, eine höhere Ausfallsicherheit für die DB2-Engine bereitzustellen, indem der Speicher gegen unzulässige Zugriffsversuche geschützt wird. Informationen zum Beheben von Fehlern beim Aktivieren dieses Schutzes finden Sie im nachfolgenden Abschnitt.

Informationen zu diesem Vorgang

Diagnostizieren von Fehlern bei Registrierdatenbankvariablen

Bei der Definition der Registrierdatenbankvariablen „DB2_MEMORY_PROTECT" *Datenbankverwaltung - Konzepte und Konfiguration - Referenzinformationen* wurde der Fehler DBI1301E zurückgegeben, der auf ungültige Werte hinweist. Dieser Fehler tritt aus einem der folgenden Gründe auf:

- Für die Registrierdatenbankvariable wurde ein ungültiger Wert angegeben. Informieren Sie sich anhand der Informationen zur Verwendung der entsprechenden Registrierdatenbankvariablen **DB2_MEMORY_PROTECT** über zulässige Werte.
- Möglicherweise werden Speicherschutzschlüssel nicht von Hardware und Betriebssystem unterstützt, sodass dieser Schutz nicht aktiviert werden kann. Speicherschutzschlüssel sind bei POWER6-Prozessoren verfügbar und werden ab AIX 5L Version 5.3, 5300-06 Technology Level, unterstützt.

Kapitel 7. Fehlerbehebung in DB2 Connect Server

Die DB2 Connect-Umgebung umfasst mehrere Software-, Hardware- und Kommunikationsprodukte. Der beste Ansatz für die Fehlerbehebung ist das Ausschließen von Möglichkeiten und eine Annäherung an die Fehlerursache in Einzelschritten.

Stellen Sie zuerst die relevanten Informationen zusammen, und bestimmen Sie auf der Grundlage dieser Informationen das für Ihren Fall zutreffende Thema. Fahren Sie dann mit dem entsprechenden Abschnitt fort.

Diagnosetools

Wenn ein Problem auftritt, stehen folgende Hilfsmittel zur Verfügung:

- Alle Diagnosedaten einschließlich von Speicherauszugsdateien, Trapdateien, Fehlerprotokollen, Benachrichtungsdateien und Alertprotokollen befinden sich in dem Pfad, der über den Konfigurationsparameter **diagpath** des Datenbankmanagers definiert ist. Dieser Parameter gibt den Pfad für das Verzeichnis mit den Diagnosedaten an.

  Ist dieser Konfigurationsparameter mit Null definiert, werden die Diagnosedaten in eines der folgenden Verzeichnisse geschrieben:

  – Linux- und UNIX-Umgebungen: `INSTHOME/sqllib/db2dump`, wobei *INSTHOME* das Ausgangsverzeichnis der Instanz angibt.
  – Unterstützte Windows-Umgebungen:
    - Ist die Umgebungsvariable **DB2INSTPROF** nicht definiert, wird das Verzeichnis `x:\SQLLIB\DB2INSTANCE` verwendet, wobei `x:\SQLLIB` auf das Laufwerk und das über die Registrierdatenbankvariable **DB2PATH** angegebene Verzeichnis verweist und **DB2INSTANCE** den Namen der Instanz angibt.

      Anmerkung: Das Verzeichnis muss nicht zwingend mit SQLLIB bezeichnet sein.
    - Wenn die DB2-Registrierdatenbankvariable **DB2INSTPROF** definiert ist, wird `x:\DB2INSTPROF\DB2INSTANCE` verwendet. Dabei ist `x:\DB2INSTPROF` der in der Registrierdatenbankvariablen **DB2INSTPROF** angegebene Pfad und **DB2INSTANCE** der Name der Instanz (standardmäßig der Wert von **DB2INSTDEF** unter 32-Bit-Windows-Betriebssystemen).
- Bei Windows-Betriebssystemen können Sie mithilfe der Ereignisanzeigefunktion (Event Viewer) das Protokoll für Verwaltungshinweise anzeigen.
- Als Diagnosetools sind zum Beispiel die Tools **db2trc**, **db2pd**, **db2support** und **db2diag** verfügbar.
- Bei Linux- und UNIX-Betriebssystemen der Befehl **ps**, der Informationen zum Status der aktiven Prozesse an die Standardausgabe weiterleitet.
- Bei UNIX-Betriebssystemen die Kerndatei, die im aktuellen Verzeichnis erstellt wird, wenn schwerwiegende Fehler auftreten. Sie enthält ein Hauptspeicherabbild des beendeten Prozesses und kann herangezogen werden, um zu ermitteln, welche Funktion den Fehler verursachte.

Zusammenstellen relevanter Informationen

Zur Fehlerbehebung gehört unter anderem das Einkreisen des Problembereichs und das Prüfen möglicher Ursachen. Zu Beginn der Analyse sollten die relevanten Informationen zusammengestellt werden, und es sollte festgestellt werden, welche Fakten bekannt sind, welche Fakten nicht bekannt sind und welche möglichen Problemursachen ausgeschlossen werden können. Es sollten mindestens folgende Fragen beantwortet werden:

- War die einleitende Verbindung erfolgreich?
- Funktioniert die Hardware einwandfrei?
- Funktionieren die Übertragungswege ordnungsgemäß?
- Wurden Änderungen am Kommunikationsnetz vorgenommen, durch die frühere Verzeichniseinträge ungültig wurden?
- Wurde die Datenbank gestartet?
- Tritt der Kommunikationsfehler zwischen mehreren Clients und dem DB2 Connect-Server (Gateway), zwischen dem DB2 Connect-Gateway und dem IBM Mainframe-Datenbankserver oder zwischen DB2 Connect Personal Edition und dem IBM Mainframe-Datenbankserver auf?
- Was lässt sich aus dem Inhalt der Nachricht und den in der Nachricht aufgeführten Token ablesen?
- Können Diagnose-Tools wie **db2trc**, **db2pd**, und **db2support** im derzeitigen Stadium hilfreich sein?
- Arbeiten andere Maschinen, die die gleichen Funktionen ausführen, einwandfrei?
- Wird im Fall einer fernen Funktion diese lokal erfolgreich ausgeführt?

Nicht erfolgreiche einleitende Verbindung

Beantworten Sie folgende Fragen, und stellen Sie sicher, dass die Schritte zur Installation ordnungsgemäß durchgeführt wurden:

1. *Wurde der Installationsvorgang erfolgreich abgeschlossen?*
   - Waren alle vorausgesetzten Softwareprodukte verfügbar?
   - War genug Hauptspeicher und Plattenspeicher verfügbar?
   - Wurde die Unterstützung für ferne Clients installiert?
   - Wurde die Installation der Kommunikationssoftware ohne Fehlerbedingungen beendet?
2. *Für UNIX-gestützte Systeme: Wurde eine Instanz des Produkts erstellt?*
   - Haben Sie als Rootbenutzer einen Benutzer und eine Gruppe als Instanzeigner und als Gruppe SYSADM erstellt?
3. *Wurden (falls dies im vorliegenden Fall zutrifft) die Lizenzinformationen erfolgreich verarbeitet?*
   - Wurde bei UNIX-Systemen die Datei 'nodelock' editiert und das von IBM angegebene Kennwort eingegeben?
4. *War die Kommunikation des IBM Mainframe-Datenbankservers und der Workstation ordnungsgemäß konfiguriert?*
   - Es gibt drei Konfigurationen, die betrachtet werden müssen:
     a. Die Konfiguration des IBM Mainframe-Datenbankservers identifiziert den Anwendungsrequester gegenüber dem Server. Das Verwaltungssystem

der IBM Mainframe-Serverdatenbank enthält Systemkatalogeinträge, die die Position, das Netzprotokoll und die Sicherheit des Requesters definieren.

b. Die Konfiguration der DB2 Connect-Workstation definiert die Gruppe von Clients gegenüber dem Server und den IBM Mainframe-Server gegenüber dem Client.

c. In der Konfiguration der Client-Workstation müssen der Name der Workstation und das Übertragungsprotokoll definiert sein.

- Wenn keine einleitende Verbindung hergestellt wurde, muss im Rahmen der Problemanalyse überprüft werden, ob die Namen aller physischen Einheiten vollständig und korrekt sind. Bei TCP/IP-Verbindungen muss überprüft werden, ob die korrekte Portnummer und der korrekte Hostname angegeben wurden.
- Sowohl der Datenbankadministrator des IBM Mainframe-Servers als auch die Netzadministratoren verfügen über Dienstprogramme, mit denen eine Problemdiagnose durchgeführt werden kann.

5. *Verfügen Sie über die für das Verwaltungssystem der IBM Mainframe-Serverdatenbank erforderliche Berechtigungsstufe zur Verwendung der IBM Mainframe-Serverdatenbank?*
   - Es müssen die Zugriffsberechtigung des Benutzers, die Regeln für Tabellenqualifikationsmerkmale und die erwarteten Ergebnisse beachtet werden.
6. *Schlägt der Versuch fehl, mit dem Befehlszeilenprozessor (CLP) SQL-Anweisungen für einen IBM Mainframe-Datenbankserver abzusetzen?*
   - Wurde der CLP anhand der entsprechenden Prozedur an den IBM Mainframe-Datenbankserver gebunden?

Probleme nach dem Herstellen einer einleitenden Verbindung

Die folgenden Fragen sollen helfen, den Problembereich einzukreisen.

1. *Liegen besondere oder ungewöhnlichen Umstände beim Betrieb vor?*
   - Handelt es sich um eine neue Anwendung?
   - Werden neue Prozeduren verwendet?
   - Wurden in letzter Zeit Änderungen vorgenommen, die Auswirkungen auf das System haben könnten? Wurden z. B. Softwareprodukte oder Anwendungen geändert, seit die Anwendung oder das Szenario zum letzten Mal erfolgreich durchgeführt wurden?
   - Bei Anwendungsprogrammen: Welche Anwendungsprogrammierschnittstelle (API) wurde zur Erstellung des Programms verwendet?
   - Wurden andere Anwendungen, die die Software oder Kommunikations-APIs verwenden, auf dem System des Benutzers ausgeführt?
   - Wurde in letzter Zeit ein Fixpack installiert? Trat das Problem auf, als ein Benutzer versuchte, eine Funktion zu verwenden, die seit ihrer Installation nicht im Betriebssystem verwendet (oder geladen) wurde, ermitteln Sie das neueste Fixpack von IBM, und laden Sie es, *nachdem* die Funktion installiert wurde.
2. *Trat dieser Fehler bereits früher auf?*
   - Gibt es dokumentierte Lösungen für frühere Fehlerbedingungen?
   - Wer waren die Betroffenen? Können diese Hinweise für mögliche Maßnahmen bieten?
3. *Wurde versucht, über Befehle der DFV-Software Informationen zum Netz abzurufen?*

- Durch die Verwendung von TCP/IP-Befehlen und -Dämonen können möglicherweise wertvolle Informationen abgerufen werden.

4. *Wurden Informationen im SQL-Kommunikationsbereich zurückgegeben, die nützlich sein könnten?*
   - Prozeduren zur Fehlerbehebung sollten Maßnahmen zur Überprüfung des Inhalts der Felder für SQLCODE- und SQLSTATE-Werte umfassen.
   - SQLSTATE-Werte ermöglichen Anwendungsprogrammierern das Testen auf Fehlerklassen, die der DB2-Familie von Datenbankprodukten gemeinsam sind. In einem Netz mit verteilten relationalen Datenbanken kann dieses Feld eine gemeinsame Basis darstellen.
5. *Wurde START DBM auf dem Server ausgeführt?* Stellen Sie zusätzlich sicher, dass die Umgebungsvariable DB2COMM korrekt eingestellt ist, sodass Clients Fernzugriff auf den Server haben.
6. *Können andere Maschinen, die die gleichen Funktionen ausführen, die Verbindung zum Server erfolgreich herstellen?* Es könnte sein, dass die maximale Anzahl von Clients erreicht ist, die versuchen, eine Verbindung zum Server herzustellen. Wenn ein anderer Client die Verbindung zum Server trennt, kann der Client, der bisher die Verbindung nicht herstellen konnte, sie nun herstellen?
7. *Hat die Maschine die richtige Adressierung?* Prüfen Sie, ob die Maschine im Netz eindeutig ist.
8. *Wurde dem Client die richtige Berechtigung für den Fernzugriff erteilt?* Die Verbindung zur Instanz kann zwar erfolgreich sein, allerdings wurde möglicherweise keine Berechtigung auf Datenbank- oder Tabellenebene erteilt.
9. *Ist dies die erste Maschine, die eine Verbindung zu einer fernen Datenbank herstellt?* In verteilten Umgebungen können Router oder Brücken zwischen Netzen die Kommunikation zwischen dem Client und dem Server blockieren. Bei TCP/IP muss beispielsweise sichergestellt werden, dass der ferne Host auf ein Pingsignal antwortet.

Nicht unterstützte DDM-Befehle

Die DDM-Befehle **BNDCPY**, **BNDDPLY**, **DRPPKG** und **DSCRDBTBL** werden von DB2 Version 9.5 für Linux, UNIX und Windows nicht unterstützt, wenn das Produkt als DRDA-Anwendungsserver (DRDA-AS) ausgeführt wird.

Symptome

Wenn ein DRDA-Anwendungsrequester (DRDA-AR) eine Verbindung zu DB2 Version 9.5 für Linux, UNIX und Windows herstellt und einen der folgenden Befehle absetzt, schlägt der betreffende Befehl fehl:

Tabelle 82. Nicht unterstützte DDM-Befehle

DDM-Befehl	DDM-Codepunkt	Beschreibung
BNDCPY	X'2011'	Kopieren eines vorhandenen RDB-Pakets (RDB = relationale Datenbank)
BNDDPLY	X'2016'	Implementieren eines vorhandenen RDB-Pakets.
DRPPKG	X'2007'	Löschen eines Pakets.
DSCRDBTBL	X'2012'	Beschreiben einer RDB-Tabelle.

Darüber hinaus werden die folgenden Codepunkte, die im SQLDTA-Deskriptor für

parameterbasierte (oder spaltenbasierte) Feldgruppeneingabe verwendet werden, ebenfalls nicht unterstützt:

Tabelle 83. Nicht unterstützte FD:OCA-Datenobjekte

FD:OCA-Datenobjekte	DDM-Codepunkt	Beschreibung
FDOEXT	X'147B'	FD:OCA-Datenbereiche (FD:OCA = Formatted Data Object Content Architecture)
FDOOFF	X'147D'	Relative FD:OCA-Datenadressen

Die häufigste Fehlernachricht in dieser Situation ist SQL30020N ("Die Ausführung schlug aufgrund eines Verteilungsprotokollfehlers (Distributed Protocol Error) fehl. Dieser Fehler beeinflusst die erfolgreiche Ausführung der nachfolgenden Befehle und SQL-Anweisungen.")

Ursachen

Distributed Data Management Architecture (DDM) ist Teil des DRDA-Protokolls. Die DDM-Befehle **BNDCPY**, **BNDDPLY**, **DRPPKG** und **DSCRDBTBL** sind in allen DRDA-Versionen enthalten, die von DB2 Version 9.5 für Linux, UNIX und Windows unterstützt werden, doch der DRDA-Anwendungsserver unterstützt diese DDM-Befehle nicht.

Ebenso unterstützen DB2 Version 9.5 for Linux-, UNIX- und Windows-DRDA-Anwendungsserver die Codepunkte **FDOEXT** und **FDOOFF** nicht. Diese Codepunkte werden im SQLDTA-Deskriptor verwendet, der an den Server gesendet wird, wenn Sie eine spaltenbasierte Feldgruppeneingabeanforderung übergeben.

Problemdiagnose

Wenn Sie einen DB2-Trace auf dem DRDA-Anwendungsserver erstellen, sehen Sie als Antwort auf diese Befehle eine Nachricht ähnlich der folgenden: `ERROR MSG = Parser: Command Not Supported`.

Problemlösung

Es gibt momentan keine unterstützten Alternativen für die DDM-Befehle **BNDCPY** und **BNDDPLY**.

Verwenden Sie zum Löschen eines Pakets die SQL-Anweisung DROP PACKAGE. Stellen Sie beispielsweise eine Verbindung zum DB2 Version 9.5 for Linux-, UNIX- und Windows-DRDA-Anwendungsserver her und senden Sie die Anweisung DROP PACKAGE in einer Anforderung EXECUTE IMMEDIATE. DB2 Version 9.5 für Linux, UNIX und Windows verarbeitet diese Anforderung erfolgreich.

Verwenden Sie zur Beschreibung einer RDB-Tabelle einen der folgenden DDM-Befehle: **DSCSQLSTT** (SQL-Anweisung beschreiben) oder **PRPSQLSTT** (SQL-Anweisung vorbereiten). Wenn Sie beispielsweise eine Beschreibung der Tabelle TAB1 benötigen, beschreiben Sie die folgende Anweisung bzw. bereiten Sie sie vor: `SELECT * FROM TAB1`.

Anmerkung: Wenn der DRDA-AR den Befehl **PRPSQLSTT** absetzt, muss auch die Instanzvariable **RTNSQLDA** mit dem Wert `TRUE` angegeben werden, da andernfalls der Deskriptor SQLDARD (SQLDA Reply Data) nicht vom Server zurückgegeben wird.

Um Probleme mit den **FDOEXT**- und **FDOOFF**-Codepunkten zu vermeiden, sollten Sie zeilenbasierte Feldgruppeneingabeanforderungen anstelle von parameterbasierten (oder spaltenbasierten) Feldgruppeneingabeanforderungen verwenden.

DB2 Connect - Häufige Probleme

In diesem Abschnitt werden die häufigsten Symptome von Verbindungsproblemen bei der Verwendung von DB2 Connect aufgelistet. Für jedes Problem werden Ihnen folgende Informationen zur Verfügung gestellt:

- Eine Kombination aus Nachrichtennummer und Rückkehrcode (oder protokollspezifischem Rückkehrcode) für die Nachricht. Jede Kombination aus Nachricht und Rückkehrcode hat eine separate Überschrift, und die Überschriften sind der Nachrichtennummer und dann dem Rückkehrcode nach geordnet.
- Es wird ein Symptom angegeben, in der Regel in Form einer Beispielnachricht.
- Es wird eine Lösung vorgeschlagen, die die wahrscheinliche Ursache des Fehlers angibt. In einigen Fällen werden eventuell mehrere Lösungen vorgeschlagen.

SQL0965 oder SQL0969

Symptom

Die Nachrichten SQL0965 und SQL0969 können mit einer Reihe unterschiedlicher Rückkehrcodes von IBM DB2 for IBM i, DB2 for z/OS und DB2 Server for VM and VSE ausgegeben werden.

Wird eine dieser Nachrichten angezeigt, müssen Sie den ursprünglichen SQL-Code in der Dokumentation für das Datenbankserverprodukt nachschlagen, das die Nachricht ausgegeben hat.

Lösung

Der von der IBM Mainframe-Datenbank empfangene SQL-Code kann nicht umgesetzt werden. Korrigieren Sie das Problem basierend auf dem Fehlercode, und wiederholen Sie den fehlgeschlagenen Befehl.

SQL5043N

Symptom

Die Unterstützung für eines oder mehrere Übertragungsprotokolle konnte nicht gestartet werden. Die Kernfunktionalität des Datenbankmanagers wurde jedoch erfolgreich gestartet.

Vielleicht wurde das TCP/IP-Protokoll auf dem DB2 Connect-Server nicht gestartet. Möglicherweise hat zuvor eine erfolgreiche Client-Verbindung bestanden.

Bei Verwendung von `diaglevel = 4` enthalten die **db2diag**-Protokolldateien möglicherweise einen Eintrag ähnlich zum Beispiel dem folgenden:

```
2001-05-30-14.09.55.321092   Instance:svtdbm5   Node:000
   PID:10296(db2tcpcm)   Appid:none
   common_communication  sqlcctcpconnmgr_child   Probe:46
   DIA3205E Die Socket-Adresse "30090", die in der
   TCP/IP-Servicedatei definiert und    für die TCP/IP-Server-
Unterstützung erforderlich ist, wird von einem anderen
Prozess verwendet.
```

Lösung

Diese Warnung ist ein Symptom dafür, dass DB2 Connect als Server für ferne Clients Schwierigkeiten beim Verarbeiten von einem oder mehreren Clientübertragungsprotokollen hat. Dabei kann es sich um TCP/IP-Protokolle oder andere Protokolle handeln. In der Nachricht wird in der Regel

angegeben, dass eines der für DB2 Connect definierten Übertragungsprotokolle nicht ordnungsgemäß konfiguriert ist.

Eine mögliche Ursache ist häufig, dass die Profilvariable DB2COMM nicht oder falsch definiert ist. Im Allgemeinen ist das Problem das Ergebnis einer Abweichung zwischen der Variablen DB2COMM und den in der Datenbankmanagerkonfiguration definierten Namen (zum Beispiel **svcename** oder **nname**).

Ein mögliches Szenario ist, dass zuvor erfolgreich eine Verbindung hergestellt wurde und dass dann die Fehlernachricht SQL5043 angezeigt wird, obwohl die Konfiguration nicht geändert wurde. Dazu kann es bei Verwendung des TCP/IP-Protokolls kommen, wenn das ferne System die Verbindung aus einem bestimmten Grund abnormal beendet. Tritt dieser Fall auf, kann auf dem Client weiterhin eine Verbindung vorhanden sein, und es ist eventuell möglich, die Verbindung durch Absetzen der unten stehenden Befehle ohne weiteres Eingreifen wiederherzustellen.

Sehr wahrscheinlich hat einer der Clients, der mit dem DB2 Connect-Server verbunden ist, noch eine Kennung am TCP/IP-Port. Geben Sie auf jeder Client-Maschine, die mit dem DB2 Connect-Server verbunden ist, die folgenden Befehle ein:

```
db2 terminate
db2stop
```

SQL30020

Symptom

SQL30020N Die Ausführung schlug aufgrund eines Verteilungsprotokollfehlers (Distributed Protocol Error) fehl. Dieser Fehler beeinflusst die erfolgreiche Ausführung der nachfolgenden Befehle und SQL-Anweisungen.

Lösungen

Wenden Sie sich bei diesem Fehler an den Service. Führen Sie den Befehl **db2support** aus, bevor Sie sich an den Service wenden.

SQL30060

Symptom

SQL30060N "<*berechtigungs-id*>" verfügt nicht über die Berechtigung, die Operation "<*operation*>" auszuführen.

Lösung

Beim Herstellen der Verbindung zu DB2 for z/OS wurden die Kommunikationsdatenbanktabellen nicht ordnungsgemäß aktualisiert.

SQL30061

Symptom

Verbindung zu falscher Position des IBM Mainframe-Datenbankservers. Es wurde keine Zieldatenbank gefunden.

Lösung

Im DCS-Verzeichniseintrag wurde eventuell der falsche Name der Serverdatenbank angegeben. In diesem Fall wird SQLCODE -30061 an die Anwendung zurückgegeben.

Überprüfen Sie den DB2-Knoten, die Datenbank und die DCS-Verzeichniseinträge. Das Feld für den Zieldatenbanknamen im DCS-Verzeichniseintrag muss mit dem Namen der Datenbank auf der Basis der Plattform übereinstimmen. Bei einer Datenbank unter DB2 for z/OS muss der zu

verwendende Name beispielsweise mit dem Namen übereinstimmen, der im Feld "LOCATION=*standortname*" des BSDS (Boot Strap Data Set) verwendet wird und der auch in der Nachricht DSNL004I (LOCATION=*standort*) angezeigt wird, wenn DDF (Distributed Data Facility) gestartet wird.

Die korrekten Befehle für einen TCP/IP-Knoten lauten wie folgt:

```
db2 catalog tcpip node <knotenname> remote <hostname_oder_-adresse>
            server <port_nr_oder_servicename>
db2 catalog dcs database <lokaler_name> as <tatsächlicher_datenbankname>
db2 catalog database <lokaler_name> as <alias> at node <knotenname>
            authentication server
```

Setzen Sie anschließend den folgenden Befehl ab, um die Verbindung zur Datenbank herzustellen:

```
db2 connect to <alias> user <benutzername> using <kennwort>
```

SQL30081N mit Rückkehrcode 79

Symptom

```
SQL30081N  Übertragungsfehler.
Verwendetes
Übertragungsprotokoll: "TCP/IP".
Verwendete Übertragungs-API: "SOCKETS".
   Position, an der
der Fehler festgestellt wurde: "".
Übertragungsfunktion, die    den Fehler feststellte:    "connect".
Protokollspezifische(r) Fehlercode(s): "79", "*", "*".
SQLSTATE=08001
```

Lösung(en)

Dieser Fehler kann auftreten, wenn ein ferner Client keine Verbindung zu einem DB2 Connect-Server herstellen kann. Er kann auch auftreten, wenn eine Verbindung vom DB2 Connect-Server zu einem IBM Mainframe-Datenbankserver hergestellt wird.

1. Möglicherweise ist die Profilvariable `DB2COMM` auf dem DB2 Connect-Server falsch eingestellt. Überprüfen Sie diese Variable. Zum Beispiel muss der Befehl `db2set db2comm=tcpip` in `sqllib/db2profile` vorhanden sein, wenn DB2 Enterprise Server Edition unter AIX ausgeführt wird.
2. Möglicherweise liegt eine Abweichung zwischen dem TCP/IP-Servicenamen und den Angaben der Portnummern auf dem IBM Data Server-Client und dem DB2 Connect-Server vor. Prüfen Sie die Einträge in den TCP/IP-Dateien `services` auf beiden Maschinen.
3. Stellen Sie sicher, dass DB2 auf dem DB2 Connect-Server gestartet wurde. Setzen Sie `diaglevel` der Datenbankmanagerkonfiguration mit dem folgenden Befehl auf 4:

   ```
   db2 update dbm cfg using diaglevel 4
   ```

   Überprüfen Sie nach dem Stoppen und Neustart von DB2 die **db2diag**-Protokolldateien, ob die DB2-TCP/IP-Kommunikation gestartet wurde. Es wird eine Ausgabe angezeigt, die der folgenden ähnelt:

   ```
   2001-02-03-12.41.04.861119   Instance:svtdbm2   Node:00
   PID:86496(db2sysc)   Appid:none
   common_communication  sqlcctcp_start_listen   Probe:80
   DIA3000I Die Protokollunterstützung für "TCPIP" wurde erfolgreich gestartet.
   ```

SQL30081N mit protokollspezifischem Fehlercode 10032

Symptom

```
SQL30081N  Übertragungsfehler.
Verwendetes
Übertragungsprotokoll: "TCP/IP".
Verwendete Übertragungs-API: "SOCKETS".
   Position, an der
der Fehler festgestellt wurde: "9.21.85.159".
   Übertragungsfunktion, die den
Fehler feststellte: "send".
Protokollspezifische(r) Fehlercode(s): "10032",    "*", "*".
SQLSTATE=08001
```

Lösung

Diese Fehlernachricht wird eventuell empfangen, wenn versucht wird, die Verbindung zu einer Maschine zu trennen, auf der die TCP/IP-Kommunikation bereits fehlgeschlagen ist. Korrigieren Sie das Problem mit dem TCP/IP-Subsystem.

Starten Sie dazu auf den meisten Maschinen einfach das TCP/IP-Protokoll erneut. Gelegentlich ist der Neustart der gesamten Maschine erforderlich.

SQL30082 RC=24 während CONNECT

Symptom

SQLCODE -30082 Angegebene Benutzer-ID oder Kennwort ist falsch.

Lösung

Stellen Sie sicher, dass in der Anweisung CONNECT das richtige Kennwort angegeben ist, falls es erforderlich ist. Das Kennwort kann nicht an die Zielserverdatenbank gesendet werden, weil es nicht verfügbar ist. Ein Kennwort muss vom IBM Data Server-Client an die Zielserverdatenbank gesendet werden. Auf bestimmten Plattformen, zum Beispiel AIX, kann das Kennwort nur abgerufen werden, wenn es in der Anweisung CONNECT bereitgestellt wird.

Kapitel 8. Fehlerbehebung bei DB2 Text Search

Die Tasks zur Fehlerbehebung und Verwaltung für DB2 Text Search umfassen das Löschen verwaister Datensammlungen der Textsuche und das Bereinigen von Textsuchindexereignissen.

Bei der Fehlerbestimmung für DB2 Text Search können Sie auf die drei folgenden Ressourcen zurückgreifen: Ereignistabelle, Befehl **db2trc** und Formatierungstool für Protokolle.

Ausführen der Tracefunktion zur Überprüfung von Fehlern bei der Textsuche

Wenn Sie einen Fehler an einen IBM Ansprechpartner melden müssen, werden Sie möglicherweise aufgefordert, die Tracefunktion einzuschalten, sodass die Informationen in eine Datei geschrieben werden, die anschließend zur Fehlersuche verwendet werden kann.

Informationen zu diesem Vorgang

Sie können die DB2-Tracefunktion (**db2trc**) verwenden, um Informationen zu den Problemen zu erfassen, die während der Ausführung von Textsuchoperationen oder Verwaltungsoperationen auftreten, die in Zusammenhang mit der Textsuche stehen.

Da die Systemleistung durch das Einschalten der Tracefunktion beeinträchtigt wird, sollten Sie sie nur verwenden, wenn Sie von einem Mitarbeiter des IBM Support Center oder von Ihrem Ansprechpartner des Technical Support dazu aufgefordert werden.

Um die Tracefunktion zu aktivieren und Informationen zu empfangen, die sich speziell auf DB2 Text Search beziehen, müssen Sie den Befehl **db2trc** mit einer Maske mit einem Komponentencode für CIE (155) wie folgt ausführen:

```
db2trc on -m '*.*.CIE.*.*'
```

Um die Diagnose schwerwiegender Fehler zu erleichtern, können Sie auch die Protokolldatei **db2diag** heranziehen.

Überwachen von Warteschlangen für DB2 Text Search-Indexaktualisierungen

Sie können Überwachungsdaten zusammenstellen, die zur Optimierung der Serverkonfiguration von DB2 Text Search verwendet werden können, wenn auf Ihrem System Probleme in Bezug auf die Indexierleistung auftreten.

Sie können die Ein- und Ausgabewarteschlangen überwachen, indem Sie den DB2 Text Search-Server mit der Markierung -monitorQueues starten. Wenn Sie den Server mit der Markierung -monitorQueues starten, dann werden Informationen zum aktuellen Status der Ein- und Ausgabewarteschlangen in die Dateien `InputQueueSizes.csv` und `OutputQueueSizes.csv` ausgegeben. Diese Dateien werden im Verzeichnis `db2tss\logs` gespeichert. Jede Datei enthält die folgenden Informationen:

- Die Uhrzeit.
- Die aktuelle Anzahl der Dokumente, die auf die Vorverarbeitung und Indexierung warten.
- Die Anzahl der Dokumente, für die momentan die Vorverarbeitung und Indexierung durchgeführt wird.
- Die aktuelle Warteschlangengröße. Die maximale Warteschlangengröße ist ein konfigurierbarer Parameter.
- Die Gesamtzahl der Dokumente, die die Warteschlange durchlaufen haben, d. h., die Anzahl der Dokumente, für die die Vorverarbeitung und Indexierung seit dem letzten Serverstart durchgeführt wurde.

Wichtig: Die CSV-Dateien werden erneut erstellt, wenn der Server gestartet wird. Wenn Sie die Informationen in diesen Dateien speichern wollen, dann führen Sie ein Backup für sie durch, bevor Sie den Server erneut starten.

Gehen Sie wie folgt vor, um den Server im Warteschlangenüberwachungsmodus zu starten:

1. Stoppen Sie den Server.
2. Bearbeiten Sie das Startscript im Verzeichnis *ECMTS_HOME*\bin.
3. Fügen Sie die Markierung -monitorQueues zu einem der Startbefehle hinzu. Beispiel:

```
"%JAVA_HOME%\bin\java" -classpath "%CLASSPATH%"
com.ibm.es.nuvo.launcher.Launcher "%CONFIG_XML%" -monitorQueues
-Xmx%HEAP_SIZE% %1 %2 %3 %4 %5 %6 %7 %8 %9 %IPv6Flag% %JVM_OPTIONS%
```

4. Speichern Sie das Startscript und führen Sie für den Server einen Neustart durch.

Verbesserung der Leistung auf der Basis des Warteschlangenstatus
Mithilfe der Informationen in der folgenden Tabelle können Sie Probleme in Bezug auf die Indexierleistung beheben.

Tabelle 84.

Warteschlangenstatus	Fehlerbehebungsstrategie
Die Eingabewarteschlange ist leer.	Es werden keine Dokumente vom Client empfangen. Vergewissern Sie sich, dass der Client Dokumente mit Push schnell genug an den Server überträgt.
Die Eingabewarteschlange ist vollständig belegt, die Ausgabewarteschlange ist leer.	Überprüfen Sie die Protokolle auf Vorverarbeitungsfehler hin und prüfen Sie die Möglichkeit, die Anzahl der Vorverarbeitungsthreads zu erhöhen. Niedrige CPU-Auslastungswerte (weniger als 60 %) auf Servern in umfangreichen Multiprozessorsystemen können ebenfalls darauf hindeuten, dass die Anzahl der Vorverarbeitungsthreads erhöht werden sollte. Die Anzahl der Vorverarbeitungsthreads sollte die Anzahl der Prozessoren nicht übersteigen.

Tabelle 84. (Forts.)

Warteschlangenstatus	Fehlerbehebungsstrategie
Sowohl Ein- als auch Ausgabewarteschlangen sind vollständig belegt.	Wenden Sie eine oder mehrere der folgenden Strategien an: • Erhöhen Sie die Anzahl der Indexthreads. • Konfigurieren Sie häufigere, kürzere Zusammenführungen, indem Sie den Wert des Parameters **MaxMergeDocs** in der Datei *db2tss*\config\collections\collection_name\collection.xml reduzieren. • Erhöhen Sie die Stapelgröße. Die Indexierung von Dokumenten in kleinen Stapeln kann die Leistung beeinträchtigen, da der Index gelöscht und die Daten für jeden indexierten Stapel auf der Platte gespeichert werden. • Erhöhen Sie den Wert des Parameters **BufferSize** in der Datei *db2tss*\config\collections\collection_name\collection.xml. • Indexieren Sie die Objektgruppen nacheinander. • Vergewissern Sie sich, dass die Plattengeschwindigkeit und die Geschwindigkeit der Netzverbindung ausreichen.

Hinweise und Tipps zur Fehlerbehebung bei DB2 Text Search

Mithilfe der Informationen in der folgenden Tabelle können Sie Probleme beheben.

Tabelle 85. Probleme und Lösungen

Problem	Lösung
Wenn Sie einen Textindex während der Ausführung einer DB2-Verbindungssitzung löschen und erneut erstellen, wird möglicherweise ein Plan für eine zuvor verwendete Abfrage (für Net Search Extender oder DB2 Text Search) im Cache zwischengespeichert und wiederverwendet. Dadurch kann es zu fehlerhaften Ergebnissen kommen. Wenn die DB2 Text Search-Steuerkomponente diesen Fehler feststellt, dann wird die Nachricht IQQG0037W zurückgegeben, in der angegeben ist, dass für die Suche ein nicht korrekter Objektgruppenname verwendet wurde.	Mit der Anweisung FLUSH PACKAGE CACHE DYNAMIC können Sie die Ausführungspläne für die SQL-Anweisungen aus dem Anweisungscache entfernen.

Tabelle 85. Probleme und Lösungen (Forts.)

Problem	Lösung
Wenn Sie Ihre DB2-Instanz mit dem Befehl **db2idrop**, **db2iupdt** oder **db2iupgrade** löschen oder aktualisieren, wird das gesamte Verzeichnis `sqllib` einschließlich des Verzeichnisses `sqllib/db2tss/config` in der Instanz entfernt. Dadurch wird die Wiederverwendung von Textsuchinstanzdatenbanken erschwert.	Führen Sie für das Verzeichnis `sqllib/db2tss/config` ein Backup durch, bevor Sie den Befehl **db2idrop**, **db2iupdt** oder **db2iupgrade** ausgeben. Verwenden Sie außerdem die Option **printAll** des Befehls **configTool**, um einen Datensatz der Konfigurationsparameter zu speichern, mit deren Hilfe Sie die neue Instanz konfigurieren können.
Durch einen Fehler aufgrund einer vollständig belegten Platte kann es zur Inkonsistenz der Textindexmetadaten kommen.	Verwenden Sie das Verwaltungstool, um zu überprüfen, ob verwaiste Objektgruppen vorhanden sind, die entfernt werden müssen. Dazu befolgen Sie die Anweisungen im Abschnitt zum „Löschen verwaister Objektgruppen für die Textsuche" in der DB2-Dokumentation zur Textsuche. Beachten Sie hierbei, dass der Textindexname, der zum Zeitpunkt der Feststellung des Fehlers aufgrund der vollständig belegten Platte verwendet wurde, möglicherweise nicht wiederverwendet werden kann.
Sie erhalten die Nachricht CIE00301 "Keine ausreichende Berechtigung" oder CIE00377 "Nur der Instanzeigner <*instanzeigner*> kann diesen Befehl verwenden", nachdem Sie einen db2ts-Befehl **START FOR TEXT** abgesetzt haben.	Für Windows-Benutzer muss sichergestellt werden, dass der db2ts-Befehl **START FOR TEXT** von einem Benutzer mit ausreichender Berechtigung abgesetzt wird. Bei Linux und UNIX muss dieser Befehl durch den Instanzeigner abgesetzt werden.

Tabelle 85. Probleme und Lösungen (Forts.)

Problem	Lösung
Sie erhalten die Nachricht CIE00311, in der Sie darüber informiert werden, dass eine interne Datei nicht geöffnet werden konnte. Diese Nachricht kann auf ein fehlendes Konfigurationsverzeichnis oder darauf hinweisen, dass eine Datei fehlt oder beschädigt wurde. Dies kann z. B. auf einen Fehler wegen einer vollständig belegten Platte in dem Dateisystem, in dem sich `db2tss/config` befindet, oder auf ein Problem während des Backups des Verzeichnisses `db2tss/config` zurückzuführen sein.	Überprüfen Sie, ob die Instanz ein Verzeichnis `config` für die Textsuche aufweist. • Wenn das Verzeichnis `config` fehlt, dann stellen Sie sicher, dass DB2 Text Search installiert und konfiguriert wurde. • Wenn das Verzeichnis `config` sich an einer anderen Position befindet, dann fügen Sie einen symbolischen Link hinzu (UNIX). Gehen Sie wie folgt vor, wenn dieser Fehler durch eine fehlende oder beschädigte Datei verursacht wurde: • Wenn es sich bei der Datei um `ciedem.dat` handelt, dann können Sie sie durch Ausführung der folgenden Schritte erneut generieren: 1. Setzen Sie den Befehl **db2ts STOP FOR TEXT** ab. 2. Suchen Sie die Datei `ciedem.dat`. Sie befindet sich normalerweise in `instancehome/sqllib/db2tss/config` (UNIX) oder in `instanceprofilepath\instancename\db2tss\config` (Windows). Sie können den Instanzprofilpfad durch Eingabe von **db2set DB2INSTPROF** suchen. 3. Wenn die Datei vorhanden ist, dann führen Sie ein Backup für sie durch und entfernen Sie sie anschließend. 4. Setzen Sie den Befehl **db2ts START FOR TEXT** ab. (Die interne Datei wird dann als eine Datei mit einer Größe von 0 Byte erneut generiert.) 5. Setzen Sie den Befehl **db2ts CLEANUP FOR TEXT** ab. • Wenn es sich nicht um die Datei `ciedem.dat` handelt, dann wenden Sie sich an den IBM Support.

Tabelle 85. Probleme und Lösungen (Forts.)

Problem	Lösung
Sie erhalten die Nachricht CIE00445N, in der Sie darüber informiert werden, dass die angeforderte Operation nicht ausgeführt werden kann. Führen Sie **REBIND** aus. Diese Nachricht kann darauf hinweisen, dass durch einen der vorherigen DB2-Befehle das Paket "NULLID.SYSSH100, NULLID.SYSSH200, NULLID.SYSSN100, NULLID.SYSSN200 oder NULLID.SYSLH202", das zur Ausführung von DB2 Text Search benötigt wird, inaktiviert wurde.	Der Benutzer muss das Paket "NULLID.SYSSH100, NULLID.SYSSH200, NULLID.SYSSN100, NULLID.SYSSN200 oder NULLID.SYSLH202" manuell erneut binden, bevor db2ts-Befehle weiter ausgeführt werden können. Die Inaktivierung des Pakets tritt normalerweise dann auf, wenn Befehle zum Widerrufen oder Gewähren ausgeführt werden. Der Benutzer kann das Auftreten dieses Fehlers verhindern, indem er den Paketstatus anhand der folgenden SQL-Anweisung überprüft: `select 1 from syscat.packages where VALID = 'N' AND ((PKGSCHEMA='NULLID' AND PKGNAME= 'SYSSH100') OR (PKGSCHEMA='NULLID' AND PKGNAME= 'SYSSH200') OR (PKGSCHEMA='NULLID' AND PKGNAME= 'SYSSN100') OR (PKGSCHEMA='NULLID' AND PKGNAME= 'SYSSN200') OR (PKGSCHEMA='NULLID' AND PKGNAME= 'SYSLH202')) FETCH FIRST 1 ROWS ONLY FOR READ ONLY;` Wenn der Rückgabewert für diese SQL-Anweisung 1 lautet, dann muss der Benutzer das Paket NULLID.SYSSH100, NULLID.SYSSH200, NULLID.SYSSN100, NULLID.SYSSN200 oder NULLID.SYSLH202 erneut binden, bevor er die Befehle von DB2 Text Search ausführt.

Kapitel 9. Durchsuchen von Wissensbasen

Effektives Suchen nach bekannten Problemen

Es steht eine Vielzahl von Ressourcen, in denen bekannte Probleme beschrieben werden, zur Verfügung, wie beispielsweise DB2 APARs, Whitepapers, IBM Redbooks, technische Hinweise (Technotes) und Handbücher. Das effektive Durchsuchen dieser (und anderer) Ressourcen ist wichtig, um schnell festzustellen, ob für das aufgetretene Problem bereits eine Lösung vorhanden ist.

Bevor Sie mit der Suche beginnen, sollten Sie eine konkrete Vorstellung von der Problemsituation haben.

Wenn Sie sich ein klares Bild von der Problemsituation verschafft haben, erstellen Sie eine Liste mit Suchbegriffen, die die Chance, vorhandene Lösungen zu finden, vergrößern. Nachfolgend sind einige Tipps aufgeführt:

1. Verwenden Sie mehrere Wörter für die Suche. Je zutreffender die verwendeten Suchbegriffe sind, desto besser sind die Suchergebnisse.
2. Beginnen Sie mit speziellen Ergebnissen und erweitern Sie diese bei Bedarf. Reichen beispielsweise die Ergebnisse nicht aus, entfernen Sie einige der weniger relevanten Suchbegriffe, und wiederholen Sie die Suche. Wenn Sie nicht sicher sind, welche Suchbegriffe Sie verwenden sollen, können Sie alternativ dazu auch eine weniger spezifische Suche mit wenigen Suchbegriffen durchführen, die so ermittelten Ergebnisse auswerten und daraufhin eine genauere Auswahl weiterer Suchbegriffe treffen.
3. In manchen Fällen ist eine Suche nach einem bestimmten Ausdruck effektiver. Wenn Sie beispielsweise "Benachrichtigungsdatei für Systemverwaltung" (inclusive Anführungszeichen) eingeben, erhalten Sie nur die Dokumente, die den genauen Ausdruck in der eingegebenen Reihenfolge der Wörter enthalten. (Im Gegensatz zu allen Dokumenten, die eine beliebige Kombination dieser drei Wörter enthalten.)
4. Verwenden Sie Platzhalter. Wenn ein bestimmter SQL-Fehler auftritt, suchen Sie nach `SQL5005<platzhalter>`", wobei `<platzhalter>` die durchsuchte Ressource angibt. Auf diese Weise erhalten Sie wahrscheinlich mehr Ergebnisse als bei der Suche nach `SQL5005`" oder "`SQL5005c`".
5. In Situationen, in denen die verwendete Instanz abnormal beendet wird und Trapdateien erstellt werden, suchen Sie nach bekannten Problemen, indem Sie die ersten zwei oder drei Funktionen im Stack-Traceback der Trap- oder Kerndatei verwenden. Wenn zu viele Ergebnisse zurückgegeben werden, fügen Sie Suchbegriffe wie "`Trap`", "`Abend`" oder "`Absturz`" hinzu.
6. Wenn Sie nach betriebssystemspezifischen Suchbegriffen suchen (z. B. Signalnummern oder Werten für Fehlernummern), suchen Sie nach dem Konstantennamen, nicht nach dem Wert. Suchen Sie beispielsweise nach "EFBIG", nicht nach der Fehlernummer 27.

Erfolg versprechende Suchbegriffe enthalten häufig die folgenden Elemente:

- Wörter, die den ausgeführten Befehl beschreiben
- Wörter, die die Symptome beschreiben
- Token des Diagnoseprogramms

Ressourcen zur Fehlerbehebung

Eine breite Palette verschiedener Informationen zur Fehlerbehebung steht zur Verfügung, um Sie bei der Verwendung von DB2-Datenbankprodukten zu unterstützen.

DB2-Dokumentation

Informationen zur Fehlerbehebung stehen in der gesamten *DB2-Informationszentrale* sowie in den PDF-Büchern der DB2-Bibliothek zur Verfügung.

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 zur technischen Unterstützung enthält Links zu den neuesten DB2-Veröffentlichungen, technischen Hinweisen (TechNotes), APARs (Authorized Program Analysis Reports), Fixpacks sowie weiteren Ressourcen. Sie können diese Wissensbasis nach möglichen Lösungen für aufgetretene Probleme durchsuchen.

Rufen Sie die DB2-Website zur technischen Unterstützung unter www.ibm.com/software/data/db2/support/db2_9/ auf.

Kapitel 10. Abrufen von DB2-Produktkorrekturen

Fixpacks enthalten Codeaktualisierungen und Korrekturen für Probleme, die IBM beim Testen des Produkts festgestellt hat bzw. die Kunden bei der Verwendung des Produkts ermittelt haben. Es wird erläutert, wie Sie die neuesten Fixpacks abrufen und die Korrekturen dann auf Ihre Datenbankumgebung anwenden können.

Abrufen von Programmkorrekturen (Fixes)

Möglicherweise können Sie das bei Ihnen aufgetretene Problem mit einer für das Produkt verfügbaren Programmkorrektur lösen. Rufen Sie Programmkorrekturen wie in diesem Abschnitt angegeben ab.

Informationen zu diesem Vorgang

1. Eine Liste der Programmkorrekturen sowie Links zum Abrufen von Fixpacks finden Sie auf den entsprechenden Webseiten, die Sie über die folgenden Webseiten aufrufen können:
   - Unterstützung für DB2 9 for Linux, UNIX und Windows
   - Programmkorrekturen für die verschiedenen Versionen von DB2 für Linux, UNIX und Windows
2. Ermitteln Sie zunächst, welches Fixpack Sie benötigen. Im Allgemeinen empfiehlt es sich, das aktuellste Fixpack zu installieren, um Fehler durch bereits bekannte und korrigierte Softwarefehler zu vermeiden.
3. Laden Sie das Fixpack herunter und extrahieren Sie die Dateien, indem Sie das sich selbst entpackende Paket doppelt anklicken. Öffnen Sie das Dokument `SERVER/doc/`*`sprache`*`/readme.txt` und rufen Sie die Installationsanweisungen über den angegebenen Link für die DB2-Informationszentrale ab.
4. Wenden Sie die Programmkorrektur an. Anweisungen finden Sie in „Anwenden von Fixpacks" im Handbuch *DB2-Server - Installation*.

Fixpacks, vorläufige Fixpacks und Testfixes

Ein APAR (Authorized Program Analysis Report) ist ein formaler Bericht über ein Problem, das durch einen vermuteten Fehler in einem aktuellen, ungeänderten Release eines IBM Programms verursacht wird. APARs beschreiben Probleme, die während Tests durch IBM festgestellt werden, sowie Probleme, die von Kunden gemeldet werden.

Der modifizierte DB2-Code, durch den das im APAR beschriebene Problem behoben wird, kann im Rahmen von Fixpacks, vorläufigen Fixpacks oder Testfixes bereitgestellt werden.

Fixpack

Ein Fixpack ist eine kumulative Sammlung von APAR-Korrekturen. Genauer: Fixpacks beziehen sich auf die APARs, die zwischen neuen Releases von DB2 anfallen. Sie sollen das Implementieren einer bestimmten Wartungsstufe ermöglichen. Fixpacks weisen die folgenden Merkmale auf:

- Sie sind kumulativ. Fixpacks für ein bestimmtes DB2-Release ersetzen bzw. enthalten alle APAR-Korrekturen, die in vorhergehenden Fixpacks und vorläufigen Fixpacks für das betreffende Release bereitgestellt wurden.

- Sie sind für alle unterstützten Betriebssysteme und DB2-Datenbankprodukte verfügbar.
- Sie enthalten eine Vielzahl von APARs.
- Sie werden auf der Website mit technischer Unterstützung für DB2 veröffentlicht und sind für die Kunden, die Produkte im Rahmen des Passport Advantage-Programms bezogen haben, allgemein verfügbar.
- Sie wurden von IBM in vollem Umfang getestet.
- Sie umfassen Begleitdokumentation, die eine Beschreibung der an den Datenbankprodukten vorgenommenen Änderungen sowie Anweisungen zum Installieren und Entfernen des Fixpacks enthält.

Anmerkung: Der Status eines APARs wird von "open" (offen) in "Closed as program error" (als Programmfehler geschlossen) geändert, sobald die entsprechende APAR-Korrektur in einem Fixpack bereitgestellt wird. Sie können den Status einzelner APARs ermitteln, indem Sie die APAR-Beschreibungen auf der Website mit technischer Unterstützung für DB2 lesen.

Vorläufiges Fixpack

Ein vorläufiges Fixpack ist eine kumulative Sammlung von wichtigen APAR-Korrekturen zwischen den Fixpacks. Um in ein vorläufiges Fixpack aufgenommen zu werden, muss ein APAR potenziell weitreichende Konsequenzen haben oder auf andere Weise von Bedeutung sein. Infrage kommende APARs werden von Experten der technischen DB2-Unterstützungsfunktion ausgewertet und genehmigt. Vorläufige Fixpacks weisen die folgenden Merkmale auf:

- Sie sind kumulativ. Vorläufige Fixpacks für ein bestimmtes DB2-Release ersetzen bzw. enthalten alle APAR-Korrekturen, die in vorhergehenden Fixpacks und vorläufigen Fixpacks für das betreffende Release bereitgestellt wurden.
- Sie sind für einen Teil der unterstützten Betriebssysteme und DB2-Datenbankprodukte verfügbar.
- Sie enthalten normalerweise 20 bis 30 neue APARs.
- Sie werden auf der Website mit technischer Unterstützung für DB2 veröffentlicht und sind für die Kunden, die Produkte im Rahmen des Passport Advantage-Programms bezogen haben, allgemein verfügbar.
- Sie wurden von IBM in vollem Umfang getestet.
- Sie umfassen Begleitdokumentation, die eine Beschreibung zum Installieren und Entfernen des Fixpacks enthält.

Vorläufige Fixpacks werden in der Produktion für einen Zeitraum von zwei Jahren nach ihrer Veröffentlichung unterstützt. Sie werden etwa in der Mitte des Zeitraums zwischen zwei Fixpacks bereitgestellt und stellen die bevorzugte Alternative zu Testfixes dar, die nicht im selben Maße wie Fixpacks getestet und unterstützt werden.

Testfix

Ein Testfix ist eine temporäre Lösung, die bestimmten Kunden als Reaktion auf ein gemeldetes Problem zu Testzwecken zur Verfügung gestellt wird. Testfixes werden gelegentlich auch als 'spezielle Builds' bezeichnet und weisen folgende Merkmale auf:

- Sie enthalten normalerweise einen einzelnen APAR.
- Sie werden von der DB2-Unterstützungsfunktion bereitgestellt und sind nicht für alle Benutzer allgemein verfügbar.
- Sie werden von IBM in begrenztem Umfang getestet.

- Sie enthalten minimale Dokumentation, wie z. B. eine Beschreibung zur Anwendung des Testfix, Informationen zu dem/den zugehörigen APAR(s) sowie Anweisungen zur Entfernung des Testfix.

Testfixes werden in Situationen bereitgestellt, in denen ein neues Problem festgestellt wurde, es keine Ausweichlösung oder Problemumgehung gibt und nicht auf die Verfügbarkeit des nächsten Fixpacks bzw. des nächsten vorläufigen Fixpacks gewartet werden kann. Wenn das Problem beispielsweise kritische Auswirkungen auf Ihr Unternehmen hat, kann ein Testfix bereitgestellt werden, durch das die Situation entschärft wird, bis eine Lösung für den APAR in einem Fixpack bzw. einem vorläufigen Fixpack zur Verfügung steht

Es wird empfohlen, die DB2-Umgebung stets auf dem Stand des aktuellen Fixpacks zu halten, um einen fehlerfreien Betrieb sicherzustellen. Wenn Sie über die Verfügbarkeit neuer Fixpacks benachrichtigt werden möchten, melden Sie sich bei "My Notifications"-E-Mail-Aktualisierungen auf der Website mit technischer Unterstützung für DB2 unter http://www.ibm.com/software/data/db2/support/db2_9/ an.

Weitere Informationen zu der Rolle und dem Zweck von DB2-Fixes und -Fixpacks finden Sie im Abschnitt zur Unterstützung von Richtlinienanweisungen.

Anwenden von Testfixes

Ein Testfix ist eine temporäre Korrektur, die bestimmten Kunden als Reaktion auf ein gemeldetes Problem zu Testzwecken zur Verfügung gestellt wird. Jeder Testfix verfügt über eine Readme-Datei. Die Testfix-Readme enthält Anweisungen zur Installation und Deinstallation des Testfix sowie eine Liste der APARs (falls zutreffend), die im Testfix enthalten sind.

Vorbereitende Schritte

Für jeden Testfix gelten eigene Voraussetzungen. Genauere Informationen enthält die Readme-Datei, die zu dem betreffenden Testfix gehört.

Informationen zu diesem Vorgang

Es gibt zwei Arten von Testfixes:

- Ein Testfix für ein einzelnes DB2-Produkt. Dieser Testfix kann auf eine vorhandene Installation des Produkts angewendet werden, oder er kann verwendet werden, um eine vollständige Produktinstallation auszuführen, wenn noch keine DB2-Installation vorhanden ist.
- Universelle Testfixes (nur Linux und UNIX). Hierbei handelt es sich um einen universellen Testfix für Installationen, bei denen mehr als ein DB2-Produkt installiert wurde.

Wurden Landessprachen installiert, benötigen Sie darüber hinaus möglicherweise einen separaten Testfix für die Landessprache. Der Testfix für die Landessprache kann nur dann angewendet werden, wenn er dieselbe Testfixstufe aufweist wie das installierte DB2-Produkt. Bei der Anwendung eines universellen Testfix müssen Sie sowohl den universellen Testfix als auch den Testfix für die Landessprache anwenden, um die DB2-Produkte zu aktualisieren.

Vorgehensweise

Fordern Sie den Testfix bei IBM Software Support an und gehen Sie wie in der Readme-Datei angegeben vor, um den Testfix zu installieren, zu testen und (falls erforderlich) zu entfernen.
Beim Installieren eines Testfix in einer Umgebung mit partitionierten Datenbanken muss das System offline sein und für alle an der Instanz teilnehmenden Computer muss ein Upgrade auf dieselbe Testfixstufe durchgeführt werden.

Kapitel 11. Weitere Informationen zur Fehlerbehebung

An einem bestimmten Punkt kann es bei der Arbeit mit DB2-Datenbankprodukten möglicherweise zu einem Problem kommen. Dieses Problem kann vom Datenbankmanager, von einer Anwendung, die für die Datenbank ausgeführt wird, oder von Ihren Benutzern im Rahmen des an Sie gerichteten Feedbacks über ein merkwürdiges Funktionsverhalten der Datenbank gemeldet werden.

Die hier vorgestellten Konzepte und Tools sollen Sie mit der Behebung von Problemen jeglicher Art bei Datenbankoperationen vertraut machen und Sie beim Prozess der Problembehebung unterstützen. Die Wichtigkeit der Erfassung der richtigen Daten zum richtigen Zeitpunkt wird hervorgehoben; FODC wird daher als erstes Tool erläutert. Andere Protokolle und Dateien, die vom Datenbankmanager zum Erfassen von Daten über die Operationen der Datenbank verwendet werden, werden zusammen mit den Diagnosetools des Betriebssystems erläutert.

Informationen zu verschiedenen Themen

Diese Abschnitte vermitteln Ihnen Informationen zu Konzepten und Begriffen, die bei einer effizienten Behebung von Fehlern, die im Zusammenhang mit dem DB2-Produkt auftreten, von Interesse sind.

- Fehlerbehebung

  Die Fehlerbehebung erfordert eine systematische Vorgehensweise. Ziel ist es dabei herauszufinden, warum etwas nicht wie erwartet funktioniert und wie sich das Problem lösen lässt.

- Informationen zum Verzeichnispfad für Diagnosedaten

  Abhängig von der jeweiligen Plattform befinden sich die DB2-Diagnoseinformationen, die in Speicherauszugsdateien, Trapdateien, Diagnoseprotokolldateien, Protokolldateien mit Benachrichtigungen für die Systemverwaltung, Alertprotokolldateien und FODC-Paketen (FODC = First Occurrence Data Collection, Datenerfassung beim ersten Vorkommen) enthalten sind, im Verzeichnispfad für Diagnosedaten, der durch den Konfigurationsparameter **diagpath** des Datenbankmanagers angegeben wird.

- Protokoll mit Benachrichtigungen für die Systemverwaltung

  Der DB2-Datenbankmanager schreibt die folgenden Arten von Informationen in das Protokoll mit Benachrichtigungen für die Systemverwaltung: den Status von DB2-Dienstprogrammen wie **REORG** und **BACKUP**, Fehler in Clientanwendungen, Änderungen von Serviceklassen, Lizenzierungsvorgänge, Protokolldateipfade und Speicherprobleme, Überwachungs- und Indexierungsvorgänge sowie Tabellenbereichsprobleme. Ein Datenbankadministrator kann diese Informationen zum Diagnostizieren von Problemen, zur Optimierung der Datenbank sowie zur Überwachung der Datenbank verwenden.

- Informationen zu DB2-Diagnoseprotokolldateien (db2diag)

  Da auch die Nachrichten des Protokolls mit Benachrichtigungen für die Systemverwaltung in einem standardisierten Nachrichtenformat in die **db2diag**-Protokolldateien protokolliert werden, ist es empfehlenswert die **db2diag**-Protokolldateien zuerst anzuzeigen, um sich einen Überblick über die für die Datenbank relevanten Vorgänge zu verschaffen.

- Plattformspezifische Fehlerprotokolle

  Auch außerhalb von DB2 steht eine Vielzahl weiterer Dateien und Dienstprogramme zur Verfügung, die Sie bei der Fehleranalyse unterstützen. Häufig sind

sie für die Ermittlung der eigentlichen Fehlerursache ebenso wichtig wie die Informationen, die in den DB2-Dateien zur Verfügung gestellt werden.

- Informationen zu Nachrichten

  Die Informationen zu den Nachrichten erleichtern Ihnen das Eingrenzen von Fehlern bzw. Problemen sowie das Beheben eines Problems mit entsprechenden Recoverymaßnahmen. Diese Informationen können auch verwendet werden, um sich darüber zu informieren, wo Nachrichten generiert und protokolliert werden.

- Informationen zu internen Rückkehrcodes

  Es gibt zwei Arten interner Rückkehrcodes: ZRC-Werte und ECF-Werte. Sie werden in der DB2-Traceausgabe und in den **db2diag**-Protokolldateien angezeigt. ZRC- und ECF-Werte sind normalerweise negative Zahlen und stellen Fehlerbedingungen dar.

- Speicherauszugsdateien

  Speicherauszugsdateien (engl. Dump Files) werden erstellt, wenn ein Fehler auftritt, für den zusätzliche Informationen verfügbar sind, die bei der Diagnose eines Problems nützlich sein könnten (z. B. interne Steuerblöcke). Jedem Datenelement, das in die Speicherauszugsdateien geschrieben wird, wird zur Unterstützung der Problembestimmung eine Zeitmarke zugeordnet. Speicherauszugsdateien liegen im Binärformat vor und sind für Ansprechpartner der DB2-Kundenunterstützung gedacht.

- Trapdateien

  DB2 generiert eine Trapdatei, falls DB2 die Verarbeitung aufgrund einer Fehlerunterbrechung (Trap), einer Segmentierungsverletzung oder einer Ausnahmebedingung nicht fortsetzen kann. Alle Signale oder Ausnahmebedingungen, die von DB2 empfangen werden, werden in der Trapdatei aufgezeichnet. Die Trapdatei enthält außerdem die Funktionsfolge, die aktiv war, als der Fehler aufgetreten ist. Diese Folge wird manchmal auch als "Funktionsaufrufstack" (engl. function call stack) oder "Stack-Trace" bezeichnet. Die Trapdatei enthält darüber hinaus Informationen zum Status des Prozesses, als das Signal oder die Ausnahmebedingung aufgefangen wurde.

- Informationen zu FODC (First Occurrence Data Capture)

  First Occurrence Data Capture (FODC) ist der Prozess, der für die Erfassung von Daten zu einer DB2-Instanz auf Basis von Szenarien verwendet wird. FODC kann manuell von einem DB2-Benutzer auf Grundlage eines bestimmten Symptoms aufgerufen werden. FODC kann auch automatisch aufgerufen werden, wenn ein vordefiniertes Szenario oder Symptom ermittelt wird. Diese Informationen reduzieren die Notwendigkeit, Fehler für Diagnoseinformationen zu reproduzieren.

- Informationen zu Ausgabedateien des Aufrufscripts (db2cos)

  Das Script 'db2cos' wird standardmäßig aufgerufen, wenn der Datenbankmanager die Verarbeitung aufgrund einer Panic-Situation, eines Traps, einer Segmentierungsverletzung oder einer Ausnahmebedingung nicht fortsetzen kann.

- Kombinieren von Diagnoseprogrammen von DB2 und des Betriebssystems

  Das Diagnostizieren von Problemen im Hinblick auf Hauptspeicher, Auslagerungsdateien, CPUs, Plattenspeicher und andere Ressourcen erfordert gründliche Kenntnisse darüber, wie das betreffende Betriebssystem die entsprechenden Ressourcen verwaltet. Die Definition eines ressourcenbezogenen Problems erfordert als Minimum Kenntnisse darüber, wie viel einer Ressource vorhanden ist und welche Ressourcengrenzen pro Benutzer gelten.

Verzeichnispfad für Diagnosedaten

DB2-Diagnoseinformationen, die in einer Speicherauszugsdatei, einer Trapdatei, einer Diagnoseprotokolldatei, einer Protokolldatei mit Benachrichtigungen für die Systemverwaltung, einer Alertprotokolldatei und einem FODC-Paket enthalten sind, befinden sich im Diagnosedatenverzeichnis, das im Konfigurationsparameter **diagpath** des Datenbankmanagers oder alternativ im Konfigurationsparameter **alt_diagpath** des Datenbankmanagers angegeben ist.

Übersicht

Mit der Angabe des Verzeichnispfads für Diagnosedaten oder des alternativen Verzeichnispfads für Diagnosedaten mithilfe des Konfigurationsparameters **diagpath** oder **alt_diagpath** des Datenbankmanagers kann bestimmt werden, welche der folgenden Verzeichnispfadmethoden für die Speicherung der Diagnosedaten verwendet werden kann.

Primärer Verzeichnispfad für Diagnosedaten
Alle Diagnosedaten für die DB2-Instanz werden in einem einzelnen Verzeichnis gespeichert, unabhängig davon, ob die Datenbank partitioniert ist. In einer Umgebung mit partitionierten Datenbanken werden die Diagnosedaten von verschiednen Partitionen innerhalb des Hosts alle in diesem einen Verzeichnispfad für Diagnosedaten gespeichert. Dieser eine Verzeichnispfad für Diagnosedaten gilt standardmäßig, wenn als Wert für **diagpath** null oder ein beliebiger gültiger Pfadname ohne die Musterkennung `$h` oder `$n` definiert wird.

Alternativer Verzeichnispfad für Diagnosedaten
Der Konfigurationsparameter **alt_diagpath** des Datenbankmanagers ist ein alternativer Verzeichnispfad für Diagnosedaten, der als sekundärer Pfad für die Speicherung von Diagnoseinformationen dient. Der durch den Parameter **alt_diagpath** angegebene Pfad wird nur verwendet, wenn der Datenbankmanager in den durch **diagpath** angegebenen Pfad keine Daten schreiben kann. Hierdurch wird sichergestellt, dass wichtige Diagnoseinformationen nicht verloren gehen. Damit der alternative Verzeichnispfad für Diagnosedaten verfügbar ist, müssen Sie den Konfigurationsparameter **alt_diagpath** festlegen. Zur Gewährleistung einer höheren Ausfallsicherheit empfiehlt es sich, für diesen Parameter einen Pfad zu definieren, der sich auf einem anderen Dateisystem befindet als der mit **diagpath** angegebene Pfad.

Aufgeteilte Verzeichnispfade für Diagnosedaten
In Umgebungen mit partitionierten Datenbanken können Diagnosedaten separat jeweils in einem Verzeichnis gespeichert werden, das dem Host und/oder der Datenbankpartition entsprechend benannt ist. So enthält jeder Diagnosedateityp innerhalb eines bestimmten Diagnoseverzeichnisses nur Diagnoseinformationen von jeweils einem Host und/oder jeweils einer Datenbankpartition. Zum Aufteilen des Verzeichnispfads für Diagnosedaten und des alternativen Verzeichnispfads für Diagnosedaten müssen Sie die Prozedur zum Aufteilen ein Mal für den Parameter **diagpath** und dann nochmals für den Parameter **alt_diagpath** ausführen.

Vorteile

Die Angabe von Verzeichnispfaden für Diagnosedaten bietet die folgenden Vorteile:

- Diagnoseinformationen von mehreren Datenbankpartitionen und Hosts können an einer zentralen Position konsolidiert und so einfach zugänglich gemacht werden, indem ein einziger Verzeichnispfad für Diagnosedaten definiert wird.

- Die Leistung beim Protokollieren von Diagnosedaten kann verbessert werden, da es bei der Protokolldatei **db2diag** zu weniger Konkurrenzsituationen kommt, wenn Sie den Verzeichnispfad der Diagnosedaten nach Host oder Datenbankpartition aufteilen.

Die Angabe eines sekundären Pfads für Diagnosedaten (**alt_diagpath**) hat folgende Vorteile:

- Erhöhte Ausfallsicherheit wichtiger Diagnoseinformationen
- Kompatibilität mit einigen Tools, die für den Parameter **diagpath** verwendet werden, wie Aufteilung

Zusammenfügen von Dateien und Sortieren von Datensätzen

Das Zusammenfügen und Sortieren von Datensätzen mehrerer Diagnosedateien desselben Typs auf der Basis von Zeitmarken kann im Fall eines aufgeteilten Verzeichnispfads für Diagnosedaten mit dem Befehl **db2diag -merge** durchgeführt werden. Weitere Informationen hierzu finden Sie im Abschnitt „db2diag - Analysetool für db2diag-Protokolle (Befehl)" in der Veröffentlichung *Command Reference*.

Speicherbedarf für Diagnosedaten

Die Erfassung von Diagnosedaten in dem durch den Parameter **diagpath** angegebenen Pfad kann große Mengen an Diagnoseinformationen generieren, vor allem, wenn Kerndateispeicherauszüge und FODC-Daten nicht in einen separaten Verzeichnispfad umgeleitet werden oder wenn eine einzelne `db2diag.log`-Datei ohne Begrenzung der Größe verwendet wird. Es muss ausreichend Speicherplatz für die Diagnosedaten zur Verfügung stehen und der Diagnosepfad muss sorgfältig verwaltet werden, damit stets genügend Speicherplatz verfügbar ist.

Die folgenden Empfehlungen können bei der Konfiguration der Diagnosedatenprotokollierung auf dem Datenserver verwendet werden, um sicherzustellen, dass die Voraussetzungen hinsichtlich des Speicherbedarfs für die Diagnosedaten erfüllt sind:

Sorgen Sie dafür, dass die Mindestvoraussetzungen hinsichtlich des Speicherbedarfs für Diagnosedaten erfüllt sind.

Die Menge des im Diagnoseverzeichnispfad verfügbaren freien Speichers sollte mindestens doppelt so groß sein wie der auf der Maschine installierte physische Speicher (Minimum freier Speicher = 2mal physischer Speicher). Wenn auf einer Maschine beispielsweise 64 GB physischer Speicher installiert sind, sollten im Dateisystem mindestens 128 GB Speicherplatz für Diagnosedaten verfügbar sein.

Leiten Sie Kerndateispeicherauszüge und FODC-Daten in einen anderen Verzeichnispfad um.

Sowohl Kerndateispeicherauszüge als auch FODC-Daten können in kurzer Zeit eine erhebliche Menge an Plattenspeicherplatz verbrauchen und senden standardmäßig Daten an den Verzeichnispfad, der durch den Konfigurationsparameter **diagpath** des Datenbankmanagers angegeben ist. Sie können Kerndateispeicherauszüge und FODC-Daten in einen anderen Verzeichnispfad oder in ein anderes Dateisystem umleiten, um im Diagnoseverzeichnispfad mehr Speicherplatz zur Verfügung zu haben. Sie können mit der Registrierdatenbankvariablen *DB2FODC* steuern, wo Kerndateien generiert werden, indem Sie die Variable *DUMPDIR* so definieren, dass sie auf einen Verzeichnispfad verweist, der nicht mit **diagpath** identisch ist. Auf ähnliche Weise können Sie steuern, wo FODC-Paketverzeichnisse er-

stellt werden, indem Sie die Variable *FODCPATH* so definieren, dass sie auf einem anderen Verzeichnispfad verweist.

Verschieben oder entfernen Sie Dateien, die nicht mehr benötigt werden.
Wenn Sie den Befehl **db2support** ausführen, ohne einen Pfad anzugeben, der von **diagpath** abweicht, wird das erstellte komprimierte Archiv im Diagnoseverzeichnispfad gespeichert. Nachdem Sie die Datei an IBM hochgeladen haben, müssen Sie das komprimierte Archiv aus dem Diagnoseverzeichnispfad entfernen, da sie andernfalls weiter den verfügbaren Plattenspeicherplatz belegt.

Konfigurieren Sie rollierende Diagnoseprotokolle und Archivprotokolldateien.
Standardmäßig wird bei der Verwendung einer einzigen Datei `db2diag.log` die DB2-Diagnoseprotokolldatei unbegrenzt größer. Wenn Sie die Verwendung rollierender Diagnoseprotokolle konfigurieren, indem Sie den Konfigurationsparameter **diagsize** des Datenbankmanagers entsprechend definieren, werden eine Reihe rollierender Diagnoseprotokolldateien und eine Reihe rollierender Protokolldateien mit Benachrichtigungen für die Systemverwaltung verwendet, die der mit **diagsize** festgelegten Größe entsprechen. Während die Protokolldateien mit Daten gefüllt werden, werden die ältesten Dateien gelöscht und neue erstellt. Um zu vermeiden, dass aufgrund des rollierenden Verfahrens (d. h. durch das Löschen der ältesten Protokolldatei) Informationen zu schnell verloren gehen, definieren Sie für **diagsize** einen Wert, der über 50 MB liegt, aber nicht mehr als 80 % des freien Speicherplatzes beträgt, den Sie mit den Parametern **diagpath** und **alt_diagpath** angeben. Sie können die Dateien des rollierenden Diagnoseprotokolls auch speichern, indem Sie sie mit dem Befehl **db2diag -archive** aus dem Pfad **diagpath** archivieren.

Konfigurieren Sie einen alternativen Diagnosepfad.
Als Absicherung gegen den Verlust wichtiger Diagnoseinformationen können Sie mithilfe des Konfigurationsparameters **alt_diagpath** des Datenbankmanagers einen alternativen Verzeichnispfad für Diagnosedaten angeben, in dem Sie Diagnoseinformationen speichern. Wenn der Datenbankmanager keine Daten in den mit **diagpath** angegebenen Pfad schreiben kann, wird der mit **alt_diagpath** angegebene Pfad zum Speichern der Diagnoseinformationen verwendet, bis **diagpath** wieder verfügbar ist. Eine noch höhere Ausfallsicherheit erreichen Sie, wenn der Parameter **alt_diagpath** auf ein anderes Dateisystem verweist als der Parameter **diagpath**.

Aufteilen eines Verzeichnispfads für Diagnosedaten nach Datenbankpartitionsservern und/oder Datenbankpartitionen

Mit der Standardeinstellung für den Verzeichnispfad für DB2-Diagnosedaten, die für den Konfigurationsparameter **diagpath** des Datenbankmanagers gilt, werden alle Diagnoseinformationen in einem einzigen Diagnosedatenverzeichnis gesammelt. Sie können den Verzeichnispfad für Diagnosedaten aufteilen, sodass separate Verzeichnisse dem Datenbankpartitionsserver und/oder der Datenbankpartition entsprechend erstellt und benannt werden. Auf diese Weise werden Diagnosespeicherauszugsdateien, die zuvor in einem einzigen Verzeichnis gespeichert wurden, nun in separaten Verzeichnissen gespeichert, die dem Datenbankpartitionsserver bzw. der Datenbankpartition entsprechen, von denen der Diagnosedatenspeicherauszug stammt.

Vorbereitende Schritte

DB2 Version 9.7 Fixpack 1 oder ein neueres Fixpack ist erforderlich.

Informationen zu diesem Vorgang

Sie können einen Verzeichnispfad für Diagnosedaten aufteilen, sodass Diagnoseinformationen dem Datenbankpartitionsserver bzw. der Datenbankpartition, von dem/der der Diagnosedatenspeicherauszug stammt, entsprechend separat gespeichert werden.

Einschränkungen

Das Aufteilen eines Verzeichnispfads für Diagnosedaten, um die verschiedenen Quellen der Diagnoseinformationen gegeneinander abzugrenzen, ist vor allem in Umgebungen mit partitionierten Datenbanken nützlich.

Vorgehensweise

- **Aufteilen eines Verzeichnispfads für Diagnosedaten nach physischem Datenbankpartitionsserver**
  - Führen Sie den folgenden Schritt aus, um den Standardverzeichnispfad für Diagnosedaten aufzuteilen:
    - Definieren Sie den Konfigurationsparameter **diagpath** des Datenbankmanagers so, dass der Standardverzeichnispfad für Diagnosedaten auf der Basis des physischen Datenbankpartitionsservers aufgeteilt wird, indem Sie den folgenden Befehl eingeben:

      ```
      db2 update dbm cfg using diagpath '"$h"'
      ```

      Mit diesem Befehl wird ein Unterverzeichnis unter dem Standardverzeichnis für Diagnosedaten mit dem Computernamen erstellt, wie im folgenden Beispiel dargestellt:

      standard-diagpfad/HOST_*name_des_datenbankpartitionsservers*
  - Führen Sie den folgenden Schritt aus, um einen benutzerdefinierten Diagnoseverzeichnispfad, zum Beispiel /home/usr1/db2dump/, aufzuteilen:
    - Definieren Sie den Konfigurationsparameter **diagpath** des Datenbankmanagers so, dass der Verzeichnispfad /home/usr1/db2dump/ für Diagnosedaten auf der Basis des Datenbankpartitionsservers aufgeteilt wird, indem Sie den folgenden Befehl eingeben:

      ```
      db2 update dbm cfg using diagpath '"/home/usr1/db2dump/ $h"'
      ```

      Anmerkung: Zwischen /home/usr1/db2dump/ und $h muss ein Leerzeichen stehen.

      Mit diesem Befehl wird ein Unterverzeichnis unter dem Verzeichnis /home/usr1/db2dump/ für Diagnosedaten mit dem Namen des Datenbankpartitionsservers erstellt, wie im folgenden Beispiel dargestellt:

      /home/usr1/db2dump/HOST_*name_des_datenbankpartitionsservers*
- **Aufteilen eines Verzeichnispfads für Diagnosedaten nach Datenbankpartition**
  - Führen Sie den folgenden Schritt aus, um den Standardverzeichnispfad für Diagnosedaten aufzuteilen:
    - Definieren Sie den Konfigurationsparameter **diagpath** des Datenbankmanagers so, dass der Standardverzeichnispfad für Diagnosedaten auf der Basis der Datenbankpartition aufgeteilt wird, indem Sie den folgenden Befehl eingeben:

      ```
      db2 update dbm cfg using diagpath '"$n"'
      ```

      Mit diesem Befehl wird ein Unterverzeichnis für jede Partition unter dem Standardverzeichnis für Diagnosedaten mit der jeweiligen Partitionsnummer erstellt, wie im folgenden Beispiel dargestellt:

standard-diagpfad/NODE*nummer*

– Führen Sie den folgenden Schritt aus, um einen benutzerdefinierten Diagnoseverzeichnispfad, zum Beispiel /home/usr1/db2dump/, aufzuteilen:

- Definieren Sie den Konfigurationsparameter **diagpath** des Datenbankmanagers so, dass der Verzeichnispfad /home/usr1/db2dump/ für Diagnosedaten auf der Basis der Datenbankpartition aufgeteilt wird, indem Sie den folgenden Befehl eingeben:

```
db2 update dbm cfg using diagpath '"/home/usr1/db2dump/ $n"'
```

Anmerkung: Zwischen /home/usr1/db2dump/ und $n muss ein Leerzeichen stehen.

Mit diesem Befehl wird ein Unterverzeichnis für jede Partition unter dem Verzeichnis /home/usr1/db2dump/ für Diagnosedaten mit der jeweiligen Partitionsnummer erstellt, wie im folgenden Beispiel dargestellt:

/home/usr1/db2dump/NODE*nummer*

- **Aufteilen eines Verzeichnispfads für Diagnosedaten nach physischem Datenbankpartitionsserver und nach Datenbankpartition**

– Führen Sie den folgenden Schritt aus, um den Standardverzeichnispfad für Diagnosedaten aufzuteilen:

- Definieren Sie den Konfigurationsparameter **diagpath** des Datenbankmanagers so, dass der Standardverzeichnispfad für Diagnosedaten auf der Basis des physischen Datenbankpartitionsservers und der Datenbankpartition aufgeteilt wird, indem Sie den folgenden Befehl eingeben:

```
db2 update dbm cfg using diagpath '"$h$n"'
```

Mit diesem Befehl wird ein Unterverzeichnis für jede logische Partition auf dem Datenbankpartitionsserver unter dem Standardverzeichnis für Diagnosedaten erstellt und dem jeweiligen Namen des Datenbankpartitionsservers sowie der jeweiligen Partitionsnummer entsprechend benannt, wie im folgenden Beispiel dargestellt:

standard-diagpfad/HOST_*name_des_datenbankpartitionsservers*/NODE*nummer*

– Führen Sie den folgenden Schritt aus, um einen benutzerdefinierten Diagnoseverzeichnispfad, zum Beispiel /home/usr1/db2dump/, aufzuteilen:

- Definieren Sie den Konfigurationsparameter **diagpath** des Datenbankmanagers so, dass der Verzeichnispfad /home/usr1/db2dump/ für Diagnosedaten auf der Basis des Datenbankpartitionsservers und der Datenbankpartition aufgeteilt wird, indem Sie den folgenden Befehl eingeben:

```
db2 update dbm cfg using diagpath '"/home/usr1/db2dump/ $h$n"'
```

Anmerkung: Zwischen /home/usr1/db2dump/ und hn muss ein Leerzeichen stehen.

Mit diesem Befehl wird ein Unterverzeichnis für jede logische Partition auf dem Datenbankpartitionsserver unter dem Verzeichnis /home/usr1/db2dump/ für Diagnosedaten erstellt und dem jeweiligen Namen des Datenbankpartitionsservers sowie der jeweiligen Partitionsnummer entsprechend benannt, wie im folgenden Beispiel dargestellt:

/home/usr1/db2dump/HOST_*name_des_datenbankpartitionsservers*/NODE*nummer*

Beispiel: Ein AIX-Datenbankpartitionsserver mit dem Namen boson hat drei Datenbankpartitionen mit den Knotennummern 0, 1 und 2. Eine Listenausgabe des Verzeichnisses könnte z. B. wie folgt aussehen:

```
usr1@boson /home/user1/db2dump->ls -R *
HOST_boson:
```

```
HOST_boson:
NODE0000 NODE0001 NODE0002

HOST_boson/NODE0000:
db2diag.log db2eventlog.000 db2resync.log db2sampl_Import.msg events usr1.nfy

HOST_boson/NODE0000/events:
db2optstats.0.log

HOST_boson/NODE0001:
db2diag.log db2eventlog.001 db2resync.log usr1.nfy stmmlog

HOST_boson/NODE0001/stmmlog:
stmm.0.log

HOST_boson/NODE0002:
db2diag.log db2eventlog.002 db2resync.log usr1.nfy
```

Nächste Schritte

Anmerkung:

- Um zu überprüfen, ob die Einstellung des Verzeichnispfads für Diagnosedaten erfolgreich aufgeteilt wurde, führen Sie den folgenden Befehl aus:

  ```
  db2 get dbm cfg | grep DIAGPATH
  ```

  Ein erfolgreich aufgeteilter Verzeichnispfad für Diagnosedaten gibt die Werte `$h`, `$n` oder `$h$n` mit führendem Leerzeichen zurück. Die zurückgegebene Ausgabe sieht etwa wie folgt aus:

  ```
  Verzeichnispfad für Diagnosedaten            (DIAGPATH) = /home/usr1/db2dump/ $h$n
  ```

Mit dem Befehl **db2diag -merge** können Sie separate **db2diag**-Protokolldateien zusammenfügen, um die Analyse und Fehlerbehebung zu vereinfachen. Weitere Informationen hierzu finden Sie im Abschnitt „db2diag - Analysetool für db2diag-Protokolle (Befehl)" in der Veröffentlichung *Command Reference* sowie im Abschnitt „Analysieren der db2diag-Protokolldateien mit dem Tool 'db2diag'" auf Seite 494.

Protokoll mit Benachrichtigungen für die Systemverwaltung

Das Protokoll mit Benachrichtigungen für die Systemverwaltung (*instanzname*.nfy) ist das Repository, dem Informationen zu zahlreichen Aktivitäten der Datenbankverwaltung und -wartung zu entnehmen sind. Ein Datenbankadministrator kann diese Informationen zur Diagnose von Problemen, zur Optimierung der Datenbank oder einfach zur Überwachung der Datenbank verwenden.

Der DB2-Datenbankmanager schreibt die folgenden Arten von Informationen in das Protokoll mit Benachrichtigungen für die Systemverwaltung auf Plattformen mit einem UNIX- oder Linux-Betriebssystem (auf Plattformen mit dem Windows-Betriebssystem werden Benachrichtigungsereignisse für die Systemverwaltung im Ereignisprotokoll aufgezeichnet):

- Status von DB2-Dienstprogrammen wie **REORG** und **BACKUP**
- Fehler von Clientanwendungen
- Serviceklassenänderungen
- Lizenzierungsaktivitäten
- Dateipfade
- Speicherprobleme
- Überwachungsaktivitäten
- Indexierungsaktivitäten

- Tabellenbereichsprobleme

Die Nachrichten für das Protokoll mit Benachrichtigungen für die Systemverwaltung werden in einem standardisierten Nachrichtenformat ebenfalls in den **db2diag**-Protokolldateien aufgezeichnet.

Hinweisnachrichten bieten zusätzliche Informationen als Ergänzung zu angegebenen SQLCODE-Werten.

Die Protokolldatei mit Benachrichtigungen für die Systemverwaltung kann in zwei verschiedenen Formen vorliegen:

Als einzelne Protokolldatei mit Benachrichtigungen für die Systemverwaltung
Eine einzige aktive Protokolldatei mit Benachrichtigungen für die Systemverwaltung mit dem Namen *instanzname*.nfy, deren Größe unbegrenzt wächst. Dies ist die Standardform, die verwendet wird, wenn der Konfigurationsparameter **diagsize** des Datenbankmanagers den Standardwert 0 hat.

Als rollierende Protokolldateien mit Benachrichtigungen für die Systemverwaltung
Eine einzige aktive Protokolldatei (mit dem Namen *instanzname*.*N*.nfy, wobei *N* die als Dateinamenindex verwendete Folgenummer ist, die mit 0 beginnt und stetig ansteigt). In dem Verzeichnis, das durch den Konfigurationsparameter **diagpath** definiert ist, kann sich jedoch eine Reihe von Protokolldateien mit Benachrichtigungen für die Systemverwaltung befinden. Jede der Dateien wächst bis zu einer begrenzten Größe an. Dann wird sie geschlossen und eine neue Datei zur Protokollierung mit einem inkrementierten Dateinamenindex erstellt und geöffnet (*instanzname*.*N+1*.nfy). Diese Protokolldateiform wird verwendet, wenn der Konfigurationsparameter **diagsize** des Datenbankmanagers einen Wert ungleich null hat.

Anmerkung: Auf der Plattform des Windows-Betriebssystems ist weder eine einzelne Protokolldatei noch eine rollierende Reihe von Protokolldateien mit Benachrichtigungen für die Systemverwaltung verfügbar.

Sie bestimmen die von Ihrem System verwendete Protokolldateiform, indem Sie den Konfigurationsparameter **diagsize** des Datenbankmanagers entsprechend definieren.

Konfigurationen

Größe, Speicherposition sowie Typen und Detaillierungsebene für aufgezeichnete Ereignisse können für Protokolldateien mit Benachrichtigungen für die Systemverwaltung durch Festlegen der folgenden Konfigurationsparameter des Datenbankmanagers konfiguriert werden:

diagsize
Der Wert für **diagsize** bestimmt, welche Form der Protokolldatei mit Benachrichtigungen für die Systemverwaltung verwendet wird. Bei dem Wert 0 wird nur eine Protokolldatei mit Benachrichtigungen für die Systemverwaltung verwendet. Bei einem Wert ungleich 0 werden rollierende Protokolldateien mit Benachrichtigungen für die Systemverwaltung verwendet. Der Wert gibt gleichzeitig die Gesamtgröße aller rollierenden Diagnoseprotokolldateien und der rollierenden Protokolldateien mit Benachrichtigungen für die Systemverwaltung an. Neue oder geänderte Werte für den Parameter **diagsize** werden erst nach einem Neustart der Instanz wirksam. Ausführliche Angaben hierzu finden Sie in dem Abschnitt "diagsize - Rol-

lierende Protokolle mit Benachrichtigungen für die Systemverwaltung und für die Diagnose (Konfigurationsparameter)".

diagpath
Die Position der Protokolldateien mit Benachrichtigungen für die Systemverwaltung, in die die Diagnoseinformationen geschrieben werden sollen, wird über den Konfigurationsparameter **diagpath** festgelegt. Ausführliche Angaben hierzu finden Sie in dem Abschnitt "diagpath - Verzeichnispfad für Diagnosedaten (Konfigurationsparameter)".

notifylevel
Die Typen von Ereignissen und die Detaillierungsebene der Informationen, die in die Protokolldateien mit Benachrichtigungen für die Systemverwaltung geschrieben werden, kann mit dem Konfigurationsparameter **notifylevel** definiert werden. Ausführliche Angaben hierzu finden Sie in dem Abschnitt "notifylevel - Benachrichtigungsstufe (Konfigurationsparameter)".

Anmerkung: Ab DB2 Version 9.7 Fixpack 1 gilt: Wenn der Konfigurationsparameter **diagsize** auf einen Wert ungleich null gesetzt wird und der Konfigurationsparameter **diagpath** so gesetzt wird, dass die Diagnosedaten auf separate Verzeichnisse aufgeteilt werden, dann gibt der Wert ungleich null des Konfigurationsparameters **diagsize** die Gesamtgröße aller rotierenden Protokolldateien mit Benachrichtigungen für die Systemverwaltung zusammen mit allen rotierenden Protokollen der Diagnoseprogramme innerhalb eines jeweiligen geteilten Diagnosedatenverzeichnisses an. Beispiel: Wenn in einem System mit 4 Datenbankpartitionen der Parameter **diagsize** auf 1 GB und der Parameter **diagpath** auf "$n" (Diagnosedaten pro Datenbankpartition aufteilen) gesetzt ist, dann kann die Gesamtgröße der Benachrichtigungsprotokolle und der Protokolle der Diagnoseprogramme maximal 4 GB (4 x 1 GB) betragen.

Interpretieren von Einträgen in der Protokolldatei mit Benachrichtigungen für die Systemverwaltung

Sie können die Protokolldatei mit Benachrichtigungen für die Systemverwaltung in einem Texteditor auf der Maschine aufrufen, auf der vermutlich ein Fehler aufgetreten ist. Die zuletzt aufgezeichneten Ereignisse befinden sich am Ende der Datei.

Im Allgemeinen enthält jeder Eintrag die folgenden Komponenten:

- Eine Zeitmarke.
- Die Position, die den Fehler meldet. Anhand von Anwendungs-IDs können Sie in den Protokollen von Servern und Clients die Einträge zuordnen, die zu einer Anwendung gehören.
- Eine Diagnosenachricht, die normalerweise mit "DIA" oder "ADM" beginnt und den Fehler erläutert.
- Alle verfügbaren unterstützenden Daten, wie zum Beispiel Datenstrukturen des SQL-Kommunikationsbereichs (SQLCA) und Zeiger auf die Position zusätzlicher Speicherauszugs- oder Trapdateien.

Das folgende Beispiel zeigt die Kopfdaten für einen Beispielprotokolleintrag, wobei alle Teile des Protokolls identifiziert werden.

Anmerkung: Nicht jeder Protokolleintrag enthält alle diese Komponenten.

```
2006-02-15-19.33.37.630000 1     Instance:DB2 2    Node:000 3
PID:940(db2syscs.exe) TID: 660 4    Appid:*LOCAL.DB2.020205091435 5
recovery manager 6   sqlpresr 7   Probe:1 8    Database:SAMPLE   9
ADM1530E 10 Die Recovery nach dem Systemabsturz wurde eingeleitet. 11
```

Legende:

1. Eine Zeitmarke für die Nachricht.
2. Der Name der Instanz, die die Nachricht generiert.
3. Bei Mehrpartitionssystemen die Datenbankpartition, die die Nachricht generiert. (In einer nicht partitionierten Datenbank ist der Wert '000'.)
4. Die Prozess-ID (PID), gefolgt vom Namen des Prozesses, gefolgt von der Thread-ID (TID). Dabei handelt es sich um die für die Generierung der Nachricht verantwortlichen Komponenten.
5. Identifikation der Anwendung, für die der Prozess ausgeführt wird. In diesem Beispiel wird der Prozess, der die Nachricht generiert, für eine Anwendung mit der ID *LOCAL.DB2.020205091435 ausgeführt.

   Dieser Wert ist identisch mit den Daten des Monitorelements **appl_id**. Ausführliche Informationen zur Interpretation dieses Werts finden Sie in der Dokumentation zum Monitorelement **appl_id**.

   Zur weiteren Identifikation einer bestimmten Anwendungs-ID haben Sie folgende Möglichkeiten:

   - Verwenden Sie den Befehl **LIST APPLICATIONS** auf einem DB2-Server oder den Befehl **LIST DCS APPLICATIONS** auf einem DB2 Connect-Gateway, um eine Liste mit Anwendungs-IDs anzuzeigen. Anhand dieser Liste können Sie Informationen über den Client ermitteln, auf dem der Fehler auftritt, wie zum Beispiel den Knotennamen und die TCP/IP-Adresse des Clients.
   - Verwenden Sie den Befehl **GET SNAPSHOT FOR APPLICATION**, um eine Liste mit Anwendungs-IDs anzuzeigen.
6. Die DB2-Komponente, die die Nachricht erstellt. Für Nachrichten, die von Benutzeranwendungen mit der API db2AdminMsgWrite erstellt werden, lautet die Angabe der Komponente „User Application".
7. Der Name der Funktion, die die Nachricht bereitstellt. Diese Funktion wird innerhalb der DB2-Komponente ausgeführt, die die Nachricht generiert. Für Nachrichten, die von Benutzeranwendungen mit der API db2AdminMsgWrite geschrieben werden, lautet die Angabe der Funktion „User Function".
8. Eindeutige interne ID. Mithilfe dieser Nummer können die Mitarbeiter der DB2-Kundenunterstützung und -Entwicklung die Stelle im DB2-Quellcode feststellen, von der die Nachricht ausgegeben wurde.
9. Die Datenbank, in der der Fehler auftrat.
10. Eine Nachricht, falls verfügbar, die den Fehlertyp und die Fehlernummer als Hexadezimalcode angibt.
11. Ein Nachrichtentext, falls verfügbar, der das protokollierte Ereignis erläutert.

Festlegen der Aufzeichnungsebene für die Protokolldatei mit Benachrichtigungen für die Systemverwaltung

In diesem Abschnitt wird beschrieben, wie Sie die Aufzeichnungsebene für die Protokolldatei mit Benachrichtigungen für die Systemverwaltung festlegen können.

Informationen zu diesem Vorgang

Welche Informationen DB2 im Protokoll mit Benachrichtigungen für die Systemverwaltung aufzeichnet, ist abhängig von der Einstellung für **NOTIFYLEVEL**.

Vorgehensweise

- Geben Sie den Befehl **GET DBM CFG** ein, um die aktuelle Einstellung zu überprüfen.

Suchen Sie nach der folgenden Variablen:

```
Aufzeichnungsebene                          (NOTIFYLEVEL) = 3
```

- Verwenden Sie den Befehl **UPDATE DBM CFG**, um die Einstellung zu ändern. Beispiel:

  ```
  DB2 UPDATE DBM CFG USING NOTIFYLEVEL X
  ```

  Dabei ist *X* die gewünschte Aufzeichnungsebene.

DB2-Diagnoseprotokolldateien (db2diag)

Während die DB2-**db2diag**-Protokolldateien für Diagnosezwecke in erster Linie für IBM Software Support konzipiert sind und zur Fehlerbehebung dienen, enthält das Protokoll mit Benachrichtigungen für die Systemverwaltung nützliche Informationen zur Fehlerbehebung, die in erster Linie für Datenbank- und Systemadministratoren konzipiert sind. Die Nachrichten für das Protokoll mit Benachrichtigungen für die Systemverwaltung werden in einem standardisierten Nachrichtenformat ebenfalls in den **db2diag**-Protokolldateien aufgezeichnet.

Übersicht

Da sowohl die DB2-Diagnosenachrichten als auch die Benachrichtigungen für die Systemverwaltung in den **db2diag**-Protokolldateien aufgezeichnet werden, stellen die **db2diag**-Protokolldateien in aller Regel die erste Informationsquelle für den Datenbankbetrieb dar. Informationen zur Interpretation der Inhalte dieser Diagnoseprotokolldateien finden Sie in den unter "Zugehörige Konzepte" angegebenen Abschnitten. Wenden Sie sich an IBM Software Support (siehe "Kontaktaufnahme mit IBM Software Support"), wenn Sie die bei Ihnen aufgetretenen Fehler trotz aller Anstrengungen nicht ohne fremde Hilfe lösen können. Neben anderen Informationsquellen, wie z. B. relevanten Protokollen, Speicherauszugsdateien und Tracedateien, fordert IBM Software Support zur Fehlerbehebung auch die **db2diag**-Protokolldateien an.

Die **db2diag**-Protokolldateien liegen in zwei verschiedenen Formaten vor:

Eine einzige Diagnoseprotokolldatei
: Eine einzige aktive Diagnoseprotokolldatei mit der Bezeichnung db2diag.log, deren Größe unbegrenzt ist. Dies ist das Standardformat, das verwendet wird, wenn der Konfigurationsparameter **diagsize** des Datenbankmanagers mit dem Standardwert 0 definiert ist.

Rollierende Diagnoseprotokolldateien
: Eine einzige aktive Protokolldatei db2diag.*N*.log (wobei *N* für die als Dateinamenindex verwendete Folgenummer steht, die mit 0 beginnt und stetig ansteigt). In dem Verzeichnis, das über den Konfigurationsparameter **diagpath** angegeben ist, befinden sich je nach Protokollgröße jedoch weitere ältere Protokolldateien. Die aktive Protokolldatei wird verwendet, bis sie die als Grenzwert definierte Größe erreicht hat und geschlossen wird. Daraufhin wird eine neue Datei erstellt und für die Protokollierung geöffnet, deren Dateiname einen um 1 erhöhten Dateinamenindex beinhaltet (db2diag.*N+1*.log). Dieses Protokolldateiformat wird verwendet, wenn der Konfigurationsparameter **diagsize** des Datenbankmanagers mit einem Wert über null definiert ist.

Sie bestimmen das von Ihrem System verwendete Protokolldateiformat, indem Sie den Konfigurationsparameter **diagsize** des Datenbankmanagers entsprechend definieren.

Konfigurationen

Größe, Position und Fehlerinhalt der **db2diag**-Protokolldateien können mit den folgenden Konfigurationsparametern des Datenbankmanagers definiert werden:

diagsize
Der Wert für **diagsize** bestimmt, welches Format der Diagnoseprotokolldatei verwendet wird. Bei einem Wert von 0 ist lediglich eine einzige Diagnoseprotokolldatei vorhanden. Bei einem Wert über 0 werden rollierende Diagnoseprotokolldateien verwendet. Der Wert gibt gleichzeitig die Gesamtgröße aller rollierenden Diagnoseprotokolldateien und der Protokolldateien mit Benachrichtigungen für die Systemverwaltung an. Neue oder geänderte Werte für den Parameter **diagsize** werden erst nach einem Neustart der Instanz wirksam. Ausführliche Angaben hierzu finden Sie in dem Abschnitt "diagsize - Rollierende Protokolle mit Benachrichtigungen für die Systemverwaltung und für die Diagnose (Konfigurationsparameter)".

diagpath
Die Position der **db2diag**-Protokolldateien, in die die Diagnoseinformationen geschrieben werden, wird über den Konfigurationsparameter **diagpath** bestimmt. Ausführliche Angaben hierzu finden Sie in dem Abschnitt "diagpath - Verzeichnispfad für Diagnosedaten (Konfigurationsparameter)".

alt_diagpath
Der Konfigurationsparameter **alt_diagpath** des Datenbankmanagers bietet einen alternativen Diagnosedatenverzeichnispfad in dem Diagnoseinformationen gespeichert werden können. Wenn der Datenbankmanager in den mit **diagpath** angegebenen Pfad keine Daten schreiben kann, wird stattdessen der mit **alt_diagpath** angegebene Pfad verwendet, um Diagnoseinformationen zu speichern.

diaglevel
Die Art der Diagnosefehler, die in die **db2diag**-Protokolldateien geschrieben werden, wird über den Konfigurationsparameter **diaglevel** angegeben. Ausführliche Angaben hierzu finden Sie in dem Abschnitt "diaglevel - Aufzeichnungsebene bei Fehlerdiagnose (Konfigurationsparameter)".

Anmerkung: Ab DB2 Version 9.7 Fixpack 1 gilt: Wenn der Konfigurationsparameter **diagsize** auf einen Wert ungleich null gesetzt wird und der Konfigurationsparameter **diagpath** so gesetzt wird, dass die Diagnosedaten auf separate Verzeichnisse aufgeteilt werden, dann gibt der Wert ungleich null des Konfigurationsparameters **diagsize** die Gesamtgröße aller rotierenden Protokolldateien mit Benachrichtigungen für die Systemverwaltung zusammen mit allen rotierenden Protokollen der Diagnoseprogramme innerhalb eines jeweiligen geteilten Diagnosedatenverzeichnisses an. Beispiel: Wenn in einem System mit 4 Datenbankpartitionen der Parameter **diagsize** auf 1 GB und der Parameter **diagpath** auf `"$n"` (Diagnosedaten pro Datenbankpartition aufteilen) gesetzt ist, dann kann die Gesamtgröße der Benachrichtigungsprotokolle und der Protokolle der Diagnoseprogramme maximal 4 GB (4 x 1 GB) betragen.

Interpretieren von Einträgen in der Protokolldatei der Diagnoseprogramme

Verwenden Sie das Analysetool **db2diag** für die **db2diag**-Protokolldateien, um die **db2diag**-Protokolldateien zu filtern und zu formatieren. Da auch die Nachrichten des Protokolls mit Benachrichtigungen für die Systemverwaltung in den **db2diag**-Protokolldateien in einem standardisierten Nachrichtenformat protokolliert werden,

ist es empfehlenswert das Protokoll **db2diag**-Protokolldateien zuerst anzuzeigen, um zu verstehen, was bei der Datenbank passierte.

Als Alternative zur Verwendung von **db2diag** können Sie einen Texteditor verwenden, um die Diagnoseprotokolldatei auf dem System anzuzeigen, auf dem vermutlich ein Fehler aufgetreten ist. Die zuletzt aufgezeichneten Ereignisse befinden sich am Ende der Datei.

Anmerkung: Das Protokoll mit Benachrichtigungen für die Systemverwaltung (*instance_name*.nfy) und die Diagnoseprotokolldatei (db2diag.log) nehmen als einzelne Protokolldateien *kontinuierlich* an Umfang zu. Wird der Konfigurationsparameter **diagsize** des Datenbankmanagers mit einem Wert ungleich null definiert, werden für das Protokoll mit Benachrichtigungen für die Systemverwaltung und die Diagnoseprotokolldatei eine Reihe rollierender Protokolldateien ('*instanzname.N*.nfy' und 'db2diag.*N*.log') verwendet, deren Gesamtumfang durch den für den Konfigurationsparameter **diagsize** definierten Wert begrenzt wird.

Das folgende Beispiel zeigt die Kopfdaten für einen Beispielprotokolleintrag, wobei alle Teile des Protokolls identifiziert werden.

Anmerkung: Nicht jeder Protokolleintrag enthält alle diese Komponenten. Nur die ersten Felder (Zeitmarke bis TID) und FUNCTION sind in allen Einträgen der **db2diag**-Protokolldateien vorhanden.

```
2007-05-18-14.20.46.973000-240 1  I27204F655 2  LEVEL: Info 3
PID : 3228 4  TID : 8796 5  PROC : db2syscs.exe 6
INSTANCE: DB2MPP 7  NODE : 002 8  DB : WIN3DB1 9
APPHDL : 0-51 10  APPID: 9.26.54.62.45837.070518182042 11
AUTHID : UDBADM 12
EDUID : 8796 13  EDUNAME: db2agntp 14  (WIN3DB1) 2
FUNCTION: 15  DB2 UDB, data management, sqldInitDBCB, probe:4820
DATA #1 : 16  String, 26 bytes
Setting ADC Threshold to:
DATA #2 : unsigned integer, 8 bytes
1048576
```

Legende:

1. Die Zeitmarke und die Zeitzone für die Nachricht.

   Anmerkung: Die Zeitmarken in den **db2diag**-Protokolldateien enthalten eine Zeitzone. Beispiel: `2006-02-13-14.34.35.965000-300`. Hierbei ist `"-300"` die Differenz zwischen der Weltzeit (Coordinated Universal Time, UTC - früher GMT) und der Ortszeit auf dem Anwendungsserver in Minuten. Das heißt, -300 bedeutet UTC - 5 Stunden, z. B. EST (Eastern Standard Time).

2. Das Feld mit der Satz-ID. Die Satz-ID in den **db2diag**-Protokolldateien gibt die relative Dateiposition, an der die aktuelle Nachricht aufgezeichnet wird (beispielsweise „27204”), sowie die Nachrichtenlänge (beispielsweise „655”) für die Plattform an, auf der das DB2-Diagnoseprotokoll erstellt wurde.

3. Die einer Fehlernachricht zugeordnete Diagnosestufe. Zum Beispiel `Info`, `Warning`, `Error`, `Severe` oder `Event`.

4. Die Prozess-ID.

5. Die Thread-ID.

6. Der Prozessname.

7. Der Name der Instanz, die die Nachricht generiert.

8. Bei Mehrpartitionssystemen die Datenbankpartition, die die Nachricht generiert. (In einer nicht partitionierten Datenbank ist der Wert '000'.)

9. Der Datenbankname

10. Die Anwendungskennung. Dieser Wert richtet sich nach dem in der **db2pd**-Ausgabe und den Speicherauszugsdateien für Sperren verwendeten Wert. Er besteht aus der Koordinatorpartitionsnummer, gefolgt von einem Gedankenstrich und der Koordinatorindexnummer.

11. Identifikation der Anwendung, für die der Prozess ausgeführt wird. In diesem Beispiel wird der Prozess, der die Nachricht generiert, für eine Anwendung mit der ID 9.26.54.62.45837.070518182042 ausgeführt.

    Eine von TCP/IP generierte Anwendungs-ID besteht aus drei Abschnitten:

    1. **IP-Adresse**: Diese wird als 32-Bit-Zahl dargestellt, die als Hexadezimalzahl mit höchstens 8 Stellen angezeigt wird.
    2. **Portnummer**: Diese wird als vierstellige Hexadezimalzahl dargestellt.
    3. Eine **eindeutige Kennung** für die Instanz dieser Anwendung.

    Anmerkung: Wenn die hexadezimale Version der IP-Adresse oder Portnummer mit 0 - 9 beginnt, wird sie jeweils in G - P umgesetzt. So wird '0' beispielsweise in 'G' umgesetzt, '1' in 'H' etc. Die IP-Adresse AC10150C.NA04.006D07064947 wird wie folgt interpretiert: Die IP-Adresse bleibt AC10150C, d. h. 172.16.21.12. Die Portnummer ist NA04. Das erste Zeichen ist 'N', das in '7' umgesetzt wird. So lautet die Portnummer im Hexadezimalformat '7A04', nach der Umsetzung in Dezimalformat '31236'.

    Dieser Wert ist identisch mit den Daten des Monitorelements *appl_id*. Ausführliche Informationen zur Interpretation dieses Werts finden Sie in der Dokumentation zum Monitorelement *appl_id*.

    Zur weiteren Identifikation einer bestimmten Anwendungs-ID haben Sie folgende Möglichkeiten:

    - Verwenden Sie den Befehl **LIST APPLICATIONS** auf einem DB2-Server oder den Befehl LIST DCS APPLICATIONS auf einem DB2 Connect-Gateway, um eine Liste mit Anwendungs-IDs anzuzeigen. Anhand dieser Liste können Sie Informationen über den Client ermitteln, auf dem der Fehler auftritt, wie zum Beispiel den Datenbankpartitionsnamen und die TCP/IP-Adresse des Clients.
    - Verwenden Sie den Befehl **GET SNAPSHOT FOR APPLICATION**, um eine Liste mit Anwendungs-IDs anzuzeigen.
    - Verwenden Sie den Befehl **db2pd -applications -db <datenbankname>**.

12. Die Berechtigungs-ID.

13. Die Engine-Dispatchable-Unit-ID.

14. Der Name der Engine-Dispatchable-Unit.

15. Der Name des Produkts ("DB2"), der Komponente („data management”) und der Funktion („sqlInitDBCB”), das bzw. die die Nachricht generiert (sowie der Testpunkt („4820”) innerhalb der Funktion).

16. Die Informationen, die von einer aufgerufenen Funktion zurückgegeben wurden. Es werden möglicherweise mehrere Datenfelder zurückgegeben.

Nach der Beschreibung des Beispieleintrags der **db2diag**-Protokolldateien folgt nun eine Liste aller möglichen Felder:

```
<zeitmarke><zeitzone>       <satzID>           LEVEL: <stufe> (<quelle>)
PID     : <pid>             TID  : <tid>       PROC : <prozName>
INSTANCE: <instanz>        NODE : <knoten>      DB   : <datenbank>
APPHDL  : <anwKennung>      APPID: <anwID>
AUTHID  : <berKennung>
EDUID   : <eduKennung>         EDUNAME: <name der engine-dispatchable-unit>
FUNCTION: <produktname>, <komponentenname>, <funktionsname>, probe:<testnr>
MESSAGE : <nachrichtenID>  <nachrichtentext>
CALLED  : <produktname>, <komponentenname>, <funktionsname>   OSERR: <fehlername> (<fehlernr>)
RETCODE : <typ>=<rückkehrcode> <fehlerbeschreibung>
ARG #N  : <typentitel>, <typenname>, <größe> bytes
... argument ...
DATA #N : <typentitel>, <typenname>, <größe> bytes
... data ...
```

Die Felder, die nicht bereits im Beispiel beschrieben wurden, sind nachfolgend aufgeführt:

- <quelle> Gibt den Ursprung des protokollierten Fehlers an. (Befindet sich am Ende der ersten Zeile im Beispiel.) Die möglichen Werte sind:
  - origin - Die Nachricht wird von der Funktion protokolliert, bei der der Fehler ursprünglich auftrat (Anfangspunkt).
  - OS - Der Fehler wurde vom Betriebssystem generiert.
  - received - Der Fehler wurde von einem anderen Prozess (Client/Server) empfangen.
  - sent - Der Fehler wurde an einen anderen Prozess (Client/Server) gesendet.
- MESSAGE Enthält die protokollierte Nachricht, bestehend aus:
  - <nachrichtenID> - Nachrichtennummer, z. B. ECF=0x9000004A oder DIA8604C
  - <nachrichtentext> - Fehlerbeschreibung

  Wenn das Feld CALLED ebenfalls aufgeführt ist, dann ist <nachrichtentext> die Folge des Fehlers, der durch die unter CALLED aufgelistete Funktion für die Funktion zurückgegeben wird, für die eine Nachricht protokolliert wird. (Diese Funktion wird im Feld FUNCTION aufgeführt.)
- CALLED Gibt die Funktion an, die einen Fehler zurückgibt, bestehend aus:
  - <produktname> - Produktname: "OS", "DB2", "DB2 Tools" oder "DB2 Common"
  - <komponentenname> - Der Komponentenname ('-' im Falle eines Systemaufrufs)
  - <funktionsname> - Name der aufgerufenen Funktion
- OSERR Betriebssystemfehler, der durch den Systemaufruf (CALLED) zurückgegeben wird, (befindet sich am Ende derselben Zeile wie CALLED) bestehend aus:
  - <fehlername> - systemspezifischer Fehlername
  - <fehlernummer> - Fehlernummer des Betriebssystems
- ARG In diesem Abschnitt sind die Argumente eines Funktionsaufrufs aufgeführt, der einen Fehler zurückgab, bestehend aus:
  - <N> - Position eines Arguments in einem Aufruf an die durch CALLED angegebene Funktion
  - <typentitel> - Bezeichnung, die dem Typennamen des N-ten Arguments zugeordnet ist
  - <typenname> - Typenname des protokollierten Arguments
  - <größe> - Größe des zu protokollierenden Arguments

- DATA Enthält zusätzliche Daten, für die möglicherweise von der Protokollierungsfunktion ein Speicherauszug erstellt wird, bestehend aus:
  - <N> - Fortlaufende Nummer des Datenobjekts, für das ein Speicherauszug erstellt wird
  - <typentitel> - Bezeichnung der Daten, für die ein Speicherauszug erstellt wird
  - <typenname> - Typenname des Datenfelds, das protokolliert wird, z. B. PD_TYPE_UINT32, PD_TYPE_STRING
  - <größe> - Größe eines Datenobjekts

Interpretation des Informationssatzes in den db2diag-Protokolldateien

Die erste Nachricht in den **db2diag**-Protokolldateien sollte immer ein Informationssatz sein.

Nachfolgend ist ein Beispiel für einen Informationssatz dargestellt:

```
2006-02-09-18.07.31.059000-300 I1H917            LEVEL: Event
PID     : 3140                 TID  : 2864       PROC : db2start.exe
INSTANCE: DB2                  NODE : 000
FUNCTION: DB2 UDB, RAS/PD component, _pdlogInt, probe:120
START   : New Diagnostic Log file
DATA #1 : Build Level, 124 bytes
Instanz "DB2" verwendet "32" Bit und DB2-Codefreigabe "SQL09010"
mit Aktualitäts-ID "01010107".
Informationstoken: "DB2 v9.1.0.190", "s060121", "", Fixpack "0".
DATA #2 : System Info, 1564 bytes
System: WIN32_NT MYSRVR Service Pack 2 5.1 x86 Family 15, model 2, stepping 4
CPU: total:1 online:1 Cores per socket:1 Threading degree per core:1
Physical Memory(MB): total:1024 free:617 available:617
Virtual  Memory(MB): total:2462 free:2830
Swap     Memory(MB): total:1438 free:2213
Die Informationen in diesem Informationssatz sind nur zum Zeitpunkt
der Erstellung dieser Datei gültig (siehe Zeitmarke dieses Satzes).
```

Der Informationssatz wird für **db2start** in jeder logischen Partition ausgegeben. Auf diese Weise werden mehrere Informationssätze generiert: einer für jede logische Partition. Da der Informationssatz Speicherwerte enthält, die für jede Partition unterschiedlich sind, können diese Informationen nützlich sein.

Festlegen der Aufzeichnungsebene für die Diagnoseprotokolldateien

Die DB2-Diagnoseprotokolldateien (**db2diag**) enthalten von DB2 aufgezeichnete Textinformationen. Diese Informationen dienen der Fehlerbehebung und sind größtenteils für Mitarbeiter von IBM Software Support bestimmt.

Informationen zu diesem Vorgang

Die Einstellung für den Konfigurationsparameter **diaglevel** des Datenbankmanagers bestimmt, welche Typen von Diagnosefehlern in den **db2diag**-Protokolldateien aufgezeichnet werden.

Vorgehensweise

- Geben Sie den Befehl **GET DBM CFG** ein, um die aktuelle Einstellung zu überprüfen.

  Suchen Sie nach der folgenden Variablen:

  ```
  Aufzeichnungsebene bei Fehlerdiagnose             (DIAGLEVEL) = 3
  ```

- Verwenden Sie den Befehl **UPDATE DBM CFG**, um den Wert dynamisch zu ändern.

Gehen Sie wie folgt vor, um einen Konfigurationsparameter des Datenbankmanagers online zu ändern:

```
db2 attach to <instance-name>
db2 update dbm cfg using <parameter-name> <value>
db2 detach
```

Beispiel:

```
DB2 UPDATE DBM CFG USING DIAGLEVEL X
```

Dabei ist `X` die gewünschte Aufzeichnungsebene. Wenn Sie eine Fehlerdiagnose für einen reproduzierbaren Fehler durchführen, schlägt ein Mitarbeiter von IBM Software Support möglicherweise vor, bei der Fehlerbehebung die Aufzeichnungsebene 4 für **diaglevel** zu verwenden.

Kombinieren von Diagnoseprogrammen der DB2-Datenbank und des Betriebssystems

Das Diagnostizieren von Problemen im Hinblick auf Hauptspeicher, Auslagerungsdateien, CPUs, Plattenspeicher und andere Ressourcen erfordert gründliche Kenntnisse darüber, wie das betreffende Betriebssystem die entsprechenden Ressourcen verwaltet. Die Definition eines ressourcenbezogenen Problems erfordert als Minimum Kenntnisse darüber, wie viel einer Ressource vorhanden ist und welche Ressourcengrenzen pro Benutzer gelten. (Die entsprechenden Grenzwerte gelten normalerweise für die Benutzer-ID des DB2-Instanzeigners.)

Im Folgenden werden einige der wichtigen Konfigurationsdaten aufgeführt, die abgerufen werden müssen:

- Programmkorrekturstufe (Patch-Level) des Betriebssystems sowie installierte Software und Upgrade-Verlauf
- Anzahl der CPUs
- Volumen des Arbeitsspeichers (RAM)
- Einstellungen für Auslagerungs- und Dateicache
- Grenzwerte für Benutzerdaten und Dateiressourcen sowie Prozessgrenzwerte pro Benutzer
- IPC-Ressourcengrenzwerte (Nachrichtenwarteschlangen, gemeinsam genutzte Speichersegmente, Semaphore)
- Typ des Plattenspeichers
- Wofür wird das System sonst noch verwendet? Muss sich DB2 die Ressourcen mit anderen Anwendungen teilen?
- Wo findet die Authentifizierung statt?

Auf den meisten Plattformen können Informationen zu Ressourcen mithilfe von einfachen Befehlen abgerufen werden. Allerdings ist es nur selten erforderlich, diese Informationen manuell abzurufen, da diese Daten und viele mehr vom Dienstprogramm **db2support** erfasst werden. Die Datei `detailed_system_info.html`, die von **db2support** generiert wird (sofern die Optionen **-s** und **-m** angegeben werden), enthält die Syntax für viele der Betriebssystembefehle, die zum Erfassen dieser Informationen verwendet werden.

Anhand der folgenden Übungen soll gezeigt werden, wie Informationen zur Systemkonfiguration und Benutzerumgebung in den verschiedenen DB2-Diagnosedateien ermittelt werden. Die erste Übung illustriert die erforderlichen Schritte zur Ausführung des Dienstprogramms **db2support**. Die nachfolgenden Übungen befas-

sen sich mit Trapdateien, die weitere von DB2 generierte Daten bereitstellen, die für ein besseres Verständnis der Benutzerumgebung und Ressourcengrenzen nützlich sein können.

Übung 1: Befehl **db2support** ausführen

1. Starten Sie die DB2-Instanz mithilfe des Befehls **db2start**.
2. Erstellen Sie ein Verzeichnis zum Speichern der Befehlsausgabe von **db2support**. (Hierbei wird davon ausgegangen, dass die Datenbank SAMPLE bereits vorhanden ist.)
3. Wechseln Sie in dieses Verzeichnis, und geben Sie den folgenden Befehl aus:

   ```
   db2support <verzeichnis> -d sample -s -m
   ```

4. Überprüfen Sie die Konsolausgabe, und achten Sie hierbei insbesondere auf die Typen der erfassten Informationen.

   Die Befehlsausgabe sollte wie folgt aussehen (unter Windows):

   ```
   ...
   "Systemdateien" erfassen
             "db2cache.prf"
             "db2cos9402136.0"
             "db2cos9402840.0"
             "db2dbamr.prf"
             "db2diag.bak"
             "db2eventlog.000"
             "db2misc.prf"
             "db2nodes.cfg"
             "db2profile.bat"
             "db2systm"
             "db2tools.prf"
             "HealthRulesV82.reg"
             "db2dasdiag.log"
        ...
   "Detaillierte Betriebssystem- und Hardwareinformationen" erfassen
   "Systemressourceninformationen (Datenträger, CPU, Speicher)" erfassen
   "Betriebssystem und -stufe" erfassen
   "JDK-Stufe" erfassen
   "DB2-Release-Informationen" erfassen
   "DB2-Installationspfadinformationen" erfassen
   "Registrierdatenbankinformationen" erfassen
   ...
   Endgültiges Ausgabearchiv erstellen
             "db2support.html"
             "db2_sqllib_directory.txt"
             "detailed_system_info.html"
             "db2supp_system.zip"
             "dbm_detailed.supp_cfg"
             "db2diag.log"
   db2support ist jetzt beendet.
     Die folgende Archivdatei wurde erstellt: 'db2support.zip'
   ```

5. Zeigen Sie die Datei `detailed_system_info.html` nun mit einem Web-Browser an. Ermitteln Sie auf jedem System jeweils die folgenden Informationen:
   - Anzahl der CPUs
   - Version des Betriebssystems
   - Benutzerumgebung
   - Ressourcengrenzwerte für Benutzer (UNIX-Befehl **ulimit**)

Übung 2: Informationen zur Umgebung in einer DB2-Trapdatei ermitteln

1. Stellen Sie sicher, dass eine DB2-Instanz gestartet ist, und geben Sie anschließend den folgenden Befehl aus:

   ```
   db2pd -stack all
   ```

Die Aufrufstacks werden in Dateien im Diagnoseverzeichnis gestellt (das über den Konfigurationsparameter **diagpath** des Datenbankmanagers definiert ist).

2. Suchen Sie in einer der Trapdateien nach folgenden Informationen:
   - DB2-Codeversion
   - Data seg top (dies ist der maximal erforderliche private Adressraum)
   - Cur data size (dies ist der maximale Grenzwert für den privaten Adressraum)
   - Cur core size (dies ist der maximale Grenzwert für die Kerndatei)
   - Signalroutinen (diese Information wird unter Umständen nicht in allen Trapdateien angezeigt)
   - Umgebungsvariablen (diese Information wird unter Umständen nicht in allen Trapdateien angezeigt)
   - Zuordnungsausgabe (zeigt die geladenen Bibliotheken)

Beispieltrapdatei aus Windows (abgeschnitten):

```
...
<DB2TrapFile version="1.0">
<Trap>
<Header>
DB2 build information: DB2 v9.1.0.190 s060121 SQL09010
timestamp: 2006-02-17-14.03.43.846000
uname: S:Windows
comment:
process id: 940
thread id: 3592
</Header>
<SystemInformation>
Number of Processors: 1
Processor Type: x86 Family 15 Model 2 Stepping 4
OS Version: Microsoft Windows XP, Service Pack 2 (5.1)
Current Build: 2600
</SystemInformation>
<MemoryInformation>
<Usage>
Physical Memory:    1023 total,     568 free.
Virtual Memory :    2047 total,    1882 free.
Paging File    :    2461 total,    2011 free.
Ext. Virtual   :       0 free.
</Usage>
</MemoryInformation>
<EnvironmentVariables>
<![CDATA[
[e] DB2PATH=C:\Program Files\IBM\SQLLIB
[g] DB2_EXTSECURITY=YES
[g] DB2SYSTEM=MYSRVR
[g] DB2PATH=C:\Program Files\IBM\SQLLIB
[g] DB2INSTDEF=DB2
[g] DB2ADMINSERVER=DB2DAS00
]]></EnvironmentVariables>
```

Korrelation zwischen DB2- und Systemereignissen bzw. -fehlern

Systemnachrichten und Fehlerprotokolle werden viel zu häufig ignoriert. Sie können bei der Lösung von Problemen Stunden, Tage oder sogar Wochen sparen, wenn Sie sich die Zeit nehmen, ganz am Anfang der Problemdefinition und -untersuchung eine einfache Task auszuführen. Diese Task besteht darin, die Einträge in verschiedenen Protokollen miteinander zu vergleichen und alles zu notieren, das miteinander in Zusammenhang zu stehen scheint, was die Zeit und die Ressourcen betrifft, auf die die Einträge verweisen.

Die besten Hinweise sind häufig in den Systemprotokollen enthalten, auch wenn diese nicht immer für die Problemdiagnose relevant sind. Wenn ein gemeldetes Systemproblem mit DB2-Fehlern korreliert, d. h. in Zusammenhang gebracht werden kann, dann ist in vielen Fällen bereits klar, worin die direkte Ursache für das DB2-Symptom liegt. Offensichtliche Beispiele sind Plattenfehler, Netzfehler und Hardwarefehler. Weniger offensichtlich sind Probleme, die auf verschiedenen Systemen gemeldet werden, wie beispielsweise auf Domänencontrollern, und die sich auf die Verbindungszeit oder die Authentifizierung auswirken können.

Systemprotokolle können Aufschluss über die Stabilität des Systems geben. Dies gilt insbesondere dann, wenn Probleme auf ganz neuen Systemen gemeldet werden. Gelegentlich auftretende Traps in einheitlichen Anwendungen können ein Anzeichen dafür sein, dass es sich bei dem zugrunde liegenden Problem um einen Hardwarefehler handelt.

Systemprotokolle bieten unter anderem auch folgende Informationen:

- Bedeutende Ereignisse, wie beispielsweise der Zeitpunkt eines Systemwarmstarts
- Zeitliche Abfolge der DB2-Traps im System (sowie Fehler, Traps und Ausnahmebedingungen anderer fehlschlagender Softwarekomponenten)
- Fehler aufgrund von Kernel-Notfällen (Panic-Situationen), unzureichendem Dateisystemspeicher und unzureichendem Auslagerungsspeicher (die verhindern können, dass das System einen neuen Prozess oder eine neue Prozessverzweigung erstellt)

Systemprotokolle können dabei helfen, Absturzeinträge in den **db2diag**-Protokolldateien als relevante Hinweise auf die Ursache auszuschließen. Wenn Sie in den DB2-Protokollen mit Benachrichtigungen für die Systemverwaltung oder in den DB2-Diagnoseprotokollen auf einen Eintrag zu einem Systemabsturz stoßen, ohne dass zuvor ein Fehler aufgetreten ist, dann ist die Recovery von DB2 nach Systemabsturz wahrscheinlich die Folge eines Systemabschlusses.

Das Prinzip der Korrelation von Informationen erstreckt sich auch auf Protokolle aus anderen Quellen und auf alle identifizierbaren Benutzersymptome. So kann es beispielsweise sehr nützlich sein, korrelierende Einträge aus dem Protokoll einer anderen Anwendung zu identifizieren und zu dokumentieren, selbst wenn Sie die Einträge nicht vollständig interpretieren können.

Das Ergebnis dieser Informationen ist ein sehr gründliches Verständnis Ihres Servers und der Gesamtheit der verschiedenen Ereignisse, die zum Zeitpunkt des Fehlers auftreten.

db2cos-Ausgabedateien (Aufrufscript)

Ein **db2cos**-Script wird standardmäßig aufgerufen, wenn der Datenbankmanager die Verarbeitung aufgrund einer Panic-Situation, eines Traps, einer Segmentierungsverletzung oder einer Ausnahmebedingung nicht fortsetzen kann. Jedes **db2cos**-Standardscript ruft **db2pd**-Befehle zum entsperrten Erfassen von Informationen auf.

Die Namen der **db2cos**-Scripts lauten **db2cos_hang**, **db2cos_trap** usw. Jedes Script verhält sich auf eine ähnliche Weise. Einzige Ausnahme ist das Script **db2cos_hang**, das über das Tool **db2fodc** aufgerufen wird.

Die **db2cos**-Standardscripts befinden sich im Verzeichnis `bin`. Unter dem Betriebssystem UNIX ist dieses Verzeichnis schreibgeschützt. Sie können die Scriptdatei **db2cos** in das Verzeichnis `adm` kopieren und die Datei an dieser Position bei Bedarf

ändern. Wird ein **db2cos**-Script im Verzeichnis `adm` gefunden, so wird es ausgeführt. Andernfalls wird das Script im Verzeichnis `bin` ausgeführt.

In einer Konfiguration mit mehreren Partition wird das Script nur für den Trapagenten auf der betreffenden Partition mit dem Trap aufgerufen. Müssen Informationen von anderen Partitionen erfasst werden, können Sie das Script 'db2cos' entsprechend aktualisieren, damit der Befehl **db2_all** verwendet wird oder die Option **-alldbpartitionnums** im Befehl **db2pd** angegeben wird, wenn sich alle Partitionen auf derselben Maschine befinden.

Die Signaltypen, die den Aufruf von 'db2cos' auslösen, können ebenfalls konfiguriert werden. Hierfür wird der Befehl **db2pdcfg -cos** verwendet. In der Standardkonfiguration wird das Script 'db2cos' ausgeführt, wenn entweder eine Panic-Situation oder ein Trap eintritt. Generierte Signale hingegen führen standardmäßig nicht zum Start des Scripts 'db2cos'.

Eine Panic-Situation, ein Trap, eine Segmentierungsverletzung oder eine Ausnahmebedingung führt zu folgenden Ereignissen in der angegebenen Reihenfolge:

1. Eine Trapdatei wird erstellt.
2. Eine Signalroutine wird aufgerufen.
3. Das Script 'db2cos' wird aufgerufen (in Abhängigkeit von den aktivierten db2cos-Einstellungen).
4. Ein Eintrag wird im Protokoll mit Benachrichtigungen für die Systemverwaltung aufgezeichnet.
5. Ein Eintrag wird in der **db2diag**-Protokolldatei aufgezeichnet.

Die mit dem Befehl **db2pd** im Script 'db2cos' erfassten Standardinformationen umfassen Angaben zum Betriebssystem, zur Version und Servicestufe des installierten DB2-Produkts, zum Datenbankmanager und zur Datenbankkonfiguration sowie Angaben zu folgenden Elementen: Status der Agenten, Speicherpools, Speichergruppen, Speicherblöcke, Anwendungen, Dienstprogramme, Transaktionen, Pufferpools, Sperren, Transaktionsprotokolle, Tabellenbereiche und Container. Darüber hinaus werden Informationen zum Status der dynamischen Cachespeicher, statischen Cachespeicher und Katalogcachespeicher, zu Tabellen- und Indexstatistiken und zum Recoverystatus geliefert sowie die reoptimierten SQL-Anweisungen und eine Liste der aktiven Anweisungen angegeben. Sollen weitere Informationen erfasst werden, können Sie das Script **db2cos** mit den entsprechenden zusätzlichen Befehlen aktualisieren.

Wird das Script **db2cos** in der Standardkonfiguration aufgerufen, werden die Ausgabedateien des Scripts in dem Verzeichnis erstellt, das mit dem Konfigurationsparameter DIAGPATH des Datenbankmanagers angegeben wird. Die Namen der Dateien haben das Format XXX.YYY.ZZZ.cos.txt, wobei XXX die Prozess-ID (PID), YYY die Thread-ID (TID) und ZZZ die Datenbankpartitionsnummer (bzw. 000 bei Einzelpartitionsdatenbanken) ist. Tritt ein Trap in mehreren Threads auf, wird das Script **db2cos** für jeden Thread separat aufgerufen. Falls eine Kombination aus PID und TID mehr als einmal vorkommt, werden die Daten an die Datei angehängt. Die Iterationen der Ausgabe lassen sich anhand von Zeitmarken unterscheiden.

In Abhängigkeit von den im Script **db2cos** angegebenen Befehlen enthalten die Ausgabedateien des Scripts **db2cos** unterschiedliche Informationen. Wurde das Standardscript nicht geändert, werden Einträge ähnlich den nachstehenden Einträgen (gefolgt von der ausführlichen Ausgabe von **db2pd**) angezeigt:

```
2005-10-14-10.56.21.523659
PID     : 782348               TID  : 1           PROC : db2cos
INSTANCE: db2inst1             NODE : 0           DB   : SAMPLE
APPHDL  :                      APPID: *LOCAL.db2inst1.051014155507
FUNCTION: oper system services, sqloEDUCodeTrapHandler, probe:999
EVENT   : Invoking /home/db2inst1/sqllib/bin/db2cos from oper system
services sqloEDUCodeTrapHandler
Trap Caught

Instanz db2inst1 verwendet 64 Bit und DB2-Codefreigabe SQL09010
...
Operating System Information:

OSName:   AIX
NodeName: n1
Version:  5
Release:  2
Machine:  000966594C00

...
```

Die **db2diag**-Protokolldateien enthalten ebenfalls Einträge, die mit dem Vorkommen in Zusammenhang stehen. Beispiel:

```
2005-10-14-10.42.17.149512-300 I19441A349         LEVEL: Event
PID     : 782348               TID  : 1           PROC : db2sysc
INSTANCE: db2inst1                 NODE : 000
FUNCTION: DB2 UDB, trace services, pdInvokeCalloutScript, probe:10
START   : Invoking /home/db2inst1/sqllib/bin/db2cos from oper system
services sqloEDUCodeTrapHandler

2005-10-14-10.42.23.173872-300 I19791A310         LEVEL: Event
PID     : 782348               TID  : 1           PROC : db2sysc
INSTANCE: db2inst1                 NODE : 000
FUNCTION: DB2 UDB, trace services, pdInvokeCalloutScript, probe:20
STOP    : Completed invoking /home/db2inst1/sqllib/bin/db2cos

2005-10-14-10.42.23.519227-300 E20102A509         LEVEL: Severe
PID     : 782348               TID  : 1           PROC : db2sysc
INSTANCE: db2inst1                 NODE : 000
FUNCTION: DB2 UDB, oper system services, sqloEDUCodeTrapHandler, probe:10
MESSAGE : ADM0503C  Ein unerwarteter interner Verarbeitungsfehler ist aufgetreten.
          ALLE DIESER INSTANZ ZUGEORDNETEN DB2-PROZESSE WURDEN BEENDET.
          Diagnoseinformationen wurden aufgezeichnet. Weitere Hilfe erhalten Sie
                    beim IBM Support.

2005-10-14-10.42.23.520111-300 E20612A642         LEVEL: Severe
PID     : 782348               TID  : 1           PROC : db2sysc
INSTANCE: db2inst1                 NODE : 000
FUNCTION: DB2 UDB, oper system services, sqloEDUCodeTrapHandler, probe:20
DATA #1 : Signal Number Recieved, 4 bytes
11
DATA #2 : Siginfo, 64 bytes
0x0FFFFFFFFFFFD5C0 : 0000 000B 0000 0000 0000 0009 0000 0000    ................
0x0FFFFFFFFFFFD5D0 : 0000 0000 0000 0000 0000 0000 0000 0000    ................
0x0FFFFFFFFFFFD5E0 : 0000 0000 0000 0000 0000 0000 0000 0000    ................
0x0FFFFFFFFFFFD5F0 : 0000 0000 0000 0000 0000 0000 0000 0000    ................
```

Speicherauszugsdateien

Speicherauszugsdateien (engl. Dump Files) werden erstellt, wenn ein Fehler auftritt, für den zusätzliche Informationen verfügbar sind, die bei der Diagnose eines Problems nützlich sein könnten (z. B. interne Steuerblöcke). Jedem Datenelement, das in die Speicherauszugsdateien geschrieben wird, wird zur Unterstützung der

Problembestimmung eine Zeitmarke zugeordnet. Speicherauszugsdateien liegen im Binärformat vor und sind für Ihre Ansprechpartner bei IBM Software Support vorgesehen.

Wenn eine Speicherauszugsdatei erstellt wird oder an sie Daten angehängt werden, wird ein Eintrag in der **db2diag**-Protokolldatei aufgezeichnet, der die Uhrzeit und den Typ der geschriebenen Daten angibt. Diese Einträge in der **db2diag**-Protokolldatei sehen in etwa wie folgt aus:

```
2007-05-18-12.28.11.277956-240 I24861950A192 LEVEL: Severe
PID:1056930 TID:225448 NODE:000 Title: dynamic memory buffer
Dump File:/home/svtdbm5/sqllib/db2dump/1056930.225448.000.dump.bin
```

Anmerkung: Bei Umgebungen mit partitionierten Datenbanken gibt die Erweiterung des Dateinamens die Partitionsnummer an. Zum Beispiel gibt der folgende Eintrag an, dass die Speicherauszugsdatei durch einen DB2-Prozess erstellt wurde, der in Partition 10 ausgeführt wurde:

```
Dump File: /home/db2/sqllib/db2dump/6881492.2.010.dump.bin
```

FODC-Informationen (FODC - First Occurrence Data Capture)

Bei FODC (First Occurrence Data Capture, Datenerfassung beim ersten Vorkommen) werden Diagnoseinformationen zu einer DB2-Instanz, einem DB2-Host oder einem DB2-Member erfasst, wenn ein Problem auftritt. Durch FODC ist es nicht mehr so häufig erforderlich, ein Problem zu reproduzieren, um Diagnoseinformationen zu erhalten, da die Diagnoseinformationen bereits beim Auftreten des Problems erfasst werden können.

FODC kann manuell mit dem Befehl **db2fodc** aufgerufen werden, wenn ein Problem festgestellt wird, oder automatisch, sobald ein vordefiniertes Szenario oder Symptom auftritt. Nach der Erfassung der Diagnoseinformationen werden diese dazu verwendet, die möglichen Ursachen des Problems zu ermitteln. In einigen Fällen können Sie die Problemursache möglicherweise selbst ermitteln, in anderen wird Unterstützung durch die Mitarbeiter von IBM Support benötigt.

Nachdem die Ausführung des Befehls **db2fodc** abgeschlossen ist, muss das Tool **db2support** ausgeführt werden, um die generierten Diagnosedateien zu sammeln und das FODC-Paket für die Übergabe an IBM Support vorzubereiten. Mit dem Befehl **db2support** wird der Inhalt aller FODC-Paketverzeichnisse gesammelt, die gefunden werden bzw. die mit dem Parameter **-fodcpath** angegeben wurden. Hierdurch soll vermieden werden, dass IBM Support zusätzliche Diagnoseinformationen anfordern muss.

Erfassung von Diagnoseinformationen auf Grundlage allgemeiner Ausfälle

Diagnoseinformationen können automatisch in einem FODC-Paket (First Occurrence Data Collection, Datenerfassung beim ersten Vorkommen) erfasst werden, sobald das Problem, das eine Instanz, einen Host oder ein Mitglied betrifft, auftritt. Die Informationen im FODC-Paket können auch manuell erfasst werden.

Automatische Erfassung von Diagnoseinformationen

Der Datenbankmanager ruft den Befehl **db2fodc** für FODC auf, der seinerseits eines der DB2-Aufrufscripts (COS) aufruft.

Um den Ausfall mit den DB2-Diagnoseprotokollen und den Fehlerbehebungsdateien zu korrelieren, wird eine Diagnosenachricht an das Protokoll mit Benachrichtigungen für die Systemverwaltung und an die **db2diag**-Protokolldateien geschrieben. Der Verzeichnisname des FODC-Pakets enthält das Präfix FODC_, die Art des Ausfalls, die Zeitmarke der Erstellung des FODC-Verzeichnisses sowie die Nummer des Members bzw. der Partition, in dem bzw. der das Problem auftrat. fDie Beschreibungsdatei des FODC-Pakets wird in das neue FODC-Paketverzeichnis gestellt.

Tabelle 86. Typen und Pakete für die automatische Generierung von FODC-Daten

Paket	Beschreibung	Aufruftyp	Ausgeführtes Script
FODC_Trap_*Zeitmarke_ Membernummer*	Es ist ein Trap aufgetreten, der die gesamte Instanz betrifft.	Automatisch	db2cos_trap(.bat)
FODC_Panic_*Zeitmarke_ Membernummer*	Die Engine stellte eine Inkohärenz fest und setzt die Verarbeitung nicht fort.	Automatisch	db2cos_trap(.bat)
FODC_BadPage_*zeitmarke _Membernummer*	Es wurde eine fehlerhafte Seite festgestellt.	Automatisch	db2cos_datacorruption(.bat)
FODC_DBMarkedBad_ *Zeitmarke_Membernummer*	Eine Datenbank wurde aufgrund eines Fehlers als fehlerhaft gekennzeichnet.	Automatisch	db2cos(.bat)
FODC_IndexError_ *Zeitmarke_PID_EDUID_ Membernummer*	Es ist ein Indexfehler aufgetreten, der die gesamte EDU betrifft.	Automatisch	db2cos_indexerror_short (.bat) oder db2cos_indexerror_long (.bat)

Manuelle Erfassung von Diagnoseinformationen

Verwenden Sie den Befehl **db2fodc** manuell, wenn Sie den Verdacht haben, dass ein Problem auftritt. Problemszenarios, für die Sie Diagnosedaten erfassen können, sind zum Beispiel erkennbare Systemblockierungen, Leistungsprobleme oder Upgrade- bzw. Instanzerstellungsoperationen, die nicht wie erwartet abgeschlossen wurden. Wenn der Befehl **db2fodc** manuell ausgeführt wird, wird ein neues FODC-Paketverzeichnis erstellt. Der Verzeichnisname des FODC-Pakets enthält das Präfix FODC_, das Problemszenario, die Zeitmarke der Erstellung des FODC-Verzeichnisses sowie das/die Member bzw. die Partitionsnummer(n), in dem/denen bzw. der/denen die FODC-Operation durchgeführt wurde.

Tabelle 87. Typen und Pakete für die manuelle Generierung von FODC-Daten

Paket	Beschreibung	Aufruftyp	Ausgeführtes Script
FODC_Clp_*Zeitmarke_ Member*	Der Benutzer hat 'db2fodc -clp' aufgerufen, um umgebungs- und konfigurationsbezogene Informationen zu erfassen, die für die Fehlerbehebung bei Instanzerstellungsproblemen verwendet werden.	Manuell	db2cos_clp script(.bat)

Tabelle 87. Typen und Pakete für die manuelle Generierung von FODC-Daten (Forts.)

Paket	Beschreibung	Aufruftyp	Ausgeführtes Script
FODC_Connections_ ***Zeitmarke_Member***	Der Benutzer hat 'db2fodc -connections' aufgerufen, um verbindungsbezogene Diagnosedaten zu erfassen, die für die Diagnose von Problemen wie z. B. plötzlich auftretenden hohen Werten bei der Anzahl der Anwendungen im Ausführungs- oder Kompilierungsstatus oder verweigerten neuen Datenbankverbindungen verwendet werden.	Manuell	db2cos_threshold script(.bat)
FODC_Cpu_ ***Zeitmarke_ Member***	Der Benutzer hat 'db2fodc -cpu' aufgerufen, um prozessorbezogene Leistungs- und Diagnosedaten zu erfassen, die zur Diagnose von Problemen wie hohen Prozessorauslastungsraten, einer hohen Anzahl aktiver Prozesse oder langen Prozessorwartezeiten verwendet werden.	Manuell	db2cos_threshold script(.bat)
FODC_Hang_ ***Zeitmarke_ Memberliste***	Der Benutzer hat 'db2fodc -hang' aufgerufen, um Daten für die Fehlerbehebung bei einer Blockierung (oder schwerwiegenden Leistungsproblemen) zu erfassen.	Manuell	db2cos_hang(.bat)
FODC_Memory_ ***Zeitmarke _Member***	Der Benutzer hat 'db2fodc -memory' aufgerufen, um speicherbezogene Diagnosedaten zu erfassen, die zur Diagnose von Problemen wie der fehlenden Verfügbarkeit von freiem Speicher, der hohen Auslastung von Auslagerungsspeicher, außergewöhnlich hoher Pagingaktivität oder einem möglichen Speicherverlust verwendet werden.	Manuell	db2cos_threshold script(.bat)
FODC_Perf_ ***Zeitmarke_ Memberliste***	Der Benutzer hat 'db2fodc -perf' aufgerufen, um Daten für die Fehlerbehebung bei Leistungsproblemen zu erfassen.	Manuell	db2cos_perf(.bat)

Tabelle 87. Typen und Pakete für die manuelle Generierung von FODC-Daten (Forts.)

Paket	Beschreibung	Aufruftyp	Ausgeführtes Script
FODC_Preupgrade_ ***Zeitmarke_Member***	Der Benutzer hat 'db2fodc -preupgrade' aufgerufen, um leistungsbezogene Informationen vor der Durchführung eines kritischen Upgrades bzw. einer kritischen Aktualisierung zu erfassen, wie zum Beispiel einem Instanzupgrade oder der Aktualisierung auf das nächste Fixpack.	Manuell	db2cos_preupgrade(.bat)
Scripts in FODC_IndexError_ ***Zeitmarke_PID_EDUID_ Memberliste***	Der Benutzer kann **db2fodc -indexerror** *FODC_IndexError-Verzeichnis* [basic | full] eingeben (der Standardwert ist 'basic'), um die **db2dart**-Befehle im Script bzw. in den Scripts aufzurufen. Für DPF: **db2_all** "<<+node#< **db2fodc** -indexerror *FODC_IndexError-Verzeichnis* [basic | full]". Die Knotennummer (node#) ist die letzte Zahl im Verzeichnisnamen *FODC_IndexError-Verzeichnis*. Bei der Verwendung von **db2fodc** -indexerror mit dem Befehl **db2_all** ist ein absoluter Pfad erforderlich.	Manuell	db2cos_indexerror_long (.bat) oder db2cos_indexerror_short (.bat)

FODC-Konfiguration (First Occurrence Data Capture)

Die Funktionsweise der FODC-Konfiguration (First Occurrence Data Capture, Datenerfassung beim ersten Vorkommen), einschließlich des Speicherpfads für das FODC-Paket, wird durch die Registrierdatenbankvariable *DB2FODC* gesteuert, die mit dem Befehl **db2set** permanent festgelegt oder mit dem Befehl **db2pdcfg** dynamisch (nur speicherintern) geändert werden kann. Das FODC-Verhalten kann auch angepasst werden, indem die Call-out-Scripts (COS) aktualisiert werden, die während der FODC-Ausführung aufgerufen werden.

Jede Partition bzw. jedes Member in der Instanz verfügt über eigene FODC-Einstellungen und Sie können die FODC-Funktionsweise auf Partitions- oder Memberebene steuern. Wenn sowohl auf der Member- und Partitionsebene als auch auf der Instanzebne FODC-Einstellungen vorhanden sind, setzen die Einstellungen auf Member- und Partitionsebene die Einstellungen auf Instanzebene außer Kraft. Bei manuellem FODC können die Einstellungen auch durch Befehlszeilenparameter außer Kraft gesetzt werden, die Sie angeben, wie zum Beispiel durch den Parameter **-fodcpath**. Wenn Sie in partitionierten oder DB2 pureScale-Datenbankumgebungen eine Liste mit Membern oder Partitionen für manuelles FODC angeben, werden die Einstellungen des ersten angegebenen Members bzw. der ersten angegebenen Partition verwendet.

Permanente Einstellungen, die mit dem Befehl **db2set** festgelegt werden, werden erst wirksam, wenn die Instanz erneut gestartet wird; dynamische Einstellungen, die mit dem Befehl **db2pdcfg** vorgenommen werden, sind sofort wirksam und bleiben bis zum Neustart der Instanz im Speicher in Kraft.

Es stehen eine Reihe von Einstellungen für die Registrierdatenbankvariable *DB2FODC* zur Verfügung, mit denen Sie die Verarbeitung von FODC-Paketen steuern können; nicht alle Einstellungen sind jedoch auf allen Plattformen verfügbar. Sie können mit der Registrierdatenbankvariablen *DB2FODC* die folgenden Verhaltensweisen steuern:

- Wo werden die generierten FODC-Pakete gespeichert (mit der Einstellung für *FODCPATH*)?
- Werden Speicherauszüge für Kerndateien generiert oder nicht (mit der Einstellung für *DUMPCORE*)?
- Wie groß können Speicherauszüge für Kerndateien werden (mit der Einstellung für *CORELIMIT*)?
- Wo werden die generierten Speicherauszugsdateien gespeichert (mit der Einstellung für *DUMPDIR*)?

FODC ruft standardmäßig das Aufrufscript **db2cos** zum Erfassen von Diagnoseinformationen auf, wenn der Datenbankmanager die Verarbeitung aufgrund einer Panik- oder Trapsituation, einer Segmentverletzung oder einer Ausnahmebedingung nicht fortsetzen kann. Zum Steuern des Aufrufscripts, das während der FODC-Verarbeitung aufgerufen wird, stehen eine Reihe von *COS*-Parametereinstellungen zur Verfügung. Sie können mit dem Parameter *COS* der Registrierdatenbankvariablen *DB2FODC* die folgenden Verhaltensweisen steuern:

- Wird das Script **db2cos** aufgerufen, wenn der Datenbankmanager die Verarbeitung nicht fortsetzen kann (Einstellung *ON* bzw. *OFF*; Standardeinstellung ist *ON*)?
- Wie häufig prüft das Script **db2cos** die Größe der generierten Ausgabedateien (Einstellung *COS_SLEEP*)?
- Wie lange soll FODC auf die Beendigung des Scripts **db2cos** warten (Einstellung *COS_TIMEOUT*)?
- Wie häufig wird das Script **db2cos** während eines Datenbankmanagertraps aufgerufen (Einstellung *COS_COUNT*)?
- Wird das Script **db2cos** aktiviert, wenn das Signal SQLO_SIG_DUMP empfangen wird (Einstellung *COS_SQLO_SIG_DUMP*)?

FODC-Paketverzeichniseinstellungen (FODCPATH)

FODC-Pakete können die Generierung großer Mengen an Diagnosedaten bewirken, für die viel Speicherplatz benötigt wird und die erheblichen Systemaufwand verursachen können. Sie können steuern, in welchen Verzeichnispfad FODC die Diagnosedaten sendet, und so einen Verzeichnispfad auswählen, in dem ausreichend freier Speicherplatz verfügbar ist.

Die folgende Reihenfolge wird bei der Bestimmung des zu verwendenden FODC-Pfads angewendet:

Automatisches FODC

Einstellung der Registrierdatenbankvariablen FODCPATH
Der Parameter **FODCPATH** für die Registrierdatenbankvariable **DB2FODC** kann auf Member- bzw. Partitionsebene oder auf Instanzebene definiert werden. FODC verwendet die Einstellung des Para-

meters **FODCPATH** für jede Partition bzw. jedes Member, falls er definiert ist. Wenn keine Einstellung auf Member- bzw. Partitionsebene vorhanden ist, wird die Einstellung auf Instanzebene verwendet.

Keine FODC-Pfadeinstellungen
Wenn Sie weder auf Member- noch auf Instanzebene eine **FODCPATH**-Einstellung angeben, sendet FODC Diagnoseinformationen an den aktuellen Diagnoseverzeichnispfad (**diagpath** oder **alt_diagpath**).

Manuelles FODC

Befehlsparameteroption db2fodc -fodcpath
Beim manuellen Aufruf des Befehls **db2fodc** können Sie die Position angeben, an der das FODC-Paketverzeichnis erstellt wird, indem Sie den Parameter **-fodcpath** im Befehl angeben. Wenn Sie den Parameter **-fodcpath** mit einem gültigen Pfadnamen angeben, wird das FODC-Paketverzeichnis in diesem Pfad erstellt.

Einstellung der Registrierdatenbankvariablen FODCPATH
Wenn Sie den Parameter **-fodcpath** nicht im Befehl **db2fodc** angeben und eine Liste mit Partitionen oder Membern angegeben haben, verwendet der Befehl **db2fodc** die **FODCPATH**-Parametereinstellung für die Registrierdatenbankvariable **DB2FODC** der ersten Partition bzw. des ersten Members in der angegebenen Liste. Wenn der Wert für diesen Parameter **FODCPATH** nicht definiert ist, verwendet **db2fodc** die **FODCPATH**-Einstellung auf Instanzebene. Wenn Sie den Parameter **-fodcpath** nicht angeben und auch keine Liste mit Partitionen oder Membern, versucht der Befehl **db2fodc** zuerst, die **FODCPATH**-Parametereinstellung für die aktuelle Partition bzw. das aktuelle Member zu verwenden; ist diese nicht definiert, wird die Einstellung auf Instanzebene verwendet.

Keine FODC-Pfadeinstellungen
Wenn Sie keinen FODC-Pfad angeben, sendet FODC die Diagnoseinformationen standardmäßig an den aktuellen Diagnoseverzeichnispfad (**diagpath** oder **alt_diagpath**).

Angenommen, Sie verwenden eine Umgebung mit partitionierten Datenbanken, die 3 Member oder Partitionen enthält (0, 1 und 2). Das folgende Beispiel zeigt, wie der FODC-Pfad mit dem Befehl **db2set** permanent auf Instanzebene für alle 3 Partitionen bzw. Member definiert wird:

```
db2set DB2FODC=FODCPATH=/home/hotel49/juntang/FODC
```

Die Einstellungen für den FODC-Pfad können auch für jedes Member auf Memberebene permanent vorgenommen werden, wodurch die Einstellungen auf Instanzebene außer Kraft gesetzt werden. Damit diese Einstellungen wirksam werden, muss die Instanz neu gestartet werden. Geben Sie zum Beispiel den folgenden Befehl ein, um den FODC-Pfad für das Member 0 zu ändern:

```
db2set DB2FODC=FODCPATH=/home/hotel49/juntang/FODC/FODC0 –i juntang 0
```

Wenn Sie nun den FODC-Pfad dynamisch für Member 1 und Member 2 ändern möchten, verwenden Sie die folgenden **db2pdcfg**-Befehle. Diese Einstellungen werden sofort wirksam und bleiben bis zum Neustart der Instanz im Speicher bestehen.

```
db2pdcfg -fodc FODCPATH=/home/hotel49/juntang/FODC/FODC1 -member 1

db2pdcfg -fodc FODCPATH=/home/hotel49/juntang/FODC/FODC2 -member 2
```

Wenn Sie wissen möchten, wie die aktuellen FODC-Einstellungen für die einzelnen Member bzw. Partitionen in einem System lauten, können Sie den Befehl **db2pdcfg -fodc -member all** verwenden (im Beispiel ist die Ausgabe gekürzt und es wird nur die Ausgabe für den FODC-Pfad dargestellt):

```
Database Member 0
FODC package path (FODCPATH)= /home/hotel49/juntang/FODC/FODC0/

Database Member 1
FODC package path (FODCPATH)= /home/hotel49/juntang/FODC/FODC1/

Database Member 2
FODC package path (FODCPATH)= /home/hotel49/juntang/FODC/FODC2/
```

Angepasste Datenerfassung

Das Verhalten der Datenerfassung mit **db2fodc -hang** und **db2fodc -perf** wird auch durch die Parameter gesteuert, die im Abschnitt TOOL OPTIONS des DB2-Aufrufscripts definiert sind, das während der FODC-Ausführung aufgerufen wird. Diese Parameter können durch Änderungen des Scripts, das während der FODC-Ausführung aufgerufen wird, angepasst werden.

Zum Anpassen der Datenerfassung unter UNIX kopieren Sie das Script in `/bin/db2cos_`*`Symptom`* in `/adm/db2cos_`*`Symptom`*. Dabei ist *Symptom* entweder `hang` oder `perf`. Wenn sich das Script in diesem neuen Verzeichnis befindet, können Sie es nach Bedarf modifizieren. Unter Windows modifizieren Sie das Standardscript `\bin\db2cos_`*`Symptom`*`.bat`. Unter UNIX versucht **db2fodc** zuerst, das Script in `/adm/db2cos_`*`Symptom`* auszuführen; wird das Script nicht gefunden, wird das ursprüngliche Script in `/bin/db2cos_`*`Symptom`* ausgeführt. Unter Windows wird stets das Script `\bin\db2cos_`*`Symptom`*`.bat` ausgeführt.

Im Rahmen von FODC erfasste Daten

Bei der Ausführung von FODC (First Occurrence Data Capture, Datenerfassung beim ersten Vorkommen) werden ein FODC-Paketverzeichnis sowie Unterverzeichnisse erstellt, in denen Diagnoseinformationen erfasst werden. Das übergeordnete Paketverzeichnis, die Unterverzeichnisse und die Dateien, in denen die Diagnosedaten erfasst werden, werden zusammen als FODC-Paket bezeichnet.

Dateien mit den von FODC erfassten Diagnoseinformationen

FODC erfasst Diagnoseinformationen aus einer Reihe verschiedener Quellen. Welche Diagnoseinformationen genau von FODC erfasst werden, ist abhängig von der Art des auftretenden Problems. Sie können Folgendes umfassen:

Protokoll mit Benachrichtigungen für die Systemverwaltung (*instanzname***.nfy)**

- Betriebssystem: Alle
- Standardposition:
  - Linux und UNIX: In dem Verzeichnis, das durch den Konfigurationsparameter **diagpath** des Datenbankmanagers angegeben ist.
  - Windows: Verwenden Sie die Ereignisanzeigefunktion (**Start** > **Systemsteuerung** > **Verwaltung** > **Ereignisanzeige**)
- Wird bei der Erstellung der Instanz automatisch generiert.
- Wenn signifikante Ereignisse auftreten, schreibt DB2 Informationen in das Protokoll mit Benachrichtigungen für die Systemverwaltung. Die Informationen sind für Datenbank- und Systemadministratoren gedacht. Der Typ der in dieser Datei aufgezeichneten Nachrichten hängt von der Einstellung des Konfigurationsparameters **notifylevel** ab.

Anmerkung: Wenn der Konfigurationsparameter **diagsize** des Datenbankmanagers auf einen Wert ungleich null gesetzt wird, werden anstelle einer einzelnen Protokolldatei mit Benachrichtigungen für die Systemverwaltung (*instanzname*.nfy) rollierende Protokolldateien (*instanzname.N*.nfy) verwendet.

DB2-Diagnoseprotokoll (db2diag.log)

- Betriebssystem: Alle
- Standardposition: In dem Verzeichnis, das durch den Konfigurationsparameter **diagpath** des Datenbankmanagers angegeben ist.
- Wird bei der Erstellung der Instanz automatisch generiert.
- Diese Textdatei enthält Diagnoseinformationen zu den in der Instanz festgestellten Fehlern und Warnungen. Diese Informationen dienen der Fehlerbehebung und sind für die Techniker von IBM Software Support konzipiert. Der Typ der in dieser Datei aufgezeichneten Nachrichten hängt von der Einstellung des Konfigurationsparameters **diaglevel** des Datenbankmanagers ab.

  Anmerkung: Wird der Konfigurationsparameter **diagsize** des Datenbankmanagers auf einen Wert ungleich null gesetzt wird, werden anstelle einer einzelnen Protokolldatei (db2diag.log) rollierende Protokolldateien (db2diag.*N*.log) verwendet.

Diagnoseprotokoll des DB2-Verwaltungsservers (DAS) (db2dasdiag.log)

- Betriebssystem: Alle
- Standardposition:
  - Linux und UNIX: Im Verzeichnis `DASHOME/das/dump`, wobei `DASHOME` das Ausgangsverzeichnis des DAS-Eigners ist.
  - Windows: Im Ordner "dump" im DAS-Ausgangsverzeichnis. Beispiel: `C:\Program Files\IBM\SQLLIB\DB2DAS00\dump`
- Wird bei der Erstellung des DAS automatisch generiert.
- Diese Textdatei enthält Diagnoseinformationen zu den vom DAS festgestellten Fehlern und Warnungen.

DB2-Ereignisprotokoll (db2eventlog.xxx, wobei xxx die Datenbankpartitionsnummer ist)

- Betriebssystem: Alle
- Standardposition: In dem Verzeichnis, das durch den Konfigurationsparameter **diagpath** des Datenbankmanagers angegeben ist.
- Wird bei der Erstellung der Instanz automatisch generiert.
- Die DB2-Ereignisprotokolldatei ist ein Umlaufprotokoll für Ereignisse auf Infrastrukturebene, die im Datenbankmanager auftreten. Die Datei hat eine festgelegte Größe und fungiert als Umlaufpuffer für die bestimmten Ereignisse, die protokolliert werden, während die Instanz ausgeführt wird. Sobald die Instanz gestoppt wird, wird das vorherige Ereignisprotokoll ersetzt und nicht angehängt. Wird die Instanz durch einen Fehler unterbrochen, wird außerdem eine Datei namens 'db2eventlog.XXX.crash' generiert. Diese Dateien sind für die Verwendung durch IBM Software Support vorgesehen.

Ausgabedateien des DB2-Aufrufscripts (db2cos)

- Betriebssystem: Alle
- Standardposition: In dem Verzeichnis, das durch den Konfigurationsparameter **diagpath** des Datenbankmanagers angegeben ist.

- Wird das Script 'db2cos' aufgrund eines FODC-Ausfalls ausgeführt, werden die db2cos-Ausgabedateien unter dem FODC-Verzeichnis gespeichert, das an der durch den Konfigurationsparameter **diagpath** des Datenbankmanagers angegebenen Position erstellt wurde.
- Werden bei Auftreten einer Panic-Situation, einer Fehlerunterbrechung (Trap) oder eines Segmentierungsverstoßes automatisch erstellt. Können auch bei bestimmten Problemszenarien erstellt werden, die mit dem Befehl **db2pdcfg** angegeben wurden.
- In der Standardkonfiguration ruft das Script 'db2cos' **db2pd**-Befehle zum entsperrten Erfassen von Informationen auf. Der Inhalt der db2cos-Ausgabedateien variiert je nach den im Script 'db2cos' enthaltenen Befehlen (z. B. Betriebssystembefehle und andere DB2-Diagnosetools). Öffnen Sie die Scriptdatei mit einem Texteditor, wenn Sie sich über die mit dem Script 'db2cos' ausgeführten Tools informieren möchten.
- Das Script 'db2cos' wird im Verzeichnis `bin/` geliefert. Unter UNIX ist dieses Verzeichnis schreibgeschützt. Um Ihre eigene veränderbare Version dieses Scripts zu erstellen, kopieren Sie das Script 'db2cos' in das Verzeichnis `adm/`. Sie können diese Version des Scripts wie gewünscht verändern. Wenn das Script sich im Verzeichnis `adm/` befindet, so ist dies die Version, die ausgeführt wird. Andernfalls wird die Standardversion im Verzeichnis `bin/` ausgeführt.

Speicherauszugsdateien

- Betriebssystem: Alle
- Standardposition: In dem Verzeichnis, das durch den Konfigurationsparameter **diagpath** des Datenbankmanagers angegeben ist.
- Werden diese Dateien während eines FODC-Ausfalls gespeichert, werden sie in das FODC-Verzeichnis platziert.
- Werden bei Auftreten bestimmter Problemszenarien automatisch erstellt.
- Für einige Fehlerbedingungen werden Zusatzinformationen in Binärdateien protokolliert, die nach der Prozess-ID des fehlgeschlagenen Prozesses benannt werden. Diese Dateien sind für die Verwendung durch IBM Software Support vorgesehen.

Trapdateien

- Betriebssystem: Alle
- Standardposition: In dem Verzeichnis, das durch den Konfigurationsparameter **diagpath** des Datenbankmanagers angegeben ist.
- Werden diese Dateien während eines FODC-Ausfalls gespeichert, werden sie in das FODC-Verzeichnis platziert.
- Werden bei abnormaler Beendigung der Instanz automatisch erstellt. Können auch beliebig mithilfe des Befehls **db2pd** erstellt werden.
- Der Datenbankmanager generiert eine Trapdatei, wenn er die Verarbeitung aufgrund einer Fehlerunterbrechung (Trap), eines Segmentierungsverstoßes oder einer Ausnahmebedingung nicht fortsetzen kann.

Kerndateien

- Betriebssystem: Linux und UNIX
- Standardposition: In dem Verzeichnis, das durch den Konfigurationsparameter **diagpath** des Datenbankmanagers angegeben ist.
- Werden diese Dateien während eines FODC-Ausfalls gespeichert, werden sie in das FODC-Verzeichnis platziert.

- Werden bei abnormaler Beendigung der DB2-Instanz vom Betriebssystem erstellt.
- Das Kernimage enthält unter anderem die meisten oder alle Hauptspeicherzuordnungen von DB2, die möglicherweise für die Problembeschreibung erforderlich sind.

FODC-Paketpfad und -inhalt

FODC erstellt das FODC-Paketverzeichnis im angegebenen FODC-Pfad. Der FODC-Pfad wird durch die Einstellung der Registrierdatenbankvariablen **FODCPATH** oder durch die Befehlsparameteroption **db2fodc -fodcpath** angegeben. Wenn Sie keinen FODC-Pfad angeben, sendet FODC die Diagnoseinformationen an den aktuellen Diagnoseverzeichnispfad (**diagpath** oder **alt_diagpath**). Eine Diagnosenachricht wird in der **db2diag**-Protokolldatei aufgezeichnet, um den für FODC verwendeten Verzeichnisnamen anzugeben. Bei der Erfassung von Diagnoseinformationen kann - abhängig von den angegebenen Parametern - eine erhebliche Menge an Diagnosedaten entstehen, daher muss im Verzeichnispfad, in dem FODC die Diagnoseinformationen speichert, ausreichend Speicherplatz verfügbar sein. Um ein Szenario zu vermeiden, bei dem FODC den gesamten verfügbaren Speicherplatz im Dateisystem belegt und damit eine Beeinträchtigung des Datenservers verursacht, empfiehlt es sich, einen FODC-Pfad anzugeben, der zum Speichern der FODC-Diagnosedaten geeignet ist.

Wird FODC automatisch ausgeführt, wird ein Paket für das Member bzw. die Partition erstellt, in dem bzw. der das Problem auftritt; tritt das Problem in mehreren Membern auf, werden mehrere Pakete in separaten FODC-Paketverzeichnissen erstellt. Für das FODC-Paketverzeichnis gilt die Namenskonvention 'FODC_*art des ausfalls_zeitmarke_membernummer*', wobei *art des ausfalls* das Problemsymptom, *zeitmarke* der Zeitpunkt des FODC-Aufrufs und *membernummer* die Nummer des Members bzw. der Partition ist, in dem bzw. der das Problem auftrat. Beispiel: Wenn eine Unterbrechung in Member 1 auftritt, kann FODC automatisch ein Paket mit einem Namen ähnlich `FODC_Trap_ 2010-11-17-20.58.30.695243_0001` erstellen.

Bei einer manuellen Ausführung von FODC wird ein Paket für das/die Member bzw. die Partition(en) erstellt, das/die Sie angeben. Die Namenskonvention für das FODC-Paketverzeichnis lautet 'FODC_*art des ausfalls (manuell)_zeitmarke_memberliste*'. Dabei gilt Folgendes: *art des ausfalls (manuell)* ist das Problemsymptom, *zeitmarke* ist der Zeitpunkt des FODC-Aufrufs und *memberliste* ist eine Liste der Member oder Partitionen, in denen das Problem auftrat. Der manuell eingegebene Befehl **db2fodc –hang –basic –member 1,2,3 –db sample** erstellt beispielsweise ein manuelles FODC-Paket für die Member 1, 2 und 3 mit einem Namen ähnlich dem folgenden: `FODC_hang_ 2010-11-17-20.58.30.695243_0001.0002.0003`.

Eines oder mehrere der folgenden Unterverzeichnisse wird im FODC-Paketverzeichnis erstellt:

- DB2CONFIG mit DB2-Konfigurationsausgabe und -dateien
- DB2PD mit **db2pd**-Ausgabe oder -Ausgabedateien
- DB2SNAPS mit DB2-Momentaufnahmen
- DB2TRACE mit DB2-Traces
- OSCONFIG mit Betriebssystemkonfigurationsdateien
- OSSNAPS mit Betriebssystemüberwachungsdaten
- OSTRACE mit Betriebssystemtraces

Abhängig von der jeweiligen FODC-Konfiguration und der Art des Ausfalls, für den der Befehl **db2fodc** ausgeführt wird, sind möglicherweise nicht alle diese Verzeichnisse vorhanden.

FODC sendet die folgenden Diagnoseinformationen an das FODC-Paketverzeichnis:

Mit `db2fodc -clp` werden die folgenden Informationen erfasst:

- Informationen zum Betriebssystem.
- Informationen zur Instanz und zur Datenbankkonfiguration.

Mit `db2fodc -hang` werden die folgenden Informationen erfasst:

- Grundlegende Informationen zum Betriebssystem. Das Problem ist möglicherweise auf die Betriebssystemversion, Programmkorrekturen usw. zurückzuführen.
- Grundlegende Informationen zur DB2-Konfiguration.
- Informationen der Betriebssystemüberwachung: vmstat, netstat, iostat usw.
  - Mindestens zwei Iterationen: mit gespeicherten Zeitmarken.
- Partielle Aufrufstacks: DB2-Stack-Traces für die wichtigsten CPU-Agenten.
- Betriebssystemtrace: Trace für AIX.
- Von **db2pd** erfasste Diagnoseinformationen.
- DB2-Trace.
- Vollständige DB2-Aufrufstacks.
- DB2-Konfigurationsinformationen (zweiter Durchlauf).
  - Mit zweiter DB2-Traceerstellung.
- Momentaufnahmeninformationen: **db2 get snapshot** für Datenbanken, Anwendungen, Tabellen usw.
  - Falls mehrere logische Knoten vorhanden sind, werden die Informationen pro Knoten erfasst.

Mit `db2fodc –indexerror` werden die folgenden Informationen erfasst:

- Basismodus
  - Das Script `db2cos_indexerror_short(.bat)` wird ausgeführt. Das Script enthält zusätzliche Details hierzu.
  - Falls anwendbare **db2dart**-Befehle im Script vorhanden sind, werden die **db2dart /DD**- und/oder **db2dart /DI**-Datenformatierungsaktionen mit einer maximalen Seitenanzahl von 100 ausgeführt.
- Vollständiger Modus
  - Die Scripts `db2cos_indexerror_short(.bat)` und `db2cos_indexerror_long(.bat)` werden ausgeführt. Die Scripts enthalten zusätzliche Details hierzu.
  - Falls anwendbare **db2dart**-Befehle im Script `db2cos_indexerror_short(.bat)` vorhanden sind, werden die **db2dart /DD**- und/oder **db2dart /DI**-Datenformatierungsaktionen mit einer maximalen Seitenanzahl von 100 ausgeführt.
  - Falls anwendbare **db2dart**-Befehle im Script `db2cos_indexerror_long(.bat)` vorhanden sind, werden die **db2dart /DD**- und/oder **db2dart /DI**-Datenformatierungsaktionen ohne Begrenzung der Seitenanzahl ausgeführt.

– Falls anwendbare **db2dart**-Befehle im Script db2cos_indexerror_long(.bat) vorhanden sind, wird der Befehl **db2dart /T** ausgeführt. Für diesen Befehl ist es erforderlich, dass die Datenbank offline ist.

Mit db2fodc –perf wird das System überwacht; gegebenenfalls werden die folgenden Informationen erfasst:

- Momentaufnahmen.
- Stapeltraces.
- Virtueller Speicher (Vmstat).
- Ein-/Ausgabeinformationen (Iostat).
- Traces.
- Abhängig vom jeweiligen Fall weitere Informationen. Das Script enthält zusätzliche Details hierzu.

Mit db2fodc -preupgrade werden die folgenden Informationen erfasst:

- Informationen zum Betriebssystem.
- Instanz- und Datenbankkonfigurationsinformationen wie die Ausgabe des Befehls **db2level**, Umgebungsvariablen, die Ausgabe des Befehls **db2 get dbm cfg** und die Datei db2nodes.cfg.
- Systemkatalogdaten und -statistikinformationen, zum Beispiel Daten des Optimierungsprogramms, die mit dem Befehl **db2support -d** *dbname* **-c -s -cl 0** erfasst werden.
- Daten der Betriebssystemüberwachung, wie zum Beispiel die Ausgabe der Befehle **netstat -v** und **ps -elf**.
- Systemdateien.
- Paketdaten, die vom Befehl DB2 LIST PACKAGES FOR SCHEMA *schemaname* SHOW DETAIL für alle Schemanamen zurückgegeben werden.
- Alle 'FODC_Preupgrade'-Verzeichnisse, die in db2dump/ gefunden werden. Diese Verzeichnisse enthalten Informationen wie zum Beispiel Leistungsdaten, die wichtigsten dynamischen SQL-Afragen und EXPLAIN-Pläne.
- Die Protokolldatei des Befehls **db2ckupgrade** in '/tmp/db2ckupgrade.log.*prozess-ID*', falls vorhanden.

Die folgenden Diagnoseinformationen werden ebenfalls erfasst, wenn Sie die Member angeben, für die die Erfassung durchgeführt werden soll:

- Momentaufnahmen (nach der Aktivierung aller Monitorschalter).
- Die Ausgabe des Befehls **db2pd** für die Parameter -everything, -agents, -applications, -mempools und -fcm.
- Die am häufigsten verwendeten dynamischen SQL-Anweisungen.
- Die Abfragepläne für SQL-Anweisungen.
- Die EXPLAIN-Pläne für statische Pakete.

Bei einem manuellen Aufruf des Befehls **db2fodc** wird eine Protokolldatei mit dem Namen db2fodc_*symptom*.log im Verzeichnis FODC_*symptom* erstellt, wobei *symptom* eine der Erfassungsarten bezeichnet, wie zum Beispiel hang oder perf. In dieser Datei speichert der Befehl **db2fodc** auch Statusinformationen und Metadaten, die das FODC-Paket im FODC-Unterverzeichnis beschreiben. Diese Datei enthält Informationen zur FODC-Art, die Zeitmarke für den Start und das Ende der Datenerfassung sowie weitere Informtionen, die zur Analyse des FODC-Pakets nützlich sind.

Automatische Generierung von FODC-Daten

Wenn eine Betriebsunterbrechung auftritt und die automatische FODC aktiviert ist, werden die Daten auf Basis der Symptome erfasst. Die erfassten Daten sind auf die Bedürfnisse bei der Diagnose der Betriebsunterbrechung abgestimmt.

Es werden eine oder mehrere Nachrichten, einschließlich der als "kritisch" definierten Nachrichten, verwendet, um den Ursprung der Betriebsunterbrechung zu markieren.

Trapdateien enthalten Informationen wie die Folgenden:

- Die Größe des freien virtuellen Speichers
- Werte, die den Konfigurationsparametern und Registrierdatenbankvariablen des Produkts zu dem Zeitpunkt zugeordnet sind, zu dem der Trap auftrat.
- Geschätzte Größe des Speichers, der vom DB2-Produkt zum Zeitpunkt des Traps genutzt wurde
- Informationen, die einen Kontext für die Betriebsunterbrechung zur Verfügung stellen

Der unformatierte Stack-Speicherauszug kann in einer ASCII-Trapdatei eingeschlossen sein.

Speicherauszugsdateien, die für die Komponenten innerhalb des Datenbankmanagers spezifisch sind, werden im entsprechenden FODC-Paketverzeichnis gespeichert.

DB2 Query Patroller und First Occurrence Data Capture (FODC)

Wenn Sie der Meinung sind, Fehler bei DB2 Query Patroller untersuchen zu müssen, stehen Ihnen Protokolle mit Informationen zu den möglichen Ursachen der aufgetretenen Schwierigkeiten bzw. Fehler zur Verfügung.

qpdiag.log

- Betriebssysteme: Alle
- Standardposition: In dem Verzeichnis, das durch den Konfigurationsparameter **diagpath** des Datenbankmanagers angegeben ist.
- Wird automatisch erstellt, wenn das Query Patroller-System aktiviert wird.
- Enthält Informations- und Diagnosesätze für Query Patroller. Diese Informationen dienen der Fehlerbehebung und sind zur Verwendung durch IBM Software Support konzipiert.

qpmigrate.log

- Betriebssysteme: Alle
- Standardposition: In dem Verzeichnis, das durch den Konfigurationsparameter **diagpath** des Datenbankmanagers angegeben ist.
- Wird automatisch durch das Dienstprogramm **qpmigrate** erstellt. Der Befehl **qpmigrate** kann implizit bei der Installation von Query Patroller ausgeführt werden (wenn Sie eine vorhandene Datenbank angeben, auf der Query Patroller ausgeführt werden soll), oder explizit nach der Installation.
- Erfasst Informationen und Fehlernachrichten bei der Migration von Query Patroller auf eine andere Version. Zur Verwendung durch Query Patroller-Administratoren konzipiert.

qpsetup.log

- Betriebssysteme: Alle
- Standardposition: In dem Verzeichnis, das durch den Konfigurationsparameter **diagpath** des Datenbankmanagers angegeben ist.
- Wird automatisch durch das Dienstprogramm **qpsetup** erstellt. Der Befehl **qpsetup** kann implizit bei der Installation von Query Patroller ausgeführt werden (wenn Sie eine vorhandene Datenbank angeben, auf der Query Patroller ausgeführt werden soll), oder explizit nach der Installation.
- Erfasst Informationen und Fehlernachrichten, die während der Ausführung des Dienstprogramms **qpsetup** auftreten. Zur Verwendung durch Query Patroller-Administratoren konzipiert.

qpuser.log

- Betriebssysteme: Alle
- Standardposition: In dem Verzeichnis, das durch den Konfigurationsparameter **diagpath** des Datenbankmanagers angegeben ist.
- Wird automatisch erstellt, wenn das Query Patroller-System aktiviert wird.
- Enthält Informationsnachrichten zu Query Patroller. Diese geben beispielsweise an, wann Query Patroller gestartet und gestoppt wird. Zur Verwendung durch Query Patroller-Administratoren konzipiert.

Überwachungs- und Prüffunktionen, die FODC (First Occurrence Data Capture) verwenden

Wenn Sie Fehler bei der Überwachungs- und Prüffunktion untersuchen müssen, stehen Ihnen Protokolle mit Informationen zu den möglichen Ursachen der aufgetretenen Schwierigkeiten zur Verfügung.

DB2-Prüfprotokoll ("db2audit.log")

- Betriebssysteme: Alle
- Standardposition:
  - Windows: Verzeichnis `$DB2PATH\`*`instanzname`*`\security`.
  - Linux und UNIX: Verzeichnis `$HOME\sqllib\security`, wobei $HOME das Ausgangsverzeichnis des Instanzeigners ist.
- Wird erstellt, wenn die Funktion **db2audit** gestartet wird.
- Enthält Prüfsätze, die durch die DB2-Prüffunktion für eine Reihe vordefinierter Datenbankereignisse generiert werden.

DB2-Governorprotokoll ("*mylog.x*", wobei *x* die Anzahl der Datenbankpartitionen angibt, auf denen der Governor ausgeführt wird)

- Betriebssysteme: Alle
- Standardposition:
  - Windows: Verzeichnis `$DB2PATH\`*`instanzname`*`\log`.
  - Linux und UNIX: Verzeichnis `$HOME\sqllib\log`, wobei $HOME das Ausgangsverzeichnis des Instanzeigners ist.
- Wird erstellt, wenn das Governordienstprogramm verwendet wird. Die Basis des Protokolldateinamens wird im Befehl **db2gov** angegeben.
- Erfasst Informationen zu Aktionen, die der Governordämon ausführt (z. B. Beenden einer Anwendung erzwingen, Governorkonfigurationsdatei lesen, Dienstprogramm starten oder beenden), sowie Fehler und Warnungen.

Ereignismonitordatei (z. B. "00000000.evt")

- Betriebssysteme: Alle
- Standardposition: Wenn Sie einen Dateiereignismonitor erstellen, werden alle Ereignisdatensätze in das Verzeichnis geschrieben, das in der Anweisung CREATE EVENT MONITOR angegeben ist.
- Wird vom Ereignismonitor generiert, wenn Ereignisse auftreten.
- Enthält Ereignisdatensätze, die dem Ereignismonitor zugeordnet sind.

Grafische Tools, die FODC (First Occurrence Data Capture) verwenden

Wenn Sie Fehler beim Befehlseditor, bei der Data Warehouse-Zentrale oder der Informationskatalogzentrale untersuchen müssen, stehen Ihnen Protokolle mit Informationen zu den möglichen Ursachen der aufgetretenen Fehler bzw. Schwierigkeiten zur Verfügung.

Befehlseditorprotokoll

- Betriebssysteme: Alle
- Standardposition: Der Name und die Position dieser Protokolldatei werden mithilfe der Befehlseditorseite der DB2-Funktionsleiste angegeben. Wenn kein Pfad angegeben wird, wird das Protokoll im Verzeichnis `$DB2PATH\sqllib\tools` unter Windows und im Verzeichnis `$HOME/sqllib/tools` unter Linux und UNIX gespeichert, wobei HOME das Ausgangsverzeichnis des Instanzeigners ist.
- Wird erstellt, wenn Sie **Befehlsprotokoll in Datei aufzeichnen** im Befehlseditor auswählen und die Datei sowie die Position angeben.
- Enthält das Verlaufsprotokoll der Befehls- und Anweisungsdurchführung des Befehlseditors.

Datei IWH2LOGC.log der Data Warehouse-Zentrale

- Betriebssysteme: Alle
- Standardposition: Verzeichnis, das durch die Umgebungsvariable VWS_LOGGING angegeben ist. Der Standardpfad ist das Verzeichnis `$DB2PATH\sqllib\logging` unter Windows und das Verzeichnis `$HOME/sqllib/logging` unter Linux und UNIX, wobei HOME das Ausgangsverzeichnis des Instanzeigners ist.
- Wird automatisch durch die Data Warehouse-Zentrale erstellt, wenn die Protokollfunktion gestoppt wird.
- Enthält Nachrichten, die von der Data Warehouse-Zentrale und dem OLE-Server erstellt wurden und die beim Stoppen der Protokollfunktion nicht gesendet werden konnten. Dieses Protokoll kann mithilfe des Fensters der Protokollanzeigefunktion in der Data Warehouse-Zentrale angezeigt werden.

Datei IWH2LOG.log der Data Warehouse-Zentrale

- Betriebssysteme: Alle
- Standardposition: Verzeichnis, das durch die Umgebungsvariable VWS_LOGGING angegeben ist. Der Standardpfad ist das Verzeichnis `$DB2PATH\sqllib\logging` unter Windows und das Verzeichnis `$HOME/sqllib/logging` unter Linux und UNIX, wobei HOME das Ausgangsverzeichnis des Instanzeigners ist.
- Wird automatisch durch die Data Warehouse-Zentrale erstellt, wenn diese nicht gestartet werden kann oder wenn die Traceerstellung aktiviert ist.
- Enthält Diagnoseinformationen für Situationen, in denen die Protokollfunktion der Data Warehouse-Zentrale nicht gestartet werden und keine

Daten in das Protokoll der Data Warehouse-Zentrale (IWH2LOGC.log) schreiben kann. Dieses Protokoll kann mithilfe des Fensters der Protokollanzeigefunktion in der Data Warehouse-Zentrale angezeigt werden.

Datei IWH2SERV.log der Data Warehouse-Zentrale

- Betriebssysteme: Alle
- Standardposition: Verzeichnis, das durch die Umgebungsvariable VWS_LOGGING angegeben ist. Der Standardpfad ist das Verzeichnis `$DB2PATH\sqllib\logging` unter Windows und das Verzeichnis `$HOME/sqllib/logging` unter Linux und UNIX, wobei HOME das Ausgangsverzeichnis des Instanzeigners ist.
- Wird automatisch durch die Server-Tracefunktion der Data Warehouse-Zentrale erstellt.
- Enthält Nachrichten zum Starten der Data Warehouse-Zentrale sowie Nachrichten, die durch die Server-Tracefunktion erstellt werden. Dieses Protokoll kann mithilfe des Fensters der Protokollanzeigefunktion in der Data Warehouse-Zentrale angezeigt werden.

EXPORT-Protokoll der Befehlsdatei der Informationskatalogzentrale

- Betriebssysteme: Alle
- Standardposition: Der Pfad der exportierten Befehlsdatei und der Name der Protokolldatei werden auf der Indexzunge **Optionen** des Exporttools in der Informationskatalogzentrale angegeben.
- Wird vom Exporttool in der Informationskatalogzentrale generiert.
- Enthält Informationen zum Export der Befehlsdatei, wie beispielsweise Uhrzeit und Datum des Startens und Stoppens des Exportprozesses. Darüber hinaus sind alle Fehlernachrichten enthalten, die während der Exportoperation festgestellt wurden.

IMPORT-Protokoll der Befehlsdatei der Informationskatalogzentrale

- Betriebssysteme: Alle
- Standardposition: Der Pfad der importierten Befehlsdatei und der Name der Protokolldatei werden im Importtool in der Informationskatalogzentrale angegeben.
- Wird vom Importtool in der Informationskatalogzentrale generiert.
- Enthält Informationen zum Import der Befehlsdatei, wie beispielsweise Uhrzeit und Datum des Startens und Stoppens des Importprozesses. Darüber hinaus sind alle Fehlernachrichten enthalten, die während der Importoperation festgestellt wurden.

Interne Rückkehrcodes

Es gibt zwei Arten interner Rückkehrcodes: ZRC-Werte und ECF-Werte. Hierbei handelt es sich um Rückkehrcodes, die normalerweise nur in den von IBM Software Support verwendeten Diagnosetools angezeigt werden.

Sie werden beispielsweise in der DB2-Traceausgabe und in den **db2diag**-Protokolldateien angezeigt.

ZRC- und ECF-Werte dienen grundsätzliche demselben Zweck, unterscheiden sich jedoch geringfügig in ihrem Format. Jeder ZRC-Wert weist die folgenden Merkmale auf:

- Klassenname
- Komponente

- Ursachencode
- Zugehöriger SQLCODE
- SQLCA-Nachrichtentoken
- Beschreibung

ECF-Werte bestehen dagegen aus folgenden Elementen:

- Gruppenname
- Produkt-ID
- Komponente
- Beschreibung

ZRC- und ECF-Werte sind normalerweise negative Zahlen und stellen Fehlerbedingungen dar. ZRC-Werte sind in Gruppen zusammengefasst, die dem Fehlertyp entsprechen, den sie darstellen. Diese Gruppierungen werden als Klassen bezeichnet. So beziehen sich ZRC-Werte, deren Namen mit „SQLZ_RC_MEMHEP" beginnen, normalerweise auf Fehler im Zusammenhang mit Speicherknappheit. ECF-Werte werden in ähnlicher Weise in Gruppen zusammengefasst, die als "Sets" bezeichnet werden.

Ein Beispiel für einen Eintrag in einer **db2diag**-Protokolldatei, der einen ZRC-Wert enthält, ist nachfolgend aufgeführt:

```
2006-02-13-14.34.35.965000-300    I17502H435       LEVEL: Error
PID     : 940                  TID  : 660         PROC : db2syscs.exe
INSTANCE: DB2                  NODE : 000         DB   : SAMPLE
APPHDL  : 0-1433                APPID: *LOCAL.DB2.050120082811
FUNCTION: DB2 UDB, data protection, sqlpsize, probe:20
RETCODE : ZRC=0x860F000A=-2045837302=SQLO_FNEX "File not found."
          DIA8411C Eine Datei "" konnte nicht gefunden werden.
```

Vollständige Details zu diesem ZRC-Wert können mithilfe des Befehls **db2diag** abgerufen werden. Beispiel:

```
c:\>db2diag -rc 0x860F000A

Input ZRC string '0x860F000A' parsed as 0x860F000A (-2045837302).

ZRC value to map: 0x860F000A (-2045837302)
        V7 Equivalent ZRC value: 0xFFFFE60A (-6646)

ZRC class :
        Critical Media Error (Class Index: 6)
Component:
        SQLO ; oper system services (Component Index: 15)
Reason Code:
        10 (0x000A)

Identifer:
        SQLO_FNEX
        SQLO_MOD_NOT_FOUND
Identifer (without component):
        SQLZ_RC_FNEX

Description:
        File not found.

Associated information:
        Sqlcode -980
SQL0980C  Datenträgerfehler. Nachfolgende SQL-Anweisungen können
```

```
nicht verarbeitet werden.

        Number of sqlca tokens : 0
        Diaglog message number: 8411
```

Dieselben Informationen werden zurückgegeben, wenn Sie die Befehle **db2diag -rc -2045837302** bzw. **db2diag -rc SQLO_FNEX** eingeben.

Ein Beispiel für die Ausgabe für einen ECF-Rückkehrcode ist nachfolgend dargestellt:

```
c:\>db2diag -rc 0x90000076

Input ECF string '0x90000076' parsed as 0x90000076 (-1879048074).

ECF value to map: 0x90000076 (-1879048074)

ECF Set :
        setecf (Set index : 1)
Product :
        DB2 Common
Component:
        OSSe
Code:
        118 (0x0076)

Identifier:
        ECF_LIB_CANNOT_LOAD

Description:
        Cannot load the specified library
```

Die wertvollsten Fehlerbehebungsinformationen in der Ausgabe des Befehls **db2diag** sind die Beschreibung und die zugehörigen Informationen (nur für ZRC-Rückkehrcodes).

Eine vollständige Liste der ZRC- bzw. ECF-Werte erhalten Sie durch die Eingabe des Befehls **db2diag -rc zrc** bzw. **db2diag -rc ecf**.

Einführung in Nachrichten

Es wird angenommen, dass Sie mit den Funktionen des Betriebssystems vertraut sind, unter dem DB2 installiert ist. Mithilfe der in den folgenden Kapiteln enthaltenen Informationen können Sie einen Fehler oder ein Problem identifizieren und durch eine entsprechende Recoveryaktion beheben. Diese Informationen können auch verwendet werden, um sich darüber zu informieren, wo Nachrichten generiert und protokolliert werden.

Nachrichtenstruktur

Die Nachrichtenhilfe beschreibt die Ursache einer Nachricht und erläutert eine Aktion, die Sie als Reaktion auf die Nachricht ausführen sollten.

Nachrichten-IDs bestehen aus einem Nachrichtenpräfix aus drei Zeichen gefolgt von einer vier- oder fünfstelligen Nachrichtennummer und einem Suffix aus einem Zeichen. Beispiel: *SQL1042C*. Eine Liste von Nachrichtenpräfixen finden Sie in „Aufrufen von Hilfe für Nachrichten” auf Seite 688 und „Andere DB2-Nachrichten” auf Seite 690. Das Suffix aus einem Buchstaben gibt die Wertigkeit der Fehlernachricht an.

Im Allgemeinen sind Nachrichten-IDs, die auf den Buchstaben *C* enden, Nachrichten über schwerwiegende Fehler. Der Endbuchstabe *E* weist auf dringende Nachrichten hin und der Endbuchstabe *N* auf Fehlernachrichten. Nachrichten-IDs auf *W* kennzeichnen Warnungen und Nachrichten-IDs auf *I* Informationsnachrichten.

Bei Nachrichten-IDs von ADM-Nachrichten kennzeichnet der Endbuchstabe *C* Nachrichten über schwerwiegende Fehler, der Buchstabe *E* dringende Nachrichten, der Buchstabe *W* wichtige Nachrichten und der Buchstabe *I* Informationsnachrichten.

Bei Nachrichten-IDs von SQL-Nachrichten kennzeichnet der Endbuchstabe *C* Nachrichten über kritische Systemfehler, der Buchstabe *N* Fehlernachrichten und der Buchstabe *W* Warnungen oder Informationsnachrichten.

Einige Nachrichten enthalten Token, die auch als Nachrichtenvariablen bezeichnet werden. Wenn eine Nachricht, die Token enthält, von DB2 generiert wird, wird jedes Token durch einen Wert ersetzt, der für die gemeldete Fehlerbedingung spezifisch ist, um den Benutzer bei der Diagnose der Ursache für die Fehlernachricht zu unterstützen. Zum Beispiel sieht die DB2-Nachricht SQL0107N wie folgt aus:

- Im Befehlszeilenprozessor:

  SQL0107N Der Name "<name>" ist zu lang. Die maximale Länge beträgt "<länge>".
- In der DB2-Informationszentrale:

  SQL0107N Der Name *name* ist zu lang. Die maximale Länge beträgt *länge*.

Diese Nachricht enthält die beiden Token "<name>" und "<länge>". Wenn diese Nachricht während der Ausführung generiert wird, werden die Token durch den tatsächlichen Namen des Objekts, das den Fehler verursacht hat, und die maximal zulässige Länge für den entsprechenden Typ von Objekt ersetzt.

In einigen Fällen ist ein Token für eine bestimmte Instanz eines Fehlers nicht anwendbar, sodass stattdessen der Wert *N zurückgegeben wird. Beispiel:

```
SQL20416N Der angegebene Wert ("*N") konnte nicht in einen Sicherheitskennsatz
konvertiert werden. Kennsätze für die Sicherheitsrichtlinie mit einer Richtlinien-ID "1"
sollten "8" Zeichen lang sein. Der Wert ist "0" Zeichen lang. SQLSTATE=23523
```

Aufrufen von Hilfe für Nachrichten

Auf die folgenden DB2-Nachrichten kann über den Befehlszeilenprozessor zugegriffen werden:

Präfix	Beschreibung
ADM	Nachrichten, die von vielen DB2-Komponenten generiert werden. Diese Nachrichten werden in die Protokolldatei mit Benachrichtigungen für die Systemverwaltung geschrieben und sind dazu gedacht, zusätzliche Informationen für Systemadministratoren bereitzustellen.
AMI	Nachrichten, die von MQ Application Messaging Interface generiert werden.
ASN	Nachrichten, die von der DB2-Replikation generiert werden.
CCA	Nachrichten, die vom Konfigurationsassistenten generiert werden.
CLI	Nachrichten, die von Call Level Interface generiert werden.
DBA	Nachrichten, die von den Datenbankverwaltungstools generiert werden.

DBI Nachrichten, die durch Installation und Konfiguration generiert werden.

DBT Nachrichten, die von den Datenbanktools generiert werden.

DB2 Nachrichten, die vom Befehlszeilenprozessor (CLP) generiert werden.

DQP Nachrichten, die von Query Patroller generiert werden.

EAS Nachrichten, die von Embedded Application Server generiert werden.

EXP Nachrichten, die vom EXPLAIN-Dienstprogramm generiert werden.

GSE Nachrichten, die von DB2 Spatial Extender generiert werden.

LIC Nachrichten, die vom DB2-Lizenzmanager generiert werden.

MQL Nachrichten, die von MQ Listener generiert werden.

SAT Nachrichten, die in einer Umgebung mit Satellitensystemen generiert werden.

SPM Nachrichten, die vom Synchronisationspunktmanager (SPM) generiert werden.

SQL Nachrichten, die vom Datenbankmanager generiert werden, wenn eine Warn- oder Fehlerbedingung erkannt wurde.

XMR Nachrichten, die vom XML-Metadatenrepository generiert werden.

Zum Aufrufen der Hilfe für eine Nachricht müssen Sie den Befehlszeilenprozessor öffnen und Folgendes eingeben:

```
? XXXnnnnn
```

Dabei ist *XXX* ein gültiges Nachrichtenpräfix und *nnnnn* eine gültige Nachrichtennummer.

Der Nachrichtentext, der zu einem bestimmten SQLSTATE-Wert gehört, kann durch folgende Eingabe abgerufen werden:

```
? nnnnn

  oder

? nn
```

Dabei ist *nnnnn* ein fünfstelliger SQLSTATE-Wert (alphanumerisch) und *nn* der zweistellige SQLSTATE-Klassencode (d. h. die ersten beiden Stellen des SQLSTATE-Werts).

Anmerkung: Die Nachrichten-ID, die als Parameter des Befehls **db2** akzeptiert wird, ist von der Groß-/Kleinschreibung unabhängig. Darüber hinaus ist das aus einem Buchstaben bestehende Suffix optional und wird ignoriert.

Daher führen die folgenden Befehle zum selben Ergebnis:

- ? SQL0000N
- ? sql0000
- ? SQL0000w

Zum Aufrufen der Hilfe für eine Nachricht über die Befehlszeile eines Systems auf UNIX-Basis, geben Sie folgenden Befehl ein:

```
db2 "? XXXnnnnn"

Dabei ist XXX ein gültiges Nachrichtenpräfix
und nnnnn eine gültige Nachrichtennummer.
```

Wenn der Nachrichtentext für Ihre Anzeige zu lang ist, verwenden Sie den folgenden Befehl (auf UNIX-basierten Systemen und anderen, die den Befehl 'more' unterstützen):

```
db2 "? XXXnnnnn" | more
```

Andere DB2-Nachrichten

Einige DB2-Komponenten geben Nachrichten zurück, die nicht online verfügbar sind und in diesem Handbuch nicht beschrieben werden. Einige solcher Nachrichtenpräfixe sind die folgenden:

AUD Nachrichten, die von der DB2-Prüffunktion (Audit) generiert werden.

DIA Diagnosenachrichten, die von vielen DB2-Komponenten generiert werden. Diese Nachrichten werden in die db2diag-Protokolldatei geschrieben und sind dazu gedacht, zusätzliche Informationen für Benutzer und DB2-Servicepersonal bei der Untersuchung von Fehlern bereitzustellen.

GOV Nachrichten, die vom Dienstprogramm DB2-Governor generiert werden.

In den meisten Fällen stellen diese Nachrichten ausreichend Informationen zur Verfügung, um die Ursache einer Warnung oder eines Fehlers zu bestimmen. Weitere Informationen zu dem Befehl oder dem Dienstprogramm, das die Nachrichten generiert hat, finden Sie im entsprechenden Handbuch, in dem der Befehl bzw. das Dienstprogramm dokumentiert ist.

Weitere Nachrichtenquellen

Wenn Sie andere Programme auf dem System ausführen, empfangen Sie möglicherweise Nachrichten mit anderen Präfixen als den in dieser Referenz genannten.

Informationen zu solchen Nachrichten finden Sie in den für das jeweilige Programmprodukt verfügbaren Informationen.

Informationen in plattformspezifischen Fehlerprotokollen

Auch außerhalb von DB2 steht eine Vielzahl weiterer Dateien und Dienstprogramme zur Verfügung, die Sie bei der Fehleranalyse unterstützen. Häufig sind sie für die Ermittlung der eigentlichen Fehlerursache ebenso wichtig wie die Informationen, die in den DB2-Dateien zur Verfügung gestellt werden.

Die anderen Dateien und Dienstprogramme bieten Zugriff auf Informationen in Protokollen und Traces zu den folgenden Bereichen:

- Betriebssysteme
- Anwendungen und Fremdanbieter
- Hardware

Abhängig von der verwendeten Betriebsumgebung können sich relevante Informationen auch in Bereichen befinden, die hier nicht beschrieben sind. Achten Sie daher auf alle in Frage kommenden Bereiche, in denen Sie nach den erforderlichen Informationen suchen müssen, wenn Sie die Fehlerbehebung auf Ihrem System durchführen.

Betriebssysteme

Jedes Betriebssystem verfügt über eigene Diagnosedateien, in denen Aktivitäten und Fehler protokolliert werden. Die am häufigsten vorkommenden (und normalerweise nützlichsten) dieser Dateien sind Fehlerberichte oder Ereignisprotokolle. Nachfolgend sind die Methoden aufgeführt, mit denen diese Informationen erfasst werden:

- AIX: Mit dem Befehl **/usr/bin/errpt -a**
- Solaris: Mit /var/adm/messages*-Dateien oder dem Befehl **/usr/bin/dmesg**
- Linux: Mit /var/log/messages*-Dateien oder dem Befehl **/bin/dmesg**
- HP-UX: Mit der Datei /var/adm/syslog/syslog.log oder dem Befehl **/usr/bin/dmesg**
- Windows: Mit den Ereignisprotokolldateien für System, Sicherheit und Anwendungen sowie mit der Datei windir\drwtsn32.log (wobei 'windir' das Windows-Installationsverzeichnis ist)

Für jedes Betriebssystem gibt es weitere Trace- und Debugdienstprogramme. Verschaffen Sie sich anhand der Dokumentation und des Unterstützungsmaterials für Ihr Betriebssystem einen Überblick, welche Informationen zusätzlich zur Verfügung stehen.

Anwendungen und Fremdanbieter

Normalerweise verfügt jede Anwendung über eigene Protokoll- und Diagnosedateien. Diese Dateien ergänzen die DB2-Informationen und vermitteln so ein genaueres Bild möglicher Problembereiche.

Hardware

Hardwareeinheiten zeichnen Informationen normalerweise in Betriebssystemfehlerprotokollen auf. In manchen Situationen sind jedoch möglicherweise zusätzliche Informationen erforderlich. In diesen Fällen müssen Sie feststellen, welche Hardwarediagnosedateien und -dienstprogramme für welche Hardwarekomponente in der verwendeten Umgebung verfügbar sind. Ein Beispiel hierfür ist, wenn DB2 eine fehlerhafte Seite oder eine Beschädigung meldet. Normalerweise wird diese Nachricht aufgrund eines Plattenfehlers ausgegeben, was bedeutet, dass die Hardwarediagnose überprüft werden muss. Verschaffen Sie sich anhand der Dokumentation und des Unterstützungsmaterials einen Überblick, welche Informationen zusätzlich zur Verfügung stehen.

Einige Informationen sind zeitkritisch, zum Beispiel die Informationen aus Hardwareprotokollen. Wenn ein Fehler auftritt, sollten Sie alle Anstrengungen unternehmen, um schnellstmöglich aus den relevanten Quellen so viele Informationen wie möglich zusammenzustellen.

Zusammenfassend lässt sich sagen, dass Sie, um einen Fehler vollständig zu verstehen und auszuwerten, möglicherweise alle Informationen erfassen müssen, die von DB2, von den Anwendungen, vom Betriebssystem und von der zugrunde liegenden Hardware bereitgestellt werden. Das Tool **db2support** automatisiert die Erfassung der meisten DB2- und Betriebssysteminformationen, die Sie benötigen; dennoch sollten Sie auch auf Informationen außerhalb dieser Erfassung achten, die bei der Untersuchung von Fehlern nützlich sein können.

Systemkerndateien (Linux und UNIX)

Wenn ein Programm abnormal beendet wird, wird eine Kerndatei (engl. Core File) durch das System erstellt, um ein Speicherimage des beendeten Prozesses zu speichern. Fehler, wie zum Beispiel Speicheradressverletzungen, unzulässige Instruktionen, Busfehler und benutzergenerierte Beendigungssignale führen zur Generierung von Kerndateien.

Die Kerndatei enthält die Zeichenfolge `core` im Dateinamen und wird standardmäßig in dem Verzeichnis abgelegt, das für den Konfigurationsparameter **diagpath** des Datenbankmanagers definiert ist, sofern diese Position nicht durch die Werte in der Registrierdatenbankvariablen DB2FODC anders konfiguriert wurde. Beachten Sie, dass sich die Systemkerndateien von DB2-Trapdateien unterscheiden.

Einstellungen zur Steuerung von Kerndateien

Kerndateien können die Generierung großer Mengen an Diagnosedaten bewirken, für die viel Speicherplatz benötigt wird und die erheblichen Systemaufwand verursachen können. Es stehen eine Reihe von Einstellungen der Registrierdatenbankvariablen *DB2FODC* zur Verfügung, mit denen Sie die Verarbeitung von Kerndateien steuern können. Sie können die Einstellungen der Registrierdatenbankvariablen *DB2FODC* mit dem Befehl **db2set** permanent festlegen oder mit dem Befehl **db2pdcfg** dynamisch (nur speicherintern) ändern. Permanente Einstellungen, die mit dem Befehl **db2set** festgelegt werden, werden erst wirksam, wenn die Instanz erneut gestartet wird; dynamische Einstellungen, die mit dem Befehl **db2pdcfg** vorgenommen werden, sind sofort wirksam und bleiben bis zum Neustart der Instanz in Kraft.

Sie können mit der Registrierdatenbankvariablen *DB2FODC* die folgenden Verhaltensweisen für Kerndateien steuern:

- Werden Kerndateien generiert oder nicht (mit der Einstellung für *DUMPCORE*)?
- Wie groß können Kerndateien werden (mit der Einstellung für *CORELIMIT*)?
- Wo werden die generierten Kerndateien gespeichert (mit der Einstellung für *DUMPDIR*)?

Grundsätzlich kann eine Kerndatei so groß werden wie die Menge des installierten physischen Speichers auf der Maschine, auf der die Kerndatei generiert wird. Wenn der verwendete Datenserver beispielsweise über 64 GB an physischem Speicher verfügt, müssen in dem Verzeichnispfad, in dem die Kerndatei gespeichert wird, mindestens 64 GB Speicherplatz verfügbar sein. Es ist möglich, die Größe der Kerndatei zu begrenzen, es empfiehlt sich jedoch, stattdessen das Verhalten der Kerndatei so zu konfigurieren, dass sie in einem Dateisystem gespeichert wird, in dem ausreichend Speicherplatz verfügbar ist. Wenn Sie die Speichermenge, die eine Kerndatei verwenden kann, begrenzen müssen, stellen Sie sicher, dass die verfügbare Speichermenge mindestens so groß ist wie die Menge an physischem Speicher auf der Maschine, da andernfalls die Gefahr besteht, dass die Kerndatei abgeschnitten wird und Diagnoseinformationen verloren gehen. Beispiel: Wenn Sie die für die Kerndateigenerierung verfügbare Speichermenge auf 64 GB begrenzen und die Kerndateien permanent in das Verzeichnis '/tmp' umleiten möchten, geben Sie den folgenden Befehl ein. Die Einstellungen werden erst wirksam, nachdem die Instanz neu gestartet wurde.

```
db2set DB2FODC="CORELIMIT=64000000000 DUMPDIR=/tmp"
```

Die Generierung einer Kerndatei kann einen erheblichen Systemaufwand verursachen, der die Systemverfügbarkeit beeinträchtigt. Wenn die während der Kerndateigenerierung auftretende Leistungsbeeinträchtigung hinsichtlich der Systemver-

fügbarkeit nicht akzeptabel ist, können Sie die Kerndateigenerierung inaktivieren. Es wird jedoch empfohlen, die Inaktivierung nicht permanent beizubehalten. Kerndateien enthalten Diagnoseinformationen, die für die erfolgreiche Fehlerbehebung erforderlich sein können. Wenn keine Diagnoseinformationen verfügbar sind, da die Kerndateigenerierung permanent inaktiviert wurde, kann möglicherweise keine Fehlerbehebung für den Datenserver durchgeführt werden. Beispiel: Für eine dynamische Inaktivierung der Kerndateigenerierung, die sofort wirksam wird und bis zum Neustart der Instanz in Kraft ist, geben Sie den folgenden Befehl ein:

```
db2pdcfg DB2FODC="DUMPCORE=OFF"
```

Zugriff auf Informationen in Systemkerndateien (Linux und UNIX)

Der Systembefehl **dbx** hilft Ihnen bei der Ermittlung, welche Funktion dazu führte, dass eine Systemkerndatei erstellt wurde. Hierbei handelt es sich um eine einfache Prüfung, die dabei hilft festzustellen, ob der Datenbankmanager den Fehler verursacht hat oder ob ein Fehler des Betriebssystems bzw. einer Anwendung für den Fehler verantwortlich ist.

Vorbereitende Schritte

- Der Befehl **dbx** muss installiert sein. Der Befehl ist vom Betriebssystem abhängig: unter AIX und Solaris verwenden Sie **dbx**; unter HP-UX verwenden Sie **xdb** und unter Linux **gdb**.
- Unter AIX müssen Sie sicherstellen, dass die volle Kernoption (Core) mithilfe des Befehls **chdev** oder über "smitty" aktiviert wurde.

Informationen zu diesem Vorgang

Anhand der folgenden Schritte können Sie die Funktion ermitteln, die den Kerndateispeicherauszug bewirkt hat.

Vorgehensweise

1. Geben Sie den folgenden Befehl über eine UNIX-Eingabeaufforderung ein:

   ```
   dbx programmname kerndateiname
   ```

   Dabei ist *programmname* der Name des Programms, dass abnormal beendet wurde, und *kerndateiname* der Name der Datei, die den Kerndateispeicherauszug enthält. Der Parameter für den *kerndateinamen* ist optional. Wenn Sie ihn nicht angeben, wird der Standardname "core" verwendet.
2. Überprüfen Sie den Aufrufstack in der Kerndatei. Informationen zur Vorgehensweise erhalten Sie, indem Sie `man dbx` in einer UNIX-Eingabeaufforderung eingeben.
3. Zum Beenden des Befehls **dbx** geben Sie **quit** in die dbx-Eingabeaufforderung ein.

Beispiel

Das folgende Beispiel zeigt, wie der Befehl **dbx** verwendet wird, um die Kerndatei für ein Programm mit dem Namen "main" zu lesen.

1. Geben Sie in eine Eingabeaufforderung Folgendes ein:

   ```
   dbx main
   ```
2. Auf Ihrem Bildschirm wird eine Ausgabe ähnlich der folgenden angezeigt:

   ```
   dbx version 3.1 for AIX.
   Type 'help' for help.
   reading symbolic information ...
   ```

```
[using memory image in core]
segmentation.violation in freeSegments at line 136
136         (void) shmdt((void *) pcAdress[i]);
```

3. Der Name der Funktion, die den Kernspeicherauszug verursacht hat, ist "freeSegments". Geben Sie **where** in die dbx-Eingabeaufforderung ein, um den Programmpfad zu der Stelle mit dem Fehler anzuzeigen.

   ```
   (dbx) where
   freeSegments(numSegs = 2, iSetId = 0x2ff7f730, pcAddress = 0x2ff7f758, line
   136
   in "main.c"
   main (0x1, 2ff7f7d4), line 96 in "main.c"
   ```

   In diesem Beispiel ist der Fehler in Zeile 136 der Funktion freeSegments aufgetreten, die in Zeile 96 im Programm `main.c` aufgerufen wurde.

4. Zum Beenden des Befehls **dbx** geben Sie **quit** in die dbx-Eingabeaufforderung ein.

Zugreifen auf Ereignisprotokolle (Windows)

In diesem Abschnitt wird erläutert, wie Sie auf die Windows-Ereignisprotokolle zugreifen können.

Informationen zu diesem Vorgang

Die Windows-Ereignisprotokolle können ebenfalls nützliche Informationen liefern. Das Systemereignisprotokoll ist in der Regel am nützlichsten bei DB2-Abstürzen oder anderen ungeklärten Fehlern im Zusammenhang mit Systemressourcen; es empfiehlt sich jedoch, alle drei Ereignisprotokolltypen abzurufen:

- System
- Anwendung
- Sicherheit

Vorgehensweise

Zeigen Sie die Ereignisprotokolle mithilfe der Windows-Ereignisanzeigefunktion an. Die Vorgehensweise zum Öffnen der Ereignisanzeige richtet sich danach, welches Windows-Betriebssystem Sie verwenden.
Klicken Sie unter Windows XP zum Starten der Ereignisanzeige beispielsweise auf **Start** —> **Systemsteuerung**. Wählen Sie **Verwaltung** aus, und klicken Sie dann **Ereignisanzeige** doppelt an.

Exportieren von Ereignisprotokollen (Windows)

In diesem Abschnitt wird der Export von Windows-Ereignisprotokollen erläutert.

Informationen zu diesem Vorgang

Sie können Ereignisprotokolle in der Windows-Ereignisanzeigefunktion in den folgenden Formaten exportieren:

- Protokolldateiformat
- Text- oder CSV-Dateiformat

Vorgehensweise

Exportieren Sie die Ereignisprotokolle aus der Windows-Ereignisanzeige.

- Daten im Protokolldateiformat (*.evt) können Sie wieder zurück in eine Ereignisanzeige laden (z. B. auf einer anderen Workstation). Dieses Format ist einfach zu

bearbeiten, da Sie die Ereignisanzeige verwenden können, um die chronologische Reihenfolge zu ändern, bestimmte Ereignisse herauszufiltern und vorwärts oder zurück zu blättern.

- Protokolle im Textdateiformat (*.txt) oder CSV-Dateiformat (*.csv) können Sie in den meisten Texteditoren öffnen. Darüber hinaus werden mit diesen Formaten mögliche Probleme aufgrund von Zeitmarken vermieden. Wenn Sie Ereignisprotokolle im .evt-Format exportieren, weisen die Zeitmarken das Weltzeitformat (Coordinated Universal Time) auf und werden in die Ortszeit der Workstation in der Anzeige konvertiert. So kann es vorkommen, dass Sie aufgrund der Zeitzonendifferenz wichtige Ereignisse übersehen. Darüber hinaus sind Textdateien einfacher zu durchsuchen.

Zugriff auf die Protokolldatei von Dr. Watson (Windows)

In diesem Abschnitt wird beschrieben, wie Sie an Windows-Systemen auf die Protokolldateien von Dr. Watson zugreifen können.

Informationen zu diesem Vorgang

Das Protokoll von Dr. Watson, `drwtsn32.log`, enthält eine chronologische Aufzeichnung aller Ausnahmebedingungen, die auf dem System aufgetreten sind. Obwohl die DB2-Trapdateien aussagekräftiger sind als das Dr. Watson-Protokoll, kann es bei der Einschätzung der allgemeinen Systemstabilität sowie zur Dokumentation der Protokollierung von DB2-Traps nützlich sein.

Vorgehensweise

Suchen Sie die Dr. Watson-Protokolldatei. Der Standardpfad lautet `<installationsverzeichnis>:\Dokumente und Einstellungen\All Users\Dokumente\DrWatson`.

Trapdateien

DB2 generiert eine Trapdatei, falls DB2 die Verarbeitung aufgrund einer Fehlerunterbrechung (Trap), einer Segmentierungsverletzung oder einer Ausnahmebedingung nicht fortsetzen kann.

Alle Signale oder Ausnahmebedingungen, die von DB2 empfangen werden, werden in der Trapdatei aufgezeichnet. Die Trapdatei enthält außerdem die Funktionsfolge, die aktiv war, als der Fehler aufgetreten ist. Diese Folge wird manchmal auch als "Funktionsaufrufstack" (engl. function call stack) oder "Stack-Trace" bezeichnet. Die Trapdatei enthält darüber hinaus Informationen zum Status des Prozesses, als das Signal oder die Ausnahmebedingung aufgefangen wurde.

Eine Trapdatei wird auch generiert, wenn beim Ausführen einer threadsicheren, abgeschirmten Routine ein Anwendungsstopp erzwungen wird. Die Generierung der Datei erfolgt beim Beenden. Es handelt sich hierbei um einen harmlosen Fehler, der nicht weiter beachtet werden muss.

Die Dateien befinden sich in dem Verzeichnis, das durch den Konfigurationsparameter **diagpath** des Datenbankmanagers angegeben wird.

Der Name der Trapdatei beginnt auf allen Plattformen mit einer Prozess-ID (PID), gefolgt von einer Thread-ID (TID), gefolgt von der Partitionsnummer (000 für Datenbanken mit nur einer Partition) und am Ende „.trap.txt”.

Es gibt auch Diagnosetraps, die der Code generiert, wenn bestimmte Bedingungen eintreten, die keine Unterbrechung der Instanz rechtfertigen, bei denen aber der Stack nützliche Informationen enthält. Der Name dieser Traps besteht aus der PID im Dezimalformat, gefolgt von der Partitionsnummer (0 für Datenbanken mit nur einer Partition).

Beispiele:

- `6881492.2.000.trap.txt` ist eine Trapdatei mit der Prozess-ID (PID) 6881492 und der Thread-ID (TID) 2.
- `6881492.2.010.trap.txt` ist eine Trapdatei deren Prozess und Thread auf Partition 10 ausgeführt werden.

Sie können Trapdateien nach Bedarf erstellen, indem Sie den Befehl **db2pd** mit der Option `-stack all` oder `-dump` verwenden. Im Allgemeinen sollte dies jedoch nur erfolgen, wenn Sie von IBM Software Support dazu aufgefordert werden.

Sie können Stack-Tracedateien mit den Befehlen **db2pd -stacks** oder **db2pd -dumps** generieren. Diese Dateien haben denselben Inhalt wie die Trapdatei, werden jedoch nur für Diagnosezwecke generiert. Die Namen dieser Dateien lauten ähnlich wie im folgenden Beispiel: `6881492.2.000.stack.txt`.

Formatieren von Trapdateien (Windows)

Sie können Trapdateien (*.TRP) mit dem Befehl **db2xprt** formatieren. Dieser Befehl formatiert die binären Trapdateien der DB2-Datenbank in vom Benutzer lesbare ASCII-Dateien.

Informationen zu diesem Vorgang

Das Tool **db2xprt** verwendet DB2-Symboldateien zum Formatieren der Trapdateien. Eine Untergruppe dieser .PDB-Dateien wird in die DB2-Datenbankprodukte integriert.

Beispiel

Wurde die Trapdatei 'DB30882416.TRP' in dem über den Konfigurationsparameter **diagpath** des Datenbankmanagers angegebenen Verzeichnis generiert, kann diese wie folgt formatiert werden:

```
db2xprt DB30882416.TRP DB30882416.FMT
```

Kapitel 12. Unterstützung

Kontaktaufnahme mit IBM Software Support

IBM Software Support bietet Unterstützung bei Produktfehlern.

Vorbereitende Schritte

IBM Software Support steht Ihnen zur Verfügung, wenn Sie über einen aktuellen IBM Softwarewartungsvertrag verfügen und für eine Weitergabe von Fehlern an IBM autorisiert sind. Informationen zu den verfügbaren Typen von Wartungsverträgen finden Sie in dem Abschnitt „Premium Support" im Handbuch *Software Support Handbook* unter folgender Adresse: techsupport.services.ibm.com/guides/services.html.

Vorgehensweise

Führen Sie die folgenden Schritte aus, um im Falle eines Problems Kontakt mit IBM Software Support aufzunehmen:

1. Grenzen Sie den Fehler ein, sammeln Sie Hintergrundinformationen zu dem Fehler und ordnen Sie dem Fehler je nach Schweregrad eine Fehlerbewertung zu. Informationen zur Unterstützung finden Sie im Abschnitt „Contacting IBM" in der Veröffentlichung *Software Support Handbook* unter der folgenden Adresse: techsupport.services.ibm.com/guides/beforecontacting.html.
2. Stellen Sie Diagnoseinformationen zusammen.
3. Leiten Sie die Informationen zu dem Fehler auf eine der folgenden Weisen an IBM Software Support weiter:
   - Online: Klicken Sie auf den Link **ESR** (Electronic Service Request) auf der Website von IBM Software Support für offene Serviceanforderungen (Open Service Request) unter folgender Adresse: www.ibm.com/software/support/probsub.html.
   - Telefonisch: Die aktuelle Telefonnummer für Ihr Land und Ihre Region finden Sie auf der Seite 'Contacts' in dem Handbuch *Software Support Handbook* unter folgender Adresse: techsupport.services.ibm.com/guides/contacts.html.

Nächste Schritte

Wenn es sich bei dem Fehler um einen Softwarefehler oder um Lücken oder Fehler in der Dokumentation handelt, wird von IBM Software Support ein Authorized Program Analysis Report (APAR) erstellt. In dem APAR wird der Fehler genauestens beschrieben. IBM Software Support wird soweit möglich eine Ausweichlösung bereitstellen, die Sie implementieren können, bis der APAR abgeschlossen ist und eine Programmkorrektur geliefert werden kann. IBM veröffentlicht behobene APARs täglich auf der Website von IBM Software Support, damit auch andere Benutzer, bei denen das gleiche Problem auftritt, von der Lösung profitieren können.

Übergeben von Daten an IBM Software Support

Sie können Daten über FTP oder über das ESR-Tool (Electronic Service Request, elektronische Serviceanforderung) an IBM Software Support senden. Gehen Sie dazu wie in diesem Abschnitt beschrieben vor.

Vorbereitende Schritte

Bei den beschriebenen Schritten wird davon ausgegangen, dass Sie bereits einen PMR (Problem Management Record) bei IBM Software Support geöffnet haben.

Informationen zu diesem Vorgang

Sie können Diagnosedaten, wie beispielsweise Protokolldateien und Konfigurationsdateien, über eine der folgenden Methoden an IBM Software Support senden:

- FTP
- ESR-Tool (Electronic Service Request, elektronische Serviceanforderung)

Vorgehensweise

- Gehen Sie wie folgt vor, um Dateien (über FTP) an EcuRep (Enhanced Centralized Client Data Repository) zu übermitteln:
  1. Komprimieren Sie die erfassten Datendateien in ZIP- oder TAR-Format, und benennen Sie das Paket der PMR-Kennung entsprechend.

     Die Datei muss den folgenden Namenskonventionen entsprechen, um dem PMR korrekt zugeordnet zu werden: xxxxx.bbb.ccc.yyy.yyy. Dabei ist xxxxx die PMR-Nummer, bbb die PMR-Zweigstellennummer, ccc der PMR-Gebietscode und yyy.yyy der Dateiname.
  2. Stellen Sie über ein FTP-Programm eine Verbindung zum Server ftp.emea.ibm.com her.
  3. Melden Sie sich mit der Benutzer-ID "anonymous" an, und geben Sie Ihre E-Mail-Adresse als Kennwort ein.
  4. Wechseln Sie in das Verzeichnis `toibm`. Beispiel: **`cd toibm`**.
  5. Wechseln Sie in eines der betriebssystemspezifischen Unterverzeichnisse. Zu diesen Unterverzeichnissen gehören zum Beispiel die Verzeichnisse: `aix`, `linux`, `unix` oder `windows`.
  6. Wechseln Sie in den Binärmodus. Geben Sie in der Eingabeaufforderung zum Beispiel **`bin`** ein.
  7. Stellen Sie Ihre Datei unter Verwendung des Befehls **`put`** auf den Server. Beachten Sie die folgende Dateinamenskonvention bei der Benennung Ihrer Datei, und stellen Sie sie auf den Server. Ihr PMR wird aktualisiert und listet nun die Speicherposition der Dateien unter Verwendung des folgenden Formats auf: `xxxx.bbb.ccc.yyy.yyy`. (`xxx` ist die PMR-Nummer (PMR - Problem Management Record), `bbb` ist die Zweigstellennummer, `ccc` ist der Gebietscode und `yyy.yyy` ist die Beschreibung des Dateityps als `tar.Z` oder `xyz.zip`.) Sie können Dateien an den FTP-Server senden, diese jedoch nicht aktualisieren. Jedes Mal, wenn Sie die Datei zu einem späteren Zeitpunkt ändern möchten, müssen Sie einen neuen Dateinamen verwenden.
  8. Geben Sie den Befehl **`quit`** ein.
- Gehen Sie wie folgt vor, um Dateien mithilfe des ESR-Tools zu übermitteln:
  1. Melden Sie sich bei ESR an.
  2. Geben Sie auf der Eingangsseite die PMR-Nummer in das Feld **Enter a report number** ein, und klicken Sie **Go** an.
  3. Blättern Sie nach unten zum Feld **Attach Relevant File**.
  4. Klicken Sie auf **Browse**, um die Protokoll-, Trace- oder sonstige Diagnosedatei zu suchen, die Sie an IBM Software Support übermitteln möchten.
  5. Klicken Sie **Submit** an. Die Datei wird über FTP an IBM Software Support übertragen und Ihrem PMR zugeordnet.

Nächste Schritte

Weitere Informationen zum EcuRep-Service finden Sie unter IBM EMEA Centralized Customer Data Store Service.

Weitere Informationen zu ESR finden Sie unter "Electronic Service Request (ESR) - Help".

Teil 3. Anhänge und Schlussteil

Anhang A. Übersicht über die technischen Informationen zu DB2

Die technischen Informationen zu DB2 stehen über die folgenden Tools und Methoden zur Verfügung:

- DB2-Informationszentrale
  - Themen (zu Tasks, Konzepten und Referenzinformationen)
  - Hilfe für DB2-Tools
  - Beispielprogramme
  - Lernprogramme
- DB2-Bücher
  - PDF-Dateien (für den Download verfügbar)
  - PDF-Dateien (auf der DB2-PDF-DVD)
  - Gedruckte Bücher
- Befehlszeilenhilfe
  - Hilfe für Befehle
  - Hilfe für Nachrichten

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 IBM 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 bearbeitet 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 mithilfe der Dokumentation nicht beheben können, fordern Sie beim zuständigen IBM Service-Center Unterstützung an.

Bibliothek mit technischen Informationen zu DB2 im Hardcopy- oder PDF-Format

Die folgenden Tabellen enthalten eine Beschreibung der DB2-Bibliothek, die im IBM Publications Center unter www.ibm.com/e-business/linkweb/publications/servlet/pbi.wss zur Verfügung steht. Englische Handbücher zu Version 9.7 im PDF-Format können über die folgende Adresse heruntergeladen werden: www.ibm.com/support/docview.wss?uid=swg27015148. Übersetzte DB2-Handbücher im PDF-Format können über die folgende Adresse heruntergeladen werden: www.ibm.com/support/docview.wss?uid=swg27015149.

In den Tabellen sind die Bücher, die in gedruckter Form zur Verfügung stehen, gekennzeichnet; möglicherweise sind diese in Ihrem Land oder Ihrer Region jedoch nicht verfügbar.

Die Formnummer wird bei jeder Aktualisierung eines Handbuchs erhöht. Anhand der nachfolgenden Liste können Sie sicherstellen, dass Sie die jeweils neueste Version des Handbuchs lesen.

Anmerkung: Die *DB2-Informationszentrale* wird häufiger aktualisiert als die PDF- und Hardcopybücher.

Tabelle 88. Technische Informationen zu DB2

Name	IBM Form	In gedruckter Form verfügbar	Letzte Aktualisierung
Administrative API Reference	SC27-2435-03	Ja	Juli 2012
Administrative Routines and Views	SC27-2436-03	Nein	Juli 2012
Call Level Interface Guide and Reference, Volume 1	SC27-2437-03	Ja	Juli 2012
Call Level Interface Guide and Reference, Volume 2	SC27-2438-03	Ja	Juli 2012
Command Reference	SC27-2439-03	Ja	Juli 2012
Dienstprogramme für das Versetzen von Daten - Handbuch und Referenz	SC12-4281-01	Ja	Juli 2012
Datenrecovery und hohe Verfügbarkeit - Handbuch und Referenz	SC12-4282-03	Ja	Juli 2012
Datenbankverwaltung - Konzepte und Konfiguration - Referenzinformationen	SC12-4283-03	Ja	Juli 2012
Datenbanküberwachung - Handbuch und Referenz	SC12-4287-03	Ja	Juli 2012
Datenbanksicherheit	SC12-4285-02	Ja	Juli 2012
DB2 Text Search	SC12-4288-03	Ja	Juli 2012

Tabelle 88. Technische Informationen zu DB2 (Forts.)

Name	IBM Form	In gedruckter Form verfügbar	Letzte Aktualisierung
Developing ADO.NET and OLE DB Applications	SC27-2444-02	Ja	Juli 2012
Developing Embedded SQL Applications	SC27-2445-02	Ja	Juli 2012
Developing Java Applications	SC27-2446-03	Ja	Juli 2012
Developing Perl, PHP, Python, and Ruby on Rails Applications	SC27-2447-02	Nein	Juli 2012
Developing User-defined Routines (SQL and External)	SC27-2448-02	Ja	Juli 2012
Getting Started with Database Application Development	GI11-9410-02	Ja	Juli 2012
Installation und Verwaltung von DB2 unter Linux und Windows - Erste Schritte	GI11-3220-00	Ja	August 2009
Globalisierung	SC12-4279-00	Ja	August 2009
DB2-Server - Installation	GC12-4276-03	Ja	Juli 2012
IBM Data Server-Clients - Installation	GC12-4275-02	Nein	Juli 2012
Fehlernachrichten, Band 1	SC12-4295-01	Nein	August 2009
Fehlernachrichten, Band 2	SC12-4296-01	Nein	August 2009
Net Search Extender - Verwaltung und Benutzerhandbuch	SC12-4298-02	Nein	September 2010
Partitionierung und Clustering	SC12-4286-02	Ja	Juli 2012
pureXML - Handbuch	SC12-4293-02	Ja	Juli 2012
Query Patroller - Verwaltung und Benutzerhandbuch	SC12-4304-00	Nein	August 2009
Spatial Extender und Geodetic Data Management Feature - Benutzer- und Referenzhandbuch	SC12-4299-02	Nein	Juli 2012
SQL Procedural Languages: Application Enablement and Support	SC27-2470-03	Ja	Juli 2012
SQL Reference, Volume 1	SC27-2456-03	Ja	Juli 2012
SQL Reference, Volume 2	SC27-2457-03	Ja	Juli 2012

Tabelle 88. Technische Informationen zu DB2 (Forts.)

Name	IBM Form	In gedruckter Form verfügbar	Letzte Aktualisierung
Fehlerbehebung und Optimieren der Datenbankleistung	SC12-4289-03	Ja	Juli 2012
Upgrade auf DB2 Version 9.7	SC12-4274-03	Ja	Juli 2012
Lernprogramm für Visual Explain	SC12-4290-00	Nein	August 2009
Neuerungen in DB2 Version 9.7	SC12-4291-03	Ja	Juli 2012
Workload-Manager - Handbuch und Referenz	SC12-4292-03	Ja	Juli 2012
XQuery - Referenz	SC12-4294-01	Nein	November 2009

Tabelle 89. Technische Informationen zu DB2 Connect

Name	IBM Form	In gedruckter Form verfügbar	Letzte Aktualisierung
DB2 Connect Personal Edition - Installation und Konfiguration	SC12-4277-03	Ja	Juli 2012
DB2 Connect-Server - Installation und Konfiguration	SC12-4278-03	Ja	Juli 2012
DB2 Connect - Benutzerhandbuch	SC12-4280-02	Ja	September 2010

Tabelle 90. Technische Informationen zu Information Integration

Name	IBM Form	In gedruckter Form verfügbar	Letzte Aktualisierung
Information Integration: Föderierte Systeme - Verwaltung	SC12-3759-02	Ja	August 2009
Information Integration: ASNCLP Program Reference for Replication and Event Publishing	SC19-1018-04	Ja	August 2009
Information Integration: Konfiguration föderierter Datenquellen	SC12-3777-02	Nein	August 2009
Information Integration: SQL Replication - Handbuch und Referenz	SC12-3782-02	Ja	August 2009
Information Integration: Replikation und Event-Publishing - Einführung	GC12-3779-02	Ja	August 2009

Bestellen gedruckter DB2-Bücher

Informationen zu diesem Vorgang

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 DVD 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 DVD 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 DVD mit der DB2-PDF-Dokumentation verfügbar sind, können in gedruckter Form bestellt werden.

Anmerkung: Über http://publib.boulder.ibm.com/infocenter/db2luw/v9r7 haben Sie Zugriff auf die DB2-Informationszentrale, wo Sie die neueste und umfassendste DB2-Dokumentation finden.

Gehen Sie wie folgt vor, um gedruckte DB2-Bücher zu bestellen:

Vorgehensweise

- 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:
  1. 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").
  2. Geben Sie bei Ihrem Anruf an, dass Sie eine DB2-Veröffentlichung bestellen möchten.
  3. Teilen Sie dem zuständigen Ansprechpartner die Titel und Formularnummern der Bücher mit, die Sie bestellen möchten. Titel und Formularnummern finden Sie unter „Bibliothek mit technischen Informationen zu DB2 im Hardcopy- oder PDF-Format” auf Seite 704.

Aufrufen der Hilfe für den SQL-Status über den Befehlszeilenprozessor

DB2-Produkte geben für Bedingungen, die aufgrund 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 Starten der Hilfe für SQL-Statuswerte müssen Sie den Befehlszeilenprozessor öffnen und Folgendes eingeben:

```
? SQL-Status oder ? Klassencode
```

Hierbei steht *SQL-Status* für einen gültigen fünfstelligen SQL-Statuswert und *Klassencode* für die ersten beiden Ziffern dieses Statuswerts.
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.

Zugriff auf verschiedene Versionen der DB2-Informationszentrale

Informationen zu diesem Vorgang

Für Themen aus DB2 Version 9.8 lautet die URL der *DB2-Informationszentrale* http://publib.boulder.ibm.com/infocenter/db2luw/v9r8/.

Für Themen aus DB2 Version 9.7 lautet die URL der *DB2-Informationszentrale* http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/.

Für Themen aus DB2 Version 9.5 lautet die URL der *DB2-Informationszentrale* http://publib.boulder.ibm.com/infocenter/db2luw/v9r5.

Für Themen aus DB2 Version 9.1 lautet die URL der *DB2Informationszentrale* http://publib.boulder.ibm.com/infocenter/db2luw/v9/.

Für Themen aus DB2 Version 8 lautet die URL der *DB2-Informationszentrale (Version 8, 'Information - Unterstützung')* http://publib.boulder.ibm.com/infocenter/db2luw/v8/.

Anzeigen von Themen in der gewünschten Sprache in der DB2-Informationszentrale

Informationen zu diesem Vorgang

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. Aktualisieren Sie 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 den Knopf im Bereich **Languages** des Dialogfensters **Tools** —> **Options** —> **Advanced** 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. Aktualisieren Sie die Seite, um die DB2-Informationszentrale in der gewünschten Sprache anzuzeigen.

Ergebnisse

Bei einigen Kombinationen aus Browser und Betriebssystem müssen Sie auch die Ländereinstellungen des Betriebssystems in die gewünschte Locale und Sprache ändern.

Aktualisieren der auf Ihrem Computer oder Intranet-Server installierten DB2-Informationszentrale

Eine lokal installierte DB2-Informationszentrale muss regelmäßig aktualisiert werden.

Vorbereitende Schritte

Eine DB2-Informationszentrale der Version 9.7 muss bereits installiert sein. Einzelheiten hierzu finden Sie unter „Installation der DB2-Informationszentrale mit dem DB2-Installationsassistenten" in *DB2-Server - Installation*. Alle für die Installation der Informationszentrale geltenden Voraussetzungen und Einschränkungen gelten auch für die Aktualisierung der Informationszentrale.

Informationen zu diesem Vorgang

Eine vorhandene DB2-Informationszentrale kann automatisch oder manuell aktualisiert werden:

- Automatische Aktualisierungen. Verwenden Sie diese Aktualisierungsmethode zur Aktualisierung vorhandener Komponenten und Sprachen der Informationszentrale. Ein zusätzlicher Vorteil von automatischen Aktualisierungen ist, dass die Informationszentrale während der Aktualisierung nur für einen sehr kurzen Zeitraum nicht verfügbar ist. Darüber hinaus können automatische Aktualisierungen so konfiguriert werden, dass sie als Teil anderer, regelmäßig ausgeführter Stapeljobs ausgeführt werden.
- Manuelle Aktualisierungen. Verwenden Sie diese Aktualisierungsmethode, wenn Sie während des Aktualisierungsprozesses Komponenten oder Sprachen hinzufügen möchten. Beispiel: Eine lokale Informationszentrale wurde ursprünglich sowohl mit englischer als auch mit französischer Sprachunterstützung installiert; nun soll auch die deutsche Sprachunterstützung installiert werden. Bei einer manuellen Aktualisierung werden sowohl eine Installation der deutschen Sprachunterstützung als auch eine Aktualisierung der vorhandenen Komponenten und Sprachen der Informationszentrale durchgeführt. Sie müssen jedoch bei einer manuellen Aktualisierung die Informationszentrale manuell stoppen, aktualisieren und erneut starten. Die Informationszentrale ist während des gesamten Aktualisierungsprozesses nicht verfügbar.

Dieser Abschnitt enthält Details zum Prozess der automatischen Aktualisierung. Anweisungen zur manuellen Aktualisierung finden Sie im Abschnitt „Manuelles Aktualisieren der auf Ihrem Computer oder Intranet-Server installierten DB2-Informationszentrale".

Vorgehensweise

Gehen Sie wie folgt vor, um die auf Ihrem Computer bzw. Intranet-Server installierte DB2-Informationszentrale automatisch zu aktualisieren:

1. Unter Linux:
   a. Navigieren Sie zu dem Pfad, in dem die Informationszentrale installiert ist. Standardmäßig ist die DB2-Informationszentrale im Verzeichnis `/opt/ibm/db2ic/V9.7` installiert.
   b. Navigieren Sie vom Installationsverzeichnis in das Verzeichnis `doc/bin`.
   c. Führen Sie das Script `update-ic` aus:
      ```
      update-ic
      ```
2. 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 `<Programme>\IBM\DB2 Information Center\Version 9.7` installiert, wobei `<Programme>` das Verzeichnis der Programmdateien (Program Files) angibt.
   c. Navigieren Sie vom Installationsverzeichnis in das Verzeichnis `doc\bin`.
   d. Führen Sie die Datei `update-ic.bat` aus:
      ```
      update-ic.bat
      ```

Ergebnisse

Die DB2-Informationszentrale wird automatisch erneut gestartet. Standen Aktualisierungen zur Verfügung, zeigt die Informationszentrale die neuen und aktualisierten Abschnitte an. Waren keine Aktualisierungen für die Informationszentrale verfügbar, wird eine entsprechende Nachricht zum Protokoll hinzugefügt. Die Protokolldatei befindet sich im Verzeichnis `doc\eclipse\configuration`. Der Name der Protokolldatei ist eine Zufallszahl. Beispiel: `1239053440785.log`.

Manuelles Aktualisieren der auf Ihrem Computer oder Intranet-Server installierten DB2-Informationszentrale

Wenn Sie die DB2-Informationszentrale lokal installiert haben, können Sie Dokumentationsaktualisierungen von IBM abrufen und installieren.

Informationen zu diesem Vorgang

Zur manuellen 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. Die Ausführung der Informationszentrale im Standalone-Modus verhindert, dass andere Benutzer in Ihrem Netz auf die Informationszentrale zugreifen, und ermöglicht das Anwenden von Aktualisierungen. Die Workstationversion der DB2-Informationszentrale wird stets im Standalone-Modus ausgeführt.
2. Verwenden Sie die Aktualisierungsfunktion, um zu prüfen, welche Aktualisierungen verfügbar sind. Falls Aktualisierungen verfügbar sind, die Sie installieren müssen, können Sie die Aktualisierungsfunktion verwenden, um diese abzurufen und zu installieren.

   Anmerkung: Wenn es in der verwendeten Umgebung erforderlich ist, die Aktualisierungen für die *DB2-Informationszentrale* auf einer Maschine zu installieren, die nicht über ein Verbindung zum Internet verfügt, spiegeln Sie die Aktualisierungssite auf ein lokales Dateisystem und verwenden Sie dabei eine Maschine, die mit dem Internet verbunden ist und auf der die *DB2-Informationszentrale* installiert ist. Wenn viele Benutzer Ihres Netzes die Dokumentationsaktualisierungen installieren sollen, können Sie die Zeit, die jeder einzelne Benutzer für die Aktualisierungen benötigt, reduzieren, indem Sie die Aktualisierungssite lokal spiegeln und ein Proxy dafür erstellen.
   Ist dies der Fall, verwenden Sie die Aktualisierungsfunktion, um die Pakete abzurufen. Die Aktualisierungsfunktion ist jedoch nur im Standalone-Modus verfügbar.
3. Stoppen Sie die im Standalone-Modus gestartete Informationszentrale und starten Sie die *DB2-Informationszentrale* auf Ihrem Computer erneut.

Anmerkung: Unter Windows 2008 und Windows Vista (und neueren Versionen) müssen die in diesem Abschnitt aufgeführten Befehle mit Administratorberechtigung ausgeführt werden. Zum Öffnen einer Eingabeaufforderung oder eines Grafiktools mit vollen Administratorberechtigungen klicken Sie mit der rechten Maustaste die Verknüpfung an und wählen Sie **Als Administrator ausführen** aus.

Vorgehensweise

Gehen Sie wie folgt vor, um die auf Ihrem Computer bzw. Intranet-Server installierte *DB2-Informationszentrale* zu aktualisieren:

1. Stoppen Sie die *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 **Beenden** aus.
   - Unter Linux: Geben Sie den folgenden Befehl ein:
     ```
     /etc/init.d/db2icdv97 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 *Programme*\IBM\DB2 Information Center\Version 9.7 installiert, wobei *Programme* das Verzeichnis der Programmdateien (Program Files) angibt.
  c. Navigieren Sie vom Installationsverzeichnis in das Verzeichnis doc\bin.
  d. Führen Sie die Datei help_start.bat aus:

     ```
     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/V9.7 installiert.
  b. Navigieren Sie vom Installationsverzeichnis in das Verzeichnis doc/bin.
  c. Führen Sie das Script help_start aus:

     ```
     help_start
     ```

Der standardmäßig auf dem System verwendete Web-Browser wird geöffnet und zeigt die Standalone-Informationszentrale an.

3. Klicken Sie den Aktualisierungsknopf ( ) an. (JavaScript muss im verwendeten Browser aktiviert sein.) 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 Installationsprozesses die gewünschten Aktualisierungen aus und klicken Sie anschließend den Knopf für die Installation der Aktualisierungen an.
5. Klicken Sie nach Abschluss des Installationsprozesses **Fertigstellen** an.
6. Stoppen Sie die im Standalone-Modus gestartete Informationszentrale:
   - Unter Windows: Navigieren Sie in das Verzeichnis doc\bin des Installationsverzeichnisses und führen Sie die Datei help_end.bat aus:

     ```
     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 stoppen. Verwenden Sie nicht die Tastenkombination Strg+C oder eine andere Methode, um help_start.bat zu stoppen.
   - Unter Linux: Navigieren Sie in das Verzeichnis doc/bin des Installationsverzeichnisses und führen Sie das Script help_end aus:

     ```
     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 stoppen. Verwenden Sie keine andere Methode, um das Script help_start zu stoppen.
7. Starten Sie die *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 **Start** aus.
   - Unter Linux: Geben Sie den folgenden Befehl ein:

     ```
     /etc/init.d/db2icdv97 start
     ```

Ergebnisse

In der aktualisierten *DB2-Informationszentrale* werden die neuen und aktualisierten Themen angezeigt.

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.

„pureXML" in *pureXML - Handbuch*
Einrichten einer DB2-Datenbank, um XML-Daten zu speichern und Basisoperationen mit dem nativen XML-Datenspeicher auszuführen.

„Visual Explain" in *Lernprogramm für Visual Explain*
Analysieren, Optimieren und Anpassen von SQL-Anweisungen zur Leistungsverbesserung mithilfe von Visual Explain.

Informationen zur Fehlerbehebung in DB2

Eine breite Palette verschiedener Informationen zur Fehlerbestimmung und Fehlerbehebung steht zur Verfügung, um Sie bei der Verwendung von DB2-Datenbankprodukten zu unterstützen.

DB2-Dokumentation
Informationen zur Fehlerbehebung stehen im Handbuch *Fehlerbehebung und Optimieren der Datenbankleistung* oder im Abschnitt mit grundlegenden Informationen zu Datenbanken in der *DB2-Informationszentrale* zur Verfügung. Die Fehlerbehebungsinformationen enthalten Themen, die Sie beim Eingrenzen und Identifizieren von Problemen mithilfe der Diagnosetools und -dienstprogramme von DB2 unterstützen. Darüber hinaus finden Sie hier Lösungen für einige der häufigsten Probleme sowie Hinweise zur Lösung von Problemen, die in den verwendeten DB2-Datenbankprodukten auftreten können.

IBM Support Portal
Im IBM Support Portal 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.

Sie können auf das IBM Support Portal über die folgende Website zugreifen: http://www.ibm.com/support/entry/portal/Overview/Software/Information_Management/DB2_for_Linux,_UNIX_and_Windows.

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 dieser 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 dieser 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 B. Bemerkungen

Die vorliegenden Informationen wurden für Produkte und Services entwickelt, die auf dem deutschen Markt angeboten werden. Die Informationen über Produkte anderer Hersteller als IBM basieren auf den zum Zeitpunkt der ersten Veröffentlichung dieses Dokuments verfügbaren Informationen und können geändert werden.

Möglicherweise bietet IBM die in dieser Dokumentation beschriebenen Produkte, Services oder Funktionen in anderen Ländern nicht an. Informationen über die gegenwärtig im jeweiligen Land verfügbaren Produkte und Services sind beim zuständigen IBM Ansprechpartner erhältlich. Hinweise auf IBM Lizenzprogramme oder andere IBM Produkte bedeuten nicht, dass nur Programme, Produkte oder Services von IBM verwendet werden können. Anstelle der IBM Produkte, Programme oder Services können auch andere, ihnen äquivalente Produkte, Programme oder Services verwendet werden, solange diese keine gewerblichen oder anderen Schutzrechte von IBM verletzen. Die Verantwortung für den Betrieb von Produkten, Programmen und Services anderer Anbieter liegt beim Kunden.

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 hier enthaltenen Informationen werden in regelmäßigen Zeitabständen aktualisiert und als Neuausgabe veröffentlicht. 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 werden lediglich als Service für den Kunden bereitgestellt 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 unabhängig voneinander 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
U59/3600
3600 Steeles Avenue East
Markham, Ontario L3R 9Z7
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 bzw. der Allgemeinen Geschäftsbedingungen von IBM, der IBM Internationalen Nutzungsbedingungen für Programmpakete oder einer äquivalenten Vereinbarung.

Alle in diesem Dokument enthaltenen Leistungsdaten stammen aus einer kontrollierten 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 Gewährleistung, 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 davon abweichen. Benutzer dieses Dokuments sollten die entsprechenden Daten in ihrer spezifischen Umgebung prüfen.

Alle Informationen zu Produkten anderer Anbieter stammen von den Anbietern der aufgeführten Produkte, deren veröffentlichen Ankündigungen oder anderen allgemein verfügbaren Quellen. 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 von IBM unterliegen Änderungen oder können zurückgenommen werden und repräsentieren nur die Ziele von IBM.

Diese Veröffentlichung kann Beispiele für Daten und Berichte des alltäglichen Geschäftsablaufes enthalten. Sie sollen nur die Funktionen des Lizenzprogramms 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 Quellensprache geschrieben sind und Programmiertechniken in verschiedenen Betriebsumgebungen veranschaulichen. Sie dürfen diese Musterprogramme kostenlos kopieren, ändern und verteilen, wenn dies zu dem Zweck geschieht, Anwendungsprogramme zu entwickeln, zu verwenden, zu vermarkten oder zu verteilen, die mit der Anwendungsprogrammierschnittstelle für die Betriebsumgebung 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. Die Musterprogramme werden ohne Wartung (auf "as-is"-Basis) und ohne jegliche Gewährleistung zur Verfügung gestellt. IBM haftet nicht für Schäden, die durch Verwendung der Musterprogramme entstehen.

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

IBM, das IBM Logo und ibm.com sind Marken oder eingetragene Marken der IBM Corporation in den USA und/oder anderen Ländern. Weitere Produkt- oder Servicenamen können Marken von oder anderen Herstellern sein. IBM oder anderen Herstellern sein. Eine aktuelle Liste der IBM Marken finden Sie auf der Webseite "Copyright and trademark information" unter www.ibm.com/legal/copytrade.shtml.

Die folgenden Namen sind Marken oder eingetragene Marken anderer Unternehmen.

- Linux ist eine eingetragene Marke von Linus Torvalds in den USA und/oder anderen Ländern.
- Java und alle auf Java basierenden Marken und Logos sind Marken oder eingetragene Marken von Oracle und/oder ihren verbundenen Unternehmen.
- UNIX ist eine eingetragene Marke von The Open Group in den USA und anderen Ländern.
- Intel, das Intel-Logo, Intel Inside, das Intel Inside-Logo, Intel Centrino, das Intel Centrino-Logo, Celeron, Intel Xeon, Intel SpeedStep, Itanium und Pentium sind Marken oder eingetragene Marken der Intel Corporation oder deren Tochtergesellschaften in den USA oder anderen Ländern.
- Microsoft, Windows, Windows NT und das Windows-Logo sind Marken der Microsoft Corporation in den USA und/oder anderen Ländern.

Weitere Unternehmens-, Produkt- oder Servicenamen können Marken anderer Hersteller sein.

Index

Sonderzeichen

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

Q

R

S

T

U

V

W

X

Z

SC12-4289-03

Spine information:

DB2 for Linux, UNIX and Windows **Version 9 Release 7**

Fehlerbehebung und Optimieren der Datenbankleistung