IBM DB2 Universal Database

Systemverwaltung: Konzept

Version 7

SC12-2879-01

IBM DB2 Universal Database

Systemverwaltung: Konzept

Version 7

SC12-2879-01

Anmerkung:

Vor Verwendung dieser Informationen und des darin beschriebenen Produkts sollten die allgemeinen Informationen unter „Anhang F. Bemerkungen" auf Seite 331 gelesen werden.

- Die IBM Homepage finden Sie im Internet unter: **ibm.com**
- IBM und das IBM Logo sind eingetragene Marken der International Business Machines Corporation.
- Das e-business Symbol ist eine Marke der International Business Machines Corporation
- Infoprint ist eine eingetragene Marke der IBM.
- ActionMedia, LANDesk, MMX, Pentium und ProShare sind Marken der Intel Corporation in den USA und/oder anderen Ländern.
- C-bus ist eine Marke der Corollary, Inc. in den USA und/oder anderen Ländern.
- Java und alle Java-basierenden Marken und Logos sind Marken der Sun Microsystems, Inc. in den USA und/oder anderen Ländern.
- Microsoft Windows, Windows NT und das Windows-Logo sind Marken der Microsoft Corporation in den USA und/oder anderen Ländern.
- PC Direct ist eine Marke der Ziff Communications Company in den USA und/oder anderen Ländern.
- SET und das SET-Logo sind Marken der SET Secure Electronic Transaction LLC.
- UNIX ist eine eingetragene Marke der Open Group in den USA und/oder anderen Ländern.
- Marken anderer Unternehmen/Hersteller werden anerkannt.

Diese Veröffentlichung ist eine Übersetzung des Handbuchs
IBM DB2 Universal Database Administration Guide: Planning,
IBM Form SC09-2946-01,
herausgegeben von International Business Machines Corporation, USA

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

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

Änderung des Textes bleibt vorbehalten.

Herausgegeben von:
SW TSC Germany
Kst. 2877
August 2001

Inhaltsverzeichnis

Zu diesem Handbuch

Das Handbuch zur Systemverwaltung bietet in seinen drei Bänden Informationen, die zur Verwendung und Verwaltung der Jahr 2000-konformen DB2-Produkte für Verwaltungssysteme für relationale Datenbanken (RDBMS) benötigt werden:

- Informationen zum Datenbankentwurf (im Band *Systemverwaltung: Konzept*)
- Informationen zur Implementierung und Verwaltung von Datenbanken (im Band *Systemverwaltung: Implementierung*)
- Informationen zur Konfiguration und Optimierung der Datenbankumgebung zum Zweck der Leistungsverbesserung (im Band *Systemverwaltung: Optimierung*).

Viele der in diesem Handbuch beschriebenen Operationen können mit Hilfe verschiedener Schnittstellen durchgeführt werden:

- Der **Befehlszeilenprozessor**, der Ihnen den Zugriff auf Datenbanken und deren Bearbeitung über eine grafische Schnittstelle ermöglicht. Von dieser Schnittstelle aus können Sie SQL-Anweisungen und DB2-Dienstprogrammfunktionen ausführen. Die Mehrzahl der Beispiele in diesem Handbuch zeigt die Verwendung dieser Schnittstelle. Weitere Informationen zur Verwendung des Befehlszeilenprozessors finden Sie im Handbuch *Command Reference*.
- Die **Anwendungsprogrammierschnittstelle**, die Ihnen die Ausführung von DB2-Dienstprogrammfunktionen innerhalb eines Anwendungsprogramms ermöglicht. Weitere Informationen zur Verwendung der Anwendungsprogrammierschnittstelle finden Sie im Handbuch *Administrative API Reference*.
- Die **Steuerzentrale**, die Ihnen die Ausführung administrativer Aufgaben wie das Konfigurieren des Systems, die Verwaltung von Verzeichnissen, das Sichern und Wiederherstellen des Systems, die zeitliche Festlegung von Jobs und die Verwaltung von Medien über eine grafische Schnittstelle ermöglicht. Die Steuerzentrale enthält außerdem eine `Replikationsverwaltung`, mit der die Replikation von Daten zwischen den Systemen mit Hilfe einer grafischen Schnittstelle eingerichtet werden kann. Darüber hinaus ermöglicht die Steuerzentrale das Ausführen von DB2-Dienstprogrammfunktionen über eine grafische Benutzerschnittstelle. Zum Aufrufen der Steuerzentrale gibt es abhängig von der Plattform verschiedene Methoden. Zum Beispiel verwenden Sie den Befehl `db2cc` in einer Befehlszeile, wählen (unter OS/2) das Symbol der Steuerzentrale im DB2-Ordner aus oder arbeiten mit dem Startmenü auf Windows-Plattformen. Wenn Sie eine einführende Hilfe

benötigen, wählen Sie **Erste Schritte** im Menü **Hilfe** des Fensters der Steuerzentrale aus. **Visual Explain** und **Performance Monitor** werden von der Steuerzentrale aus aufgerufen.

Es gibt darüber hinaus noch weitere Tools, die Sie zur Durchführung von Verwaltungsaufgaben verwenden können. Dazu gehören:

- Die `Prozedurzentrale`, die zum Speichern kleiner Anwendungen dient, die als Prozeduren (Scripts) bezeichnet werden. Diese Prozeduren können SQL-Anweisungen, DB2-Befehle und auch Betriebssystembefehle enthalten.
- Die `Alert-Zentrale`, die zur Überwachung von Nachrichten dient, die sich aus anderen DB2-Operationen ergeben.
- Das Programm `Tools - Einstellungen`, mit dem Sie die Einstellungen für die Steuerzentrale, die Alert-Zentrale und die Replikation ändern können.
- Das `Journal`, das zur zeitlichen Terminierung von Jobs dient, die im nicht-überwachten Modus ausgeführt werden sollen.
- Die `Data Warehouse-Zentrale`, die zur Verwaltung von Warehouse-Objekten dient.

Zielgruppe

Dieses Handbuch ist hauptsächlich für Datenbankadministratoren, Systemadministratoren, Sicherheitsadministratoren und Systembediener gedacht, die eine Datenbank für den lokalen oder fernen Zugriff entwerfen, implementieren und pflegen müssen. Es wendet sich auch an Programmierer und andere Benutzer, die Kenntnisse über die Verwaltung und Bedienung des relationalen Datenbankverwaltungssystems von DB2 benötigen.

Aufbau dieses Handbuchs

Das vorliegende Handbuch enthält Informationen zu folgenden Hauptthemen:

DB2 Universal Database

- Kapitel 1. Verwalten von DB2 Universal Database, enthält eine Einführung sowie eine Übersicht zu DB2 Universal Database.

Datenbankkonzepte

- Kapitel 2. Allgemeine Konzepte relationaler Datenbanken, enthält eine Übersicht über Datenbankobjekte, einschließlich Speicherobjekten und Systemobjekten.
- Kapitel 3. Systeme zusammengeschlossener Datenbanken, behandelt Systeme zusammengeschlossener Datenbanken, d. h. Datenbankverwaltungssysteme (DBMS), die Anwendungen und Benutzer unterstützen, die SQL-Anweisungen übergeben, die auf zwei oder mehr DBMSs oder Datenbanken in einer einzigen Anweisung verweisen.

- Kapitel 4. Parallele Datenbanksysteme, enthält eine Einführung in die Arten von Parallelität, die mit Hilfe von DB2 implementiert werden können.
- Kapitel 5. Data Warehouses, enthält eine Übersicht über den Einsatz von Data Warehouses und Data Warehouse-Funktionen.
- Kapitel 6. Informationen zu Spatial Extender, stellt Spatial Extender vor und enthält Erläuterterungen zum Zweck des Produkts sowie zu den Daten, die von ihm verarbeitet werden.

Datenbankentwurf

- Kapitel 7. Entwerfen des logischen Datenbankaufbaus, behandelt die Konzepte und Richtlinien zum logischen Entwurf einer Datenbank.
- Kapitel 8. Entwerfen der physischen Datenbank, behandelt die Richtlinien zum physischen Entwurf einer Datenbank und enthält Überlegungen im Hinblick auf die Datenspeicherung.
- Kapitel 9. Entwerfen verteilter Datenbanken, beschreibt die Möglichkeit des Zugriffs auf mehrere Datenbanken in einer einzigen Transaktion.
- Kapitel 10. Entwerfen für Transaktionsmanager, behandelt die Verwendung von Datenbanken in einer Umgebung für verteilte Transaktionsverarbeitung wie CICS.
- Kapitel 11. Einführung in die Unterstützung der hohen Verfügbarkeit und der Funktionsübernahme, bietet eine Übersicht über die von DB2 bereitgestellte Unterstützung von Funktionsübernahmen zur Implementierung hoher Verfügbarkeit.

Anhänge

- Anhang A. Namenskonventionen, enthält die Regeln, die bei der Benennung von Datenbanken und Objekten zu beachten sind.
- Anhang B. Planen der Datenbankmigration, enthält Informationen zur Migration von Datenbanken auf Version 7.
- Anhang C. Inkompatibilitäten zwischen Releases, behandelt die entstandenen Inkompatibilitäten von Release zu Release bis einschließlich Version 7.
- Anhang D. Unterstützung von Landessprachen, gibt eine Einführung in die DB2-Unterstützung von Landessprachen und enthält Informationen zu Ländern, Sprachen und Codepages.
- Anhang E. Verwenden der DB2-Bibliothek, enthält Informationen zu der Struktur der DB2-Bibliothek, einschließlich Assistenten, Online-Hilfefunktion, Nachrichten und Handbücher.

Kurzübersicht über die anderen Bände des Handbuchs zur Systemverwaltung

Systemverwaltung: Implementierung

Der Band *Systemverwaltung: Implementierung* behandelt die Implementierung des entwickelten Datenbankentwurfs. Die einzelnen Kapitel und Anhänge des Bandes werden im Folgenden kurz vorgestellt:

Systemverwaltung mit der Steuerzentrale

- Das Kapitel über das Verwalten von DB2 mit GUI-Tools beschreibt die Tools der grafischen Benutzerschnittstelle (GUI), die zur Verwaltung der Datenbank verwendet werden.

Implementieren des Datenbankentwurfs

- Das Kapitel "Vor dem Erstellen einer Datenbank" beschreibt die Voraussetzungen, die vor der Erstellung einer Datenbank erfüllt sein müssen.
- Das Kapitel über das Erstellen einer Datenbank beschreibt die Aufgaben der Erstellung einer Datenbank sowie der zugehörigen Datenbankobjekte.
- Das Kapitel über das Ändern einer Datenbank behandelt, was vor der Änderung einer Datenbank zu tun ist und welche Aufgaben im Zusammenhang mit dem Ändern oder Löschen einer Datenbank oder zugehöriger Datenbankobjekte erledigt werden müssen.

Datenbanksicherheit

- Das Kapitel zur Steuerung des Datenbankzugriffs beschreibt die Möglichkeiten zur Steuerung des Zugriffs auf die Ressourcen einer Datenbank.
- Das Kapitel zur Protokollierung von DB2-Aktivitäten beschreibt Methoden zur Erkennung und Überwachung unerwünschten bzw. unvorhergesehenen Zugriffs auf Daten.

Versetzen von Daten

- Das Kapitel zu den Dienstprogrammen zum Versetzen von Daten ist eine kurze Einführung in die verschiedenen Möglichkeiten zum Versetzen von Daten, die Sie an das Handbuch *Versetzen von Daten Dienstprogramme und Referenz* verweist.

Wiederherstellung

- Die Informationen über das Wiederherstellen einer Datenbank enthalten eine kurze Einführung in die Konzepte der Sicherung, Wiederherstellung und aktualisierenden Wiederherstellung von Datenbanken. Eingehendere Informationen finden Sie in *Datenwiederherstellung und hohe Verfügbarkeit Handbuch und Referenz*.

Anhänge

- Der Anhang zur Verwendung der DCE Verzeichnisservices enthält Informationen zu den Einsatzmöglichkeiten der DCE Verzeichnisservices.
- Der Anhang über den Benutzer-Exit zur Datenbankwiederherstellung behandelt die Verwendung von Benutzer-Exit-Programmen für Datenbankprotokolldateien und beschreibt einige Beispiele von Benutzer-Exit-Programmen.
- Der Anhang über das Absetzen von Befehlen an mehrere Datenbankpartitions-Server behandelt die Verwendung der Shell-Prozeduren *db2_all* und *rah* zum Senden von Befehlen an alle Partitionen in einer Umgebung mit partitionierten Datenbanken.
- Der Anhang über die Arbeitsweise von DB2 für Windows NT mit der Windows NT-Sicherheit beschreibt, wie DB2 mit der Windows NT-Sicherheit funktioniert.
- Der Anhang zur Verwendung des Windows NT-Systemmonitors enthält Informationen über das Registrieren von DB2 beim Windows NT-Systemmonitor und über die Nutzung der Leistungsinformationen.
- Der Anhang zur Arbeit mit Windows NT- oder Windows 2000-Datenbankpartitions-Servern enthält Informationen über die verfügbaren Dienstprogramme zur Arbeit mit Datenbankpartitions-Servern unter Windows NT oder Windows 2000.
- Der Anhang zur Konfiguration mehrerer logischer Knoten beschreibt, wie mehrere logische Knoten in einer Umgebung mit partitionierten Datenbanken konfiguriert werden.
- Der Anhang zur Hochgeschwindigkeitskommunikation zwischen Knoten beschreibt, wie VIA (Virtual Interface Architecture) zur Verwendung mit DB2 Universal Database eingerichtet wird.
- Der Anhang den LDAP-Verzeichnisservices enthält Informationen zu den Einsatzmöglichkeiten der LDAP-Verzeichnisservices.
- Der Anhang zur Erweiterung der Steuerzentrale enthält Informationen zur Erweiterung der Steuerzentrale durch Hinzufügen neuer Knöpfe in der Menüleiste, einschließlich neuer Aktionen, Hinzufügen neuer Objektdefinitionen und Hinzufügen neuer Aktionsdefinitionen.

Systemverwaltung: Optimierung

Der Band *Systemverwaltung: Optimierung* behandelt Themen zur Systemleistung. Damit sind die Themen gemeint, die das Einrichten, Testen und Optimieren der Leistung von Anwendungen sowie der Leistung des Produkts DB2 Universal Database betreffen. Die einzelnen Kapitel und Anhänge des Bandes werden im Folgenden kurz vorgestellt:

Einführung in die Optimierung

- Das Kapitel zu den Leistungselementen führt in die Konzepte und Überlegungen zur Verwaltung und Optimierung der Leistung von DB2 UDB ein.
- Die Übersicht zur Archtitektur und Verarbeitung stellt die zugrundeliegende Architektur und die Prozesse von DB2 Universal Database vor.

Optimieren der Anwendungsleistung

- Die Überlegungen zu den Anwendungen beschreiben einige Techniken zur Verbesserung der Datenbankleistung beim Entwurf der Anwendungen.
- Die Überlegungen zur Umgebung beschreiben einige Techniken zur Verbesserung der Datenbankleistung bei der Einrichtung der Datenbankumgebung.
- Das Kapitel zu den Systemkatalogstatistiken beschreibt einige Techniken zur Erfassung von Statistiken über die Daten und ihre Verwendung zur Gewährleistung einer optimalen Leistung.
- Das Kapitel zum SQL-Compiler beschreibt die Verarbeitung einer SQL-Anweisung durch den SQL-Compiler bei der Kompilierung.
- Das Kapitel zur SQL-EXPLAIN-Einrichtung beschreibt die Einrichtung EXPLAIN, die Ihnen ermöglicht, die Pfade und Methoden anzuzeigen, die der SQL-Compiler ausgewählt hat, um auf die Daten zuzugreifen.

Optimieren und Konfigurieren des Systems

- Das Kapitel zur Leistung bei der Ausführung gibt einen Überblick über die Verwendung von Speicher durch den Datenbankmanager und enthält Informationen zu weiteren Faktoren, die sich auf die Leistung zur Laufzeit auswirken.
- Das Kapitel zum Programm Governor gibt eine Einführung in die Verwendung des Programms Governor, mit dem einige Aspekte der Datenbankverwaltung gesteuert werden können.
- Das Kapitel zum Skalieren Ihrer Konfiguration enthält einige Gesichtspunkte und Hinweise, die bei der Erweiterung des Datenbanksystems von Bedeutung sind.
- Das Kapitel zur Umverteilung von Daten auf Datenbankpartitionen behandelt die Punkte, die in einer Umgebung mit partitionierten Datenbanken bei der Neuverteilung von Daten auf die Partitionen zu beachten sind.
- Das Kapitel zu Vergleichstests (Benchmark-Tests) gibt einen Überblick über Vergleichstests und behandelt verschiedene Aspekte ihrer Durchführung.
- Das Kapitel zur DB2-Konfiguration behandelt die Dateien zur Konfiguration des Datenbankmanagers und der Datenbanken sowie die Werte für die Konfigurationsparameter.

Anhänge

- Der Anhang zu DB2-Registrierungsvariablen und DB2-Umgebungsvariablen beschreibt Werte für die Profilregistrierdatenbank und für Umgebungsvariablen.
- Der Anhang zu EXPLAIN-Tabellen und -Definitionen beschreibt die Tabellen, die von der DB2-EXPLAIN-Einrichtung verwendet werden, und die Erstellung dieser Tabellen.
- Der Anhang zu den SQL-Explain-Programmen beschreibt die Verwendung der DB2-EXPLAIN-Programme: db2expln und dynexpln.
- Der Anhang zu db2exfmt - Tool für EXPLAIN-Tabellenformat - beschreibt, wie mit dem DB2-EXPLAIN-Programm EXPLAIN-Tabellendaten formatiert werden.

Teil 1. DB2 Universal Database

Kapitel 1. Verwalten von DB2 Universal Database

Mit DB2 verfügen Sie über die Flexibilität, eine große Bandbreite von Hardwarekonfigurationen betreiben zu können. DB2 ermöglicht Ihnen, eine spezielle DB2-Produktkonfiguration optimal auf Ihre Hardware- und Anwendungsanforderungen abzustimmen.

DB2 unterstützt viele verschiedene Komplexitätsebenen in Datenbankumgebungen, und Sie erhalten Informationen und Anleitungen, die jeweils speziell auf eine bestimmte Umgebung zugeschnitten sind. Diese sind in detaillierter Form sowohl im Handbuch *Systemverwaltung* als auch in anderen Handbüchern der DB2-Bibliothek enthalten (siehe „Anhang E. Verwenden der DB2-Bibliothek" auf Seite 307). In einigen Fällen treffen ganze Abschnitte dieser Handbücher nur eine bestimmte Umgebung zu. Im Vorwort zu diesem Handbuch („Zu diesem Handbuch") erfahren Sie, welche Kapitel in diesem und den anderen Bänden des Handbuchs *Systemverwaltung* (*Systemverwaltung: Implementierung* und *Systemverwaltung: Optimierung*) zur Umsetzung Ihrer Geschäftsanforderung relevant sind.

Wenn Sie keine Erfahrung mit Verwaltungssystemen für relationale Datenbanken (RDBMS) oder mit DB2 haben, sind die Informationen im Abschnitt „Allgemeine Konzepte relationaler Datenbanken" für Sie sicherlich hilfreich. Wenn Sie mit diesen Konzepten bereits vertraut sind oder keine Auffrischung benötigen, können Sie sich direkt den Abschnitten zuwenden, die weiterführende Themen wie zum Beispiel folgende im Einzelnen behandeln:

- Systeme zusammengeschlossener Datenbanken. In diesem Abschnitt werden Datenbankverwaltungssysteme (DBMS) behandelt, die Anwendungen und Benutzer unterstützen, die SQL-Anweisungen übergeben, die auf zwei oder mehr DBMSs oder Datenbanken in einer einzigen Anweisung verweisen.
- Systeme paralleler Datenbanken. Dieser Abschnitt enthält eine Einführung in die Arten von Parallelität, die mit Hilfe von DB2 implementiert werden können. Komponenten einer Task, wie zum Beispiel einer Datenbankabfrage, können parallel ausgeführt werden, um die Leistung erheblich zu erhöhen.
- Verteilte Transaktionsverarbeitung. Dieser Abschnitt enthält Informationen darüber, wie in einer einzigen Transaktion auf mehrere Datenbanken zugegriffen werden kann und wie Datenbanken in einer Umgebung mit verteilter Transaktionsverarbeitung eingesetzt werden können.

DB2 kann den spezialisiertesten Anforderungen des Datenmanagements Rechnung tragen, wie zum Beispiel folgenden:

- *Replikation*, die ein Kopieren von Daten auf mehrere ferne Datenbanken in regelmäßigen Abständen ermöglicht. Wenn Aktualisierungen aus einer Masterdatenbank automatisch in andere Datenbanken kopiert werden müssen, können Sie mit Hilfe der Replikationsfunktionen von DB2 angeben, welche Daten zu kopieren sind, in welche Datenbanktabellen die Daten zu kopieren sind und wie oft die Aktualisierungen zu kopieren sind. Falls Sie auf die Replikationseinrichtungen von DB2 zurückgreifen wollen, lesen Sie die Informationen im *Replikation Benutzer- und Referenzhandbuch*. Dieses Handbuch führt in die Konzepte der DB2-Datenreplikation ein und beschreibt die Planung, Konfiguration und Verwaltung einer Replikationsumgebung.
- *Data Warehouses*, die eine Erstellung von Speichern „informativer Daten" bzw. von Daten ermöglichen, die aus den Betriebsdaten extrahiert und für Endbenutzer zur Entscheidungsfindung aufbereitet werden. Zum Beispiel könnte ein Data Warehouse-Tool alle Verkaufsdaten aus der Betriebsdatenbank kopieren, Berechnungen durchführen, um die Daten zusammenzufassen, und die zusammengefassten Daten in eine Zieltabelle einer getrennten Datenbank schreiben. Die getrennte Datenbank (das sog. *Warehouse*) kann abgefragt werden, ohne die Betriebsdatenbanken zu beeinträchtigen. Detaillierte Informationen zu Data Warehouses finden Sie im Handbuch *Data Warehouse-Zentrale Verwaltung*.
- Ein *geografisches Informationssystem* (GIS - Geographic Information System), das durch den Spatial Extender erstellt werden kann. Ein GIS ist ein Komplex von Objekten, Daten und Anwendungen, der es ermöglicht, räumliche Informationen über geografische Merkmale zu generieren und zu analysieren. Im Spatial Extender kann ein geografisches Merkmal durch eine Zeile in einer Tabelle oder Sicht bzw. durch einen Teil einer solchen Zeile dargestellt werden. Detaillierte Informationen zur Verwendung des Spatial Extender finden Sie im Handbuch *Spatial Extender Benutzer- und Referenzhandbuch*.

Das Handbuch *Systemverwaltung: Konzept* behandelt auch das Thema des Datenbankentwurfs, der die für DB2 relevanten Aspekte des logischen und physischen Datenbankdesigns umfasst. Andere konzeptionelle Themen, wie zum Beispiel die Planung einer Datenbankmigration, die Ermittlung von Inkompatibilitäten, die sich auf die Anwendungen auswirken können (eine *Inkompatibilität* ist ein Teil von DB2 Universal Database, das anders funktioniert als in einem früheren Release von DB2.

Falls auf eine solche in einer vorhandenen Anwendung zurückgegriffen wird, führt sie zu einem unerwarteten Ergebnis, macht eine Änderung der Anwendung erforderlich oder verringert die Leistung) und die Nutzung der Unterstützung nationaler Zeichensätze werden ebenfalls behandelt.

Der Band *Systemverwaltung: Implementierung* enthält Informationen zur Implementierung des entwickelten Datenbankentwurfs. Zu den behandelten Themen gehören das Erstellen und Ändern einer Datenbank, die Datenbanksicherheit und die Verwaltung von DB2 über die Steuerzentrale, eine grafische Benutzerschnittstelle für DB2.

Der Band *Systemverwaltung: Optimierung* behandelt Themen und Aspekte der Leistung, Leistungstests und Methoden der Leistungsoptimierung für Anwendungen und DB2.

Teil 2. Datenbankkonzepte

Kapitel 2. Allgemeine Konzepte relationaler Datenbanken

Dieser Abschnitt behandelt die folgenden Themen:

Übersicht über Datenbankobjekte

Dieser Abschnitt enthält eine Übersicht über die folgenden wichtigen Datenbankobjekte:

- Exemplare
- Datenbanken
- Knotengruppen
- Tabellen
- Sichten
- Indizes
- Schemata
- Systemkatalogtabellen

Abb. 1 auf Seite 10 verdeutlicht die Beziehungen zwischen einigen dieser Objekte. Sie zeigt außerdem, dass Tabellen, Indizes und lange Daten (LOB-Daten) in Tabellenbereichen gespeichert werden.

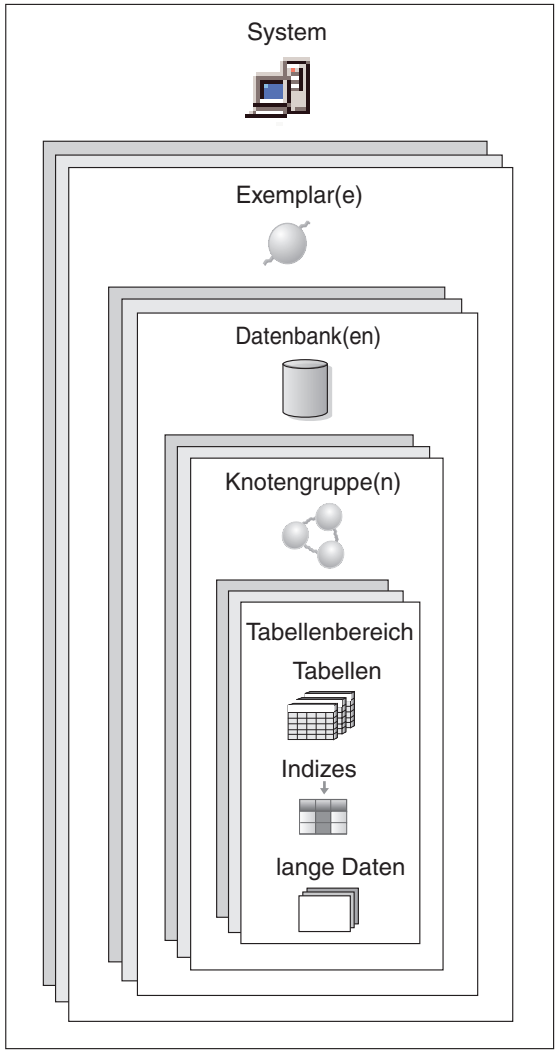

Abbildung 1. Beziehungen zwischen einigen Datenbankobjekten

Exemplare

Ein *Exemplar* (zuweilen auch als *Datenbankmanager* bezeichnet) ist der DB2-Code, der die Daten verwaltet. Er steuert, welche Operationen an den Daten ausgeführt werden können, und verwaltet die Systemressourcen, die ihm zugeordnet sind. Jedes Exemplar stellt eine vollständige Umgebung dar. Es enthält sämtliche Datenbankpartitionen, die für ein bestimmtes paralleles Datenbanksystem definiert sind (siehe „Kapitel 4. Parallele Datenbanksysteme” auf Seite 39).

Ein Exemplar verfügt über eigene Datenbanken (auf die andere Exemplare keinen Zugriff haben), und alle zugehörigen Datenbankpartitionen benutzen gemeinsame Systemverzeichnisse. Es verfügt außerdem über ein von anderen Exemplaren auf derselben Maschine (System) getrenntes Sicherheitssystem.

Datenbanken

Eine *relationale Datenbank* stellt Daten als Sammlung von Tabellen dar. Eine Tabelle besteht auf einer definierten Anzahl von Spalten und einer beliebigen Anzahl von Zeilen. Jede Datenbank enthält einen Satz von Katalogtabellen, die die logische und physische Struktur der Daten beschreiben, eine Konfigurationsdatei, die die der Datenbank zugeordneten Parameterwerte enthält, und ein Wiederherstellungsprotokoll, das laufende Transaktionen und archivierbare Transaktionen enthält.

Knotengruppen

Eine *Knotengruppe* ist eine Gruppe aus einer oder mehreren Datenbankpartitionen. Wenn Sie Tabellen für die Datenbank erstellen wollen, erstellen Sie zunächst die Knotengruppe, in der die Tabellenbereiche gespeichert werden. Anschließend erstellen Sie die Tabellenbereiche, in denen die Tabellen gespeichert werden. Im Abschnitt „Knotengruppen und Datenpartitionierung" auf Seite 40 finden Sie weitere Informationen zu Knotengruppen. Eine Definition des Begriffs Datenbankpartition finden Sie in „Kapitel 4. Parallele Datenbanksysteme" auf Seite 39. Im Abschnitt „Tabellenbereiche" auf Seite 15 finden Sie weitere Informationen zu Tabellenbereichen.

Tabellen

Eine relationale Datenbank stellt Daten als Sammlung von Tabellen dar. Eine *Tabelle* besteht aus Daten, die logisch in Spalten und Zeilen angeordnet sind. Alle Datenbank- und Tabellendaten werden Tabellenbereichen zugeordnet. Im Abschnitt „Tabellenbereiche" auf Seite 15 finden Sie weitere Informationen zu Tabellenbereichen. Die Daten in der Tabelle besitzen eine logische Beziehung, und zwischen Tabellen können Abhängigkeitsbedingungen definiert werden. Daten können nach mathematischen Prinzipien und mit Hilfe als *Relationen* bezeichneter Operationen in Sichten zusammengefasst und bearbeitet werden.

Der Zugriff auf Tabellendaten erfolgt mit Hilfe der Structured Query Language (SQL, siehe Handbuch *SQL Reference*), einer standardisierten Sprache zur Definition und Bearbeitung von Daten in einer relationalen Datenbank. Eine *Abfrage* wird in Anwendungen oder von Benutzern verwendet, um Daten aus einer Datenbank abzurufen. In der Abfrage wird unter Verwendung von SQL eine Anweisung in folgendem Format erstellt:

```
SELECT <datenname> FROM <tabellenname>
```

Sichten

Eine *Sicht* bietet eine effektive Methode zur Darstellung von Daten, ohne sie pflegen zu müssen. Eine Sicht ist keine wirkliche Tabelle und erfordert keine permanente Speicherung. Es wird eine „virtuelle Tabelle" erstellt und verwendet.

Eine Sicht kann alle oder einige der Spalten oder Zeilen der Tabelle enthalten, auf der sie basiert. Zum Beispiel können Sie eine Tabelle mit Abteilungsdaten (DEPARTMENT) und eine Tabelle mit Mitarbeiterdaten (EMPLOYEE) in einer Sicht verknüpfen, so dass Sie alle Mitarbeiter in einer bestimmten Abteilung auflisten können.

Abb. 2 zeigt die Beziehung zwischen Tabellen und Sichten.

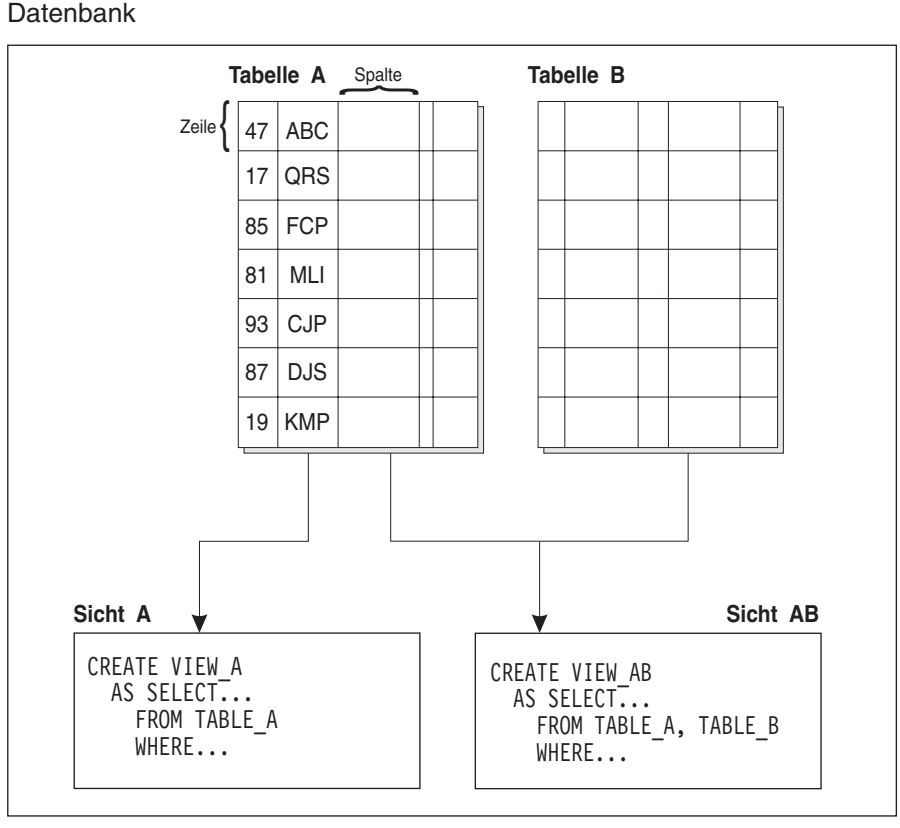

Abbildung 2. Beziehung zwischen Tabellen und Sichten

Indizes

Ein *Index* ist ist eine Menge von Schlüsseln, die jeweils auf Zeilen in einer Tabelle zeigen. Zum Beispiel hat Tabelle A in Abb. 3 einen Index, der auf den Personalnummern in der Tabelle basiert. Dieser Schlüsselwert dient als Zeiger auf die Zeile in der Tabelle: die Personalnummer 19 zeigt auf Mitarbeiter KMP. Ein Index ermöglicht einen effizienteren Zugriff auf Zeilen einer Tabelle, indem er einen direkten Pfad zu den Daten mit Hilfe von Zeigern erstellt.

Das *SQL-Optimierungsprogramm* wählt automatisch den effizientesten Weg für den Zugriff auf Daten in Tabellen aus. Das Optimierungsprogramm bezieht Indizes bei der Ermittlung des schnellsten Pfads zu den Daten mit in Betracht.

Es können eindeutige Indizes erstellt werden, um die Eindeutigkeit des Indexschlüssels sicherzustellen. Ein *Indexschlüssel* ist eine Spalte oder eine geordnete Gruppe von Spalten, auf denen ein Index definiert wird. Mit Hilfe eines eindeutigen Index wird gewährleistet, dass der Wert jedes Indexschlüssels in der indizierten Spalte bzw. den indizierten Spalten eindeutig ist. Im Abschnitt „Geschäftsregeln für Daten" auf Seite 22 werden Schlüssel und Indizes eingehender behandelt.

Abb. 3 illustriert die Beziehung zwischen einem Index und einer Tabelle.

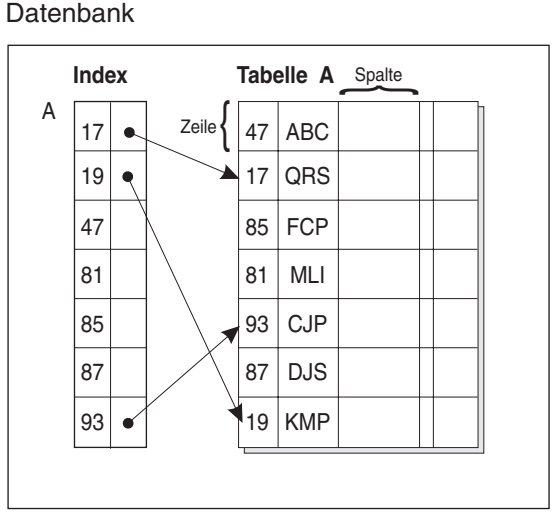

Abbildung 3. Beziehung zwischen einem Index und einer Tabelle

Schemata

Ein *Schema* ist eine Kennung, wie zum Beispiel eine Benutzer-ID, die bei der Gruppierung von Tabellen und anderen Datenbankobjekten hilfreich ist. Ein Schema kann einem Eigner zugeordnet sein, der den Zugriff auf die in ihm enthaltenen Daten und Objekte steuern kann.

Ein Schema ist außerdem ein Objekt in der Datenbank. Es kann automatisch erstellt werden, wenn das erste Objekt in einem Schema ersteltl wird. Ein solches Objekt kann ein beliebiges Objekt sein, das durch einen Schemennamen qualifiziert werden kann, wie zum Beispiel eine Tabelle, ein Index, eine Sicht, ein Paket, ein einzigartiger Datentyp, eine Funktion oder ein Auslöser. Sie müssen über die Berechtigung IMPLICIT_SCHEMA verfügen, wenn das Schema automatisch erstellt werden soll. Sie können das Schema außerdem auch explizit erstellen.

Ein Schemenname wird als erster Teil eines zweiteiligen Objektnamens verwendet. Bei der Erstellung eines Objekts können Sie es einem bestimmten Schema zuordnen. Wenn Sie kein Schema angeben, wird das Objekt dem Standardschema zugeordnet, das in der Regel die Benutzer-ID der Person ist, die das Objekt erstellt hat. Der zweite Teil des Namens ist der Name des Objekts. Zum Beispiel könnte ein Benutzer namens Smith eine Tabelle des Namens SMITH.PAYROLL besitzen.

Systemkatalogtabellen

Jede Datenbank enthält einen Satz von *Systemkatalogtabellen*, die die logische und physische Struktur der Daten beschreiben. DB2 erstellt und pflegt einen umfangreichen Satz von Systemkatalogtabellen für jede einzelne Datenbank. Diese Tabellen enthalten Informationen über die Definitionen der Definitionen von Datenbankobjekten, wie zum Beispiel Benutzertabellen, Sichten und Indizes sowie Sicherheitsinformationen über die Berechtigungen, die Benutzer für diese Objekte besitzen. Sie werden bei der Erstellung der Datenbank erstellt und im Rahmen des normalen Datenbankbetriebs aktualisiert. Sie können nicht explizit erstellt oder gelöscht werden, jedoch kann ihr Inhalt mit Hilfe von Katalogsichten abgefragt und angezeigt werden.

Übersicht über Speicherobjekte

Mit Hilfe der folgenden Datenbankobjekte können Sie definieren, wie Daten auf dem System gespeichert werden und wie die (für den Zugriff auf die Daten relevante) Leistung verbessert werden kann:

- Tabellenbereich
- Behälter
- Pufferpool

Tabellenbereiche

Eine Datenbank wird in Bestandteilen verwaltet, die als *Tabellenbereiche* bezeichnet werden. Ein Tabellenbereich ist ein Platz zum Speichern von Tabellen. Bei der Erstellung einer Tabelle können Sie sich entschließen, bestimmte Objekte wie Indizes und große Objekte (LOB-Daten) von den übrigen Tabellendaten getrennt zu speichern. Ein Tabellenbereich kann außerdem über eine oder mehrere physische Speichereinheiten verteilt angelegt werden. Das folgende Diagramm veranschaulicht etwas von der Flexibilität, die Sie durch die Verteilung von Daten auf Tabellenbereiche erhalten:

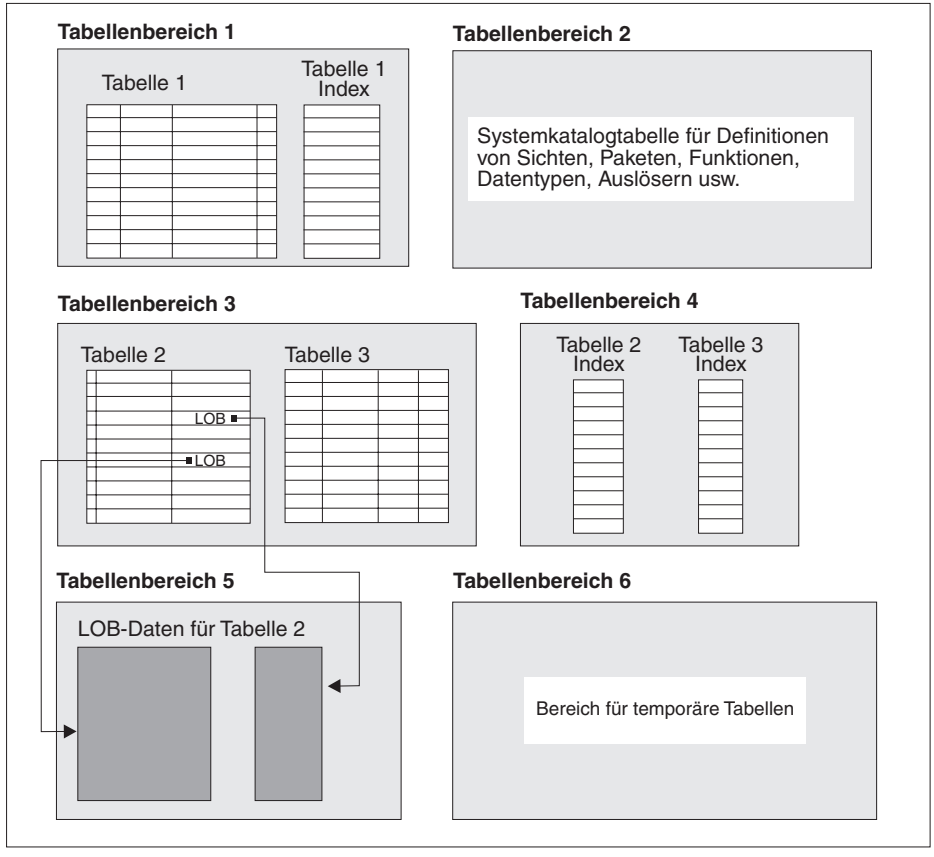

Abbildung 4. Tabellenbereiche

Tabellenbereiche befinden sich in Knotengruppen (siehe „Knotengruppen" auf Seite 11). Die Definitionen und Attribute von Tabellenbereichen werden im Systemkatalog der Datenbank aufgezeichnet (siehe „Systemkatalogtabellen" auf Seite 14).

Behälter werden Tabellenbereichen zugeordnet. Ein *Behälter* ist eine Zuordnung physischen Speichers (z. B. eine Datei oder eine Einheit).

Ein Tabellenbereich kann entweder ein vom System verwalteter Bereich (SMS - System Managed Space) oder ein von der Datenbank verwalteter Bereich (DMS - Database Managed Space) sein. Bei einem SMS-Tabellenbereich ist jeder Behälter ein Verzeichnis im Dateibereich des Betriebssystems, wobei der Speicherbereich vom Dateimanager des Betriebssystems gesteuert wird. Bei einem DMS-Tabellenbereich ist jeder Behälter entweder eine im Voraus zugeordnete Datei fester Größe oder eine physische Einheit wie eine Platte, wobei der Datenbankmanager den Speicherbereich steuert.

Abb. 5 illustriert die Beziehung zwischen Tabellen, Tabellenbereichen und den beiden Typen von Speicherbereichen. Sie zeigt außerdem, dass Tabellen, Indizes und lange Daten (LOB-Daten) in Tabellenbereichen gespeichert werden.

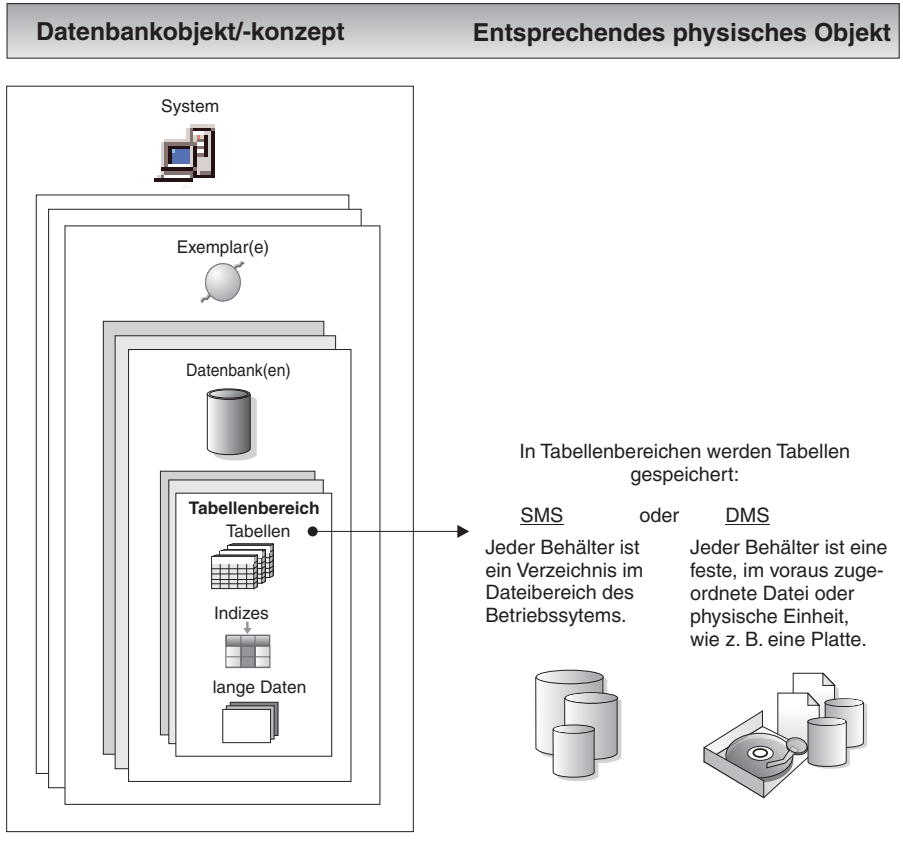

Abbildung 5. Tabellenbereiche und Tabellen

Abb. 6 auf Seite 18 zeigt die drei Arten von Tabellenbereichen: *regular*, *temporary*, und *long*.

Tabellen, die Benutzerdaten enthalten, werden in regulären Tabellenbereichen gespeichert. Der Standardtabellenbereich für Benutzer besitzt den Namen USERSPACE1. Indizes werden ebenfalls in regulären Tabellenbereichen gespeichert. Die Systemkatalogtabellen werden in einem regulären Tabellenbereich angelegt. Der Standardtabellenbereich für Systemkatalogtabellen heißt SYSCATSPACE.

Tabellen, die Spalten mit sehr langen Zeichenfolgen (Long) oder lange Objekte (LOB) wie zum Beispiel Multimediaobjekte enthalten, werden in Tabellenbereichen für lange Objektdaten (LONG) gespeichert.

Temporäre Tabellenbereiche werden als Systemtabellenbereiche oder Benutzertabellenbereiche klassifiziert. *Temporäre Systemtabellenbereiche* dienen zur Speicherung interner temporärer Daten, die für SQL-Operationen wie Sortieren, Reorganisieren von Tabellen, Erstellen von Indizes und Verknüpfen von Tabellen erforderlich sind. Obwohl eine beliebige Anzahl von temporären Systemtabellenbereichen erstellt werden kann, ist es zu empfehlen, nur einen solchen Tabellenbereich zu erstellen und dabei die Seitengröße zu verwenden, die von der Mehrheit der Tabellen verwendet wird. Der Standardtabellenbereich für temporäre Systemdaten heißt TEMPSPACE1. *Temporäre Benutzertabellenbereiche* dienen zur Speicherung deklarierter globaler temporärer Tabellen, in denen temporäre Daten abgelegt werden. Temporäre Benutzertabellenbereiche werden *nicht* standardmäßig bei der Erstellung einer Datenbank erstellt.

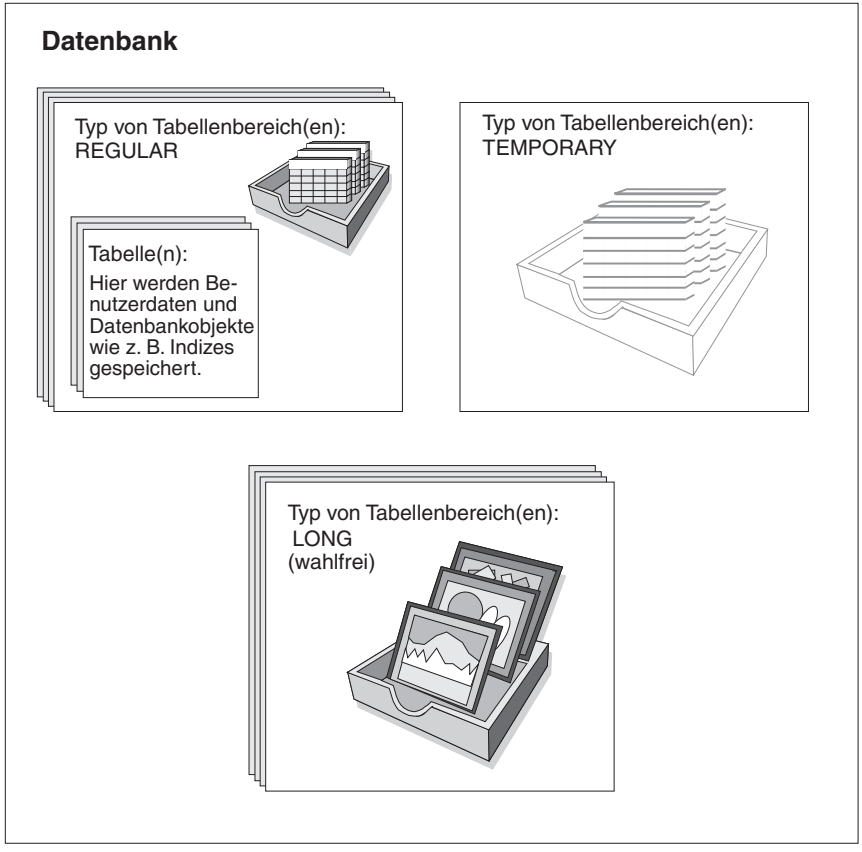

Abbildung 6. Drei Typen von Tabellenbereichen

Behälter

Ein *Behälter* ist eine physische Speichereinheit. Er kann durch einen Verzeichnisnamen, einen Einheitennamen oder einen Dateinamen bezeichnet werden.

Ein Behälter ist einem Tabellenbereich zugeordnet. Ein einzelner Tabellenbereich kann sich über viele Behälter erstrecken, jedoch kann ein Behälter jeweils nur zu einem Tabellenbereich gehören.

Abb. 7 auf Seite 19 veranschaulicht die Beziehung zwischen Tabellen und einer Tabelle innerhalb einer Datenbank und den zugeordneten Behältern und Platten.

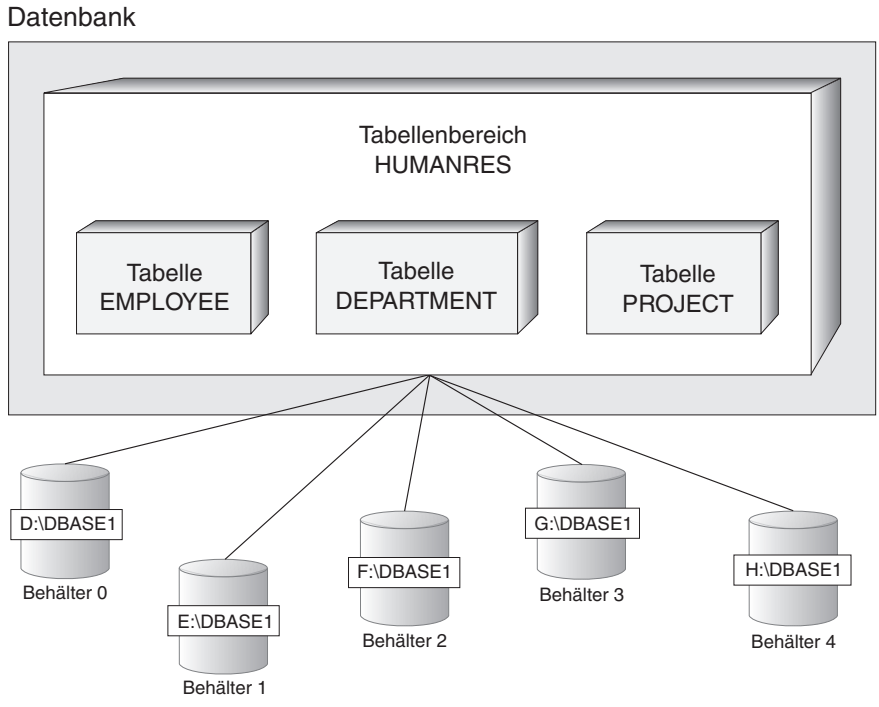

Abbildung 7. Tabellenbereiche und Tabellen in einer Datenbank

Die Tabellen EMPLOYEE, DEPARTMENT und PROJECT befinden sich im Tabellenbereich HUMANRES, der sich über die Behälter 0, 1, 2, 3 und 4 erstreckt. Dieses Beispiel zeigt jeden Behälter auf einer getrennten Platte.

Daten aller Tabellen werden in allen Behältern in einer Tabelle reihum verteilt gespeichert. Dies sorgt für eine gleichmäßige Verteilung der Daten auf die Behälter, die zu einem bestimmten Tabellenbereich gehören. Die Anzahl von Seiten, die der Datenbankmanager in einen Behälter schreibt, bevor er zu einem anderen Behälter wechselt, wird als Speicherbereich *Speicherbereich* (Extentsize) bezeichnet.

Pufferpool

Ein *Pufferpool* ist die Menge an Hauptspeicher, die beim Lesen von Daten vom Datenträger oder beim Ändern von Daten zum Zwischenspeichern von Tabellen- und Indexdatenseiten zugeordnet wird. Der Pufferpool dient zur Erhöhung der Systemleistung. Auf Daten im Hauptspeicher kann wesentlich schneller zugegriffen werden als auf Daten auf einem Datenträger. Daher ist die Leistung umso höher, je weniger oft der Datenbankmanager Daten von einem Datenträger lesen bzw. auf ihn schreiben (E/A) muss. (Sie können mehr als einen Pufferpool erstellen, wenngleich für die meisten Fälle nur ein Pufferpool erforderlich ist.)

Die Konfiguration des Pufferpools ist der wichtigste Einzelbereich der Optimierung, weil hier die Verzögerung durch eine langsame Ein-/Ausgabe (E/A) verringert werden kann.

Abb. 8 veranschaulicht die Beziehung zwischen einem Pufferpool und Behältern.

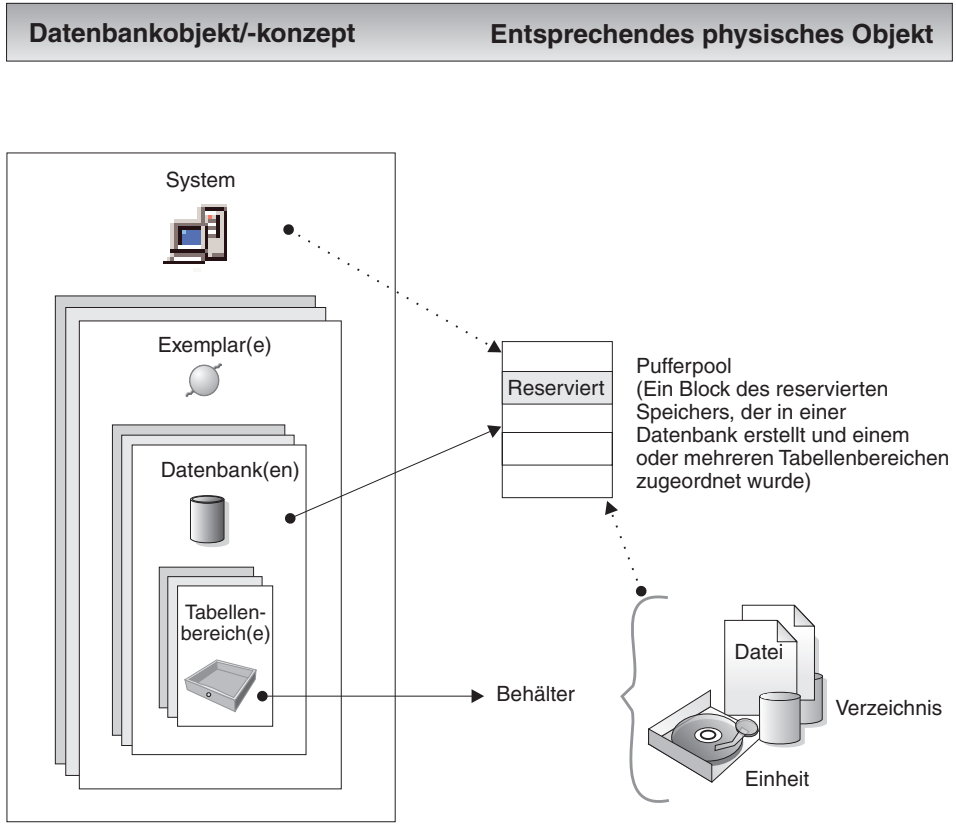

Abbildung 8. Pufferpool und Behälter

Übersicht über Systemobjekte

Bei der Erstellung eines DB2-Exemplars oder einer DB2-Datenbank wird eine entsprechende Konfigurationsdatei mit den Standardparameterwerten erstellt. Diese Parameterwerte können zur Verbesserung der Leistung geändert werden.

Konfigurationsparameter

Konfigurationsdateien enthalten Parameter, die Werte definieren, wie zum Beispiel die den DB2-Produkten zugeordneten Ressourcen und die Diagnoseebene. Es gibt zwei Arten von Konfigurationsdateien: eine Konfigurationsdatei des Datenbankmanagers für jedes DB2-Exemplar und eine Konfigurationsdatei der Datenbank für jede einzelne Datenbank (siehe Abb. 9 auf Seite 22).

Eine *Konfigurationsdatei des Datenbankmanagers* wird erstellt, wenn ein DB2-Exemplar erstellt wird. Die in ihr enthaltenen Parameter betreffen Systemressourcen auf Exemplarebene und sind von keiner Datenbank, die zu diesem Exemplar gehört, abhängig. Die Werte vieler dieser Parameter können von den Standardwerten des System zur Leistungsoptimierung oder Kapazitätserhöhung abhängig von der Konfiguration des Systems geändert werden.

Darüber hinaus gibt es auch für jede Client-Installation eine Konfigurationsdatei des Datenbankmanagers. Diese Datei enthält Informationen über den Client Enabler für eine bestimmte Workstation. Eine Untergruppe der für einen Server verfügbaren Parameter gelten für den Client.

Eine *Konfigurationsdatei der Datenbank* wird erstellt, wenn eine Datenbank erstellt wird. Sie befindet sich dort, wo sich die Datenbank befindet. Pro Datenbank gibt es eine Konfigurationsdatei. Die in ihr enthaltenen Parameter geben neben anderen Merkmalen die Menge der Ressourcen an, die der zugehörigen Datenbank zugeordnet sind. Die Werte vieler dieser Parameter können zur Leistungsoptimierung und Kapazitätserhöhung geändert werden. Abhängig von der Art der Aktivitäten in einer bestimmten Datenbank können unterschiedliche Änderungen erforderlich sein.

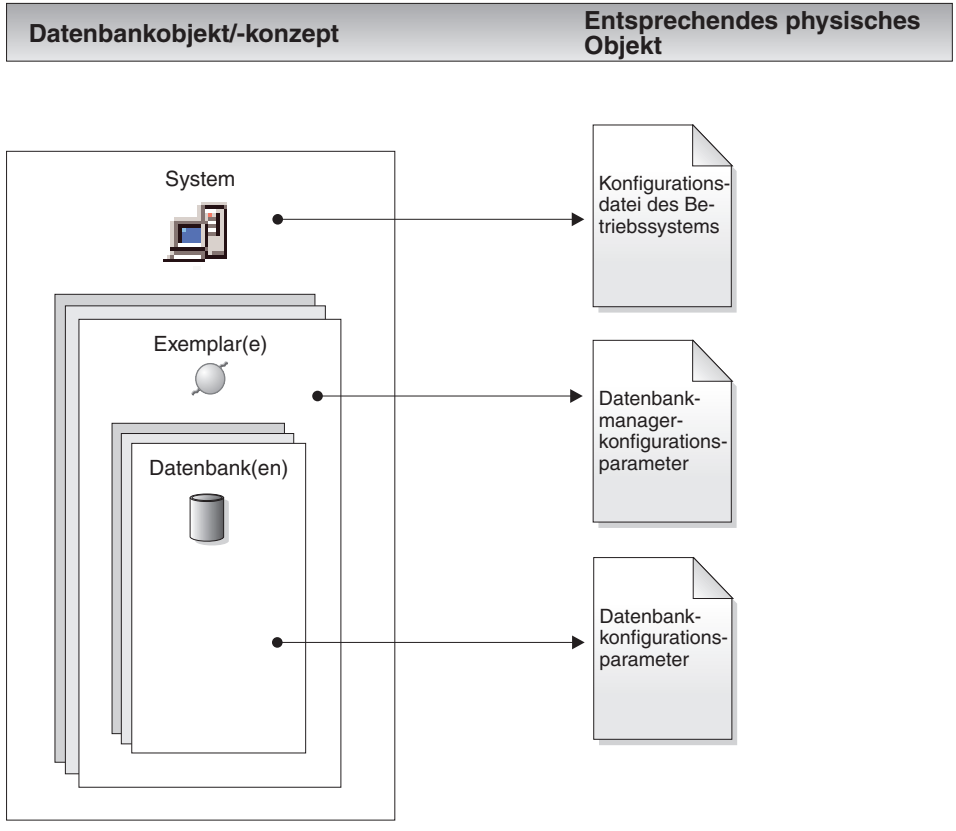

Abbildung 9. Konfigurationsparameterdateien

Geschäftsregeln für Daten

In jeder Geschäftsumgebung müssen Daten häufig bestimmten Rahmenbedingungen oder Regeln genügen. Zum Beispiel muss eine Personalnummer eindeutig sein. DB2 stellt *Integritätsbedingungen* bereit, die zur Implementierung solcher Regeln definiert werden können.

DB2 stellt die folgenden Arten von Integritätsbedingungen zur Verfügung:

- Integritätsbedingung NOT NULL
- Eindeutige Integritätsbedingung
- Integritätsbedingung über Primärschlüssel
- Integritätsbedingung über Fremdschlüssel
- Prüfung auf Integritätsbedingung

Integritätsbedingung NOT NULL

Die Integritätsbedingung NOT NULL verhindert, dass Nullwerte in eine Spalte eingegeben werden.

Eindeutige Integritätsbedingung

Eindeutige Integritätsbedingungen stellen sicher, dass die Werte in einer Gruppe von Spalten eindeutig sind und für alle Zeilen in der Tabelle nicht null sind. Zum Beispiel könnte eine typische eindeutige Integritätsbedingung in einer Tabelle DEPARTMENT mit Daten über Abteilungen darin bestehen, dass die Abteilungsnummer eindeutig und nicht null ist.

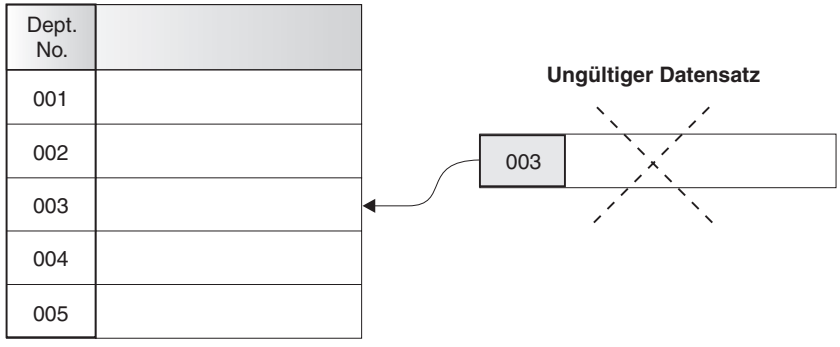

Abbildung 10. Eindeutige Integritätsbedingungen verhindern mehrfach auftretende Daten.

Der Datenbankmanager beachtet die Integritätsbedingung bei Operationen zum Einfügen und Aktualisieren von Daten, um die Datenintegrität zu gewährleisten.

Integritätsbedingung über Primärschlüssel

Jede Tabelle kann einen und nur einen Primärschlüssel besitzen. Ein Primärschlüssel ist eine Spalte oder eine Kombination von Spalten, die die gleichen Merkmale wie eine eindeutige Integritätsbedingung besitzen. Ein Primärschlüssel und Integritätsbedingungen über Fremdschlüssel dienen zur Definition von Beziehungen zwischen Tabellen.

Da ein Primärschlüssel zur Angabe einer Zeile in einer Tabelle verwendet wird, sollte er eindeutig sein und möglichst wenigen Hinzufüge- oder Löschoperationen unterliegen. Eine Tabelle kann nicht mehr als einen Primärschlüssel, jedoch mehrere eindeutige Schlüssel besitzen. Primärschlüssel sind wahlfrei und können definiert werden, wenn eine Tabelle erstellt oder geändert wird. Sie besitzen zudem den weiteren Vorteil, dass sie für eine Reihenfolge der Daten sorgen, wenn Daten exportiert oder reorganisiert werden.

In den folgenden Tabellen sind DEPTNO und EMPNO die Primärschlüssel der Tabellen DEPARTMENT und EMPLOYEE.

Tabelle 1. Tabelle DEPARTMENT

DEPTNO (Primärschlüssel)	**DEPTNAME**	**MGRNO**
A00	Spiffy Computer Service Division	000010
B01	Planning	000020
C01	Information Center	000030
D11	Manufacturing Systems	000060

Tabelle 2. Tabelle EMPLOYEE

EMPNO (Primärschlüssel)	**FIRSTNAME**	**LASTNAME**	**WORKDEPT (Fremdschlüssel)**	**PHONENO**
000010	Christine	Haas	A00	3978
000030	Sally	Kwan	C01	4738
000060	Irving	Stern	D11	6423
000120	Sean	O'Connell	A00	2167
000140	Heather	Nicholls	C01	1793
000170	Masatoshi	Yoshimura	D11	2890

Integritätsbedingung über Fremdschlüssel

Integritätsbedingungen über Fremdschlüssel (die auch als referenzielle Integritätsbedingungen bezeichnet werden) geben Ihnen die Möglichkeit, erforderliche Beziehungen zwischen und innerhalb von Tabellen zu definieren.

Zum Beispiel könnte eine typische Integritätsbedingung über Fremdschlüssel festlegen, dass jeder Mitarbeiter in der Tabelle EMPLOYEE ein Mitglied einer bestehenden, in der Tabelle DEPARTMENT definierten Abteilung sein muss.

Zur Herstellung dieser Beziehung würden Sie die Abteilungsnummer (WORKDEPT) der Tabelle EMPLOYEE als Fremdschlüssel und die Abteilungsnummer (DEPTNO) der Tabelle DEPARTMENT als Primärschlüssel definieren.

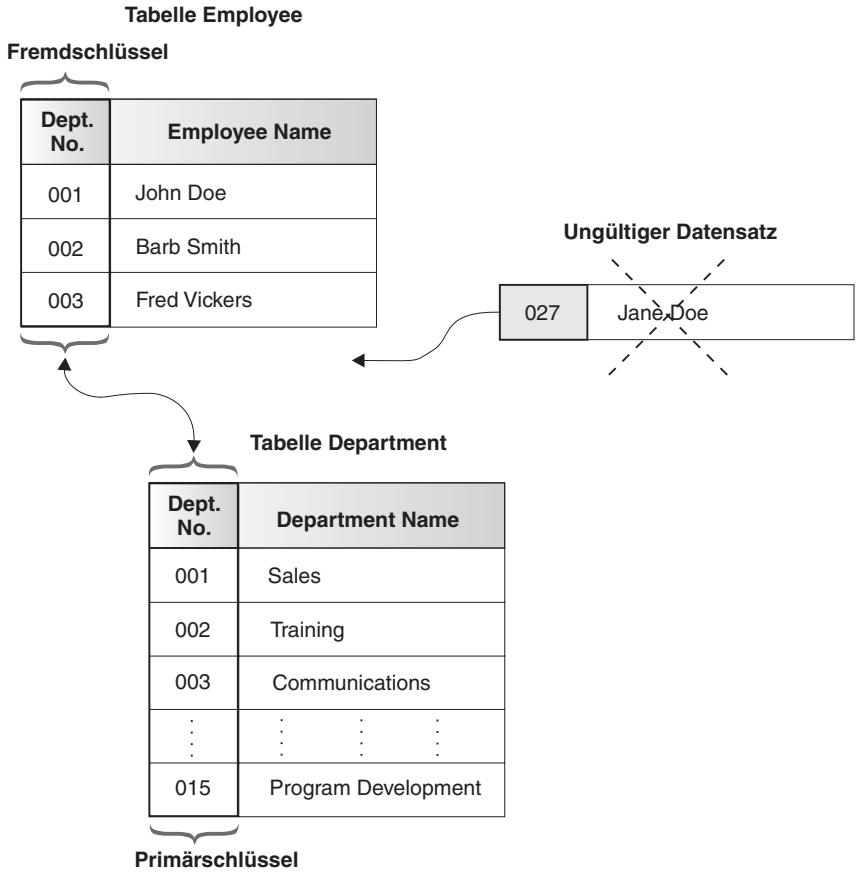

Abbildung 11. Integritätsbedingungen über Fremd- und Primärschlüssel definieren Beziehungen und schützen Daten.

Prüfung auf Integritätsbedingung

Eine Prüfung auf Integritätsbedingung ist eine Datenbankregel, mit der die zulässigen Werte in einer oder mehreren Spalten jeder Zeile einer Tabelle angegeben werden.

Zum Beispiel können Sie einer Tabelle EMPLOYEE definieren, dass die Spalte für die Jobbezeichnung nur die Werte "Sales", "Manager" oder "Clerk" enthalten kann. Unter dieser Integritätsbedingung ist jeder Datensatz mit einem anderen Wert in der Spalte für die Jobbezeichnung ungültig und würde zurückgewiesen, um die Regeln über den Typ der zulässigen Daten in der Tabelle durchzusetzen.

In einer Datenbank können außerdem *Auslöser* genutzt werden. Auslöser sind komplizierter und potenziell leistungsfähiger als Integritätsbedingungen. Sie definieren eine Reihe von Aktionen, die in Verbindung mit einer bzw. ausgelöst durch eine Klausel INSERT, UPDATE oder DELETE an einer angegebenen

Basistabelle ausgeführt wird. Auslöser können zur Unterstützung allgemeiner Formen der Datenintegrität und Geschäftsregeln verwendet werden. Zum Beispiel kann ein Auslöser die Kreditgrenze eines Kunden überprüfen, bevor ein Auftrag akzeptiert wird, oder er kann in einer Bankanwendung eingesetzt werden, um einen Alert auszulösen, wenn eine Geldentnahme aus einem Konto nicht dem gewöhnlichen Geldentnahmemuster eines Kunden entspricht. Weitere Informationen zu Auslösern (Triggers) finden Sie im Handbuch *Application Development Guide*.

Wiederherstellen einer Datenbank

Eine Datenbank kann aufgrund von Hardware- oder Softwarefehlern (oder beiden) unbrauchbar werden. Die verschiedenen Fehlerszenarios können verschiedene Maßnahmen zur Wiederherstellung erforderlich machen. Sie sollten eine erprobte Strategie zur Hand haben, um Ihre Datenbank gegen die Möglichkeit fehlerbedingter Ausfälle zu schützen.

Umfassende Informationen zur Sicherung, Wiederherstellung sowie zur Entwicklung einer guten Sicherungs- und Wiederherstellungsstrategie finden Sie in *Datenwiederherstellung und hohe Verfügbarkeit Handbuch und Referenz*.

Das Konzept einer Datenbanksicherung ist mit jeder anderen Datensicherung vergleichbar: Erstellen einer Kopie der Daten und Speichern der Kopie auf einem anderen Datenträger für den Fall eines Ausfalls oder einer Beschädigung der Originaldaten. Den einfachsten Fall einer Sicherung bildet das Verfahren, bei dem zunächst die Datenbank gestoppt wird, um sicherzustellen, dass keine weiteren Transaktionen auftreten, und anschließend die Daten gesichert (Backup) werden.

Wenn Ihre Datenbank beschädigt oder defekt wird, können Sie sie vom Sicherungsimage neu erstellen. Diese Neuerstellung der Datenbank wird als Wiederherstellung bezeichnet.

Wenn eine Datenbank während der Verarbeitung von Transaktionen abstürzt, wird ein Prozess eingeleitet, der als Wiederherstellung nach einem Systemabsturz bezeichnet wird, wenn die Datenbank erneut gestartet wird. Die *Wiederherstellung nach Systemabsturz* ist der Prozess, durch den die Datenbank in einen konsistenten und brauchbaren Status zurückgesetzt wird. Dies geschieht durch Rückgängigmachen der unvollständigen Transaktionen und Beenden der festgeschriebenen Aktionen, die sich noch im Hauptspeicher befanden, als der Systemabsturz auftrat.

Wenn die Datenbank beschädigt oder defekt ist, stehen zwei Wiederherstellungsmethoden zur Verfügung: Versionswiederherstellung oder aktualisierende Wiederherstellung.

Eine *Versionswiederherstellung* ist die Restaurierung einer früheren Version der Datenbank mit Hilfe eines Abbilds (Image) der Datenbank, das im Rahmen einer Sicherungsoperation erstellt wurde. Eine Datenbanksicherung ermöglicht Ihnen, eine Datenbank in dem Status wiederherzustellen, in dem sie sich zum Zeitpunkt der Sicherung befand. Alle Arbeitseinheiten, die vom Zeitpunkt der Sicherung ausgehend bis zum Eintreten des Fehlers ausgeführt wurden, sind jedoch verloren.

Wenn Sie in der Lage sein wollen, eine Datenbank über den Punkt hinaus zu restaurieren, an dem die Sicherungskopie erstellt wurde, müssen Sie mit der *aktualisierenden Wiederherstellung* arbeiten. Voraussetzung für die Methode der aktualisierenden Wiederherstellung ist, dass Sie eine Sicherungkopie der Datenbank erstellen und die Protokolldateien archivieren (indem Sie den Konfigurationsparameter *logretain* und/oder *userexit* der Datenbank aktivieren).

Jede Datenbank generiert Wiederherstellungsprotokolle, die zur Datenwiederherstellung nach Anwendungs- oder Systemfehlern verwendet werden. In Kombination mit den Datenbanksicherungen dienen sie zur Wiederherstellung der Datenbank bis exakt zu dem Zeitpunkt, zu dem der Fehler auftrat. Protokolldateien werden automatisch erstellt, wenn eine Datenbank erstellt wird. Protokolldateien können nicht direkt geändert werden.

Ein weiteres wichtiges Wiederherstellungsobjekt ist die Datei des Wiederherstellungsprotokolls. Die Datei des Wiederherstellungsprotokolls enthält eine Zusammenfassung der Sicherungsinformationen, die verwendet werden können, wenn die Datenbank insgesamt oder teilweise bis zu einem bestimmten Zeitpunkt wiederhergestellt werden muss. Sie dient zur Protokollierung wiederherstellungsrelevanter Ereignisse wie Sicherungs-, Wiederherstellungs- und Datenladeoperationen.

Anmerkung: Alle Informationen zur Sicherung und Wiederherstellung sowie die entsprechenden Informationen aus dem Handbuch Command Reference und dem Handbuch API Reference wurden unter dem Titel *Datenwiederherstellung und hohe Verfügbarkeit Handbuch und Referenz* zusammengefasst. Das Dokument *Datenwiederherstellung und hohe Verfügbarkeit Handbuch und Referenz* ist Ihre primäre Einzelquelle für Informationen zur Sicherung und Wiederherstellung.

Reorganisieren von Tabellen in einer Datenbank

Eine Tabelle kann infolge zahlreicher Aktualisierungen eine wachsende Fragmentierung aufweisen, wodurch die Leistung verringert wird. Wenn Sie Statistikdaten gesammelt haben und keine sichtbare Leistungsverbesserung festgestellt haben, kann eine Reorganisation der Tabellendaten hilfreich sein. Bei der Reorganisation von Tabellendaten werden die Daten in eine physische Reihenfolge gemäß einem angegebenen Index gebracht und der freie Speicherplatz, der fragmentierten Daten eigen ist, entfernt. Dies kann zu einem schnelleren Zugriff auf die Daten und somit zu einer Verbesserung der Leistung führen.

Vor der Reorganisation von Tabellen wird empfohlen, den Befehl REORGCHK auszuführen und Statistikdaten über die Tabelle zu sammeln. Die Ausführung dieses Befehls hilft bei der Bestimmung, ob eine Reorganisation der Tabellendaten angebracht ist. Informationen zum Befehl REORGCHK finden Sie im Handbuch *Command Reference*.

Übersicht über die Sicherheit unter DB2

Zum Schutz von Daten und Ressourcen auf einem Datenbank-Server verwendet DB2 eine Kombination aus externen Sicherheitsservices und internen Zugriffssteuerinformationen. Wenn Sie auf den Datenbank-Server zugreifen wollen, müssen Sie einige Sicherheitsüberprüfungen durchlaufen, bevor Ihnen Zugriff auf Datenbankdaten oder Ressourcen gewährt wird. Der erste Schritt in der Datenbanksicherheit wird als *Authentifizierung* (Identitätsüberprüfung) bezeichnet, bei der ein Benutzer nachweisen muss, dass er der ist, der er behauptet zu sein. Der zweite Schritt wird als *Berechtigung* (Authorization) bezeichnet, bei der der Datenbankmanager entscheidet, ob ein überprüfter Benutzer die angeforderte Aktion ausführen oder auf die angeforderten Daten zugreifen darf.

Authentifizierung

Die Authentifizierung (Identifikationsüberprüfung) wird mit Hilfe einer Sicherheitseinrichtung außerhalb von DB2 vollzogen. Die Sicherheitseinrichtung kann Teil des Betriebssystems, ein separates Produkt oder, in bestimmten Fällen, auch nicht vorhanden sein. Auf UNIX-Systemen ist die Sicherheitseinrichtung in das Betriebssystem selbst integriert. DCE-Sicherheitsservices (DCE Security Services) ist ein separates Produkt, das die Sicherheitseinrichtung in einer verteilten Umgebung zur Verfügung stellt. Unter den Betriebssystemen Windows 95 oder Windows 3.1 gibt es keine Sicherheitseinrichtungen.

Die Sicherheitseinrichtung benötigt zwei Elemente zur Authentifizierung eines Benutzers: eine Benutzer-ID und ein Kennwort. Die Benutzer-ID identifiziert den Benutzer der Sicherheitseinrichtung gegenüber. Durch die Eingabe des richtigen Kennworts (eine Information, die nur dem Benutzer und der Sicherheitseinrichtung bekannt ist) wird die Identität des Benutzers (entsprechend der Benutzer-ID) bestätigt.

Nach erfolgreicher Authentifizierung:

- Der Benutzer muss für DB2 mit Hilfe eines SQL-Berechtigungsnamens oder einer Berechtigungs-ID (*authid*) identifiziert werden. Der Name kann mit der Benutzer-ID übereinstimmen oder ein zugeordneter Wert sein. Beispielsweise wird auf UNIX-basierten Systemen eine DB2-Berechtigungs-ID (authid) abgeleitet, indem eine UNIX-Benutzer-ID, die den DB2-Namenskonventionen entspricht, in Großbuchstaben umgesetzt wird. Im Produkt der DCE-Sicherheitsservices ist die DB2-Berechtigungs-ID (authid) in der DCE-Registrierdatenbank enthalten und wird daraus extrahiert, wenn die Authentifizierung erfolgreich beendet wurde.
- Eine Liste von Gruppen, zu denen der Benutzer gehört, wird abgerufen. Die Gruppenzugehörigkeit kann bei der Berechtigung des Benutzers verwendet werden. Gruppen sind Definitionseinheiten von Sicherheitseinrichtungen, denen auch DB2-Berechtigungsnamen zugeordnet sein müssen. Diese Zuordnung erfolgt mit einer ähnlichen Methode wie für Benutzer-IDs.

  DB2 ruft eine Liste von Gruppen bis zu einer maximalen Anzahl von 64 Gruppen ab. Wenn ein Benutzer Mitglied in mehr als 64 Gruppen ist, werden nur die ersten 64 Gruppen, denen gültige DB2-Berechtigungsnamen zugeordnet sind, der DB2-Gruppenliste hinzugefügt. Es wird kein Fehler gemeldet. Alle weiteren Gruppen über die ersten 64 hinaus werden von DB2 ignoriert.

DB2 verwendet die Sicherheitseinrichtung zur Authentifizierung von Benutzern auf eine von zwei Arten:

- DB2 verwendet eine erfolgreiche Anmeldung durch das Sicherheitssystem als Nachweis der Identität und lässt Folgendes zu:
  - Verwendung lokaler Befehle zum Zugriff auf lokale Daten
  - Verwendung von Fernverbindungen, bei denen der Server die Authentifizierung auf dem Client akzeptiert

- DB2 akzeptiert eine Kombination aus Benutzer-ID und Kennwort. DB2 verwendet eine erfolgreiche Gültigkeitsprüfung dieses Paares durch die Sicherheitseinrichtung als Identitätsnachweis und erlaubt folgendes:
  - Verwendung von Fernverbindungen, bei denen der Server einen Beweis der Authentifizierung verlangt
  - Verwendung von Operationen, bei denen der Benutzer einen Befehl unter einer anderen als der zur Anmeldung verwendeten Identität ausführen will

DB2-Administratoren können andere Personen auf EEE-Systemen (Enterprise - Extended Edition) unter AIX und Windows NT durch die Variable DB2CHGPWD_EEE der Profilregistrierdatenbank dazu berechtigen, Kennwörter zu ändern. Der Standardwert für diese Variable nicht definiert (d. h. `NOT SET` - inaktiviert.) DB2CHGPWD_EEE akzeptiert die Booleschen Standardwerte, die von anderen DB2-Profilvariablen verwendet werden.

Der DB2-Administrator ist für die zentrale Verwaltung der Kennwörter aller Knoten verantwortlich. Dazu kann er unter Windows NT einen Windows NT-Domänen-Controller oder unter AIX den NIS verwenden.

Anmerkung: Wenn die Kennwörter nicht zentral verwaltet werden, kann die Aktivierung der Variable DB2CHGPWD_EEE möglicherweise dazu führen, dass die Kennwörter nicht über alle Knoten hinweg konsistent sind. Wenn Sie also die Funktion für die Kennwortänderung verwenden, wird Ihr Benutzerkennwort nur auf dem Knoten geändert, mit dem Sie verbunden sind.

Mit DB2 UDB unter AIX können fehlgeschlagene Kennworteingaben beim Betriebssystem protokolliert werden. Dadurch lässt sich ermitteln, wann ein Client die Anzahl zulässiger Anmeldeversuche überschritten hat, die im Parameter LOGINRETRIES angegeben ist.

Zusätzliche Informationen zu Gültigkeitsprüfungen bei Systemanmeldungen, die besonders wichtig sind, wenn Sie ferne Clients haben, die auf die Datenbank zugreifen, finden Sie in "Auswählen einer Authentifizierungsart für den Server" im Handbuch *Systemverwaltung: Implementierung*.

Berechtigung

Die Berechtigung (Authorization) ist der Prozess, durch den DB2 Informationen über einen überprüften DB2-Benutzer abruft, die angeben, welche Datenbankoperationen der Benutzer durchführen und auf welche Datenobjekte er zugreifen darf. Bei jeder Benutzeranforderung kann es mehr als eine Berechtigungsprüfung geben, je nachdem, welche Objekte und Operationen beteiligt sind.

Die Berechtigung wird mit Hilfe von DB2-Einrichtungen durchgeführt. DB2-Tabellen und DB2-Konfigurationsdateien werden zur Speicherung der Berechtigungen verwendet, die jedem Berechtigungsnamen zugeordnet sind. Der Berechtigungsname eines authentifizierten Benutzers und die Namen der Gruppen, zu denen der Benutzer gehört, werden mit den gespeicherten Berechtigungen verglichen. Anhand dieses Vergleichs entscheidet DB2, ob der angeforderte Zugriff gewährt wird.

Es gibt zwei Arten der Berechtigungen, die von DB2 gespeichert werden: Zugriffsrechte und Berechtigungsstufen. Ein *Zugriffsrecht* definiert eine einzelne Berechtigung für einen Berechtigungsnamen, Datenbankressourcen zu erstellen oder auf sie zuzugreifen. Zugriffsrechte werden in den Datenbankkatalogen gespeichert. *Berechtigungsstufen* bilden eine Methode, Zugriffsrechte und Kontrolle über Operationen zur Wartung des Datenbankmanagers und über Dienstprogrammoperationen auf einer höheren Stufe in Gruppen zusammenzufassen. Datenbankspezifische Berechtigungen werden in den Datenbankkatalogen gespeichert, während Systemberechtigungen durch Gruppenzugehörigkeit zugeordnet und in der Konfigurationsdatei des Datenbankmanagers für das jeweilige Exemplar gespeichert werden.

Gruppen stellen eine zweckmäßige Methode dar, für einen Benutzerverbund Berechtigungen durchzuführen, ohne jedem Benutzer einzeln Zugriffsrechte erteilen bzw. entziehen zu müssen. Sofern nicht anders angegeben, können Gruppenberechtigungsnamen überall dort verwendet werden, wo Berechtigungsnamen zu Berechtigungszwecken verwendet werden. Im allgemeinen wird die Gruppenzugehörigkeit für dynamisches SQL und Berechtigungen für Nichtdatenbankobjekte (z. B. Befehle auf Exemplarebene und Dienstprogramme) und nicht für statisches SQL berücksichtigt. Ein Ausnahmefall zu diesem allgemeinen Fall ist das Erteilen von Zugriffsrechten für PUBLIC, die bei der Verarbeitung von statischem SQL berücksichtigt werden. Spezielle Fälle, in denen Gruppenzugehörigkeit nicht gültig ist, werden in der DB2-Dokumentation entsprechend vermerkt.

Weitere Informationen finden Sie in "Zugriffsrechte und Berechtigungen" im Handbuch *Systemverwaltung: Implementierung*.

Übersicht über Authentifizierung und Berechtigung in einer zusammengeschlossenen Datenbank

Da ein DB2-System zusammengeschlossener Datenbanken auf Daten in mehreren Datenbankverwaltungssystemen zugreifen kann, sind möglicherweise zusätzliche Maßnahmen zur Absicherung der Daten erforderlich.

Bei der Planung der Lösung zur Authentifizierung müssen Sie berücksichtigen, dass die Benutzer sich möglicherweise bei den Datenquellen und bei DB2 authentifizieren müssen. In einem System zusammengeschlossener Datenbanken kann die Authentifizierung auf DB2-Client-Workstations, auf DB2-Servern, an den Datenquellen (DB2, DB2 für OS/390, anderen DRDA-Servern, Oracle) oder einer Kombination von DB2 (Client oder DB2-Server) und Datenquellen stattfinden. Selbst in DCE-Umgebungen, können spezielle Maßnahmen erforderlich sein, wenn für die Anmeldung an den Datenquellen eine Benutzer-ID und ein Kennwort abgefragt wird. Weitere Informationen finden Sie in "Authentifizierungsverarbeitung in zusammengeschlossenen Datenbanken" im Handbuch *Systemverwaltung: Implementierung*.

Entsprechend müssen die Benutzer eine Berechtigungsprüfung an Datenquellen und für DB2 durchlaufen. Die jeweilige Datenquelle (DB2, Oracle, DB2 für OS/390 usw.) verwaltet die Sicherheit der Objekte, die von ihr gesteuert werden. Wenn ein Benutzer eine Operation mit einem Kurznamen ausführt, muss er für die Tabelle oder Sicht, auf die sich der Kurzname bezieht, eine Berechtigungsprüfung durchlaufen.

Kapitel 3. Systeme zusammengeschlossener Datenbanken

Ein *System zusammengeschlossener Datenbanken* ist ein Datenbankverwaltungssystem (DBMS), das Anwendungen und Benutzer unterstützt, die mit SQL-Anweisungen arbeiten, in denen auf zwei oder mehr Datenbankverwaltungssysteme oder Datenbanken in einer einzelnen Anweisung verwiesen wird. Ein Beispiel hierfür ist eine Verknüpfung zwischen Tabellen in zwei unterschiedlichen DB2-Datenbanken. Diese Anweisungsart wird als eine *verteilte Anforderung* bezeichnet.

Ein System zusammengeschlossener Datenbanken unter DB2 Universal Database unterstützt verteilte Anforderungen über Datenbanken und Datenbankverwaltungssysteme hinweg. Sie können z. B. eine UNION-Operation zwischen einer DB2-Tabelle und einer Oracle-Sicht ausführen. Zu den unterstützten Datenbankverwaltungssystemen gehören DB2, Produkte der DB2-Produktfamilie (wie z. B. DB2 für OS/390 und DB2 für IBM AS/400) und Oracle.

In DB2-Systemen zusammengeschlossener Datenbanken besteht für Datenbankobjekte *Positionstransparenz*. Wenn Informationen (in Tabellen und Sichten) versetzt werden, können Verweise auf diese Informationen (so genannte *Kurznamen*) ohne Änderung an den Anwendungen, die die Informationen anfordern, aktualisiert werden. DB2-Systeme zusammengeschlossener Datenbanken bieten eine *Kompensation* für Datenbankverwaltungssysteme, die nicht das gesamte SQL von DB2 oder bestimmte Optimierungsfunktionen unterstützen. Operationen, die auf einem solchen Datenbankverwaltungssystem nicht ausgeführt werden können (z. B. rekursives SQL), werden unter DB2 ausgeführt.

Ein DB2-System zusammengeschlossener Datenbanken arbeitet *halbautonom*: DB2-Abfragen, die Verweise auf Oracle-Objekte enthalten, können übergeben werden, während Oracle-Anwendungen auf denselben Server zugreifen. Ein DB2-System zusammengeschlossener Datenbanken hat weder den alleinigen Zugriff auf Oracle oder andere DBMS-Objekte noch beschränkt es ihn (abgesehen von den Einschränkungen wegen der Integrität und durch Sperren).

Ein DB2-System zusammengeschlossener Datenbanken besteht aus einem Exemplar unter DB2 UDB, einer Datenbank, die als *zusammengeschlossene Datenbank* dient, und einer oder mehreren *Datenquellen*. Die zusammengeschlossene Datenbank enthält Katalogeinträge, die die Datenquellen und ihre Kenndaten angeben.

Eine Datenquelle besteht aus einem DBMS und Daten. Anwendungen stellen eine Verbindung zur zusammengeschlossenen Datenbank her, wie zu jeder anderen DB2-Datenbank. In Abb. 12 ist eine Umgebung zusammengeschlossener Datenbanken dargestellt.

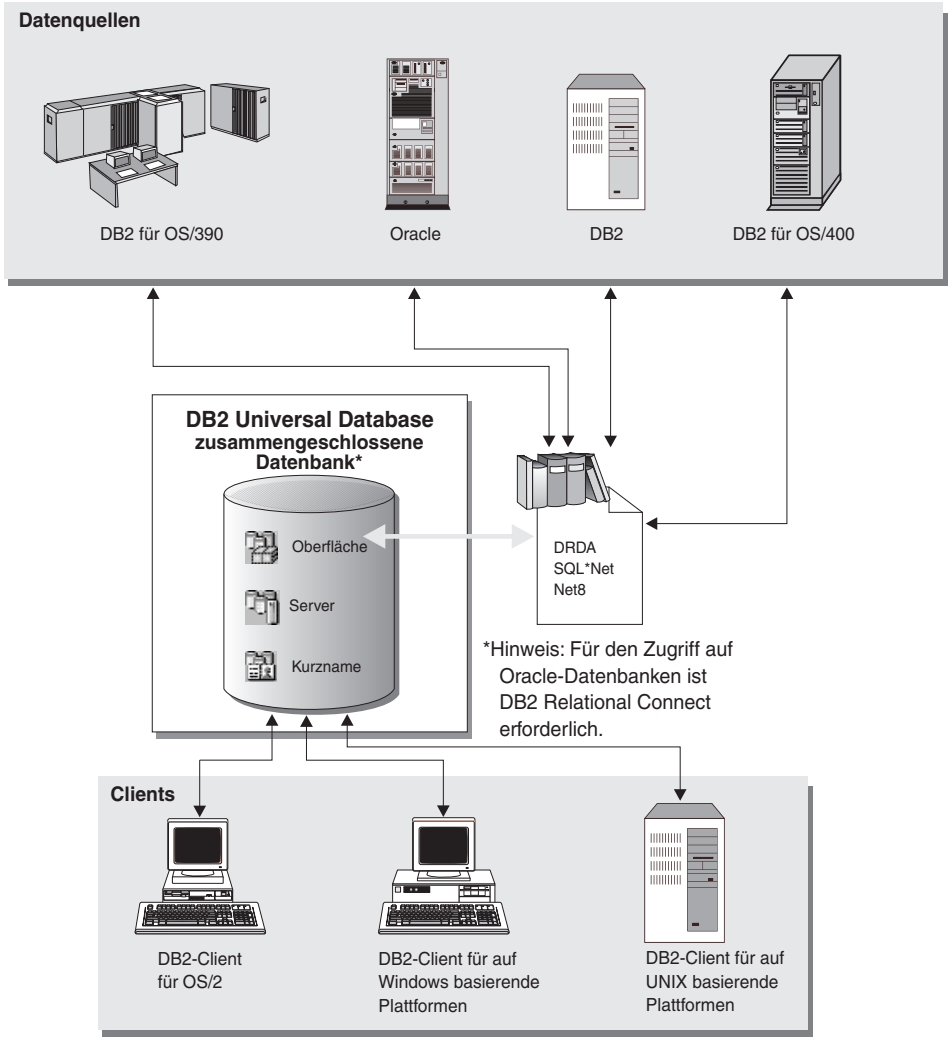

Abbildung 12. Ein System zusammengeschlossener Datenbanken

Die Katalogeinträge der zusammengeschlossenen DB2-Datenbank enthalten Informationen zu Datenquellenobjekten: den Namen dieser Objekte, die in diesen Objekten enthaltenen Informationen und die Bedingungen für die Verwendung der Objekte. Da dieser DB2-Katalog Informationen zu Objekten in vielen Datenbankverwaltungssystemen speichert, wird er auch *globaler Katalog* genannt. Objektattribute werden in dem Katalog gespeichert. Die tatsächlichen Datenbankverwaltungssysteme, auf die verwiesen wird, Module, die für die Kommunikation mit der Datenquelle verwendet werden, und DBMS-Datenobjekte (wie z. B. Tabellen), auf die zugegriffen wird, befinden sich außerhalb der Datenbank. (Eine Ausnahme: Eine zusammengeschlossene Datenbank kann eine Datenquelle für das System zusammengeschlossener Datenbanken sein.) Sie können zusammengeschlossene Objekte mit Hilfe der Steuerzentrale oder mit Hilfe von DDL-Anweisungen in SQL erstellen. Folgende Objekte sind für zusammengeschlossene Datenbanken erforderlich:

Oberflächen
Geben die Module (DLL, library usw.) an, die für den Zugriff auf eine bestimmte Klasse oder Kategorie von Datenquelle verwendet werden.

Server Definieren Datenquellen. Zu den Server-Daten gehören der Name der Oberfläche, der Server-Name, die Server-Art, die Server-Version, Berechtigungsinformationen und die Server-Optionen.

Kurznamen
Kennungen, die in der zusammengeschlossenen Datenbank gespeichert sind und auf bestimmte Datenquellenobjekte (wie z. B. Tabellen, Aliasnamen, Sichten) verweisen. Anwendungen verweisen in Abfragen auf Kurznamen genauso, wie sie auf Tabellen und Sichten verweisen.

Sie können je nach Bedarf zusätzliche Objekte erstellen:

- Benutzerzuordnungen, um Authentifizierungsfunktionen zu implementieren.
- Datentypzuordnungen, um die Abhängigkeit zwischen einem Datenquellentyp und einem DB2-Typ individuell zu definieren.
- Funktionszuordnungen, um eine lokale Funktion einer Datenquellenfunktion zuzuordnen.
- Indexspezifikationen, um die Leistung zu erhöhen.

Wenn ein System zusammengeschlossener Datenbanken eingerichtet ist, kann auf die Informationen an Datenquellen zugegriffen werden, so als ob diese sich in einer einzigen großen Datenbank befänden. Benutzer und Anwendungen senden Abfragen an eine zusammengeschlossene Datenbank, die dann Daten von Systemen der DB2-Produktfamilie und Oracle-Systemen nach Bedarf abruft. Benutzer und Anwendungen geben in Abfragen Kurznamen an. Diese Kurznamen stellen Verweise auf Tabellen und Sichten bereit, die sich in Datenquellen befinden. Aus der Perspektive eines Endbenutzers sind Kurznamen mit Aliasnamen vergleichbar.

Es gibt viele Faktoren, die sich auf die Leistung eines Systems zusammengeschlossener Datenbanken auswirken. Der wichtigste Faktor ist, sicherzustellen, dass korrekte und aktuelle Informationen zu Datenquellen und ihren Objekten im globalen Katalog der zusammengeschlossenen Datenbank gespeichert werden. Diese Informationen werden vom DB2-Optimierungsprogramm verwendet und können entscheidend dafür sein, dass Auswertungsoperationen auf die ferne Quelle der zu bearbeitenden Daten ausgelagert und dort ausgeführt werden. Im Handbuch *Systemverwaltung: Optimierung* finden Sie zusätzliche Informationen zur Leistung von Systemen zusammengeschlossener Datenbanken.

Der Betrieb eines DB2-Systems zusammengeschlossener Datenbanken unterliegt einigen Einschränkungen. Verteilte Anforderungen sind auf Lesezugriffsoperationen beschränkt. Außerdem können Sie die Dienstprogrammoperationen (LOAD, REORG, REORGCHK, IMPORT, RUNSTATS usw.) nicht für Kurznamen ausführen.

Sie können allerdings eine Durchgriffsfunktion verwenden, um DDL- und DML-Anweisungen direkt an Datenbankmanager zu übergeben, die die SQL-Programmversion verwenden, die dieser Datenquelle zugeordnet ist.

Systeme zusammengeschlossener Datenbanken tolerieren parallele Umgebungen. Leistungsvorteile sind in dem Maße begrenzt, in dem eine Abfrage einer zusammengeschlossenen Datenbank semantisch in Verweise auf lokale Objekte (Tabellen, Sichten) und Verweise auf Kurznamen aufgesplittet werden kann. Anforderungen für Kurznamendaten werden sequenziell verarbeitet. Lokale Objekte können parallel verarbeitet werden. Zum Beispiel würde in einer Abfrage `SELECT * FROM A, B, C, D`, in der A und B lokale Tabellen, C und D aber Kurznamen sind, die auf Tabellen in Oracle-Datenquellen verweisen, ein möglicher Plan die Tabellen A und B mit einer parallelen Verknüpfung verknüpfen. Die Ergebnisse würden anschließend sequenziell mit den Kurznamen C und D verknüpft.

Einrichten eines Systems zusammengeschlossener Datenbanken

DB2 Enterprise Edition (EE) und DB2 Enterprise - Extended Edition (EEE) können zusammengeschlossene Datenbanken unterstützen. Gehen Sie wie folgt vor, um ein System zusammengeschlossener Datenbanken einzurichten:

1. Wählen Sie die Installationsoption *Distributed Join für DB2-Datenquellen* von DB2 EE oder EEE während der Installation aus.
2. Wenn Oracle-Datenbanken in das System zusammengeschlossener Datenbanken aufgenommen werden, installieren Sie DB2 Relational Connect. Weitere Informationen finden Sie im Handbuch *Installation und Konfiguration Ergänzung*.
3. Setzen Sie den Konfigurationsparameter *federated* des Datenbankmanagers auf "YES".
4. Erstellen Sie Oberflächen, Server und Kurznamen (weitere Informationen finden Sie in "Erstellen einer Datenbank" im Handbuch *Systemverwaltung: Implementierung*).
5. Erstellen Sie zusätzliche Objekte, oder legen Sie Optionen nach Bedarf fest (weitere Informationen finden Sie in "Implementieren des Datenbankentwurfs" im Handbuch *Systemverwaltung: Implementierung*).

Kapitel 4. Parallele Datenbanksysteme

DB2 erweitert den Datenbankmanager auf die parallele Mehrknotenumgebung. Eine *Datenbankpartition* ist ein Teil einer Datenbank, der aus seinen eigenen Daten, Indizes, Konfigurationsdateien und Transaktionsprotokollen besteht. Eine Datenbankpartition wird manchmal auch als Knoten oder Datenbankknoten bezeichnet. (Knoten (Node) war der Begriff, der im Produkt DB2 Parallel Edition für AIX Version 1 verwendet wurde.)

Eine *Einzelpartitionsdatenbank* ist eine Datenbank, die nur über eine Datenbankpartition verfügt. Sämtliche Daten in der Datenbank werden in dieser Partition gespeichert. In diesem Fall bieten Knotengruppen (siehe „Knotengruppen" auf Seite 11), obwohl sie vorhanden sind, keine zusätzlichen Leistungsvorteile.

Eine *partitionierte Datenbank* ist eine Datenbank mit zwei oder mehr Datenbankpartitionen. Die Tabellen können in einer oder mehreren Datenbankpartitionen gespeichert werden. Wenn eine Tabelle in einer Knotengruppe mit mehreren Partitionen gespeichert ist, heißt dies, dass einige ihrer Zeilen in einer Partition gespeichert werden, während sich andere Zeilen in anderen Partitionen befinden.

In der Regel existiert eine einzelne Datenbankpartition auf jedem physischen Knoten, und die Prozessoren auf jedem System werden vom Datenbankmanager in jeder Datenbankpartition zur Verwaltung des jeweiligen Teils der Gesamtdaten in der Datenbank verwendet.

Da die Daten auf Partitionen aufgeteilt sind, können Sie das Potenzial mehrerer Prozessoren auf mehreren physischen Knoten zur Erfüllung von Informationsanforderungen nutzen. Anforderungen zur Abfrage und Aktualisierung von Daten werden automatisch in Unteranforderungen zerlegt und in den betroffenen Datenbankpartitionen parallel ausgeführt. Die Tatsache, dass Datenbanken auf Datenbankpartitionen verteilt sind, ist für Benutzer, die SQL-Anweisungen ausführen, transparent.

Die Benutzerinteraktion erfolgt über nur eine Datenbankpartition, die als *Koordinatorknoten* für den jeweiligen Benutzer bezeichnet wird. Der Koordinator ist in derselben Datenbankpartition aktiv wie die Anwendung bzw. im Fall einer fernen Anwendung in der Datenbankpartition, mit der die Anwendung verbunden ist. Eine beliebige Datenbankpartition kann als Koordinatorknoten verwendet werden.

Knotengruppen und Datenpartitionierung

Sie können benannte Untergruppen einer oder mehrerer Datenbankpartitionen in einer Datenbank definieren. Jede Untergruppe, die Sie definieren, wird als *Knotengruppe* bezeichnet. Jede Untergruppe, die mehr als eine Datenbankpartition enthält, wird als *Mehrpartitionsknotengruppe* bezeichnet. Mehrpartitionsknotengruppen können nur mit Datenbankpartitionen definiert werden, die zum selben Exemplar gehören.

Abb. 13 auf Seite 41 zeigt ein Beispiel einer Datenbank mit fünf Partitionen mit folgenden Knotengruppen:

- Knotengruppe 1 umfasst alle Datenbankpartitionen außer einer.
- Knotengruppe 2 enthält eine Datenbankpartition.
- Eine Knotengruppe enthält zwei Datenbankpartitionen. (Knotengruppe 3).
- Die Datenbankpartition innerhalb Knotengruppe 2 wird gemeinsam mit Knotengruppe 1 (und überlappend) benutzt.
- Es gibt eine einzelne Datenbankpartition innerhalb Knotengruppe 3, die gemeinsam mit Knotengruppe 1 (und überlappend) benutzt wird.

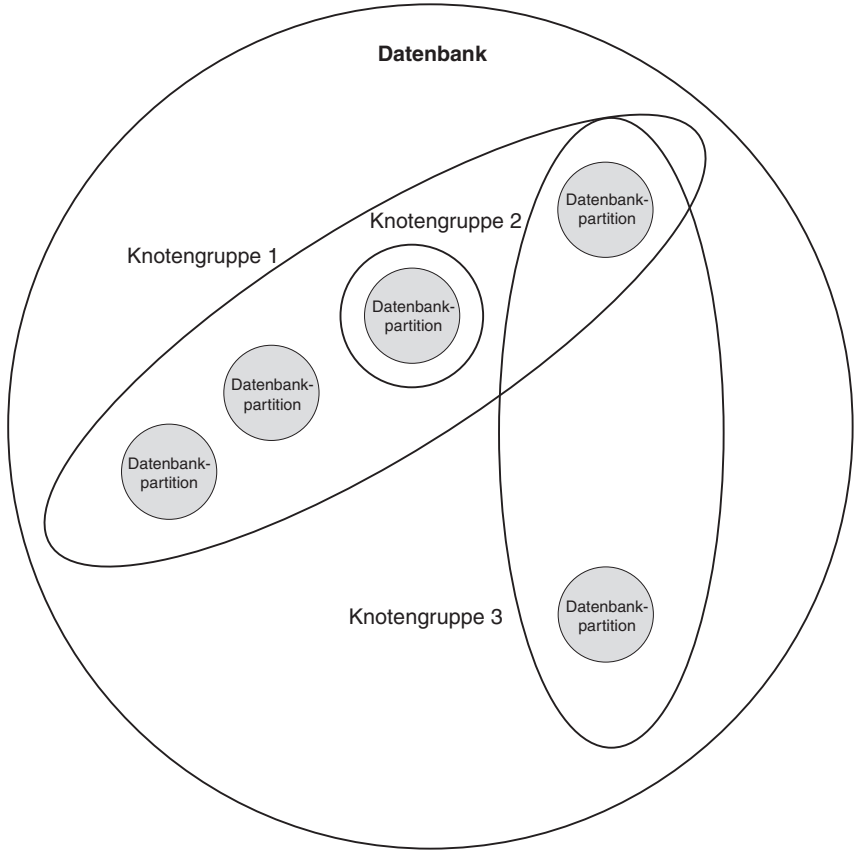

Abbildung 13. Knotengruppen in einer Datenbank

Eine neue Knotengruppe wird mit Hilfe der Anweisung CREATE NODEGROUP erstellt. Weitere Informationen finden Sie im Handbuch *SQL Reference*. Die Daten werden über alle Partitionen in einer Knotengruppe verteilt. Wenn Sie Knotengruppen mit mehreren Partitionen verwenden, müssen Sie eine Reihe von Gesichtspunkten im Hinblick auf den Aufbau der Knotengruppen beachten. Weitere Informationen finden Sie in „Entwerfen von Knotengruppen" auf Seite 116.

Arten der Parallelität

Komponenten einer Task, wie zum Beispiel einer Datenbankabfrage, können parallel ausgeführt werden, um die Leistung erheblich zu erhöhen. Die Art der Task, die Datenbankkonfiguration und die Hardwareumgebung bestimmen, wie DB2 eine Task parallel ausführt. Diese Aspekte stehen in Wechselbeziehung zueinander und sollten bei der Arbeit am physischen und logischen Entwurf einer Datenbank im Zusammenhang betrachtet werden. In diesem Abschnitt werden die folgenden Arten von Parallelität beschrieben, die von DB2 unterstützt werden:

- Ein-/Ausgabeparallelität
- Abfrageparallelität
- Dienstprogrammparallelität

Ein-/Ausgabeparallelität

Wenn mehrere Behälter für einen Tabellenbereich vorhanden sind, kann der Datenbankmanager die *parallele Ein-/Ausgabe* nutzen. Parallele E/A bezeichnet den Vorgang, bei dem Lese- bzw. Schreiboperationen mit zwei oder mehr Ein-/Ausgabeeinheiten gleichzeitig durchgeführt werden. Auf diese Weise können Verbesserungen des Durchsatzes erzielt werden.

Die E/A-Parallelität ist eine Komponente der jeweiligen Hardwareumgebungen, die im Abschnitt „Hardwareumgebungen" auf Seite 46 beschrieben werden. In Tabelle 3 auf Seite 55 sind die Hardwareumgebungen aufgeführt, die am besten für parallele Ein-/Ausgabe geeignet sind.

Abfrageparallelität

Es gibt zwei Arten der Abfrageparallelität: abfrageübergreifende und abfrageinterne Parallelität.

Abfrageübergreifende Parallelität bezieht sich auf die Möglichkeit, dass mehrere Anwendungen gleichzeitig eine Datenbank abfragen können. Jede Abfrage wird unabhängig von allen anderen Abfragen ausgeführt, jedoch führt DB2 sie alle gleichzeitig aus. DB2 unterstützt diese Art der Parallelität seit jeher.

Abfrageinterne Parallelität bezeichnet die gleichzeitige Verarbeitung von Teilen einer einzelnen Abfrage mit Hilfe der *partitionsinternen Parallelität* bzw. der *partitionsübergreifenden Parallelität* (oder beiden).

Der Begriff *Abfrageparallelität* wird im gesamten Handbuch verwendet.

Partitionsinterne Parallelität

Der Begriff *partitionsinterne Parallelität* bezeichnet die Fähigkeit, eine Abfrage in mehrere Teile zu untergliedern. (Einige Dienstprogramme arbeiten ebenfalls mit dieser Art der Parallelität. Siehe „Dienstprogrammparallelität" auf Seite 45.)

Bei der partitionsinternen Parallelität wird das, was im Allgemeinen als eine einzige Datenbankoperation betrachtet wird (z. B. eine Indexerstellung, das Laden von Daten, SQL-Abfragen) in mehrere Teile unterteilt, von denen viele oder alle parallel *innerhalb einer einzigen Datenbankpartition* ausgeführt werden können.

Abb. 14 zeigt eine Abfrage, die in vier Teile aufgeteilt ist, die parallel ausgeführt werden können, wobei die Ergebnisse schneller geliefert werden als bei serieller Ausführung der Abfrage. Die Teile sind jeweils Kopien voneinander. Zur Nutzung der partitionsinternen Parallelität muss die Datenbank entsprechend konfiguriert werden. Sie können den Grad der Parallelität selbst auswählen oder ihn vom System festlegen lassen. Als Grad der Parallelität stellt die Anzahl der Teile einer Abfrage dar, die parallel ausgeführt werden.

In Tabelle 3 auf Seite 55 sind die Hardwareumgebungen aufgeführt, die am besten für partitionsinterne Parallelität geeignet sind.

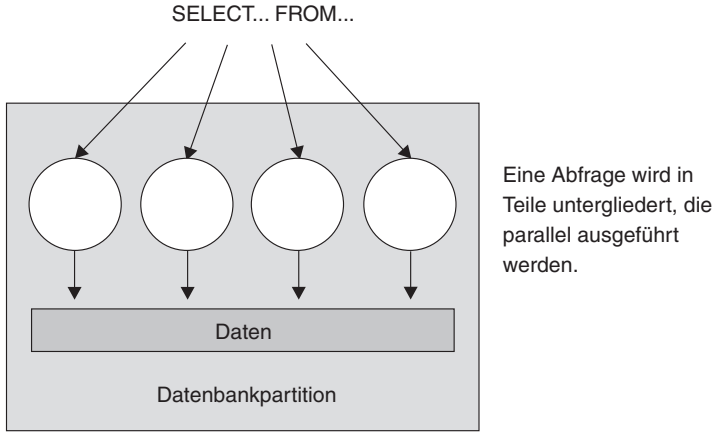

Abbildung 14. Partitionsinterne Parallelität

Partitionsübergreifende Parallelität

Der Begriff *partitionsübergreifende Parallelität* bezeichnet die Fähigkeit, eine Abfrage in mehreren Teilen über mehrere Partitionen einer partitionierten Datenbank auf einer oder mehreren Maschinen zu verteilen. Die Abfrage wird parallel ausgeführt. (Einige Dienstprogramme arbeiten ebenfalls mit dieser Art der Parallelität. Siehe „Dienstprogrammparallelität" auf Seite 45.)

Bei der partitionsübergreifenden Parallelität wird das, was im Allgemeinen als eine einzige Datenbankoperation betrachtet wird, (z. B. eine Indexerstellung, das Laden von Daten, SQL-Abfragen) in mehrere Teile unterteilt, von denen viele oder alle parallel *über mehrere Partitionen einer partitionierten Datenbank hinweg auf einer oder mehreren Maschinen* ausgeführt werden können.

Abb. 15 zeigt eine Abfrage, die in vier Teile aufgeteilt ist, die parallel ausgeführt werden können, wobei die Ergebnisse schneller zurückgeliefert werden als bei serieller Ausführung in einer Einzelpartition.

Der Grad der Parallelität wird im Wesentlichen durch die Anzahl der Partitionen, die Sie erstellen, und die Art und Weise der Definition der Knotengruppen bestimmt.

In Tabelle 3 auf Seite 55 sind die Hardwareumgebungen aufgeführt, die am besten für partitionsübergreifende Parallelität geeignet sind.

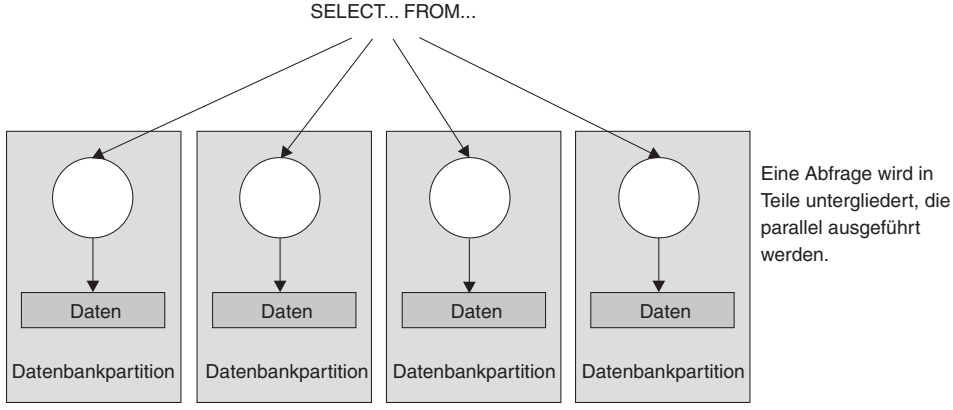

Abbildung 15. Partitionsübergreifende Parallelität

Gleichzeitiges Verwenden der partitionsinternen und partitionsübergreifenden Parallelität

Sie können die partitionsinterne Parallelität und die partitionsübergreifende Parallelität gleichzeitig nutzen. Diese Kombination bietet zwei Dimensionen der Parallelität, wodurch sich ein weiterer wesentlicher Anstieg der Verarbeitungsgeschwindigkeit für Abfragen ergibt:

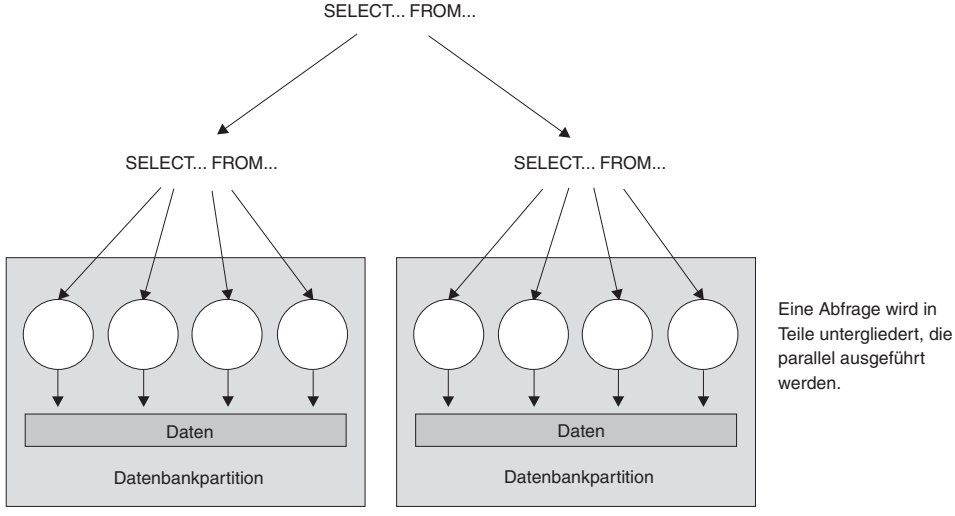

Abbildung 16. Gleichzeitige partitionsübergreifende und partitionsinterne Parallelität

Dienstprogrammparallelität

DB2-Dienstprogramme können mit partitionsinterner Parallelität arbeiten. Sie können außerdem die Vorteile der partitionsübergreifenden Parallelität nutzen. Wenn mehrere Datenbankpartitionen vorhanden sind, werden die Dienstprogramme in jeder der Partitionen parallel ausgeführt.

Das Dienstprogramm LOAD kann die Vorteile der partitionsinternen Parallelität und der E/A-Parallelität nutzen. Das Laden von Daten in eine Tabelle ist eine CDPU-intensive Operation. Das Dienstprogramm LOAD nutzt die Möglichkeit mehrerer Prozessoren bei Operationen wie dem Analysieren und Formatieren von Daten. Darüber hinaus kann es parallele E/A-Server zum gleichzeitigen Schreiben der Daten in Behälter verwenden. Im Handbuch *Versetzen von Daten Dienstprogramme und Referenz* finden Sie Informationen darüber, wie Sie die Parallelität für das Dienstprogramm LOAD aktivieren können.

In einer Umgebung mit partitionierten Datenbanken nutzt das Dienstprogramm AutoLoader die Vorteile der partitionsinternen, partitionsübergreifenden und der E/A-Parallelität durch paralleles Aufrufen des Befehls LOAD in jeder Datenbankpartition, in der sich die Tabelle befindet. Weitere Informationen zum Dienstprogramm AutoLoader finden Sie im Handbuch *Versetzen von Daten Dienstprogramme und Referenz*.

Während der Indexerstellung erfolgt das Durchsuchen und das nachfolgende Sortieren der Daten parallel. DB2 nutzt sowohl die E/A-Parallelität als auch die partitionsinterne Parallelität bei der Erstellung eines Index. Dadurch wird die Indexerstellung bei der Verarbeitung der Anweisung CREATE INDEX, beim Neustart (wenn ein Index als ungültig markiert ist) und bei der Reorganisation von Daten beschleunigt.

Sichern und Wiederherstellen von Daten sind Operationen, die wesentlich von der E/A-Leistung abhängig sind. DB2 nutzt sowohl die E/A-Parallelität als auch partitionsinterne Parallelität bei der Durchführung von BACKUP- und RESTORE-Operationen. Der Befehl BACKUP nutzt die E/A-Parallelität, indem er von mehreren Tabellenbereichsbehältern parallel liest und asynchron auf mehrere Sicherungsmedien parallel schreibt. In den Abschnitten zu den Befehlen BACKUP DATABASE und RESTORE DATABASE im Handbuch *Command Reference* finden Sie Informationen, wie Sie die Parallelität für diese beiden Dienstprogramme aktivieren können.

Hardwareumgebungen

Dieser Abschnitt enthält eine Übersicht über folgende Hardwareumgebungen:

- Einzelpartition auf Einzelprozessormaschine
- Einzelpartition auf Mehrprozessormaschine (SMP)
- Konfigurationen mit mehreren Partitionen
  - Partitionen mit einem Prozessor (MPP)
  - Partitionen mit mehreren Prozessoren (Gruppe von SMP-Systemen)
  - Logische Datenbankpartitionen (in DB2 Parallel Edition für AIX Version 1 auch als MLN (Multiple Logical Nodes) bezeichnet)

Für jede Umgebung werden die Kapazität und Skalierbarkeit behandelt. *Kapazität* bezieht sich hier auf die Anzahl der Benutzer und Anwendungen, die auf die Datenbank zugreifen können. Dies wird großenteils durch Hauptspeicher, Agenten, Sperren, E/A-Operationen und Speicherverwaltung bestimmt. *Skalierbarkeit* bezeichnet die Fähigkeit einer Datenbank, bei zunehmender Größe weiterhin die gleichen Betriebsmerkmale und Antwortzeiten zu zeigen.

Einzelpartition auf einer Einzelprozessormaschine

Diese Umgebung verfügt über Hauptspeicher und Platte, enthält aber nur eine CPU (siehe Abb. 17). Für diese Umgebung werden viele verschiedene Bezeichnungen verwendet, wie zum Beispiel eigenständige Datenbank, Client-/Server-Datenbank, serielle Datenbank, Einprozessorsystem oder nicht parallele bzw. Einzelknotenumgebung.

Die Datenbank in dieser Umgebung erfüllt die Anforderungen einer Abteilung oder eines kleinen Büros, wobei die Daten und Systemressourcen (einschließlich des Einzelprozessors bzw. einer CPU) von einem einzelnen Datenbankmanager verwaltet werden.

In Tabelle 3 auf Seite 55 sind die Arten der Parallelität aufgeführt, die am besten in dieser Hardwareumgebung genutzt werden können.

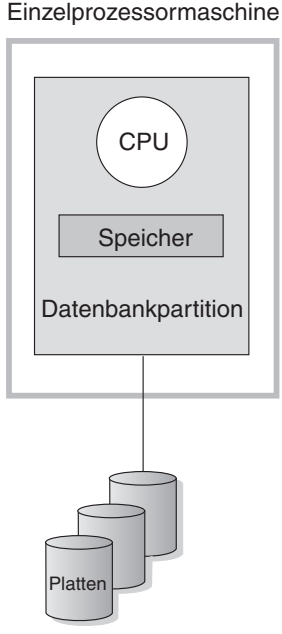

Abbildung 17. Einzelpartition auf einer Einzelprozessormaschine

Kapazität und Skalierbarkeit
In dieser Umgebung können Sie weitere Platten hinzufügen. Durch den Einsatz eines oder mehrerer E/A-Server für jede Platte kann mehr als eine E/A-Operation gleichzeitig stattfinden.

Ein Einzelprozessorsystem ist durch die Menge an Plattenspeicherplatz, den der Prozessor handhaben kann, eingeschränkt. Ungeachtet aller zusätzlichen Komponenten wie Arbeitsspeicher oder Festplattenspeicher, die Sie zur schnelleren Verarbeitung von Benutzeranforderungen vielleicht hinzufügen, kann eine einzelne CPU bei weiterer Zunahme der Auslastung Benutzeranforderungen nicht mehr schneller verarbeiten. Wenn Sie das Maximum an Kapazität und Skalierbarkeit erreicht haben, können Sie die Umrüstung auf ein Einzelpartitionssystem mit mehreren Prozessoren in Erwägung ziehen.

artition auf Mehrprozessormaschine

Diese Umgebung besteht gewöhnlich aus mehreren gleich starken Prozessoren innerhalb derselben Maschine (siehe Abb. 18 auf Seite 49) und wird als *symmetrisches Mehrprozessorsystem (SMP-System)* bezeichnet (SMP - Symmetric Multi-Processor). Ressourcen wie Plattenspeicherplatz und Hauptspeicher werden *gemeinsam* benutzt.

Wenn mehrere Prozessoren verfügbar sind, können verschiedene Datenbankoperationen schneller durchgeführt werden. DB2 kann auch die Arbeit einer einzigen Abfrage auf die verfügbaren Prozessoren verteilen, um die Verarbeitungsgeschwindigkeit zu erhöhen. Andere Datenbankoperationen wie das Laden von Daten, das Sichern und Wiederherstellen von Tabellenbereichen sowie die Erstellung von Indizes auf vorhandenen Daten können die Verfügbarkeit mehrerer Prozessoren ausnutzen.

In Tabelle 3 auf Seite 55 sind die Arten der Parallelität aufgeführt, die am besten in dieser Hardwareumgebung genutzt werden können.

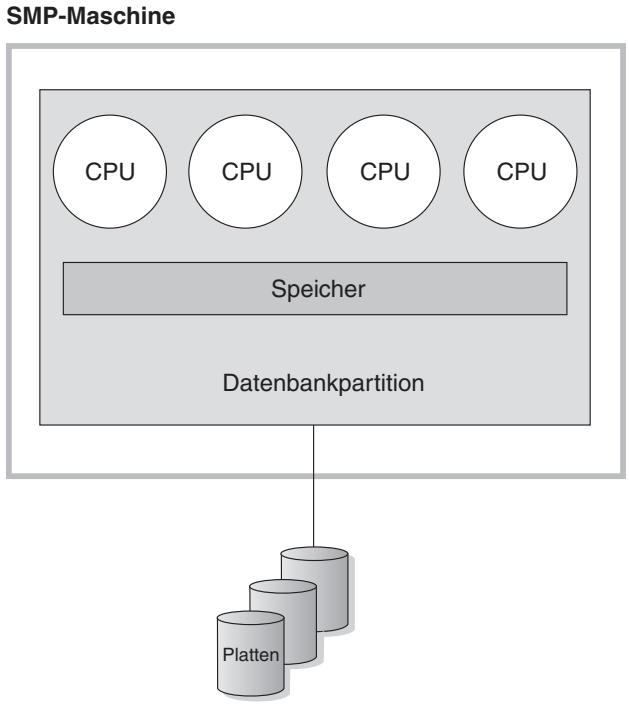

Abbildung 18. Einzelpartitionsdatenbank auf einem symmetrischen Mehrprozessorsystem (SMP)

Kapazität und Skalierbarkeit

In dieser Umgebung können Sie weitere Prozessoren hinzufügen. Da aber verschiedene Prozessoren versuchen könnten, gleichzeitig auf dieselben Daten zuzugreifen, können mit der Zunahme der Geschäftsoperationen Einschränkungen auftreten. Durch die gemeinsame Benutzung von Hauptspeicher und Platten werden effektiv sämtliche Datenbankdaten gemeinsam benutzt.

Sie können die E/A-Kapazität der Datenbankpartition, die Ihrem Prozessor zugeordnet ist, durch eine größere Anzahl von Platten erhöhen. Sie können E/A-Server speziell zur Verarbeitung von E/A-Anforderungen einrichten. Durch den Einsatz eines oder mehrerer E/A-Server für jede Platte kann mehr als eine E/A-Operation gleichzeitig stattfinden.

Wenn Sie das Maximum an Kapazität und Skalierbarkeit erreicht haben, können Sie die Umrüstung auf ein System mit mehreren Partitionen in Erwägung ziehen.

Konfigurationen mit mehreren Partitionen

Sie können eine Datenbank in mehrere Partitionen aufteilen, die sich jeweils auf einer eigenen Maschine befinden. Mehrere Maschinen mit mehreren Datenbankpartitionen können in einer Gruppe zusammengefasst werden. In diesem Abschnitt werden die folgenden Partitionskonfigurationen beschrieben:

- Partitionen auf Systemen mit einem Prozessor
- Partitionen auf Systemen mit mehreren Prozessoren
- Logische Datenbankpartitionen

Partitionen mit einem Prozessor

In dieser Umgebung gibt es viele Datenbankpartitionen. Jede Partition befindet sich auf einer eigenen Maschine mit eigenem Prozessor, eigenem Hauptspeicher und eigenen Platten (Abb. 19 auf Seite 51). Alle Maschinen sind über eine Kommunikationseinrichtung verbunden. Für eine solche Umgebung werden viele verschiedene Bezeichnungen gebraucht, wie zum Beispiel Cluster, Cluster von Einzelprozessoren, Umgebung mit exklusiver Parallelverarbeitung (MPP - Massively Parallel Processing) oder Konfiguration ohne gemeinsame Nutzung von Ressourcen. Die letztgenannte Bezeichnung charakterisiert die Anordnung der Ressourcen. Im Gegensatz zu einer SMP-Umgebung findet in einer MPP-Umgebung keine gemeinsame Nutzung von Hauptspeicher oder Platten statt. In der MPP-Umgebung fallen daher auch die Einschränkungen aufgrund der gemeinsamen Nutzung von Hauptspeicher und Platten fort.

Durch die Partitionierung in einer Umgebung kann eine Datenbank trotz ihrer physischen Verteilung auf mehrere Partitionen eine logische Gesamtheit bilden. Die Tatsache, dass die Daten auf Partitionen verteilt sind, bleibt für die Mehrheit der Benutzer transparent. Die Arbeit kann unter den Datenbankmanagern aufgeteilt werden. Die Datenbankmanager in den einzelnen Partitionen verrichten die Arbeit jeweils am eigenen Teil der Datenbank.

In Tabelle 3 auf Seite 55 sind die Arten der Parallelität aufgeführt, die am besten in dieser Hardwareumgebung genutzt werden können.

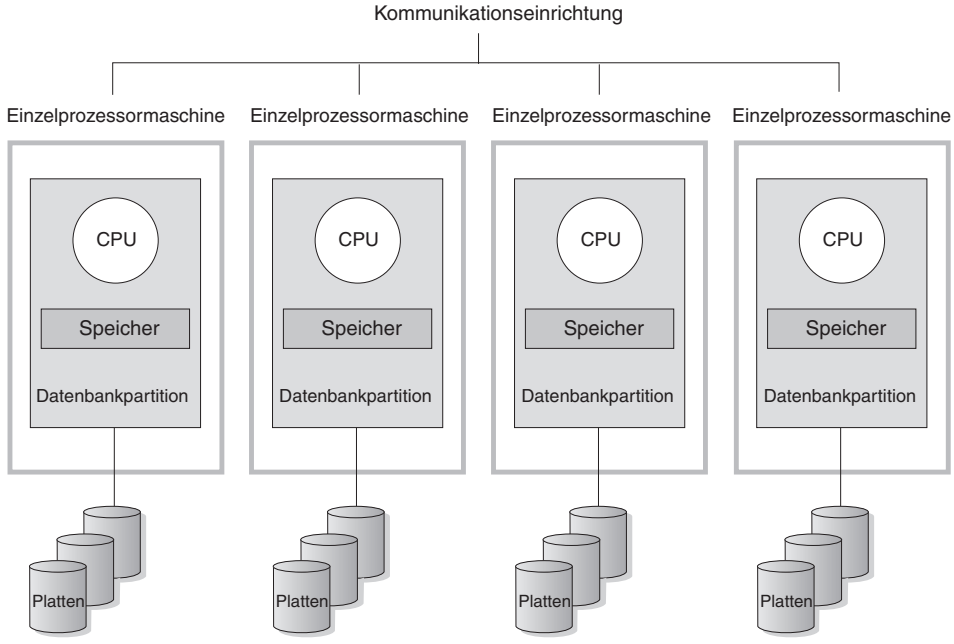

Abbildung 19. System zur exklusiven Parallelverarbeitung (MPP-System)

Kapazität und Skalierbarkeit: In dieser Umgebung können Sie weitere Datenbankpartitionen (Knoten) Ihrer Konfiguration hinzufügen. Auf einigen Plattformen, z. B. RS/6000 SP, ist die mögliche maximale Anzahl 512 Knoten. Jedoch kann es praktische Grenzen für die Verwaltung einer höheren Anzahl von Maschinen und Exemplaren geben.

Wenn Sie das Maximum an Kapazität und Systemgrößenflexibilität erreicht haben, können Sie die Umrüstung auf ein System in Erwägung ziehen, in dem jede Partition mehrere Prozessoren hat.

Partitionen mit mehreren Prozessoren

Eine Alternative zu einer Konfiguration, in der jede Partition einen einzigen Prozessor hat, ist eine Konfiguration, in der eine Partition über mehrere Prozessoren verfügt. Eine solche Konfiguration wird als *SMP-Cluster* bezeichnet (Abb. 20).

Diese Art der Konfiguration verbindet die Vorteile von SMP- und MPP-Parallelität. Dies bedeutet, dass eine Abfrage in einer Einzelpartition mit Hilfe mehrerer Prozessoren durchgeführt werden kann. Darüber hinaus kann eine Abfrage auch parallel in mehreren Partitionen durchgeführt werden.

In Tabelle 3 auf Seite 55 sind die Arten der Parallelität aufgeführt, die am besten in dieser Hardwareumgebung genutzt werden können.

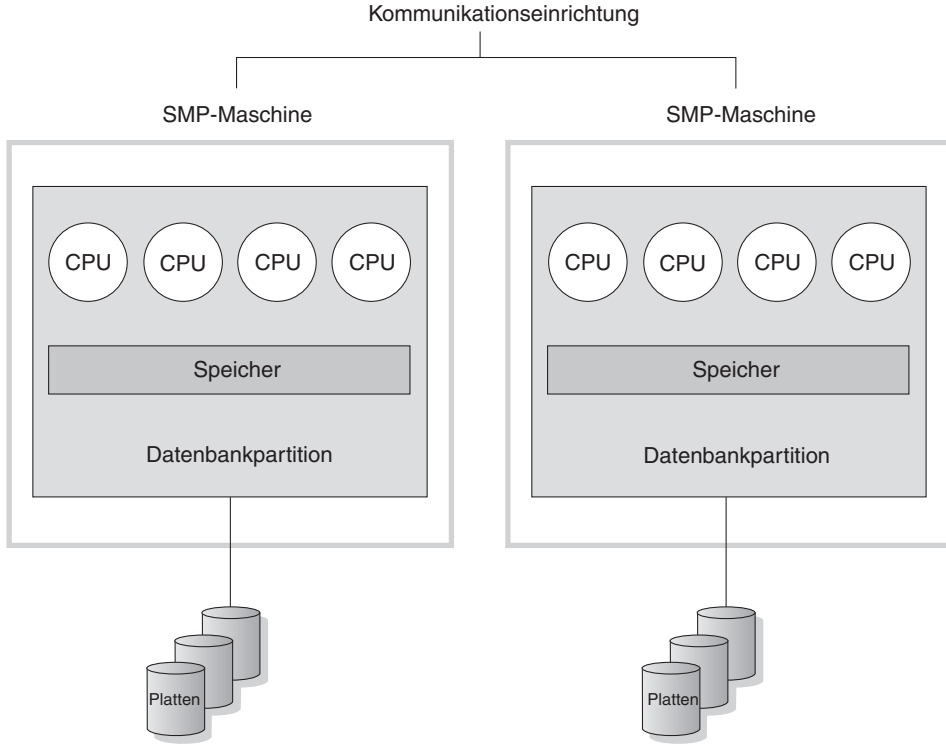

Abbildung 20. Gruppe von SMP-Maschinen

Kapazität und Skalierbarkeit: In dieser Umgebung können Sie weitere Datenbankpartitionen und den vorhandenen Datenbankpartitionen weitere Prozessoren hinzufügen.

Logische Datenbankpartitionen

Eine logische Datenbankpartition unterscheidet sich von einer physischen Partition darin, dass ihr nicht die Steuerung einer ganzen Maschine unterliegt. Obwohl die Maschine über gemeinsam benutzte Ressourcen verfügt, nutzen Datenbankpartitionen die Ressourcen nicht gemeinsam. Prozessoren werden gemeinsam benutzt, Platten und Hauptspeicher jedoch nicht.

Logische Datenbankpartitionen bieten Skalierbarkeit. Mehrere Datenbankmanager, die in mehreren logischen Partitionen ausgeführt werden, können die verfügbaren Ressourcen eventuell erschöpfender nutzen, als dies ein einzelner Datenbankmanager könnte. Abb. 21 veranschaulicht die Möglichkeit, mehr Skalierbarkeit auf einer SMP-Maschine zu gewinnen, indem weitere Partitionen hinzugefügt werden. Dies gilt insbesondere für Maschinen mit vielen Prozessoren. Durch die Partitionierung der Datenbank können Sie jede Partition getrennt verwalten und wiederherstellen.

Große SMP-Maschine

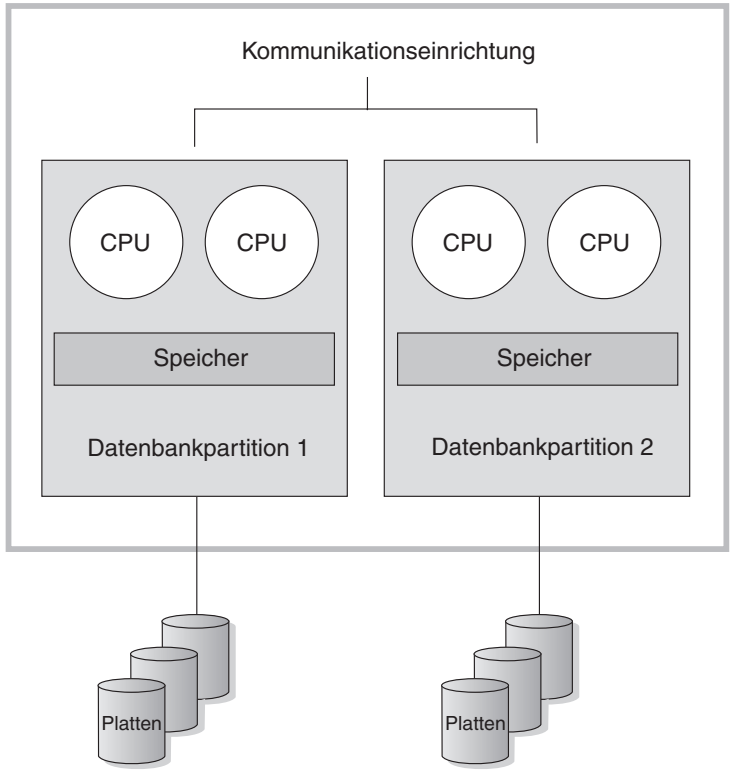

Abbildung 21. Partitionierte Datenbank auf einem symmetrischen Mehrprozessorsystem

Abb. 22 veranschaulicht die Möglichkeit, die in Abb. 21 auf Seite 53 gezeigte Konfiguration zu vervielfachen, um die Verarbeitungsleistung zu erhöhen.

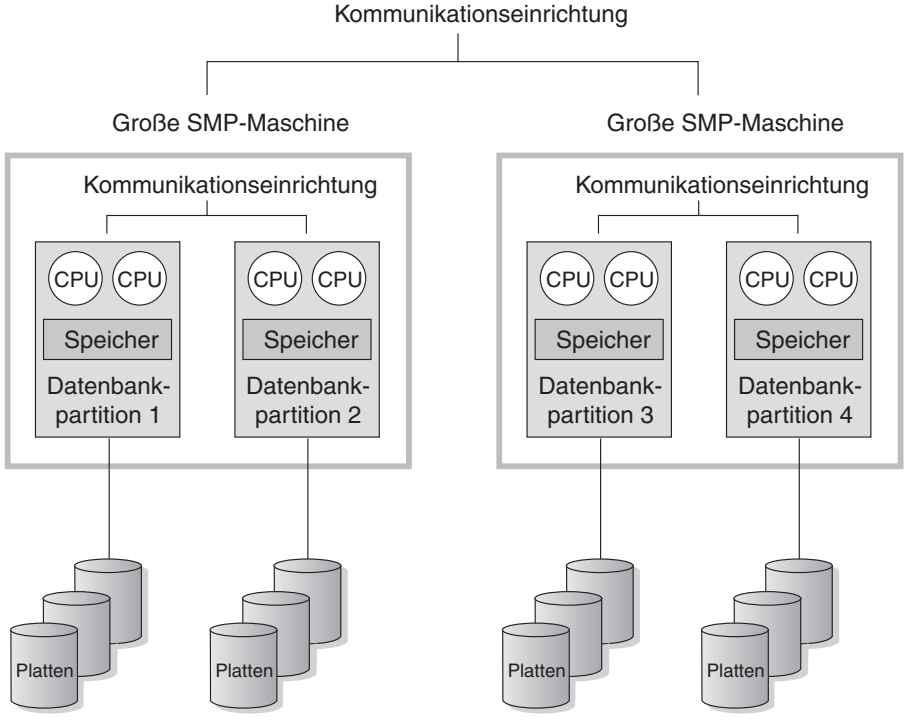

Abbildung 22. Partitionierte Datenbank auf in Gruppen zusammengefassten SMP-Systemen

In Tabelle 3 auf Seite 55 sind die Arten der Parallelität aufgeführt, die am besten in dieser Hardwareumgebung genutzt werden können.

Anmerkung: Die Möglichkeit, zwei oder mehr Partitionen nebeneinander auf derselben Maschine zu haben (ungeachtet der Anzahl der Prozessoren), bietet eine größere Flexibilität bei der Einrichtung hochverfügbarer Konfigurationen und der Implementierung von Übernahmestrategien bei Systemausfällen. Bei Ausfall einer Maschine kann eine Datenbankpartition automatisch von einer anderen Maschine, die bereits eine andere Partition derselben Datenbank enthält, übernommen und wieder gestartet werden. Weitere Informationen finden Sie in „Kapitel 11. Einführung in die Unterstützung der hohen Verfügbarkeit und der Funktionsübernahme” auf Seite 207.

Zusammenfassung der am besten geeigneten Arten von Parallelität für jede Hardwareumgebung

Die folgende Tabelle gibt eine Übersicht über die Arten der Parallelität, mit denen die verschiedenen Hardwareumgebungen am besten genutzt werden können.

Tabelle 3. Mögliche Arten der Parallelität in jeder Hardwareumgebung

Hardwareumgebung	**E/A-Parallelität**	**Abfrageinterne Parallelität**	
		Partitions-interne Parallelität	**Partitionsübergreifende Parallelität**
Einzelpartition, Einzelprozessor	Ja	Nein (1)	Nein
Einzelpartition, mehrere Prozessoren (SMP)	Ja	Ja	Nein
Mehrere Partitionen, ein Prozessor (MPP)	Ja	Nein (1)	Ja
Mehrere Partitionen, mehrere Prozessoren (Cluster von SMP-System)	Ja	Ja	Ja
Logische Datenbankpartitionen	Ja	Ja	Ja
Anmerkung: (1) Es kann vorteilhaft sein, den Grad der Parallelität (mit Hilfe eines der Konfigurationsparameter) auch auf einem Einzelprozessorsystem auf einen Wert größer als 1 zu setzen, besonders wenn die von Ihnen ausgeführten Abfragen die CPU nicht voll auslasten (z. B. wenn die Abfragen E/A-lastig sind).			

Kapitel 5. Data Warehouses

DB2 Universal Database stellt die Data Warehouse-Zentrale zur Verfügung, eine Komponente, die die Data Warehouse-Verarbeitung automatisiert. Mit Hilfe der Data Warehouse-Zentrale können Sie die Daten definieren, die in das Warehouse aufgenommen werden sollen. Anschließend können Sie über die Data Warehouse-Zentrale automatisch Aktualisierungen der Daten im Warehouse per Zeitplan terminieren.

Dieser Abschnitt enthält eine Übersicht über den Einsatz von Data Warehouses und Data Warehouse-Funktionen. Genauere Informationen zu Data Warehouses und Informationen zur Data Warehouse-Zentrale finden Sie im Handbuch *Data Warehouse-Zentrale Verwaltung* und in der Online-Hilfefunktion der Data Warehouse-Zentrale.

Was ist ein Data Warehouse?

Die Systeme mit *Betriebsdaten* (d. h. Daten, die für die täglichen Transaktionen des Geschäftsbetriebs benötigt werden) enthalten Informationen, die für Geschäftsanalysen nützlich sind. Zum Beispiel können Analytiker Informationen darüber nutzen, welche Produkte in welchen Gebieten zu welcher Jahreszeit verkauft wurden, um Abweichungen zu erkennen oder zukünftige Verkäufe zu projektieren.

Jedoch ergeben sich beim direkten Zugriff auf Betriebsdaten durch Analytiker verschiedene Probleme:

- Sie verfügen möglicherweise nicht über die Fachkenntnisse zum Abfragen der Betriebsdatenbank. Zum Beispiel ist zum Abfragen von IMS-Datenbanken ein Anwendungsprogramm erforderlich, das eine spezielle Art von Datenbearbeitungssprache verwendet. Im allgemeinen haben diejenigen Programmierer, die über die nötigen Fachkenntnisse zum Abfragen der Betriebsdatenbank verfügen, eine Vollzeitbeschäftigung mit der Verwaltung der Datenbank und der zugehörigen Anwendungen.
- Die Leistung ist für viele Betriebsdatenbanken, wie zum Beispiel Datenbanken einer Bank, von entscheidender Bedeutung. Das System kann Benutzer, die Sofortabfragen ausführen, nicht bedienen.
- Die Betriebsdaten liegen in der Regel nicht in dem zur Geschäftsanalyse günstigsten Format vor. Zum Beispiel sind Verkaufsdaten, die nach Produkt, Gebiet und Jahreszeit zusammengefasst sind, wesentlich nützlicher für Analytiker als die Rohdaten.

Data Warehouses schaffen diesen Problemen Abhilfe. In *Data Warehouses*werden Speicher *informativer Daten* eingerichtet, d. h. Daten, die aus den Betriebsdaten extrahiert und zur Entscheidungsfindung für Endbenutzer aufbereitet werden. Zum Beispiel könnte ein Data Warehouse-Tool alle Verkaufsdaten aus der Betriebsdatenbank kopieren, Berechnungen durchführen, um die Daten zusammenzufassen, und die zusammengefassten Daten in eine Zieltabelle einer von der Betriebsdatenbank getrennten Datenbank schreiben. Die getrennte Datenbank (das sog. *Warehouse*) kann dann von Endbenutzern abgefragt werden, ohne die Betriebsdatenbanken zu beeinträchtigen.

In den folgenden Abschnitten werden Objekte (Themenbereiche, Warehouse-Quellen, Warehouse-Ziele, Agenten, Agenten-Sites, Schritte und Prozesse) beschrieben, die zur Erstellung und Pflege eines Data Warehouse verwendet werden.

Themenbereiche

Ein *Themenbereich* identifiziert und gruppiert die Prozesse, die zu einem logischen Bereich des Geschäftsbetriebs gehören. Wenn Sie zum Beispiel ein Warehouse mit Marketing- und Verkaufsdaten aufbauen, können Sie einen Themenbereich Verkauf und einen Themenbereich Marketing definieren. Anschließend können Sie die Prozesse, die zum Verkauf gehören, unter dem Themenbereich Verkauf hinzufügen. Analog können Sie die Definitionen, die sich auf die Marketing-Daten beziehen, unter dem Themenbereich Marketing hinzufügen.

Warehouse-Quellen

Warehouse-Quellen geben die Tabellen und Dateien an, denen die Daten für das Warehouse entnommen werden. Die Data Warehouse-Zentrale verwendet die Angaben in den Warehouse-Quellen, um auf Daten zuzugreifen und sie auszuwählen. Die Quellen können beinahe jede relationale oder nicht relationale Quelle (Tabelle, Sicht oder Datei) sein, von der eine Verbindung zum Warehouse hergestellt werden kann.

Warehouse-Ziele

Warehouse-Ziele sind Datenbanktabellen oder Dateien, die Daten enthalten, die zur Verwendung durch Endbenutzer aufbereitet wurden. Ähnlich wie Warehouse-Quellen können auch Warehouse-Ziele Daten für Schritte der Data Warehouse-Zentrale bereitstellen.

Warehouse-Agenten und Agenten-Sites

Agenten der Data Warehouse-Zentrale verwalten den Datenfluss zwischen den Datenquellen und den Ziel-Warehouses. Agenten sind für die Betriebssysteme Windows NT, AIX, OS/2, OS/390, OS/400 und SUN Solaris verfügbar. Diese Agenten verwenden ODBC-Treiber (Open Database Connectivity) oder DB2 CLI zur Kommunikation mit verschiedenen Datenbanken.

Die Übertragung von Daten zwischen Quellen und Ziel-Warehouses kann von mehreren Agenten übernommen werden. Die Anzahl der Agenten, die Sie verwenden, hängt von der vorhandenen Konnektivitätskonfiguration und dem geplanten Volumen der Daten ab, die in das Warehouse fließen sollen. Zusätzliche Exemplare eines Agenten können generiert werden, wenn mehrere Prozesse, die den gleichen Agenten benötigen, gleichzeitig ausgeführt werden.

Agenten können lokal oder fern sein. Ein *lokaler Warehouse-Agent* ist ein Agent, der auf derselben Maschine wie der Warehouse-Server installiert ist. Ein *ferner Warehouse-Agent* ist ein Agent, der auf einer anderen Maschine installiert ist, die über eine Konnektivität zum Warehouse-Server verfügt.

Eine *Agenten-Site* ist ein logischer Name für eine Workstation, auf der Agentensoftware installiert ist. Der Name der Agenten-Site ist nicht mit dem TCP/IP-Host-Namen identisch. Eine einzelne physische Maschine kann nur einen TCP/IP-Host-Namen besitzen. Jedoch können auf einer einzelnen Maschine mehrere Agenten-Sites definiert werden. Jede Agenten-Site wird durch einen logischen Namen angegeben.

Die *Standardagenten-Site*, die den Namen Standard-VW-Agenten-Site besitzt, ist ein lokaler Agent auf Windows NT, der von der Data Warehouse-Zentrale während der Initialisierung der Warehouse-Steuerungsdatenbank definiert wird.

Schritte und Prozesse

Ein *Schritt* ist eine logische Definitionseinheit in der Data Warehouse-Zentrale, die Folgendes definiert:

- Die Struktur der Ausgabetabelle oder Ausgabedatei
- Den Mechanismus (entweder SQL oder ein Programm) zum Füllen der Ausgabetabelle oder Ausgabedatei
- Den Zeitplan, nach dem die Ausgabetabelle bzw. Ausgabedatei gefüllt wird

Schritte übertragen Daten und setzen sie um, indem sie SQL-Anweisungen ausführen oder Programme aufrufen. Wenn Sie einen Schritt ausführen, findet die Datenübertragung zwischen der Warehouse-Quelle und dem Warehouse-Ziel sowie die Umsetzung dieser Daten statt.

Ein *Prozess* enthält eine Reihe von Schritten, die Umsetzungs- und Datenübertragungstasks ausführen. In der Regel füllt ein Prozess ein Warehouse-Ziel in einer Warehouse-Datenbank, indem er Daten aus einer oder mehreren Warehouse-Quellen extrahiert, bei denen es sich um Datenbanktabellen oder Dateien handeln kann. Jedoch können Sie einen Prozess auch zum Starten von Programmen definieren, wobei weder Warehouse-Quellen noch Warehouse-Ziele angegeben werden.

Ein Schritt kann auf Anforderung ausgeführt oder zur Ausführung zu einem bestimmten Zeitpunkt terminiert werden. Ein Schritt kann zur einmaligen Ausführung oder zur wiederholten Ausführung, z. B. jeden Freitag, eingeplant werden. Schritte können auch so terminiert werden, dass sie nacheinander ausgeführt werden, d. h., wenn ein Schritt beendet wird, wird der nächste Schritt gestartet. Schritte können zudem abhängig von der (erfolgreichen oder nicht erfolgreichen) Beendigung eines anderen Schritts eingeplant werden. Wenn Sie einen Prozess über einen Zeitplan terminieren, wird der erste Schritt des Prozesses zur definierten Zeit ausgeführt.

Wenn ein Schritt oder ein Prozess ausgeführt wird, kann er Daten auf folgende Arten speichern:

- Ersetzen aller Daten im Warehouse-Ziel durch neue Daten
- Anhängen der neuen Daten an die vorhandenen Daten
- Anhängen einer getrennten Edition von Daten

Nehmen Sie zum Beispiel an, dass die Data Warehouse-Zentrale die folgenden Aufgaben ausführen soll:

1. Extrahieren von Daten aus verschiedenen Datenbanken
2. Konvertieren der Daten in ein einziges Format
3. Schreiben der Daten in eine Tabelle in einem Data Warehouse

In diesem Beispiel würden Sie einen Prozess erstellen, der einzelne Schritte enthält. Jeder Schritt würde eine getrennte Task, wie zum Beispiel das Extrahieren der Daten aus den Datenbanken oder die Konvertierung der Daten in das richtige Format, ausführen. Dann würden Sie einen weiteren Schritt verwenden, um die Zieltabelle zu füllen, die die umgesetzten Daten enthält.

In den folgenden Abschnitten werden die verschiedenen Arten von Schritten beschrieben, die in der Data Warehouse-Zentrale verfügbar sind. Weitere Informationen zu Schritten finden Sie im Handbuch *Data Warehouse-Zentrale Verwaltung*.

SQL-Schritte

Ein SQL-Schritt arbeitet mit einer SQL-Anweisung SELECT, um Daten aus einer Warehouse-Quelle zu extrahieren, und generiert eine Anweisung INSERT, um die Daten in die Warehouse-Zieltabelle einzufügen.

Programmschritte

Es gibt verschiedene Typen von Programmschritten: DB2 for AS/400-Programme, DB2 for OS/390-Programme, DB2 for UDB-Programme, Visual Warehouse 5.2 DB2-Programme, OLAP Server-Programme, Dateiprogramme und Replikation. Diese Schritte führen vordefinierte Programme und Dienstprogramme aus.

Umsetzungsschritte

Umsetzungsschritte sind gespeicherte Prozeduren und benutzerdefinierte Funktionen, die Umsetzungen für das Warehouse oder statistische Funktionen angeben, die zur Umsetzung von Daten verwendet werden können. Mit Umsetzungsprogrammen können Sie Daten bereinigen, umkehren und umlagern, Primärschlüssel und Periodentabellen generieren sowie verschiedene statistische Berechnungen durchführen.

In einem Umsetzungsschritt geben Sie eines der statistischen oder Warehouse-Umsetzungsprogramme an. Wenn der Prozess ausgeführt wird, schreibt der Umsetzungsschritt Daten in ein oder mehrere Warehouse-Ziele.

Benutzerdefinierte Programmschritte

Ein *benutzerdefinierter Programmschritt* ist eine logische Definitionseinheit innerhalb der Data Warehouse-Zentrale, die eine Anwendung darstellt, die von der Data Warehouse-Zentrale gestartet werden soll. Ein Warehouse-Agent kann einen benutzerdefinierten Programmschritt zu folgenden Zeiten starten:

- Während des Füllens eines Warehouse-Ziels
- Nach dem Füllen eines Warehouse-Ziels
- Unabhängig

Zum Beispiel können Sie ein benutzerdefiniertes Programm schreiben, das den folgenden Prozess ausführt:

1. Exportieren von Daten aus einer Tabelle
2. Bearbeiten dieser Daten
3. Schreiben der Daten in eine vorläufige Ausgaberessource oder ein Warehouse-Ziel

Aufgaben bei der Warehouse-Erstellung

Zur Erstellung eines Data Warehouse gehören folgende Aufgaben:

- Definieren eines Themenbereichs, der die Prozesse angibt und gruppiert, die in dem Warehouse verwendet werden sollen
- Erkunden der Quellendaten (bzw. Betriebsdaten) und Definieren von Warehouse-Quellen
- Erstellen einer Datenbank zur Verwendung als Warehouse und Definieren von Warehouse-Zielen
- Angeben, wie Quellendaten übertragen und in das richtige Format für die Warehouse-Datenbank umgesetzt werden, durch Definieren eines Prozesses
- Testen der definierten Schritte und Festlegen eines Zeitplans zur automatischen Ausführung
- Verwalten des Warehouse durch Definieren von Sicherheit und Überwachen der Datenbanknutzung

- Erstellen eines Informationskatalogs der Daten im Warehouse, falls das Paket DB2 Warehouse Manager eingesetzt wird. Ein Informationskatalog ist eine Datenbank, die Metadaten über den Geschäftsbetrieb enthält. Die Metadaten unterstützen Benutzer bei der Angabe und Auffindung von Daten und Informationen, die ihnen in der Organisation zur Verfügung stehen. Endbenutzer des Warehouse können den Katalog durchsuchen, um festzustellen, welche Tabellen abzufragen sind.
- Definieren eines Sternschemenmodells für die Daten im Warehouse. Ein Sternschema ist ein spezieller Aufbau, der aus mehreren Dimensionstabellen (die Aspekte eines Unternehmens beschreiben) und einer Fakttabelle (die Fakten über das Unternehmen enthält) besteht. Wenn Sie zum Beispiel Getränke herstellen, enthalten einige Dimensionstabellen Daten zu Produkten, Märkten und Zeit. Die Fakttabelle enthält Transaktionsinformationen über die Produkte, die in jedem Gebiet je nach Jahreszeit bestellt werden.
- Verknüpfen der Fakttabelle und der Dimensionstabellen, um die Einzeldaten aus den Dimensionstabellen mit den Bestellinformationen zu kombinieren. Zum Beispiel könnten Sie die Produktdimension mit der Fakttabelle verknüpfen, um Informationen darüber hinzuzufügen, wie jedes Produkt entsprechend den Bestellungen verpackt wurde.

Weitere Informationen zu diesen Aufgaben und anderen Themen erhalten Sie, indem Sie das *Lernprogramm für das Informationsmanagement* durcharbeiten, sich den *DB2 Universal Database Kurzüberblick* ansehen oder die Informationen im Handbuch *Data Warehouse-Zentrale Verwaltung* nachlesen.

Kapitel 6. Informationen zu Spatial Extender

In diesem Abschnitt wird Spatial Extender vorgestellt. Dazu werden der Zweck des Produkts erläutert und die Daten beschrieben, die von ihm verarbeitet werden. Detaillierte Informationen zur Verwendung von Spatial Extender finden Sie im Handbuch *Spatial Extender Benutzer- und Referenzhandbuch*.

Der Zweck von Spatial Extender

Mit Spatial Extender kann ein *geografisches Informationssystem* (GIS) erstellt werden: ein Komplex von Objekten, Daten und Anwendungen, die das Generieren und Analysieren räumlicher Informationen über geografische Merkmale ermöglicht. *Geografische Merkmale* sind zum Beispiel die Objekte, die die Oberfläche der Erde bilden, und Objekte, die sie bedecken. Sie stellen sowohl die natürliche Umwelt (zum Beispiel Flüsse, Wälder, Hügel und Wüsten) als auch die kulturelle Umwelt (Städte, Wohnhäuser, Bürogebäude, Landmarken usw.) dar.

Räumliche Informationen sind Fakten wie zum Beispiel folgende:

- Die Position geografischer Merkmale in Bezug auf ihre Umgebung (zum Beispiel Punkte in einer Stadt, an denen sich Krankenhäuser und Kliniken befinden, oder die Nähe von Wohngebieten der Stadt zu lokalen Erdbebenzonen)
- Arten von Beziehungen, in denen die geografische Merkmale miteinander stehen (zum Beispiel Informationen, dass ein bestimmtes Flusssystem in einer bestimmten Region eingeschlossen ist oder dass bestimmte Brücken in dieser Region Nebenflüsse des Flusssystems überqueren)
- Maßangaben, die für ein oder mehrere geografische Merkmale gelten (zum Beispiel die Entfernung zwischen einem Bürogebäude und seiner Parzellengrenze, oder die Länge der Grenzlinie eines Vogelschutzgebiets)

Räumliche Informationen können entweder als solche oder in Kombination mit herkömmlichen Datenbankausgaben bei der Entwicklung von Projekten sowie bei der geschäftlichen und geschäftspolitischen Entscheidungsfindung hilfreich sein. Nehmen Sie zum Beispiel an, dass ein leitender Mitarbeiter des Bezirkssozialamts überprüfen muss, ob Antragsteller und Empfänger von Sozialhilfe tatsächlich in dem Bezirk wohnen, für den das Amt zuständig ist. Spatial Extender kann diese Informationen aus der Lage des Bezirks und aus den Adressen der Antragsteller und Empfänger ableiten.

Oder nehmen Sie an, der Besitzer einer Restaurantkette plant, Geschäfte in nahe gelegenen Städten zu eröffnen. Zur Bestimmung, wo die neuen Restaurants zu eröffnen sind, benötigt der Besitzer Antworten auf solche Fragen wie: In welchen Teilen dieser Städte gibt es Konzentrationen der Zielgruppe von Menschen, die solche Restaurants besuchen würden? Wo verlaufen die größeren Straßen? Wo ist die Kriminalitätsrate am niedrigsten? Wo befinden sich Konkurrenzunternehmen? Spatial Extender kann räumliche Informationen in visuellen Anzeigen darstellen, um solche Fragen zu beantworten, und das zugrundeliegende Datenbankverwaltungssystem kann Beschriftungen und Text zur Erläuterung der Anzeigen generieren.

Daten zur Darstellung geografischer Merkmale

Dieser Abschnitt enthält eine Übersicht über die Daten, die zur Gewinnung räumlicher Informationen generiert, gespeichert und verarbeitet werden können. Folgende Themen werden behandelt:

- Wie Daten geografische Merkmale darstellen
- Die Eigenschaften räumlicher Daten
- Möglichkeiten zur Erstellung räumlicher Daten

Wie Daten geografische Merkmale darstellen

In Spatial Extender kann ein geografisches Merkmal durch eine Zeile in einer Tabelle oder Sicht bzw. durch einen Teil einer solchen Zeile dargestellt werden. Betrachten Sie zum Beispiel die beiden folgenden geografischen Merkmale: Bürogebäude und Wohnungen. In Abb. 23 auf Seite 65 stellt jede Zeile der Tabelle BRANCHES eine Zweigstelle einer Bank dar. Abweichend davon stellt jede Zeile der Tabelle CUSTOMERS als Ganzes einen Kunden der Bank dar. Jedoch kann ein Teil jeder Zeile, insbesondere die Zellen mit der Adresse eines Kunden, als Darstellung der Wohnung des Kunden aufgefasst werden.

Diese Tabellen enthalten Daten, die die Zweigstellen und Kunden der Bank identifizieren und beschreiben. Solche Daten werden als *attributive Daten* bezeichnet.

BRANCHES

ID	NAME	ADDRESS	CITY	STATE	ZIP
937	Airzone-Multern	92467 Airzone Blvd	San Jose	CA	95141

CUSTOMERS

ID	LAST NAME	FIRST NAME	ADDRESS	CITY	STATE	ZIP	CHECKING	SAVINGS
59-6396	Kriner	Endela	9 Concourt Circle	San Jose	CA	95141	A	A

Abbildung 23. Tabellenzeile, die ein geografisches Merkmal darstellt. Tabellenzeile, deren Adressdaten ein geografisches Merkmal darstellen. Die Datenzeile in der Tabelle BRANCHES stellt eine Zweigstelle einer Bank dar. Die Zellen für Adressdaten in der Tabelle CUSTOMERS stellen die Wohnung eines Kunden dar.

Eine Untermenge der attributiven Daten (die Werte, die Zweigstellen- oder Kundenadressen darstellen) kann in Werte umgesetzt werden, die räumliche Informationen zur Verfügung stellen. Wie zum Beispiel in der Abbildung zu sehen ist, lautet eine Zweigstellenadresse 92467 Airzone Blvd., San Jose CA 95141. Eine Kundenadresse lautet 9 Concourt Circle, San Jose CA 95141. Spatial Extender kann diese Adressen in Werte umsetzen, die angeben, wo sich die Zweigstelle und die Wohnung des Kunden in Bezug auf die jeweilige Umgebung befinden. Abb. 24 zeigt die Tabellen BRANCHES und CUSTOMERS mit neuen Spalten, die solche Werte enthalten.

BRANCHES

ID	NAME	ADDRESS	CITY	STATE	ZIP	LOCATION
937	Airzone-Multern	92467 Airzone Blvd	San Jose	CA	95141	

CUSTOMERS

ID	LAST NAME	FIRST NAME	ADDRESS	CITY	STATE	ZIP	LOCATION	CHECKING	SAVINGS
59-6396	Kriner	Endela	9 Concourt Circle	San Jose	CA	95141		A	A

Abbildung 24. Tabellen mit hinzugefügten Spalten für räumliche Daten. In jeder Tabelle ist die Spalte LOCATION für Koordinaten vorgesehen, die den Adressen entsprechen.

Wenn Adressen und ähnliche Angaben als Ausgangspunkt für räumliche Informationen genutzt werden, werden sie als *Quellendaten* bezeichnet. Da die Werte, die aus ihnen abgeleitet werden, räumliche Informationen zur Verfügung stellen, werden diese abgeleiteten Werte als *räumliche Daten* bezeichnet. Im nächsten Abschnitt werden räumliche Daten beschrieben und die ihnen zugehörigen Datentypen vorgestellt.

Die Eigenschaften räumlicher Daten

Ein wesentlicher Teil räumlicher Daten besteht aus Koordinaten. Eine *Koordinate* ist eine numerische Angabe einer Position in Relation zu einem Bezugspunkt. Zum Beispiel sind Breitengrade Koordinaten, die Positionen in Relation zum Äquator angeben. Längengrade sind Koordinaten, die Positionen in Relation zum Nullmeridian von Greenwich angeben. Auf diese Weise ist die Lage des Yellowstone National Park durch die geografische Breite (44,45 Grad nördlich des Äquators) und die geografische Länge (110,40 Grad westlich des Nullmeridians von Greenwich) definiert.

Längen, Breiten und ihre Bezugspunkte sowie andere zugehörige Parameter werden kollektiv als *Koordinatensystem* bezeichnet. Es gibt außerdem Koordinatensysteme, die auf anderen Werten als geografische Länge und Breite basieren. Diese Koordinatensysteme verfügen über eigene Positionsmaße, Bezugspunkte und zusätzliche distinktive Parameter.

Das einfachste räumliche Datenelement besteht aus zwei Koordinaten, die die Position eines einzelnen geografischen Merkmals definieren. Der Ausdruck *Datenelement* bezieht sich auf den Wert oder die Werte, die eine Zelle in einer relationalen Tabelle einnehmen. Ein umfassenderes räumliches Datenelement besteht aus mehreren Koordinaten, die einen linearen Pfad, zum Beispiel eine Straße oder einen Flusslauf, definieren. Eine dritte Art besteht aus Koordinaten, die den Umfang eines Gebiets, zum Beispiel die Grenzlinie einer Landparzelle oder einer Schwemmebene definieren.

Jedes räumliche Datenelement ist ein Exemplar eines räumlichen Datentyps. Der Datentyp für zwei Koordinaten, die eine Position definieren, ist ST_Point, der Datentyp für Koordinaten, die lineare Pfade definieren, ist ST_LineString und der Datentyp für Koordinaten, die Umfänge definieren, ist ST_Polygon. Diese Typen sind, zusammen mit den anderen Datentypen für räumliche Daten, strukturierte Datentypen, die zu einer Hierarchie gehören.

Möglichkeiten zur Erstellung räumlicher Daten

Räumliche Daten können auf folgende Arten gewonnen werden:

- Ableiten aus attributiven Daten
- Ableiten aus anderen räumlichen Daten
- Importieren

Verwenden attributiver Daten als Quellendaten

Das Ableiten räumlicher Daten aus attributiven Daten (z. B. Adressen) wird als *geografische Codierung (Geocoding)* bezeichnet. In Abb. 24 auf Seite 65 sind zwei Spalten zu sehen, eine in der Tabelle BRANCHES und eine in der Tabelle CUSTOMERS, die für räumliche Daten vorgesehen sind. Stellen Sie sich vor, dass Spatial Extender die Adressen in diesen Tabellen geografisch codiert und das Ergebnis (Koordinaten, die den Adressen entsprechen) in die Spalten einfügt. Abb. 25 veranschaulicht das Ergebnis.

BRANCHES

ID	NAME	ADDRESS	CITY	STATE	ZIP	LOCATION
937	Airzone-Multern	92467 Airzone Blvd	San Jose	CA	95141	1653 3094

CUSTOMERS

ID	LAST NAME	FIRST NAME	ADDRESS	CITY	STATE	ZIP	LOCATION	CHECKING	SAVINGS
59-6396	Kriner	Endela	9 Concourt Circle	San Jose	CA	95141	953 1527	A	A

Abbildung 25. Tabellen, die aus Quellendaten abgeleitete räumliche Daten enthalten. Die Spalte LOCATION in der Tabelle CUSTOMERS enthält Koordinaten, die durch eine geografische Codierung aus der Adresse in den Spalten ADDRESS, CITY, STATE und ZIP abgeleitet wurden. Analog enthält die Spalte LOCATION in der Tabelle BRANCHES Koordinaten, die durch eine geografische Codierung aus der Adresse in den Spalten ADDRESS, CITY, STATE und ZIP dieser Tabelle abgeleitet wurden.

Spatial Extender arbeitet mit einer Funktion namens *geocoder*, um attributive Daten geografisch zu codieren und die resultierenden räumlichen Daten in Spalten abzulegen.

Verwenden anderer räumlicher Daten als Quellendaten

Räumliche Daten können nicht nur aus attributiven Daten, sondern auch aus anderen räumlichen Daten generiert werden. Nehmen Sie zum Beispiel an, dass die Bank, deren Zweigstellen in der Tabelle BRANCHES definiert sind, in Erfahrung bringen will, wie viele Kunden innerhalb einer Entfernung von fünf Meilen von einer Zweigstelle wohnen. Bevor die Bank diese Informationen aus der Datenbank erhalten kann, muss sie der Datenbank die Definition einer Zone hinzufügen, die innerhalb eines Radius von fünf Meilen um jede Zweigstelle liegt. Eine Spatial Extender-Funktion namens ST_Buffer kann eine solche Definition erstellen. Mit Hilfe der Koordinaten der einzelnen Zweigstellen als Eingabe kann diese Funktion Koordinaten generieren, die die Grenzlinien der gewünschten Zonen festlegen. Abb. 26 auf Seite 68 zeigt die Tabelle BRANCHES mit Informationen, die von ST_Buffer geliefert wurden.

BRANCHES

ID	NAME	ADDRESS	CITY	STATE	ZIP	LOCATION	SALES_AREA
937	Airzone-Multern	92467 Airzone Blvd	San Jose	CA	95141	1653 3094	1002 2001, 1192 3564, 2502 3415, 1915 3394, 1002 2001

Abbildung 26. Tabelle mit neuen räumlichen Daten, die aus vorhandenen räumlichen Daten abgeleitet wurden. Die Koordinaten in der Spalte SALES_AREA wurden durch die Funktion ST_Buffer aus den Koordinaten in der Spalte LOCATION abgeleitet.

Neben der Funktion ST_Buffer bietet Spatial Extender auch verschiedene andere Funktionen, die neue räumliche Daten aus vorhandenen räumlichen Daten ableiten.

Importieren räumlicher Daten

Eine dritte Methode zur Gewinnung räumlicher Daten ist das Importieren der Daten aus Dateien, die eines der von Spatial Extender unterstützten Formate besitzen. Diese Dateien enthalten Daten, die in der Regel auf Karten angewendet werden: Planungslinien, Schwemmebenen, Erdbebenverwerfungen usw. Durch die kombinierte Verwendung solcher Daten mit räumlichen Daten, die erstellt werden, können Sie die Ihnen zur Verfügung stehenden geografischen Informationen erweitern. Wenn zum Beispiel eine Bebauungsbehörde die Risiken bestimmen muss, denen ein Wohngebiet ausgesetzt ist, kann sie mit Hilfe von ST_Buffer eine Zone um das Wohngebiet definieren und anschließend Daten über Schwemmebenen und Erdbebenverwerfungen importieren, um festzustellen, welche dieser Problembereiche die Zone überlappen.

Teil 3. Datenbankentwurf

Kapitel 7. Entwerfen des logischen Datenbankaufbaus

Dieser Abschnitt behandelt die folgenden Schritte zum Entwurf einer Datenbank:

- „Festlegen der in der Datenbank aufzuzeichnenden Daten"
- „Definieren von Tabellen für jede Art von Relation" auf Seite 73
- „Definieren der Spalten für alle Tabellen" auf Seite 76
- „Definieren einer oder mehrerer Spalten als Primärschlüssel" auf Seite 79
- „Sicherstellen, dass gleiche Werte die gleiche Entität darstellen" auf Seite 83
- „Normalisieren der Tabellen" auf Seite 84
- „Planen der Integritätsbedingungen" auf Seite 90
- „Weitere Überlegungen zum Datenbankentwurf" auf Seite 99.

Das Ziel des Entwurfs einer Datenbank ist es, eine Darstellung der Umgebung zu finden, die leicht verständlich ist und als Basis für zukünftige Erweiterungen dienen kann. Darüber hinaus soll der Aufbau der Datenbank die Wahrung der Konsistenz und der Integrität der Daten unterstützen. Dies wird durch eine Datenbankstruktur erreicht, mit deren Hilfe Redundanz verringert und Anomalien eliminiert werden, die während der Aktualisierung der Datenbank auftreten können.

Diese Schritte sind Teil des *logischen* Datenbankentwurfs. Der Entwurf einer Datenbank ist kein linearer Prozess. Wahrscheinlich müssen im Laufe des Entwurfs Schritte wiederholt ausgeführt oder überarbeitet werden.

Die *physische* Implementierung des Datenbankentwurfs wird in „Kapitel 8. Entwerfen der physischen Datenbank" auf Seite 103 und "Implementieren des Datenbankentwurfs" im Handbuch *Systemverwaltung: Implementierung* beschrieben.

Festlegen der in der Datenbank aufzuzeichnenden Daten

Der erste Schritt bei der Entwicklung eines Datenbankentwurfs ist die Festlegung, welche Typen von Daten in den Tabellen der Datenbank gespeichert werden sollen. Eine Datenbank umfasst Informationen über die *Entitäten* innerhalb einer Organisation oder Firma sowie über die Beziehungen bzw. Abhängigkeiten zwischen ihnen. In einer relationalen Datenbank werden *Entitäten* als *Tabellen* dargestellt.

Eine *Entität* ist eine Person, ein Objekt oder ein Begriff, über die Informationen gespeichert werden sollen. Einige der Entitäten, die in den Beispiel-

tabellen beschrieben werden, sind zum Beispiel Mitarbeiter, Abteilungen und Projekte. (Eine Beschreibung der Beispieldatenbank finden Sie im Handbuch *SQL Reference*.)

In der Beispieltabelle EMPLOYEE hat die Entität "EMPLOYEE" (Mitarbeiter) bestimmte *Attribute* oder Merkmale, wie Personalnummer, Name, Abteilung (Work Department) und Gehalt (Salary). Diese Merkmale bilden die *Spalten* EMPNO, FIRSTNME, LASTNAME, WORKDEPT und SALARY der Tabelle.

Ein konkretes *Vorkommen* der Entität "Mitarbeiter" (employee) besteht aus den Werten aller Spalten für einen Mitarbeiter. Jeder Mitarbeiter hat eine eindeutige Personalnummer (EMPNO), die zur Identifizierung dieses Vorkommens der Entität „Mitarbeiter" verwendet werden kann. Jede Zeile in einer Tabelle stellt ein Vorkommen einer Entität oder Relation dar. Die Werte in der ersten Zeile der folgenden Tabelle beschreiben beispielsweise eine Mitarbeiterin namens Haas.

Tabelle 4. Vorkommen der Entität Mitarbeiter und zugehöriger Attribute

EMPNO	FIRSTNME	LASTNAME	WORKDEPT	JOB
000010	Christine	Haas	A00	President
000020	Michael	Thompson	B01	Manager
000120	Sean	O'Connell	A00	Clerk
000130	Dolores	Quintana	C01	Analyst
000030	Sally	Kwan	C01	Manager
000140	Heather	Nicholls	C01	Analyst
000170	Masatoshi	Yoshimura	D11	Designer

Es gibt einen wachsenden Bedarf an Unterstützung für neuere Datenbankanwendungen wie zum Beispiel Multimediaanwendungen. Es kann sinnvoll sein, Attribute zur Unterstützung von Multimediaobjekten, z. B. Dokumenten, Videodaten oder gemischten Medien, Bildern und Tondaten, zu berücksichtigen.

Innerhalb einer Tabelle hat jede Spalte einer Zeile eine bestimmte Relation oder Beziehung zu allen anderen Spalten dieser Zeile. Einige Relationen, die in den Beispieltabellen zum Ausdruck kommen, sind folgende:

- Mitarbeiter sind Abteilungen zugeordnet.
  - Dolores Quintana gehört zu Abteilung C01 (WORKDEPT).
- Mitarbeiter führen eine Jobbezeichnung.
  - Dolores arbeitet als „Analyst".
- Abteilungen berichten anderen Abteilungen.
  - Abteilung C01 berichtet an Abteilung A00.

- Abteilung B01 berichtet an Abteilung A00.

- Mitarbeiter arbeiten an Projekten.
  - Dolores und Heather arbeiten beide an Projekt IF1000.
- Mitarbeiter leiten Abteilungen.
  - Sally leitet die Abteilung C01.

"Mitarbeiter (Employee) und "Abteilung" (Department) sind Entitäten. Sally Kwan ist Teil eines Vorkommens der Entität "Mitarbeiter", und C01 ist Teil des Vorkommens der Entität "Abteilung". Für dieselben Spalten in jeder Zeile einer Tabelle gelten dieselben Arten von Relationen. Zum Beispiel gibt eine Zeile einer Tabelle die Relation an, dass Sally Kwan die Abteilung C01 leitet. Eine andere Zeile enthält die Relation, dass Sean O'Connell Sachbearbeiter (Clerk) in der Abteilung A00 ist.

Die Informationen innerhalb einer Tabelle sind von der Art der Relation, die dargestellt werden soll, vom Maß der benötigten Flexibilität und von der gewünschten Geschwindigkeit, mit der die Daten abgerufen werden können, abhängig.

Neben der Feststellung, welche Beziehungen zwischen Entitäten innerhalb Ihres Unternehmens in der Datenbank erfasst werden sollen, müssen auch andere Arten von Informationen bestimmt werden, z. B. die Geschäftsregeln, denen die Daten unterliegen.

Definieren von Tabellen für jede Art von Relation

In einer Datenbank können verschiedene Arten von Beziehung definiert werden. Betrachten Sie zum Beispiel die möglichen Relationen zwischen Mitarbeitern (Employees) und Abteilungen (Departments). Ein Mitarbeiter kann in nur einer Abteilung arbeiten. Eine solche Relation ist *einwertig* für den Mitarbeiter. Auf der anderen Seite kann eine Abteilung mehrere Mitarbeiter haben. Diese Relation ist *mehrwertig* für die Abteilungen. Die Relation zwischen Mitarbeiter (einwertig) und Abteilungen (mehrwertig) ist eine *Eins-zu-viele-Beziehung*. In diesem Abschnitt werden die folgenden Arten von Beziehungen behandelt:

- „Eins-zu-viele- und Viele-zu-eins-Beziehungen"
- „Viele-zu-viele-Beziehungen" auf Seite 74
- „Eins-zu-eins-Beziehungen" auf Seite 75

Eins-zu-viele- und Viele-zu-eins-Beziehungen

Zur Definition von Tabellen für die einzelnen Eins-zu-viele- und Viele-zu-eins-Beziehungen empfiehlt sich folgende Methode:

1. Fassen Sie alle Beziehungen in Gruppen zusammen, bei denen jeweils die "Viele"-Seite der Relation dieselbe Entität ist.
2. Definieren Sie eine Tabelle für alle Relationen in der Gruppe.

Im folgenden Beispiel ist die Viele-Seite der ersten und zweiten Relation ″Employees″ (Mitarbeiter), so dass eine Tabelle für die Mitarbeiter (Tabelle EMPLOYEE) definiert wird.

Tabelle 5. Viele-zu-eins-Beziehungen

Entität	Beziehung	Entität
Mitarbeiter	sind zugeordnet zu	Abteilungen
Mitarbeiter	arbeiten als	Jobbezeichnung
Abteilungen	berichten an	(Verwaltungs-) Abteilungen

In der dritten Relation ist die Entität ″Departments″ (Abteilungen) auf der Viele-Seite, so dass eine Tabelle DEPARTMENT definiert wird.

Die folgenden Tabellen veranschaulichen, wie diese Relationen dargestellt werden:

Die Tabelle EMPLOYEE:

EMPNO	WORKDEPT	JOB
000010	A00	President
000020	B01	Manager
000120	A00	Clerk
000130	C01	Analyst
000030	C01	Manager
000140	C01	Analyst
000170	D11	Designer

Die Tabelle DEPARTMENT:

DEPTNO	ADMRDEPT
C01	A00
D01	A00
D11	D01

Viele-zu-viele-Beziehungen

Eine Relation, die in beiden Richtungen mehrwertig ist, wird als eine Viele-zu-viele-Beziehung bezeichnet. Ein Mitarbeiter (Employee) kann an mehr als einem Projekt arbeiten, und ein Projekt kann mehr als einen Mitarbeiter beschäftigen. Auf die Fragen „Woran arbeitet Dolores Quintana?” und „Wer arbeitet an Projekt IF1000?” gibt es jeweils mehrere Antworten.

Eine Viele-zu-viele-Beziehung kann in einer Tabelle mit einer Spalte für jede Entität (Mitarbeiter „EMPNO" und Projekt „PROJNO") wie im folgenden Beispiel dargestellt werden.

Die folgende Abbildung zeigt, wie eine Viele-zu-viele-Beziehung (ein Mitarbeiter kann an vielen Projekten arbeiten, und ein Projekt kann viele Mitarbeiter beschäftigen) in einer Tabelle dargestellt werden kann:

Die Tabelle EMP_ACT (Mitarbeitertätigkeiten):

EMPNO	PROJNO
000030	IF1000
000030	IF2000
000130	IF1000
000140	IF2000
000250	AD3112

Eins-zu-eins-Beziehungen

Eins-zu-eins-Beziehungen sind in beide Richtungen einwertig. Ein Manager leitet eine Abteilung. Eine Abteilung hat nur einen Manager. Auf die Fragen „Wer ist der Manager der Abteilung C01?" und „Welche Abteilung wird von Sally Kwan geleitet?" gibt es jeweils nur eine Antwort. Die Beziehung kann entweder der Tabelle DEPARTMENT oder der Tabelle EMPLOYEE zugeordnet werden. Da alle Abteilungen Manager haben, aber nicht alle Mitarbeiter Manager sind, ist die logische Konsequenz, die Manager der Tabelle DEPARTMENT wie im folgenden Beispiel zuzuordnen.

Die folgende Abbildung zeigt, wie eine Eins-zu-eins-Beziehung in einer Tabelle dargestellt wird:

Die Tabelle DEPARTMENT:

DEPTNO	MGRNO
A00	000010
B01	000020
D11	000060

Definieren der Spalten für alle Tabellen

Eine Spalte in einer relationalen Tabelle wird folgendermaßen definiert:

1. Wählen Sie einen Namen für die Spalte.

   Jede Spalte in einer Tabelle muss einen für diese Tabelle eindeutigen Namen haben. Einzelheiten zu den Spaltennamen finden Sie in „Anhang A. Namenskonventionen" auf Seite 217.

2. Definieren Sie die Art der Daten, die für die Spalte gültig sein sollen.

   Der *Datentyp* und die *Länge* legen den Typ und die maximale Länge der Daten fest, die als Werte der Spalte gültig sind. Datentypen können aus den im Datenbankmanager vordefinierten Typen ausgewählt oder vom Benutzer als so genannte benutzerdefinierte Typen selbst erstellt werden. Informationen über die von DB2 bereitgestellten Datentypen und über benutzerdefinierte Typen finden Sie im Handbuch *SQL Reference*.

   Kategorien von Datentypen sind zum Beispiel: numerische Daten, Zeichenfolge, Doppelbytezeichenfolge (oder Grafikzeichenfolge), Datum/Uhrzeit und Binärzeichenfolge.

   Der Datentyp *großes Objekt* (LOB) unterstützt Multimedia-Objekte wie Dokumente, Video-, Abbild- und Tondaten. Diese Objekte werden mit Hilfe der folgenden Datentypen implementiert:

   - Eine Zeichenfolge des Typs *großes Binärobjekt* (BLOB). Beispiele für BLOBs sind Fotografien von Mitarbeitern, Stimmenaufnahmen und Videoaufnahmen.
   - Eine Zeichenfolge des Typs *großes Zeichenobjekt* (CLOB), in der eine Folge von Zeichen entweder aus Einzelbyte- oder Mehrbytezeichen oder aus einer Kombination aus beidem bestehen kann. Ein Beispiel für ein CLOB ist ein Lebenslauf eines Mitarbeiters.
   - Eine Zeichenfolge des Typs *Doppelbytezeichen für großes Objekt* (DBCLOB), in der eine Folge von Zeichen aus Doppelbytezeichen besteht. Ein Beispiel für ein DBCLOB ist ein Lebenslauf in japanischer Sprache.

   Weitere Einzelheiten zur Unterstützung großer Objekte (LOBs) finden Sie im Handbuch *SQL Reference*.

   Ein *benutzerdefinierter Datentyp* (UDT - User-Defined Type) ist ein Datentyp, der von einem vorhandenen Typ abgeleitet ist. Es können Datentypen definiert werden, die von vorhandenen Typen abgeleitet sind und über ähnliche Merkmale wie die vorhandenen Typen verfügen, die jedoch als separat und nicht kompatibel behandelt werden sollen.

Ein *strukturierter Typ* ist ein benutzerdefinierter Typ, dessen Struktur in der Datenbank definiert ist. Er enthält eine Reihe benannter *Attribute*, die jeweils über einen Datentyp verfügen. Ein strukturierter Typ kann als *untergeordneter Typ* eines anderen strukturierten Typs definiert sein, der dementsprechend als zugehöriger *übergeordneter Typ* bezeichnet wird. Ein untergeordneter Typ übernimmt alle Attribute des dazugehörigen übergeordneten Typs und kann zusätzlich über weitere Attribute verfügen. Die Gruppe strukturierter Typen, die zu einem gemeinsamen übergeordneten Typ gehören, wird als *Typhierarchie* bezeichnet. Ein übergeordneter Typ, der keinem anderen Typ untergeordnet ist, wird als *Stammtyp* der Typhierarchie bezeichnet.

Ein strukturierter Typ kann als Typ einer Tabelle oder einer Sicht verwendet werden. Die Namen und Datentypen der Attribute der strukturierten Typen werden zusammen mit der Objektkennung als Namen und Datentypen für die Spalten dieser *typisierten Tabelle* oder *typisierten Sicht* verwendet. Zeilen einer typisierten Tabelle oder typisierten Sicht können als Darstellung von Exemplaren des strukturierten Typs verstanden werden.

Ein strukturierter Typ kann nicht als Datentyp für eine Spalte einer Tabelle oder Sicht verwendet werden. Außerdem besteht keine Möglichkeit, ein vollständiges Exemplar eines strukturierten Typs in eine Host-Variable eines Anwendungsprogramms zu übernehmen.

Ein *Referenztyp* ist ein Begleittyp zum strukturierten Typ. Wie ein einzigartiger Datentyp ist auch der Referenztyp ein Skalartyp, dessen Darstellung mit der Darstellung eines integrierten Datentyps identisch ist. Diese Darstellung wird von allen Typen der Typhierarchie gemeinsam benutzt. Die Darstellung des Referenztyps wird beim Erstellen des Stammtyps einer Typhierarchie definiert. Bei Verwendung eines Referenztyps wird ein strukturierter Typ als Parameter des Typs angegeben. Dieser Parameter wird als *Zieltyp* der Referenz bezeichnet.

Das Ziel einer Referenz ist immer eine Zeile in einer typisierten Tabelle oder Sicht. Bei Verwendung eines Referenztyps kann ein *Bereich* (Scope) definiert werden. Der Bereich identifiziert eine Tabelle (die *Zieltabelle*) oder eine Sicht (die *Zielsicht*), die die Zielzeile eines Referenzwerts enthält. Die Zieltabelle oder -sicht muss denselben Typ aufweisen wie der Zieltyp des Referenztyps. Ein Exemplar eines Referenztyps mit Bereich identifiziert eindeutig eine Zeile in einer typisierten Tabelle oder Sicht, die als *Zielzeile* dieses Referenztyps bezeichnet wird.

Eine *benutzerdefinierte Funktion* (UDF - User-Defined Function) kann zu zahlreichen Zwecken verwendet werden, darunter auch zum Aufrufen von Routinen, die Vergleiche oder Umwandlungen zwischen benutzerdefinierten Datentypen ermöglichen. Benutzerdefinierte Funktionen erweitern und ergänzen die Unterstützung, die von integrierten SQL-Funktionen zur Verfügung gestellt wird, und können überall dort verwendet werden, wo auch integrierte Funktionen verwendet werden können. Es gibt zwei Arten benutzerdefinierter Funktionen:

- Externe Funktionen, die in einer Programmiersprache geschrieben sind
- Quellenfunktionen, die zum Aufrufen anderer benutzerdefinierter Funktionen verwendet werden

Beispielsweise gibt es die beiden numerischen Datentypen der europäischen Schuhgröße und der amerikanischen Schuhgröße. Beide Typen stellen Schuhgrößen dar, aber sie sind nicht miteinander kompatibel, da die Maßeinheiten verschieden sind und nicht direkt miteinander verglichen werden können. Eine benutzerdefinierte Funktion kann aufgerufen werden, um die Maßangaben von einer Schuhgröße in eine andere umzuwandeln.

Weitere Einzelheiten zur Vertiefung der benutzerdefinierten und strukturierten Datentypen sowie der Referenztypen und benutzerdefinierten Funktionen finden Sie im Handbuch *SQL Reference*.

3. Geben Sie an, welche Spalten eventuell Standardwerte enthalten müssen.

   Einige Spalten können nicht in allen Zeilen sinnvolle Werte enthalten. Dafür kann es folgende Gründe geben:

   - Ein Spaltenwert ist für die Zeile nicht zutreffend.

     Zum Beispiel kann eine Spalte, die die Mittelinitialen von Mitarbeiternamen enthält, keinen sinnvollen Wert enthalten, wenn der in der Zeile eingetragene Mitarbeiter keine Mittelinitiale hat.
   - Ein Wert ist zwar zutreffend, aber er ist noch nicht bekannt.

     Ein Beispiel wäre die Spalte MGRNO (Managernummer), die vorübergehend keine gültige Managernummer enthält, weil der vorige Manager versetzt wurde, aber bislang noch kein neuer Manager bestimmt wurde.

   In beiden Fällen haben Sie die Wahl, einen Nullwert (einen speziellen Wert, der angibt, dass der Spaltenwert unbekannt oder unzutreffend ist) zuzulassen oder einen vom Nullwert abweichenden Standardwert durch den Datenbankmanager oder die Anwendung zuordnen zu lassen.

   Einzelheiten zu Nullwerten (NULL Values) und Standardwerten (Default Values) finden Sie im Handbuch *SQL Reference*.

Definieren einer oder mehrerer Spalten als Primärschlüssel

Ein *Schlüssel* ist eine Gruppe von Spalten, deren Hilfe eine Zeile oder Zeilen identifiziert werden oder auf sie zugegriffen wird. Der Schlüssel wird in der Beschreibung einer Tabelle, eines Index oder einer referenziellen Integritätsbedingung angegeben. Eine Spalte kann zu mehreren Schlüsseln gehören.

Ein *eindeutiger Schlüssel* ist ein Schlüssel mit der Integritätsbedingung, dass nicht zwei seiner Werte gleich sein dürfen. Die Spalten eines eindeutigen Schlüssels können keine Nullwerte enthalten. Zum Beispiel kann eine Spalte mit der Personalnummer als eindeutiger Schlüssel definiert werden, da jeder Wert in der Spalte nur einen Mitarbeiter identifiziert. Innerhalb eines Unternehmens können nicht zwei Mitarbeiter dieselbe Personalnummer haben. Der Mechanismus, mit dem die Eindeutigkeit sichergestellt wird, wird als *eindeutiger Index* bezeichnet. Der eindeutige Index einer Tabelle ist eine Spalte oder eine geordnete Gruppe von Spalten, deren Werte jeweils eine Zeile eindeutig identifizieren (funktionell bestimmen). Ein eindeutiger Index kann Nullwerte enthalten.

Der *Primärschlüssel* ist einer der für eine Tabelle definierten eindeutigen Schlüssel, der jedoch als Schlüssel mit der höchsten Priorität ausgewählt ist. Es kann nur einen Primärschlüssel für eine Tabelle geben.

Für den Primärschlüssel wird automatisch ein *Primärindex* erstellt. Der Primärindex wird vom Datenbankmanager zum effizienten Zugriff auf Tabellenzeilen verwendet und ermöglicht es dem Datenbankmanager, die Eindeutigkeit des Primärschlüssels zu erhalten. (Sie können außerdem Indizes für Spalten definieren, die nicht zum Primärschlüssel gehören, um bei der Verarbeitung von Abfragen einen effizienten Zugriff auf Daten bieten zu können.)

Wenn die Tabelle keinen „natürlichen" eindeutigen Schlüssel besitzt, oder wenn zur Unterscheidung eindeutiger Zeilen die Eingangsfolge verwendet wird, kann auch die Verwendung einer Zeitmarke als Bestandteil des Schlüssels sinnvoll sein. (Siehe auch „Definieren von IDENTITY-Spalten" auf Seite 81.)

Primärschlüssel für einige der Beispieltabellen sind folgende:

Tabelle	**Schlüsselspalte**
Tabelle EMPLOYEE	EMPNO
Tabelle DEPARTMENT	DEPTNO
Tabelle PROJECT	PROJNO

Das folgende Beispiel zeigt einen Teil der Tabelle PROJECT mit ihrem Primärschlüssel.

Tabelle 6. Ein Primärschlüssel in der Tabelle PROJECT

PROJNO (Primärschlüssel)	PROJNAME	DEPTNO
MA2100	Weld Line Automation	D01
MA2110	Weld Line Programming	D11

Wenn jede Spalte einer Tabelle doppelte Werte enthält, kann kein Primärschlüssel aus nur einer Spalte definiert werden. Ein Schlüssel, der aus mehreren Spalten besteht, wird als *zusammengesetzter Schlüssel* bezeichnet. Die Kombination der Spaltenwerte muss eine Entität eindeutig definieren. Wenn ein zusammengesetzter Schlüssel nicht einfach definiert werden kann, ist eventuell die Erstellung einer neuen Spalte mit eindeutigen Werten in Betracht zu ziehen.

Das folgende Beispiel zeigt einen Primärschlüssel, der mehrere Spalten (einen zusammengesetzten Schlüssel) enthält:

Tabelle 7. Ein zusammengesetzter Primärschlüssel in der Tabelle EMP_ACT

EMPNO (Primärschlüssel)	PROJNO (Primärschlüssel)	ACTNO (Primärschlüssel)	EMPTIME	EMSTDATE (Primärschlüssel)
000250	AD3112	60	1.0	1982-01-01
000250	AD3112	60	.5	1982-02-01
000250	AD3112	70	.5	1982-02-01

Identifizieren möglicher Schlüsselspalten

Ein Schlüsselkandidat wird durch die kleinste Anzahl von Spalten festgelegt, die eine Entität eindeutig definieren. Es kann mehrere Schlüsselkandidaten geben. In Tabelle 2 auf Seite 24 gibt es scheinbar mehrere Schlüsselkandidaten. Die Spalten EMPNO, PHONENO und LASTNAME identifizieren den Mitarbeiter eindeutig.

Bei der Auswahl eines Primärschlüssels aus einer Gruppe möglicher Schlüssel sollten die Kriterien Permanenz, Eindeutigkeit und Stabilität beachtet werden.

- Permanenz heißt, dass der Primärschlüssel für die Zeile stets vorhanden ist.
- Eindeutigkeit heißt, dass der Schlüsselwert für eine Zeile sich von den Schlüsselwerten aller anderen Zeilen unterscheidet.
- Stabilität heißt, dass die Werte für Primärschlüssel nie geändert werden.

Von den drei möglichen Schlüsselspalten im genannten Beispiel erfüllt nur die Spalte EMPNO alle Kriterien. Es kann zum Beispiel vorkommen, dass ein Mitarbeiter keine Telefonnummer (PHONENO) hat, wenn er in die Firma eintritt. Nachnamen (LASTNAME) können sich ändern und sind, wenn auch vielleicht zu einem bestimmten Zeitpunkt, so doch nicht unbedingt immer eindeutig. Die Personalnummer (EMPNO) ist die beste Wahl für den Primärschlüssel. Ein Mitarbeiter erhält nur einmal eine eindeutige Nummer, und diese Nummer wird in der Regel nicht geändert, solange der Mitarbeiter in der Firma bleibt. Da jeder Mitarbeiter unbedingt eine Nummer haben muss, gilt auch das Kriterium der Permanenz für die Werte dieser Spalte.

Definieren von IDENTITY-Spalten

Eine *IDENTITY-Spalte* stellt für DB2 eine Methode dar, automatisch einen eindeutigen numerischen Wert für jede Zeile in einer Tabelle zu generieren. Eine Tabelle kann eine einzelne Spalte besitzen, die mit dem Attribut IDENTITY definiert ist. Beispiele für IDENTITY-Spalten sind Auftragsnummern, Personalnummern, Bestandsnummern und Vorfallsnummern.

Werte für eine IDENTITY-Spalte können immer oder standardmäßig generiert werden.

- Die Eindeutigkeit einer IDENTITY-Spalte, die als *immer generiert* (generated always) definiert ist, wird von DB2 garantiert. Die Werte der Spalte werden immer von DB2 generiert. Die Angabe eines expliziten Werts durch Anwendungen wird nicht zugelassen.
- Eine IDENTITY-Spalte, die als *standardmäßig generiert* (generated by default) definiert ist, gibt Anwendungen eine Möglichkeit, einen Wert in die IDENTITY-Spalte explizit einzugeben. Wenn kein Wert angegeben wird, generiert DB2 einen, kann jedoch die Eindeutigkeit des Werts in diesem Fall nicht garantieren. DB2 garantiert die Eindeutigkeit nur für die Gruppe von Werten, die von DB2 generiert wird. Die Definition *standardmäßig generiert* ist zur Verwendung bei Datenweitergaben (Data Propagation), bei der der Inhalt einer vorhandenen Tabelle kopiert wird, oder zum Laden von Inhalt aus einer Tabelle bzw. erneutes Laden in eine Tabelle gedacht.

IDENTITY-Spalten eignen sich ideal zur Generierung eindeutiger Primärschlüsselwerte. Anwendungen können IDENTITY-Spalten nutzen, um Problemen beim gemeinsamen Zugriff und Leistungsproblemen vorzubeugen, die entstehen können, wenn eine Anwendung einen eigenen eindeutigen Zähler außerhalb der Datenbank generiert. Zum Beispiel besteht eine gängige Implementierung auf Anwendungsebene darin, eine 1-zeilige Tabelle mit einem Zähler zu pflegen. Jede Transaktion sperrt diese Tabelle, erhöht den Wert des Zählers und schreibt die Änderung fest.

Das heißt, zu einem gegebenen Zeitpunkt kann nur eine Transaktion den Wert des Zählers erhöhen. Wenn im Unterschied dazu der Zähler durch eine IDENTITY-Spalte realisiert wird, können wesentliche höhere Ebenen des gemeinsamen Zugriffs erreicht werden, weil der Zähler nicht durch Transaktionen gesperrt wird. Eine nicht festgeschriebene Transaktion, die den Wert des Zählers erhöht hat, hindert nachfolgende Transaktionen nicht daran, den Zähler ebenfalls zu erhöhen.

Der Wert des Zählers für die IDENTITY-Spalte wird unabhängig von der Transaktion erhöht (oder verringert). Wenn eine bestimmte Transaktion einen IDENTITY-Zähler zweimal erhöht, stellt diese Transaktion möglicherweise einen Abstand zwischen den beiden generierten Nummern fest, weil vielleicht andere Transaktionen zur gleichen Zeit denselben IDENTITY-Zähler ebenfalls erhöhen (d. h. Zeilen in die gleiche Tabelle einfügen). Wenn eine Anwendung unbedingt einen aufeinander folgenden Bereich von Nummern erhalten muss, sollte die Anwendung eine exklusive Sperre für die Tabelle definieren, in der die IDENTITY-Spalte enthalten ist. Diese Entscheidung muss jedoch gegen die daraus folgende Einbuße an gemeinsamen Zugriff abgewogen werden. Weiterhin ist es möglich, dass eine bestimmte IDENTITY-Spalte so aussehen kann, als habe sie Sprünge in den Nummern generiert, weil eine Transaktion, die einen Wert für die IDENTITY-Spalte generierte, rückgängig gemacht wurde, oder weil die Datenbank, die einen Bereich von Werten in einem Cache gespeichert hatte, deaktiviert wurde, bevor alle im Cache gespeicherten Werte zugeordnet werden konnten.

Die fortlaufenden Nummern, die durch die IDENTITY-Spalte generiert werden, besitzen die folgenden zusätzlichen Merkmale:

- Die Werte können jeden Exact Numeric-Datentyp ohne Kommastellen, d. h. SMALLINT, INTEGER, BIGINT oder DECIMAL mit null Kommastellen, besitzen. (Gleitkommazahlen mit einfacher und doppelter Genauigkeit werden als Approximate Numeric-Datentypen bezeichnet.)
- Fortlaufende Werte können sich durch ein beliebiges angegebenes INTEGER-Inkrement unterscheiden. Das Standardinkrement ist 1.
- Der Wert des Zählers für die IDENTITY-Spalte ist wiederherstellbar. Wenn ein Fehler auftritt, wird der Wert des Zählers aus den Protokollen rekonstruiert, wodurch sichergestellt wird, dass weiterhin eindeutige Werte generiert werden.
- Werte von IDENTITY-Spalten können im Cache gespeichert werden, um eine bessere Leistung zu erzielen.

Sicherstellen, dass gleiche Werte die gleiche Entität darstellen

Sie können die Attribute derselben Menge von Entitäten in mehreren Tabellen beschreiben. Zum Beispiel enthält die Tabelle EMPLOYEE die Nummer der Abteilung (WORKDEPT), der ein Mitarbeiter zugeteilt ist, und die Tabelle DEPARTMENT zeigt, welcher Manager jeder Abteilungsnummer (DEPTNO) zugeordnet ist. Um beide Mengen von Attributen gleichzeitig abzurufen, können die beiden Tabellen über die beiden übereinstimmenden Spalten verknüpft werden, wie im folgenden Beispiel gezeigt wird. Der Wert der Spalte WORKDEPT und der Wert der Spalte DEPTNO stellen dieselbe Entität dar und bilden daher einen *Verknüpfungspfad* zwischen den Tabellen DEPARTMENT und EMPLOYEE.

Die Tabelle DEPARTMENT:

DEPTNO	DEPTNAME	MGRNO	ADMRDEPT
D21	Administration Support	000070	D01

Die Tabelle EMPLOYEE:

EMPNO	FIRSTNAME	LASTNAME	WORKDEPT	JOB
000250	Daniel	Smith	D21	Clerk

Beim Abrufen von Informationen über eine Entität aus mehreren Tabellen muss sichergestellt werden, dass gleiche Werte auch die gleiche Entität bezeichnen. Die verknüpfenden Spalten können unterschiedliche Namen haben (wie WORKDEPT und DEPTNO im vorangegangenen Beispiel), oder sie können auch denselben Namen haben (wie die Spalten mit dem Namen DEPTNO in den Tabellen DEPARTMENT und PROJECT).

Normalisieren der Tabellen

Eine *Normalisierung* hilft bei der Vermeidung von Redundanzen und Inkonsistenzen in Tabellendaten. Sie bezeichnet den Prozess der Reduzierung von Tabellen auf eine Menge von Spalten, in der alle Spalten, die selbst zu keinem Schlüssel gehören, von der Primärschlüsselspalte abhängig sind. Wenn dies nicht der Fall ist, können die Daten durch Aktualisierungen inkonsistent werden.

In diesem Abschnitt werden kurz die Kriterien für die erste, zweite, dritte und vierte Normalform vorgestellt. Die fünfte Normalform einer Tabelle, die in zahlreichen Veröffentlichungen zum Entwurf von Datenbanken dargestellt wird, ist hier nicht beschrieben.

Form **Beschreibung**

Erste An jeder Zeilen- und Spaltenposition in der Tabelle existiert nur *ein* Wert und zu keiner Zeit eine Gruppe von Werten (siehe „Erste Normalform").

Zweite
Jede Spalte, die nicht Teil des Schlüssels ist, ist vom Schlüssel abhängig (siehe „Zweite Normalform" auf Seite 85).

Dritte Jede Spalte ohne Schlüsselfunktion ist von anderen Spalten ohne Schlüsselfunktion unabhängig und nur vom Schlüssel abhängig (siehe „Dritte Normalform" auf Seite 87).

Vierte Keine Zeile enthält zwei oder mehr unabhängige, mehrwertige Fakten über eine Entität (siehe „Vierte Normalform" auf Seite 88).

Erste Normalform

Eine Tabelle ist in der *ersten Normalform*, wenn sich in jeder Zelle nur ein Wert und nie eine Gruppe von Werten befindet. Eine Tabelle in der ersten Normalform erfüllt nicht unbedingt auch die Kriterien für höhere Normalformen.

Die folgende Tabelle verstößt beispielsweise gegen die Kriterien der ersten Normalform, weil in der Spalte LAGER mehrere Werte für jedes Vorkommen von ARTIKEL auftreten.

Tabelle 8. Tabelle, die die erste Normalform verletzt

ARTIKEL (Primärschlüssel)	LAGER
P0010	Lager A, Lager B, Lager C
P0020	Lager B, Lager D

Das folgende Beispiel zeigt die gleiche Tabelle in der ersten Normalform.

Tabelle 9. Tabelle in der ersten Normalform

ARTIKEL (Primär-schlüssel)	**LAGER (Primärschlüssel)**	**BESTAND**
P0010	Lager A	400
P0010	Lager B	543
P0010	Lager C	329
P0020	Lager B	200
P0020	Lager D	278

Zweite Normalform

Eine Tabelle ist in der *zweiten Normalform*, wenn jede Spalte, die nicht zum Schlüssel gehört, vom *gesamten* Schlüssel abhängig ist.

Die Kriterien der zweiten Normalform sind nicht erfüllt, wenn eine Spalte ohne Schlüsselfunktion von einem *Teil* eines zusammengesetzten Schlüssels abhängig ist, wie im folgenden Beispiel zu sehen ist:

Tabelle 10. Tabelle, die die zweite Normalform verletzt

ARTIKEL (Primär-schlüssel)	**LAGER (Primär-schlüssel)**	**BESTAND**	**LAGER_ADRESSE**
P0010	Lager A	400	Amselfelderstraße 24
P0010	Lager B	543	Industrieweg 6
P0010	Lager C	329	Kastanienallee 71
P0020	Lager B	200	Industrieweg 6
P0020	Lager D	278	Ulmenstraße 8

Der Primärschlüssel ist ein zusammengesetzter Schlüssel, der aus den Spalten ARTIKEL und LAGER besteht. Da die Spalte LAGER_ADRESSE jedoch nur vom Wert der Spalte LAGER abhängig ist, verstößt die Tabelle gegen die Regel der zweiten Normalform.

Dieser Tabellenentwurf hat folgende Nachteile:

- Die Lageradresse wird in jeden Datensatz für einen Artikel, der in dem entsprechenden Lagerhaus gelagert wird, wiederholt.
- Wenn die Adresse eines Lagerhauses geändert wird, muss jede Zeile, die sich auf einen Artikel in diesem Lagerhaus bezieht, aktualisiert werden.
- Aufgrund dieser Redundanz könnten die Daten inkonsistent werden, wenn für dasselbe Lagerhaus in verschiedenen Datensätzen unterschiedliche Adressen auftreten.
- Wenn zu einem Zeitpunkt keine Artikel in dem Lagerhaus gelagert werden, ist vielleicht keine Zeile mehr vorhanden, in der die Adresse des Lagerhauses gespeichert werden kann.

Die Lösung besteht in einer Aufspaltung der Tabelle in die beiden folgenden Tabellen:

Tabelle 11. Die Tabelle LAGERBESTAND erfüllt die zweite Normalform

ARTIKEL (Primärschlüssel)	**LAGER (Primärschlüssel)**	**BESTAND**
P0010	Lager A	400
P0010	Lager B	543
P0010	Lager C	329
P0020	Lager B	200
P0020	Lager D	278

Tabelle 12. Die Tabelle LAGERHAUS erfüllt die zweite Normalform

LAGER (Primärschlüssel)	**LAGER_ADRESSE**
Lager A	Amselfelderstraße 24
Lager B	Industrieweg 6
Lager C	Kastanienallee 71
Lager D	Ulmenstraße 8

Es muss beachtet werden, dass die Erstellung zweier Tabellen in der zweiten Normalform Auswirkungen auf die Leistung haben kann. Anwendungen, die Berichte über den Lagerort von Artikeln erstellen, müssen beide Tabellen verknüpfen, um die relevanten Informationen abrufen zu können.

Weitere Einzelheiten zur Datenbankleistung finden Sie im Abschnitt zum Optimieren der Anwendungsleistung im Handbuch *Systemverwaltung: Optimierung*.

Dritte Normalform

Eine Tabelle ist in der dritten Normalform, wenn jede Spalte ohne Schlüsselfunktion von anderen Spalten ohne Schlüsselfunktion unabhängig und nur vom Schlüssel abhängig ist.

Die erste Tabelle des folgenden Beispiels enthält die Spalten EMPNO und WORKDEPT. Nehmen Sie an, dass die Spalte DEPTNAME hinzugefügt wird (siehe Tabelle 14). Die neue Spalte ist von der Spalte WORKDEPT abhängig, während der Primärschlüssel nur die Spalte EMPNO umfasst. Die Tabelle verstößt nun gegen die Kriterien der dritten Normalform. Durch Ändern der Spalte DEPTNAME für einen einzelnen Mitarbeiter, z. B. John Parker, wird nicht auch der Abteilungsname für andere Mitarbeiter dieser Abteilung geändert. Beachten Sie, dass es nun zwei unterschiedliche Abteilungsnamen (DEPTNAME) für Abteilungsnummer (WORKDEPT) E11 gibt. Die sich daraus ergebende Inkonsistenz zeigt sich in der aktualisierten Version der Tabelle.

Tabelle 13. Nicht normalisierte Tabelle EMPLOYEE_DEPARTMENT vor der Aktualisierung

EMPNO (Primärschlüssel)	**FIRSTNAME**	**LASTNAME**	**WORKDEPT**	**DEPTNAME**
000290	John	Parker	E11	Operations
000320	Ramlal	Mehta	E21	Software Support
000310	Maude	Setright	E11	Operations

Tabelle 14. Nicht normalisierte Tabelle EMPLOYEE_DEPARTMENT nach der Aktualisierung. Die Daten der Tabelle sind nicht mehr konsistent.

EMPNO (Primärschlüssel)	**FIRSTNAME**	**LASTNAME**	**WORKDEPT**	**DEPTNAME**
000290	John	Parker	E11	Installation Mgmt
000320	Ramlal	Mehta	E21	Software Support
000310	Maude	Setright	E11	Operations

Die Tabelle kann normalisiert werden, indem eine neue Tabelle mit Spalten für WORKDEPT und DEPTNAME erstellt wird. Eine Aktualisierung wie das Ändern des Abteilungsnamens (DEPTNAME) ist nun wesentlich einfacher. Es muss nur eine Tabelle aktualisiert werden.

Eine SQL-Abfrage, mit der der Abteilungsname zusammen mit dem Namen des Mitarbeiters abgerufen wird, ist dagegen etwas komplizierter, da die beiden Tabellen verknüpft werden müssen. Sie wird wahrscheinlich eine etwas längere Zeit zur Ausführung benötigen als eine Abfrage auf eine einzelne Tabelle. Ferner ist zusätzlicher Speicherplatz erforderlich, da die Spalte WORKDEPT (Abteilung) in beiden Tabellen enthalten sein muss.

Die folgenden Tabellen werden als Ergebnis dieser Normalisierung definiert:

Tabelle 15. Tabelle EMPLOYEE nach Normalisierung der Tabelle EMPLOYEE_DEPARTMENT

EMPNO (Primärschlüssel)	FIRSTNAME	LASTNAME	WORKDEPT
000290	John	Parker	E11
000320	Ramlal	Mehta	E21
000310	Maude	Setright	E11

Tabelle 16. Tabelle DEPARTMENT nach Normalisierung der Tabelle EMPLOYEE_DEPARTMENT

DEPTNO (Primärschlüssel)	DEPTNAME
E11	Operations
E21	Software Support

Vierte Normalform

Eine Tabelle ist in der vierten Normalform, wenn keine Zeile zwei oder mehr unabhängige, mehrwertige Fakten bezüglich einer Entität enthält.

Betrachten Sie zum Beispiel folgende Entitäten: Mitarbeiter (Employee), Fachkenntnisse (Skill) und Sprachen (Language). Ein Mitarbeiter kann über Fachkenntnisse auf mehreren Gebieten verfügen und mehrere Sprachen beherrschen. Es gibt zwei Beziehungen, eine zwischen Mitarbeitern und Fachkenntnissen und eine zwischen Mitarbeitern und Sprachen. Eine Tabelle verstößt gegen die Kriterien der vierten Normalform, wenn sie beide Beziehungen darstellt, wie im folgenden Beispiel gezeigt:

Tabelle 17. Tabelle, die die vierte Normalform verletzt

EMPNO (Primärschlüssel)	SKILL (Primärschlüssel)	LANGUAGE (Primärschlüssel)
000130	Data Modelling	English
000130	Database Design	English
000130	Application Design	English

Tabelle 17. Tabelle, die die vierte Normalform verletzt (Forts.)

EMPNO (Primärschlüssel)	SKILL (Primärschlüssel)	LANGUAGE (Primärschlüssel)
000130	Data Modelling	Spanish
000130	Database Design	Spanish
000130	Application Design	Spanish

Statt dessen sollten die beiden Beziehungen in zwei Tabellen dargestellt werden:

Tabelle 18. Tabelle EMPLOYEE_SKILL in der vierten Normalform

EMPNO (Primärschlüssel)	SKILL (Primärschlüssel)
000130	Data Modelling
000130	Database Design
000130	Application Design

Tabelle 19. Tabelle EMPLOYEE_LANGUAGE in der vierten Normalform

EMPNO (Primärschlüssel)	LANGUAGE (Primärschlüssel)
000130	English
000130	Spanish

Wenn jedoch die Attribute gegenseitige Abhängigkeiten zeigen, (d. h., der Mitarbeiter verwendet bestimmte Sprachen nur im Zusammenhang mit bestimmten Fachgebieten), dann sollte die Tabelle *nicht* geteilt werden.

Eine gute Strategie bei der Entwicklung eines Datenbankentwurfs besteht darin, alle Daten in Tabellen in der vierten Normalform anzuordnen und anschließend zu entscheiden, ob das Ergebnis eine akzeptable Leistung liefert. Ist dies nicht der Fall, können Sie die Daten in Tabellen in der dritten Normalform neu anordnen und die Leistung erneut begutachten.

Planen der Integritätsbedingungen

Eine *Integritätsbedingung* ist eine Regel, die vom Datenbankmanager angewendet wird. Vier Arten der Behandlung von Integritätsbedingungen werden in diesem Abschnitt beschrieben:

Eindeutige Integritätsbedingungen
: Stellen die Eindeutigkeit der Schlüsselwerte in einer Tabelle sicher. Alle Änderungen an den Spalten, die den Primärschlüssel bilden, werden auf Eindeutigkeit überprüft.

Referenzielle Integrität
: Implementiert referenzielle Integritätsbedingungen für die Operationen INSERT (Einfügen), UPDATE (Aktualisieren) und DELETE (Löschen). Dies ist der Status einer Datenbank, in der alle Werte aller Fremdschlüssel gültig sind.

Prüfungen auf Integritätsbedingungen in Tabellen
: Stellen sicher, dass keine geänderten Daten gegen Bedingungen verstoßen, die bei der Erstellung oder der Änderung der Tabelle angegeben wurden.

Auslöser
: Definieren eine Reihe von Aktionen, die ausgeführt werden, wenn sie durch eine Aktualisierungs-, Lösch- oder Einfügeoperation für eine angegebene Tabelle aufgerufen werden.

Eindeutige Integritätsbedingungen

Eine *eindeutige Integritätsbedingung* ist die Regel, die sicherstellt, dass Schlüsselwerte innerhalb der Tabelle eindeutig sind. Jede Spalte, die Bestandteil des Schlüssels in einer eindeutigen Integritätsbedingung ist, muss mit NOT NULL (keine Nullwerte möglich) definiert werden. Eindeutige Integritätsbedingungen werden in der Anweisung CREATE TABLE oder ALTER TABLE mit Hilfe der Klausel PRIMARY KEY oder UNIQUE definiert.

Eine Tabelle kann eine beliebige Anzahl eindeutiger Integritätsbedingungen enthalten. Jedoch kann nur eine eindeutige Spaltenfolge als Primärschlüssel für eine Tabelle definiert werden. Außerdem kann eine Tabelle nicht mehr als eine eindeutige Integritätsbedingung für dieselbe Gruppe von Spalten besitzen.

Wenn eine eindeutige Integritätsbedingung definiert ist, erstellt der Datenbankmanager einen eindeutigen Index (falls erforderlich) und markiert ihn als primären oder eindeutigen systembedingten Index. Die Einhaltung der Integritätsbedingung wird mit Hilfe des eindeutigen Index gewährleistet. Wenn eine eindeutige Integritätsbedingung für eine Spalte definiert ist, wird die Überprüfung auf Eindeutigkeit bei Aktualisierung mehrerer Zeilen an das Ende der Aktualisierungsoperation verlegt.

Die Spalten, die einer eindeutigen Integritätsbedingung unterliegen, können auch als Primärschlüssel in einer referenziellen Integritätsbedingung verwendet werden.

Referenzielle Integrität

Der Datenbankmanager sorgt für die Erhaltung der referenziellen Integrität mit Hilfe *referenzieller Integritätsbedingungen*, die festlegen, dass alle Werte eines bestimmten Attributs oder einer Tabellenspalte auch in einer anderen Tabelle oder Spalte vorhanden sind. Eine referenzielle Integritätsbedingung kann zum Beispiel fordern, dass jeder Mitarbeiter in der Tabelle EMPLOYEE in einer Abteilung sein muss, die in der Tabelle DEPARTMENT aufgeführt ist. Kein Mitarbeiter kann in einer Abteilung sein, die nicht existiert.

Referenzielle Integritätsbedingungen können in einer Datenbank implementiert werden, um zu gewährleisten, dass die referenzielle Integrität erhalten bleibt, und um dem Optimierungsprogramm zu ermöglichen, durch Ausnutzung dieser besonderen Abhängigkeitsbeziehungen Abfragen effizienter zu verarbeiten. Bei der Planung im Hinblick auf referenzielle Integrität sollten Sie alle Beziehungen zwischen Datenbanktabellen angeben. Sie können eine Abhängigkeitsbeziehung durch Definition eines Primärschlüssels und referenzieller Integritätsbedingungen festlegen.

Betrachten Sie die folgenden, durch eine Beziehung verbundenen Tabellen:

Tabelle 20. Tabelle DEPARTMENT

DEPTNO (Primärschlüssel)	**DEPTNAME**	**MGRNO**
A00	Spiffy Computer Service Div.	000010
B01	Planning	000020
C01	Information Center	000030
D11	Manufacturing Systems	000060

Tabelle 21. Tabelle EMPLOYEE

EMPNO (Primär-schlüssel)	FIRSTNAME	LASTNAME	WORKDEPT (Fremd-schlüssel)	PHONENO
000010	Christine	Haas	A00	3978
000030	Sally	Kwan	C01	4738
000060	Irving	Stern	D11	6423
000120	Sean	O'Connell	A00	2167
000140	Heather	Nicholls	C01	1793
000170	Masatoshi	Yoshimura	D11	2890

Viele der folgenden Konzepte, die zum Verständnis der referenziellen Integrität hilfreich sind, werden mit Bezug auf diese Tabellen diskutiert.

Ein *eindeutiger Schlüssel* ist eine Spalte oder Spaltengruppe, in denen Werte einer Zeile in keiner anderen Zeile nochmals vorkommen. Für die Tabelle kann ein eindeutiger Schlüssel als Primärschlüssel definiert werden. Ein eindeutiger Schlüssel kann auch als *übergeordneter Schlüssel* bezeichnet werden, wenn sich ein Fremdschlüssel auf ihn bezieht.

Ein *Primärschlüssel* ist ein eindeutiger Schlüssel, der Teil der Definition der Tabelle ist. Jede Tabelle kann einen und nur einen Primärschlüssel besitzen. In den vorangegangenen Beispielen sind die Spalten DEPTNO und EMPNO Primärschlüssel der Tabellen DEPARTMENT und EMPLOYEE.

Ein *Fremdschlüssel* ist eine Spalte oder eine Spaltengruppe in einer Tabelle, die sich auf einen eindeutigen Schlüssel oder den Primärschlüssel derselben oder einer anderen Tabelle bezieht. Ein Fremdschlüssel wird zur Herstellung einer Abhängigkeitsbeziehung mit einem eindeutigen Schlüssel oder dem Primärschlüssel verwendet, um die referenzielle Integrität zwischen Tabellen zu gewährleisten. Die Spalte WORKDEPT in der Tabelle EMPLOYEE ist ein Fremdschlüssel, weil sie auf den Primärschlüssel, die Spalte DEPTNO, in der Tabelle DEPARTMENT bezogen ist.

Ein *zusammengesetzter Schlüssel* ist ein Schlüssel, der aus mehr als einer Spalte besteht. Sowohl Primär- als auch Fremdschlüssel können zusammengesetzte Schlüssel sein. Wenn zum Beispiel Abteilungen eindeutig durch eine Kombination aus Unternehmensbereichsnummer und Abteilungsnummer identifiziert würden, wären zwei Spalten erforderlich, um den Schlüssel für die Tabelle DEPARTMENT zu bilden.

Ein *übergeordneter Schüssel* ist ein Primärschlüssel oder eindeutiger Schlüssel einer referenziellen Integritätsbedingung. Als *Integritätsbedingung über Primärschlüssel* wird der übergeordnete Standardschlüssel der referenziellen Integritätsbedingung festgelegt, wenn keine Gruppe übergeordneter Schlüsselspalten angegeben wird.

Eine *übergeordnete Tabelle* ist eine Tabelle mit einem übergeordneten Schlüssel, auf den mindestens ein Fremdschlüssel in derselben oder einer anderen Tabelle verweist. Eine Tabelle kann in beliebig vielen Beziehungen eine übergeordnete Tabelle sein. Die Tabelle DEPARTMENT, die einen Primärschlüssel in der Spalte DEPTNO hat, ist zum Beispiel eine übergeordnete Tabelle zur Tabelle EMPLOYEE, die den Fremdschlüssel WORKDEPT enthält.

Eine *übergeordnete Zeile* ist eine Zeile einer übergeordneten Tabelle, deren Wert des übergeordneten Schlüssels mit mindestens einem Wert eines Fremdschlüssels in einer abhängigen Tabelle übereinstimmt. Eine Zeile in einer übergeordneten Tabelle ist nicht unbedingt eine übergeordnete Zeile. Die vierte Zeile (D11) der Tabelle DEPARTMENT ist die übergeordnete Zeile der dritten und der sechsten Zeile in der Tabelle EMPLOYEE. Die zweite Zeile (B01) der Tabelle DEPARTMENT ist jedoch keine übergeordnete Zeile zu irgendeiner anderen Zeile.

Eine *abhängige Tabelle* ist eine Tabelle, die einen oder mehrere Fremdschlüssel enthält. Eine abhängige Tabelle kann gleichzeitig auch eine übergeordnete Tabelle sein. Eine Tabelle kann in einer beliebigen Anzahl von Beziehungen eine abhängige Tabelle sein. Die Tabelle EMPLOYEE enthält den Fremdschlüssel WORKDEPT und ist von der Tabelle DEPARTMENT abhängig, deren Primärschlüssel die Spalte DEPTNO ist.

Eine *abhängige Zeile* ist eine Zeile einer abhängigen Tabelle, die einen Nichtnullwert im Fremdschlüssel enthält, der mit dem Wert eines übergeordneten Schlüssels übereinstimmt. Der Wert des Fremdschlüssels stellt einen Verweis der abhängigen Zeile auf die übergeordnete Zeile dar. Da Fremdschlüssel Nullwerte akzeptieren kann, ist eine Zeile einer abhängigen Tabelle nicht unbedingt auch eine abhängige Zeile.

Eine Tabelle ist eine *untergeordnete Tabelle* zu einer anderen Tabelle, wenn sie selbst eine abhängige Tabelle oder eine untergeordnete Tabelle zu einer abhängigen Tabelle ist. Eine untergeordnete Tabelle enthält einen Fremdschlüssel, der bis zu einem übergeordneten Schlüssel einer Tabelle zurückverfolgt werden kann.

Ein *Referenzzyklus* ist ein Pfad von Abhängigkeitsbeziehungen, durch den eine Tabelle mit sich selbst verknüpft wird. Wenn eine Tabelle direkt mit sich selbst verknüpft ist, handelt es sich um eine *auf sich selbst verweisende* Tabelle. Wenn die Tabelle EMPLOYEE eine weitere Spalte mit dem Namen MGRID enthielte, in der die EMPNO (Personalnummern) für jeden Manager eines Mitarbeiters aufgeführt wären, wäre die Tabelle EMPLOYEE eine auf sich selbst verweisende Tabelle. MGRID wäre ein Fremdschlüssel für die Tabelle EMPLOYEE.

Eine auf sich selbst verweisende Tabelle ist innerhalb derselben Beziehung sowohl übergeordnete als auch abhängige Tabelle. Eine auf sich selbst verweisende Zeile ist eine Zeile, die sowohl eine übergeordnete als auch eine abhängige Zeile zu sich selbst ist. Die Integritätsbedingung, die in diesem Fall vorhanden ist, wird als auf sich selbst verweisende Integritätsbedingung bezeichnet. Wenn beispielsweise der Wert für den Fremdschlüssel in einer Zeile einer auf sich selbst verweisenden Tabelle mit dem Wert des eindeutigen Schlüssels in dieser Zeile übereinstimmt, dann verweist die Zeile auf sich selbst.

Eine *referenzielle Integritätsbedingung* ist eine Festlegung, dass Nichtnullwerte eines bestimmten Fremdschlüssels nur gültig sind, wenn sie auch als Wert für einen eindeutigen Schlüssel einer bestimmten Tabelle vorkommen. Der Zweck referenzieller Integritätsbedingungen ist die Gewährleistung, dass Datenbankbeziehungen erhalten bleiben und Dateneingaberegeln befolgt werden.

Auswirkungen auf SQL-Operationen

Die Implementierung referenzieller Integritätsbedingungen hat besondere Auswirkungen für einige SQL-Operationen, bei denen unterschieden wird, ob es sich um übergeordnete oder abhängige Tabellen handelt. In diesem Abschnitt werden die Auswirkungen einer Wahrung der referenziellen Integrität auf SQL-Operationen mit den Anweisungen INSERT, DELETE, UPDATE und DROP behandelt.

DB2 sorgt *nicht* automatisch für die systemübergreifende Erhaltung referenzieller Integritätsbedingungen. Wenn also referenzielle Integritätsbedingungen systemübergreifend realisiert werden sollen, müssen die Anwendungen die erforderliche Logik enthalten.

Die folgenden Themen werden behandelt:

- „Regeln zur Anweisung INSERT"
- „Regeln zur Anweisung DELETE"
- „Regeln zur Anweisung UPDATE" auf Seite 97.

Regeln zur Anweisung INSERT: Sie können jederzeit eine Zeile in einer übergeordnete Tabelle einfügen, ohne dass eine Aktion in abhängigen Tabellen durchgeführt wird. In einer abhängigen Tabelle hingegen können Sie nur dann eine Zeile einfügen, wenn es eine Zeile in der übergeordneten Tabelle mit einem übergeordneten Schlüsselwert gibt, der mit dem Fremdschlüsselwert der einzufügenden Zeile übereinstimmt, sofern dieser Fremdschlüsselwert nicht NULL ist. Der Wert eines zusammengesetzten Fremdschlüssels ist NULL, wenn irgendeine Komponente des Werts NULL ist.

Diese Regel gilt implizit, wenn ein Fremdschlüssel definiert wird.

Bei dem Versuch, eine Zeile in einer Tabelle mit referenziellen Integritätsbedingungen einzufügen, ist die Operation INSERT nicht zulässig, wenn einer der Nichtnullwerte des Fremdschlüssels nicht im übergeordneten Schlüssel der übergeordneten Tabelle vorkommt. Wenn die Operation INSERT bei dem Versuch, mehrere Zeilen einzufügen, aufgrund einer Zeile fehlschlägt, wird keine der Zeilen eingefügt.

Regeln zur Anweisung DELETE: Wenn eine Zeile aus einer übergeordneten Tabelle gelöscht wird, prüft DB2, ob es abhängige Zeilen in abhängigen Tabellen mit übereinstimmenden Fremdschlüsselwerten gibt. Werden abhängige Zeilen gefunden, sind mehrere Aktionen möglich. Sie können bestimmen, welche Aktion ausgeführt wird, indem Sie bei der Erstellung der abhängigen Tabelle eine *Löschbedingung* angeben.

Die Löschbedingungen, die für eine abhängige Tabelle (die Tabelle, die den Fremdschlüssel enthält) für den Fall, dass ein Primärschlüssel gelöscht wird, angegeben werden können, sind folgende:

RESTRICT Diese Löschbedingung verhindert, dass eine Zeile in der übergeordneten Tabelle gelöscht wird, wenn eine oder mehrere abhängige Zeilen gefunden werden. Wenn sowohl die übergeordneten als auch die abhängigen Zeilen gelöscht werden sollen, müssen die abhängigen Zeilen zuerst gelöscht werden. Löschen zuerst der übergeordneten Zeilen verletzt die referenzielle Integritätsbedingung und ist nicht zulässig.

NO ACTION

Sichert das Vorhandensein einer übergeordneten Zeile für jede untergeordnete Zeile, nachdem alle referenziellen Integritätsbedingungen angewendet wurden.

CASCADE

Diese Löschbedingung legt fest, dass durch Löschen einer Zeile der übergeordneten Tabelle automatisch auch alle abhängigen Zeilen in der abhängigen Tabelle gelöscht werden. Diese Löschbedingung ist sinnvoll, wenn eine Zeile in der abhängigen Tabelle ohne die Zeile in der übergeordneten Tabelle keine Bedeutung mehr hätte.

Das Löschen der übergeordneten Zeile bewirkt zunächst, dass die abhängigen Zeilen, die auf einen Primärschlüssel verweisen, automatisch gelöscht werden. Daher ist es nicht nötig, die abhängigen Zeilen zuerst zu löschen. Wenn einige dieser abhängigen Zeilen selbst übergeordnete Zeilen zu abhängigen Zeilen sind, wird die Löschbedingung auf diese Abhängigkeitsbeziehungen ebenfalls angewendet. DB2 verwaltet gestaffelte Löschoperationen.

SET NULL

Diese Löschbedingung bewirkt, dass durch das Löschen einer Zeile der übergeordneten Tabelle die Werte des Fremdschlüssels in allen abhängigen Zeilen auf NULL gesetzt werden. Die anderen Spalten der Zeilen bleiben unverändert.

Wenn bei der Erstellung keine Löschbedingung explizit definiert wird, wird die Löschbedingung NO ACTION verwendet.

Jede Tabelle, die in eine Löschoperation mit einbezogen werden kann, wird als durch übergreifendes Löschen verknüpft bezeichnet. Die folgenden Einschränkungen gelten für Verknüpfungen durch übergreifendes Löschen.

- Eine Tabelle kann nicht durch übergreifendes Löschen mit sich selbst in einem Referenzzyklus mit mehr als einer Tabelle verknüpft sein.
- Wenn eine Tabelle durch übergreifendes Löschen mit einer anderen Tabelle über mehr als eine Abhängigkeitsbeziehung verknüpft ist, müssen für diese Abhängigkeitsbeziehungen dieselben Löschbedingungen, nämlich entweder CASCADE oder NO ACTION, gelten.
- Wenn eine auf sich selbst verweisende Tabelle von einer anderen Tabelle in einer CASCADE-Beziehung abhängig ist, muss für die auf sich selbst verweisende Beziehung auch die Löschbedingung CASCADE gelten.

In einer abhängigen Tabelle können zu jeder Zeit Zeilen gelöscht werden, ohne dass eine Aktion in der übergeordneten Tabelle erforderlich wird. Zum Beispiel könnte in der Abteilung-Mitarbeiter-Beziehung (DEPARTMENT-EMPLOYEE) ein Mitarbeiter in den Ruhestand gehen, so dass seine Zeile aus der Tabelle EMPLOYEE ohne Auswirkung auf die Tabelle DEPARTMENT gelöscht würde. (In der umgekehrten Mitarbeiter-Abteilung-Beziehung (EMPLOYEE-DEPARTMENT) ist die Abteilungsmanager-ID ein Fremdschlüssel mit Verweis auf den übergeordneten Schlüssel der Tabelle EMPLOYEE ist. Wenn ein Manager in den Ruhestand tritt, *hat* dies natürlich eine Auswirkung auf die Tabelle DEPARTMENT.)

Regeln zur Anweisung UPDATE: DB2 lässt nicht zu, dass ein eindeutiger Schlüssel einer übergeordneten Zeile aktualisiert wird. Wenn ein Fremdschlüssel in einer abhängigen Tabelle aktualisiert wird und der Fremdschlüssel nicht NULL ist, muss er mit einem Wert des übergeordneten Schlüssels der übergeordneten Tabelle der Abhängigkeitsbeziehung übereinstimmen. Wenn eine referenzielle Integritätsbedingung durch eine Operation UPDATE verletzt wird, tritt ein Fehler auf, und keine Zeile wird aktualisiert.

Für die Aktualisierung eines Werts in einer Spalte des übergeordneten Schlüssels gilt folgendes:

- Wenn irgendeine Zeile in der abhängigen Tabelle mit dem Ursprungswert des Schlüssels übereinstimmt, wird die UPDATE-Operation zurückgewiesen, wenn als Aktualisierungsregel RESTRICT definiert ist.
- Wenn irgendeine Zeile der abhängigen Tabelle nach Ausführung der UPDATE-Anweisung keinen entsprechenden übergeordneten Schlüssel hat (außer nach Auslösern), wird die Aktualisierung zurückgewiesen, wenn als Aktualisierungsregel NO ACTION definiert ist.

Zur Aktualisierung des Werts eines übergeordneten Schlüssels, der sich in einer übergeordneten Zeile befindet, müssen Sie zunächst die Beziehung zu allen abhängigen Zeilen in den abhängigen Tabellen durch eine der folgenden Operationen beseitigen:

- Löschen der abhängigen Zeilen
- Aktualisieren der Fremdschlüssel in den abhängigen Tabellen, um sie mit anderen gültigen Schlüsselwerten zu versehen

Wenn es keine Abhängigkeit zum Schlüsselwert in der Zeile mehr gibt, ist die Zeile keine übergeordnete Zeile einer referenziellen Abhängigkeitsbeziehung mehr und kann aktualisiert werden.

Wenn ein Teil eines Fremdschlüssels aktualisiert wird und kein Teil des Fremdschlüssels einen Nullwert (NULL) enthält, muss der neue Wert des Fremdschlüssels als eindeutiger Schlüsselwert in der übergeordneten Tabelle vorhanden sein. Wenn kein Fremdschlüssel von einem bestimmten eindeutigen Schlüssel abhängig ist, d. h., wenn die Zeile mit dem eindeutigen Schlüssel *keine* übergeordnete Zeile ist, kann ein Teil des eindeutigen Schlüssels aktualisiert werden. Jedoch kann in diesem Fall nicht mehr als eine Zeile gleichzeitig zur Aktualisierung ausgewählt werden, da es sich um einen eindeutigen Schlüssel handelt und mehrere gleiche Werte in verschiedenen Zeilen nicht zulässig sind.

Prüfungen auf Integritätsbedingungen in Tabellen

Geschäftsregeln, die in Ihrem Datenbankentwurf identifiziert wurden, können mit Hilfe von Prüfungen auf Integritätsbedingungen in Tabellen (Table Check Constraints) implementiert werden. *Prüfungen auf Integritätsbedingungen für Tabellen* geben Suchbedingungen an, die für jede Zeile einer Tabelle angewendet werden. Diese Integritätsbedingungen werden automatisch aktiviert, wenn eine Anweisung UPDATE oder INSERT für eine Tabelle ausgeführt wird. Sie werden durch die Anweisung CREATE TABLE oder ALTER TABLE definiert.

Eine *Prüfung auf Integritätsbedingung* kann zur Gültigkeitsprüfung der Daten verwendet werden. Zum Beispiel müssen Werte für eine Abteilungsnummer innerhalb des Bereichs von 10 bis 100 liegen, oder die Jobbezeichnung für einen Mitarbeiter (Employee) kann nur ″Verkauf″ (Sales), ″Manager″ oder ″Sachbearbeiter″ (Clerk) lauten, oder ein Mitarbeiter, der länger als acht Jahre in der Firma ist, muss ein Gehalt über 45.500 Dollar beziehen.

Informationen zu den Auswirkungen von Prüfungen auf Integritätsbedingungen in Tabellen auf die Befehle IMPORT und LOAD finden Sie im Handbuch *Versetzen von Daten Dienstprogramme und Referenz*.

Auslöser

Ein *Auslöser* (Trigger) ist eine definierte Gruppe von Aktionen, die immer dann ausgeführt werden, wenn eine DELETE-, INSERT- oder UPDATE-Operation für eine bestimmte Tabelle ausgeführt wird. Auslöser können zur Unterstützung von Geschäftsregeln definiert werden. Auslöser können außerdem zur automatischen Aktualisierung von Übersichts- oder Prüfdaten verwendet werden. Da ein Auslöser in der Datenbank gespeichert wird, müssen die durch den Auslöser ausgeführten Aktionen nicht mehr in jedem Anwendungsprogramm codiert werden. Der Auslöser wird einmal codiert, in der Datenbank gespeichert und automatisch durch DB2 je nach Bedarf aufgerufen, wenn eine Anwendung auf die Datenbank zugreift. Dadurch ist gewährleistet, dass die auf die Daten bezogenen Geschäftsregeln zu jeder Zeit in Kraft bleiben. Wenn eine Geschäftsregel geändert wird, müssen nur die entsprechenden Auslöser angepasst werden.

Innerhalb einer ausgelösten SQL-Anweisung kann eine benutzerdefinierte Funktion (UDF) aufgerufen werden. Dadurch ist es möglich, dass die ausgelöste Aktion eine Nicht-SQL-Operation durchführen kann, wenn der Auslöser aktiviert wird. Es kann beispielsweise eine E-Mail-Nachricht als Alert-Mechanismus verschickt werden. Weitere Informationen zu Auslösern (Triggers) finden Sie in "Erstellen eines Auslösers" in Handbuch *Systemverwaltung: Implementierung* und im Handbuch *Application Development Guide*.

Weitere Überlegungen zum Datenbankentwurf

Beim Entwurf einer Datenbank muss vorausgeplant werden, auf welche Tabellen Benutzer Zugriff haben sollten. Der Zugriff auf Tabellen wird mit Hilfe von Berechtigungen erteilt oder entzogen. Die höchste Ebene der Berechtigung ist die Berechtigung zur Systemverwaltung (SYSADM). Ein Benutzer mit der Berechtigung SYSADM kann andere Berechtigungen, einschließlich der Berechtigung für den Datenbankadministrator (DBADM), erteilen.

Darüber hinaus gibt es weitere Gesichtspunkte, die im Datenbankentwurf berücksichtigt werden müssen, wie zum Beispiel *Prüfaktivitäten*, *Protokolldaten*, *Übersichtstabellen*, *Sicherheit*, *Datentypisierung*, und *Möglichkeiten zur Parallelverarbeitung*.

Zur Erfassung von *Prüfdaten* muss eventuell jede Aktualisierung an den Daten über einen bestimmten Zeitraum hinweg aufgezeichnet werden. Sie können zum Beispiel eine Prüfdatentabelle jedes Mal dann aktualisieren, wenn das Gehalt eines Mitarbeiters geändert wird. Aktualisierungen dieser Tabelle könnten automatisch durchgeführt werden, wenn ein entsprechender Auslöser (Trigger) definiert wird. Prüfaktivitäten können außerdem über die DB2-Prüffunktion ausgeführt werden. Weitere Informationen finden Sie in "Protokollieren von DB2-Aktivitäten" im Handbuch *Systemverwaltung: Implementierung*.

Aus Gründen der Leistung kann es sinnvoll sein, nur auf eine ausgewählte Menge von Daten zuzugreifen, während die Basisdaten als *Protokoll (History)* verwaltet werden. Sie sollten in Ihren Datenbankentwurf die Anforderungen zur Verwaltung solcher Protokolldaten aufnehmen, z. B. die Anzahl der Monate oder Jahre, die Daten verfügbar sein müssen, bevor sie gelöscht werden können.

Es kann außerdem sinnvoll sein, *Übersichtsdaten* zu erstellen. Zum Beispiel könnten Sie über eine Tabelle verfügen, die Ihre gesamten Mitarbeiterdaten enthält. Angenommen, Sie möchten diese Daten nach Unternehmensbereichen oder Abteilungen in separate Tabellen aufteilen. In diesem Fall wäre es hilfreich, Übersichtstabellen für die einzelnen Unternehmensbereiche oder Abteilungen anzulegen, die auf den Daten der Originaltabelle basieren. Weitere Informationen zu Übersichtstabellen finden Sie in "Erstellen einer Übersichtstabelle" im Handbuch *Systemverwaltung: Implementierung*.

In Ihrem Datenbankentwurf sollten auch die *Sicherheitsaspekte* berücksichtigt werden. Sie können zum Beispiel den Entwurf so gestalten, dass der Zugriff von Benutzern auf bestimmte Arten von Daten über Sicherheitstabellen unterstützt wird. Sie können Zugriffsebenen für verschiedene Arten von Daten und verschiedene Benutzer mit Zugriffberechtigung für diese Daten definieren. Vertrauliche Daten, wie Personal- und Gehaltsdaten, könnten strikten Sicherheitsbeschränkungen unterliegen. Weitere Informationen zur Sicherheit und zu Berechtigungen finden Sie in "Steuern des Datenbankzugriffs" im Handbuch *Systemverwaltung: Implementierung*.

Sie können Tabellen erstellen, denen ein *strukturierter Typ* zugeordnet ist. Mit solchen typisierten Tabellen können Sie eine hierarchische Struktur mit einer definierten Abhängigkeitsbeziehung zwischen diesen Tabellen erstellen, die als *Typhierachie* bezeichnet wird. Eine Typhierarchie besteht aus einem Stammtyp sowie aus übergeordneten und untergeordneten Datentypen.

Die Darstellung eines *Referenztyps* wird beim Erstellen des Stammtyps einer Typhierarchie definiert. Das Ziel einer Referenz ist immer eine Zeile in einer typisierten Tabelle oder Sicht.

Weitere Informationen zur Implementierung eines Datenbankentwurfs mit typisierten Zeilen und Tabellen finden Sie in "Implementieren des Datenbankentwurfs" im Handbuch *Systemverwaltung: Implementierung*. Das Handbuch *Versetzen von Daten Dienstprogramme und Referenz* enthält Informationen zum Versetzen von Daten zwischen typisierten Tabellen in einer hierarchischen Struktur.

Mit wachsendem Geschäftsumfang können die zusätzlichen Kapazitäten und Leistungsvorteile von DB2 Enterprise - Extended Edition erforderlich werden. In dieser Umgebung sind die Partitionen Ihrer Datenbank über verschiedene Maschinen bzw. Systeme verteilt, wobei jedes für das Speichern und Abrufen nur eines Teils der Gesamtdatenbank zuständig ist. Jede Partition (oder Knoten) arbeitet bei der Verarbeitung von SQL- oder Dienstprogrammoperationen parallel.

Themen und Überlegungen zu parallelen Operationen werden an den entsprechenden Stellen im gesamten Handbuch berücksichtigt.

Kapitel 8. Entwerfen der physischen Datenbank

Nachdem Sie den logischen Datenbankentwurf entwickelt haben (siehe „Kapitel 7. Entwerfen des logischen Datenbankaufbaus" auf Seite 71), gibt es eine Reihe von Themen, die hinsichtlich der physischen Umgebung, in der die Datenbank und die Tabellen angelegt werden, zu berücksichtigen sind. Hierzu gehören Kenntnisse bezüglich der Dateien, die für die Unterstützung und Verwaltung Ihrer Datenbank erstellt werden, eine Vorstellung davon, wie viel Speicherbereich zum Speichern Ihrer Daten erforderlich ist, und die Festlegung, wie die zum Speichern Ihrer Daten benötigten Tabellenbereiche verwendet werden sollen.

Die folgenden Themen werden behandelt:

- „Datenbankverzeichnisse"
- „Ermitteln des Platzbedarfs für Tabellen" auf Seite 106
- „Zusätzlicher Platzbedarf" auf Seite 114
- „Entwerfen von Knotengruppen" auf Seite 116
- „Entwerfen und Auswählen von Tabellenbereichen" auf Seite 125
- „Überlegungen zum Entwerfen einer zusammengeschlossenen Datenbank" auf Seite 151

Datenbankverzeichnisse

Beim Erstellen einer Datenbank erstellt DB2 ein separates Unterverzeichnis, um Steuerdateien (z. B. Protokollkopfdateien) zu speichern und um Standardtabellenbereichen Behälter zuzuordnen. Objekte, die der Datenbank zugeordnet sind, werden nicht immer im Datenbankverzeichnis gespeichert. Sie können an unterschiedlichen Positionen gespeichert werden, zum Beispiel auf Einheiten.

Die Datenbank wird in dem Exemplar erstellt, das durch die Umgebungsvariable DB2INSTANCE definiert ist, oder in dem Exemplar, zu dem Sie explizit eine Verbindung hergestellt haben (mit dem Befehl ATTACH). Eine Einführung in Exemplare finden Sie in "Verwenden mehrerer Exemplare des Datenbankmanagers" im Handbuch *Systemverwaltung: Implementierung*.

Das auf UNIX-Plattformen verwendete Benennungsschema sieht folgendermaßen aus:

```
angegebener-pfad/$DB2INSTANCE/NODEnnnn/SQL00001
```

Das unter den Betriebssystemen OS/2 und Windows verwendete Benennungsschema ist folgendes:

```
D:\$DB2INSTANCE\NODEnnnn\SQL00001
```

Dabei gilt folgendes:

- `angegebener-pfad` ist die wahlfreie, benutzerdefinierte Speicherposition zum Installieren des Exemplars.
- `NODEnnnn` ist die Knotenkennung in einer partitionierten Datenbankumgebung. Der erste Knoten ist `NODE0000`.
- "D:" ist ein "Laufwerkbuchstabe", der den Datenträger angibt, auf dem das Stammverzeichnis angelegt ist.

`SQL00001` enthält Objekte, die der ersten erstellten Datenbank zugeordnet sind. Nachfolgend erstellte Datenbanken erhalten höhere Nummern: `SQL00002` usw.

Die Unterverzeichnisse werden in einem Verzeichnis erstellt, das denselben Namen trägt wie das Exemplar des Datenbankmanagers, mit dem Sie bei der Erstellung der Datenbank (durch ATTACH) verbunden sind. (In den Betriebssystemen OS/2 und Windows werden die Unterverzeichnisse unter dem Stammverzeichnis eines bestimmten Datenträgers erstellt, der durch den „Laufwerkbuchstaben" identifiziert wird.) Diese Exemplar- und Datenbankunterverzeichnisse werden innerhalb des Pfads erstellt, der mit dem Befehl CREATE DATABASE angegeben wurde, und ihre Verwaltung erfolgt automatisch durch den Datenbankmanager. Je nach Plattform kann jedes Exemplar einem Exemplareigner gehören, der die Berechtigung des Systemadministrators (SYSADM) über die zu diesem Exemplar gehörenden Datenbanken hat.

Sie sollten, um potenzielle Probleme zu vermeiden, keine Verzeichnisse mit demselben Benennungsschema erstellen und Verzeichnisse, die bereits vom Datenbankmanager erstellt wurden, nicht manuell ändern.

Datenbankdateien

Die folgenden Dateien gehören zu einer Datenbank:

Dateiname	Beschreibung
SQLDBCON	In dieser Datei sind die Optimierungsparameter und Markierungen (Flags) für die Datenbank gespeichert. Informationen zur Änderung von Konfigurationsparametern der Datenbank finden Sie im Handbuch *Systemverwaltung: Optimierung*.
SQLOGCTL.LFH	Diese Datei dient dem Zweck, alle Datenbankprotokolldateien zu verfolgen und zu steuern.

Syyyyyyy.LOG

Die Datenbankprotokolldateien werden von 0000000 bis 9999999 durchnummeriert. Die Anzahl dieser Dateien wird durch die Konfigurationsparameter *logprimary* und *logsecond* der Datenbank gesteuert. Die Größe der einzelnen Dateien wird durch den Konfigurationsparameter *logfilsiz* der Datenbank bestimmt.

Bei der Umlaufprotokollierung werden die Dateien erneut verwendet, und die Nummern werden beibehalten. Bei der Archivprotokollierung erhöht sich die Zahl der Dateien in Folge, wenn Protokolle archiviert und neue Protokolle angelegt werden. Wenn 9999999 erreicht wird, beginnt die Nummerierung von vorn.

Standardmäßig werden die Protokolldateien im Verzeichnis SQLOGDIR gespeichert. Das Verzeichnis SQLOGDIR befindet sich im Unterverzeichnis SQL*nnnnn*.

SQLINSLK Diese Datei hilft bei der Sicherstellung, dass eine Datenbank nur von einem Exemplar des Datenbankmanagers verwendet wird.

SQLTMPLK Diese Datei hilft bei der Sicherstellung, dass eine Datenbank nur von einem Exemplar des Datenbankmanagers verwendet wird.

SQLSPCS.1 Diese Datei enthält die Definition und den aktuellen Status aller Tabellenbereiche innerhalb der Datenbank.

SQLSPCS.2 Diese Datei ist eine Sicherungskopie von `SQLSPCS.1`. Ohne eine dieser beiden Dateien ist kein Zugriff auf die Datenbank möglich.

SQLBP.1 Diese Datei enthält die Definition aller Pufferpools, die in der Datenbank verwendet werden.

SQLBP.2 Diese Datei ist eine Sicherungskopie von `SQLBP.1`. Ohne eine dieser beiden Dateien ist kein Zugriff auf die Datenbank möglich.

DB2RHIST.ASC

Diese Datei ist die Protokolldatei der Datenbank. In ihr werden Verwaltungsoperationen an der Datenbank protokolliert, z. B. Sicherungs- und Wiederherstellungsoperationen.

DB2RHIST.BAK

Diese Datei ist eine Sicherungskopie von `DB2RHIST.ASC`.

Anmerkungen:

1. Nehmen Sie *auf keinen Fall* direkte Änderungen an diesen Dateien vor. Der Zugriff auf diese Dateien kann nur indirekt über die dokumentierten APIs und mit Tools erfolgen, die diese APIs implementieren (einschließlich des Befehlszeilenprozessors und der Steuerzentrale).
2. Verschieben Sie diese Dateien nicht.
3. Löschen Sie diese Dateien nicht.
4. Das Sichern einer Datenbank oder eines Tabellenbereichs über die API **sqlubkp** (Backup Database), einschließlich der Implementierungen dieser API durch den Befehlszeilenprozessor und die Steuerzentrale, ist die einzige Sicherungsmethode, die unterstützt wird.

Ermitteln des Platzbedarfs für Tabellen

Das Abschätzen der Größe von Datenbankobjekten ist ein ungenaues Unterfangen. Der Systemaufwand, der durch Datenträgerfragmentierung, freien Speicherbereich und die Verwendung von Spalten variabler Länge bedingt ist, macht eine Abschätzung schwierig, weil es eine große Bandbreite an Möglichkeiten für Spatlentypen und Zeilenlängen gibt. Erstellen Sie nach dem Abschätzen der Größe Ihrer Datenbank eine Testdatenbank, und füllen Sie sie mit repräsentativen Daten.

Über die Steuerzentrale können Sie auf eine Reihe von Dienstprogrammen zugreifen, die Sie bei der Ermittlung der Größenanforderungen verschiedener Datenbankobjekte unterstützen:

- Sie können ein Objekt auswählen und das Dienstprogramm "Größe schätzen" verwenden. Mit diesem Dienstprogramm können Sie die aktuelle Größe eines vorhandenen Objekts, wie z. B. einer Tabelle, ermitteln. Sie können dann das Objekt ändern, und das Dienstprogramm berechnet neue Schätzwerte für das Objekt. Dieses Dienstprogramm hilft bei der Abschätzung des ungefähren Speicherbedarfs unter Berücksichtigung zukünfiger Zuwächse. Es bietet mehr als einen einzelnen Schätzwert für die Größe des Objekts. Es gibt auch mögliche Größenbereiche für das Objekt an: die minimale Größe, die auf den aktuellen Werten beruht, und die mögliche Maximalgröße.
- Sie können die Abhängigkeiten zwischen Objekten bestimmen, indem Sie das Fenster "Zugehörige anzeigen" verwenden.
- Sie können ein beliebiges Datenbankobjekt im Exemplar auswählen und "DDL generieren" anfordern. Diese Funktion verwendet das Dienstprogramm **db2look**, um Datendefinitionsanweisungen für die Datenbank zu generieren. Informationen zu diesem Dienstprogramm enthält das Handbuch *Command Reference*.

In allen diesen Fällen steht Ihnen entweder der Knopf "SQL anzeigen" oder der Knopf "Befehl anzeigen" zur Verfügung. Sie können die generierten SQL-Anweisungen oder Befehle auch in Prozedurdateien sichern, um diese später zu verwenden. In allen diesen Dienstprogrammen werden Sie durch eine Online-Hilfefunktion unterstützt.

Sie sollten diese Einrichtungen bei der Planung Ihrer physischen Datenbankanforderungen berücksichtigen.

Bei der Abschätzung der Größe einer Datenbank müssen die Auswirkungen der folgenden Elemente berücksichtigt werden:

- „Systemkatalogtabellen"
- „Benutzertabellendaten" auf Seite 108
- „Langfelddaten" auf Seite 110
- „Daten großer Objekte (LOB-Daten)" auf Seite 110
- „Indexbereich" auf Seite 111

Der Platzbedarf folgender Elemente wird nicht behandelt:

- Datei für das lokale Datenbankverzeichnis
- Datei für das Systemdatenbankverzeichnis
- Bei der Dateiverwaltung entstehender Systemaufwand, den das Betriebssystem benötigt, zum Beispiel:
  - Dateiblockgröße
  - Speicherbereich für die Verzeichnissteuerung

Systemkatalogtabellen

Systemkatalogtabellen werden erstellt, wenn eine Datenbank erstellt wird. Der Umfang dieser Systemtabellen nimmt in dem Maße zu, wie der Datenbank Datenbankobjekte und Zugriffsrechte hinzugefügt werden. Zu Beginn belegen diese Tabellen einen Plattenspeicherplatz von ungefähr 3,5 MB.

Die Größe des Speicherbereichs, der für die Katalogtabellen zugeordnet wird, hängt von der Art des Tabellenbereichs und der Größe des durch EXTENTSIZE definierten Bereichs des Tabellenbereichs mit den Katalogtabellen ab. Wenn z. B. ein DMS-Tabellenbereich mit einem EXTENTSIZE-Wert von 32 verwendet wird, wird dem Katalogtabellenbereich anfangs ein Speicherbereich von 20 MB zugeordnet. Weitere Informationen finden Sie in „Entwerfen und Auswählen von Tabellenbereichen" auf Seite 125.

Anmerkung: Für Datenbanken mit mehreren Partitionen befinden sich die Katalogtabellen nur in der Partition, von der aus der Befehl CREATE DATABASE abgesetzt wurde. Plattenspeicherplatz für die Katalogtabellen ist nur für diese Partition erforderlich.

Benutzertabellendaten

Standardmäßig werden die Tabellendaten auf Seiten von jeweils 4 KB gespeichert. Jede Seite enthält (unabhängig von der Seitengröße) 76 Byte Systemaufwand für den Datenbankmanager. Dadurch bleiben 4020 Byte für die Benutzerdaten (oder Zeilen), obwohl keine Zeile auf einer 4-KB-Seite die Länge von 4005 Byte überschreiten kann. Eine Zeile kann sich *nicht* über mehrere Seiten erstrecken. Wenn Sie Seiten mit jeweils 4 KB verwenden, sind maximal 500 Spalten möglich.

Tabellendatenseiten enthalten *keine* Daten für Spalten mit den Datentypen LONG VARCHAR, LONG VARGRAPHIC, BLOB, CLOB oder DBCLOB. Die Zeilen einer Tabellendatenseite enthalten jedoch einen Deskriptor für diese Spalten. (Informationen zur Ermittlung des Platzbedarfs für die Tabellenobjekte, die diese Datentypen enthalten, finden Sie in den Abschnitten „Langfelddaten" auf Seite 110 und „Daten großer Objekte (LOB-Daten)" auf Seite 110.)

Zeilen werden normalerweise an der ersten passenden Stelle in einer Tabelle eingefügt. Die Datei wird nach dem ersten verfügbaren Speicherbereich durchsucht (unter Verwendung einer Liste der freien Speicherbereiche), der groß genug ist, um die neue Zeile aufzunehmen. Das Aktualisieren einer vorhandenen Zeile erfolgt an der ursprünglichen Stelle, es sei denn, der freie Bereich der 4-KB-Seite reicht nicht aus, um die aktualisierte Zeile aufzunehmen. In diesem Fall wird an der Position, an der sich die Originalzeile befand, ein Datensatz erstellt, der auf die neue Speicherposition der aktualisierten Zeile in der Tabellendatei verweist.

Bei Verwendung der Anweisung ALTER TABLE APPEND ON werden Daten immer angehängt und es werden keine Informationen zum freien Speicherbereich der Datenseiten aufgezeichnet. Weitere Informationen zu dieser Anweisung finden Sie im Handbuch *SQL Reference*.

Die Anzahl der 4-KB-Seiten für jede Benutzertabelle in der Datenbank kann nach folgender Berechnung abgeschätzt werden:

```
ABRUNDEN(4020/(durchschnittszeilengröße + 10)) = datensätze_pro_seite
```

Das Ergebnis wird in folgende Berechnung eingefügt:

```
(zahl_der_datensätze/datensätze_pro_seite) * 1,1 = anzahl_der_seiten
```

Dabei ist die durchschnittliche Zeilenlänge die Summe der durchschnittlichen Spaltengrößen (Informationen über die Größe jeder Spalte finden Sie in der Beschreibung der Anweisung CREATE TABLE im Handbuch *SQL Reference*.), und der Faktor ″1,1″ dient zur Kalkulation des Systemaufwands.

Anmerkung: Diese Formel ergibt nur einen Schätzwert. Die Genauigkeit des Schätzwerts nimmt ab, wenn die Datensatzlänge durch Fragmentierung und Überlaufsätze variiert.

Sie können auch Pufferpools oder Tabellenbereiche mit einer Seitengröße von 8 KB, 16 KB oder 32 KB erstellen. Alle Tabellen, die innerhalb eines Tabellenbereichs einer bestimmten Größe erstellt wurden, besitzen eine übereinstimmende Seitengröße. Die maximale Größe für ein einzelnes Tabellen- oder Indexobjekt beträgt 512 GB bei einer einer Seitengröße von 32 KB. Wenn Sie Seiten mit einer Größe von 8 KB, 16 KB oder 32 KB verwenden, sind maximal 1012 Spalten möglich. Für eine 4-KB-Seite beträgt die maximale Spaltenanzahl 500. Die maximalen Zeilenlängen variieren ebenfalls je nach Seitengröße:

- Bei einer Seitengröße von 4 KB beträgt die maximale Zeilenlänge 4005 Byte.
- Bei einer Seitengröße von 8 KB beträgt die maximale Zeilenlänge 8101 Byte.
- Bei einer Seitengröße von 16 KB beträgt die maximale Zeilenlänge 16 293 Byte.
- Bei einer Seitengröße von 32 KB beträgt die maximale Zeilenlänge 32 677 Byte.

Durch größere Seiten wird es möglich, die Zahl der Indexstufen in einem Index zu verringern. Wenn Sie mit OLTP-Anwendungen arbeiten, die wahlfreie Lese- und Schreibzugriffe durchführen, ist eine geringere Seitengröße empfehlenswert, weil dadurch weniger Pufferspeicher durch unerwünschte Zeilen belegt wird. Wenn Sie mit DSS-Anwendungen (Decision Support System) arbeiten, die jeweils auf eine große Anzahl aufeinander folgender Zeilen zugreifen, sind größere Seiten empfehlenswert, weil dadurch weniger E/A-Anforderungen erforderlich sind, um eine bestimmte Anzahl Zeilen zu lesen. Eine Ausnahme bildet der Fall, wenn die Zeilengröße kleiner ist als die Seitengröße dividiert durch 255. In diesem Fall wird auf jeder Seite ungenutzter Speicherbereich verschenkt. (Jede Seite kann maximal nur 255 Zeilen aufnehmen.) Zur Verringerung des unbenutzten Speicherbereichs kann eine geringere Seitengröße besser geeignet sein.

Eine Sicherung kann nicht in einer anderen Seitengröße wiederherstellt werden.

Sie können keine IXF-Datendateien importieren, die mehr als 755 Spalten beinhalten. Weitere Informationen zum Importieren von Daten in Tabellen und IXF-Datendateien finden Sie im Handbuch *Versetzen von Daten Dienstprogramme und Referenz*.

Deklarierte temporäre Tabellen können nur in einem eigenen Tabellenbereichstyp für temporäre Benutzertabellen erstellt werden. Es gibt keinen Standardtabellenbereich für temporäre Benutzertabellen. Temporäre Tabellen können keine Daten mit LONG-Typen enthalten. Die Tabellen werden implizit gelöscht, wenn eine Anwendung die Verbindung zur Datenbank trennt. Schätzungen über den entsprechenden Platzbedarf sollten dies berücksichtigen.

Langfelddaten

Langfelddaten werden in einem separaten Tabellenobjekt gespeichert, das anders als bei anderen Datentypen strukturiert ist (siehe „Benutzertabellendaten" auf Seite 108 und „Daten großer Objekte (LOB-Daten)").

Die Daten werden in Bereichen von 32 KB gespeichert, die sich aus Segmenten zusammensetzen, deren Größen sich aus dem Produkt der Zweierpotenzen und 512 Byte errechnen. (Diese Segmente können also aus 512 Byte, 1024 Byte, 2048 Byte usw. bis 32 768 Byte bestehen.)

Langfelddatentypen (LONG VARCHAR oder LONG VARGRAPHIC) werden auf eine Weise gespeichert, die eine problemlose Neuverwendung freien Speicherbereichs ermöglicht. Die Informationen zur Zuordnung und zu freien Speicherbereichen werden in Zuordnungsseiten von jeweils 4 KB gespeichert, die gelegentlich im Objekt erscheinen.

Der Bereich des freien Speicherplatzes in den Objekten ist von der Größe der Langfelddaten sowie davon abhängig, ob diese Größe bei jedem Vorkommen der Daten relativ konstant ist. Bei Dateneinträgen von mehr als 255 Byte kann dieser nicht genutzte Speicherplatz bis zu 50 Prozent der Größe der Langfelddaten ausmachen.

Wenn Zeichendaten kleiner als die Seitengröße sind und sie zusammen mit dem Rest der Daten in den Datensatz passen, sollten anstelle der Datentypen LONG VARCHAR oder LONG VARGRAPHIC die Datentypen CHAR, GRAPHIC, VARCHAR oder VARGRAPHIC verwendet werden.

Daten großer Objekte (LOB-Daten)

Daten großer Objekte (LOB-Daten) werden in zwei separaten Tabellenobjekten gespeichert, die anders als bei anderen Datentypen strukturiert sind (siehe „Benutzertabellendaten" auf Seite 108 und „Langfelddaten").

Bei der Abschätzung des für LOB-Daten erforderlichen Speicherbereichs müssen Sie die beiden Tabellenobjekte berücksichtigen, die zum Speichern von Daten dieser Datentypen verwendet werden:

- **LOB-Datenobjekte**

  Die Daten werden in Bereichen von 64 MB gespeichert, die sich aus Segmenten zusammensetzen, deren Größen sich aus dem Produkt der Zweier-

potenzen und 1024 Byte errechnen. (Diese Segmente können also aus 1024 Byte, 2048 Byte, 4096 Byte usw. bis 64 MB bestehen.)

Zur Verringerung des für LOB-Daten erforderlichen Plattenspeicherplatzes können Sie den Parameter COMPACT in der Klausel für die *LOB-Optionen* der Anweisungen CREATE TABLE und ALTER TABLE angeben. Die Option COMPACT minimiert die Größe des für die LOB-Daten erforderlichen Speicherplatzes dadurch, dass die LOB-Daten in kleinere Segmente aufgeteilt werden können. Dieser Prozess beinhaltet keine Datenkomprimierung, sondern verwendet lediglich den kleinstmöglichen Speicherbereich bis zur nächsten 1-KB-Grenze. Die Angabe der Option COMPACT kann zu einer geringeren Leistung führen, wenn Daten an LOB-Werte angehängt werden.

Der Umfang des freien Speicherbereichs innerhalb der LOB-Datenobjekte wird vom Umfang der Aktualisierungs- und Löschaktivitäten sowie von der Größe der eingefügten LOB-Werte beeinflusst.

- **LOB-Zuordnungsobjekte**

  Die Informationen zur Zuordnung und zu freien Speicherbereichen werden in Zuordnungsseiten von jeweils 4 KB gespeichert, die von den eigentlichen Daten getrennt sind. Die Anzahl dieser 4-KB-Seiten ist vom Umfang der Daten (einschließlich des nicht genutzten Speicherbereichs) abhängig, die für die LOB-Daten zugeordnet wurden. Der Systemaufwand errechnet sich wie folgt: eine 4-KB-Seite pro 64 GB plus eine 4-KB-Seite pro 8 MB.

Wenn Zeichendaten kleiner als die Seitengröße sind und sie zusammen mit dem Rest der Daten in den Datensatz passen, sollten anstelle der Datentypen BLOB, CLOB oder DBCLOB die Datentypen CHAR, GRAPHIC, VARCHAR oder VARGRAPHIC verwendet werden.

Indexbereich

Für jeden Index kann der erforderliche Speicherbereich wie folgt abgeschätzt werden:

```
(Durchschnittliche Indexschlüsselgröße + 8) * Anzahl der Zeilen * 2
```

Dabei gilt folgendes:

- Die „durchschnittliche Indexschlüsselgröße" ist die Bytezahl jeder Spalte im Indexschlüssel. Im Abschnitt zur Anweisung CREATE TABLE im Handbuch *SQL Reference* finden Sie Informationen zum Berechnen der Bytezahl für Spalten mit unterschiedlichen Datentypen. (Verwenden Sie bei der Abschätzung der durchschnittlichen Spaltengröße für VARCHAR- und VARGRAPHIC-Spalten die aktuelle Datengröße plus ein Byte. Verwenden Sie nicht die deklarierte Maximalgröße.)
- Der Faktor 2 ist für den Systemaufwand, z. B. für Nichtblattseiten und freien Speicherbereich.

Anmerkung: Fügen Sie für jede Spalte, die einen Nullwert (NULL) zulässt, ein zusätzliches Byte für den Nullanzeiger hinzu.

Bei der Indexerstellung ist temporärer Speicherplatz erforderlich. Der maximal während der Indexerstellung erforderliche temporäre Speicherplatz kann folgendermaßen abgeschätzt werden:

```
(Durchschnittliche Indexschlüsselgröße + 8) * Anzahl der Zeilen * 3,2
```

Dabei wird der Faktor 3,2 für den Indexaufwand sowie für den Speicherbereich einkalkuliert, der für während der Indexerstellung anfallende Sortierungen erforderlich ist.

Anmerkung: Für nichteindeutige Indizes werden zum Speichern doppelter Schlüsseleinträge nur vier Byte benötigt. Die oben gezeigten Berechnungen gehen von eindeutigen Indizes aus. Der zum Speichern eines Index erforderliche Speicherbereich kann durch die oben gezeigte Formel eventuell zu groß abgeschätzt werden.

Die beiden folgenden Formeln können zum Abschätzen der Anzahl von Blattseiten eines Index verwendet werden. (Die zweite Formel liefert einen etwas genaueren Schätzwert.) Die Genauigkeit dieser Schätzwerte hängt im Wesentlichen davon ab, wie gut die verwendeten Durchschnittswerte die tatsächlichen Daten widerspiegeln.

Anmerkung: Für SMS-Tabellenbereiche beträgt der minimale Speicherbedarf 12 KB. Für DMS-Tabellenbereiche entspricht der minimale Speicherbedarf dem Wert von EXTENTSIZE.

- Die Durchschnittszahl von Schlüsseln pro Blattseite kann mit folgender Formel grob abgeschätzt werden:

  ```
  (0,9 * (U - (M*2))) * (D + 1)
  ----------------------------
        K + 6 + (4 * D)
  ```

  Dabei gilt folgendes:

  – U, der verwendbare Speicherbereich auf einer Seite, entspricht ungefähr der Seitengröße minus 100. Bei einer Seitengröße von 4096 ist U gleich 3996.
  – M = U / (8 + *minimale Schlüsselgröße*)
  – D = durchschnittliche Anzahl mehrfacher Werte pro Schlüsselwert
  – K = *durchschnittliche Schlüsselgröße*

  Beachten Sie, dass die Werte für *minimale Schlüsselgröße* und *durchschnittliche Schlüsselgröße* ein Byte extra für jeden Teil des Schlüssels haben müssen, der einen Nullwert enthalten kann, und ein zusätzliches Byte für die Länge jedes Teils des Schlüssels mit variabler Länge.

Wenn INCLUDE-Spalten vorhanden sind, müssen sie in den Werten für *minimale Schlüsselgröße* und *durchschnittliche Schlüsselgröße* berücksichtigt werden.

0,9 kann durch einen beliebigen, mit (100 - pctfree)/100 berechneten Wert ersetzt werden, wenn während der Indexerstellung ein anderer Prozentwert für freien Speicherbereich (pctfree) angegeben wurde als der Standardwert zehn.

- Die Durchschnittszahl von Schlüsseln pro Blattseite kann mit folgender Formel etwas genauer abgeschätzt werden:

  ```
  L = Blattseitenzahl = X / (Durchschnittszahl von Schlüsseln pro Blattseite)
  ```

  Dabei ist X die Gesamtzahl der Zeilen in der Tabelle.

  Einen Schätzwert für die Originalgröße eines Indexes können Sie wie folgt berechnen:

  ```
  L + 2L/(Durchschnittszahl von Schlüsseln pro Blattseite)) * Seitengröße
  ```

  Für DMS-Tabellenbereiche addieren Sie die Größen aller Indizes einer Tabelle zusammen, und runden Sie auf ein Vielfaches des Werts für EXTENTSIZE für den Tabellenbereich auf, in dem sich der Index befindet.

  Sie sollten weiteren Speicherbereich für das Anwachsen des Index durch INSERT/UPDATE-Vorgänge bereitstellen, die zur Teilung von Seiten führen können.

  Durch die folgenden Berechnungen erhalten Sie einen genaueren Schätzwert für die Originalgröße des Indexes sowie einen Schätzwert für die Anzahl der Indexstufen. (Dies ist möglicherweise von besonderem Interesse, wenn in der Indexdefinition INCLUDE-Spalten verwendet werden.) Die Durchschnittszahl von Schlüsseln pro Nichtblattseite kann mit folgender Formel grob abgeschätzt werden:

  ```
  (0,9 * (U - (M*2))) * (D + 1)
  ----------------------------
        K + 12 + (8 * D)
  ```

  Dabei gilt folgendes:

  – U, der verwendbare Speicherbereich auf einer Seite, entspricht ungefähr der Seitengröße minus 100. Bei einer Seitengröße von 4096 ist U gleich 3996.
  – D ist die durchschnittliche Anzahl mehrfacher Werte pro Schlüsselwert auf Nichtblattseiten (dieser Wert ist deutlich kleiner als für Blattseiten. Sie können ihn zur Vereinfachung der Berechnung auf 0 setzen).
  – M = U / (8 + *minimale Schlüsselgröße* für Nichtblattseiten)
  – K = *durchschnittliche Schlüsselgröße* für Nichtblattseiten

Die *minimale Schlüsselgröße* und die *durchschnittliche Schlüsselgröße* für Nichtblattseiten stimmen mit den Werten für Blattseiten überein, sofern keine INCLUDE-Spalten vorkommen. INCLUDE-Spalten werden auf Nichtblattseiten nicht gespeichert.

Sie sollten `0,9` nur durch `(100 - pctfree)/100` ersetzen, wenn dieser Wert größer als `0,9` ist, weil auf Nichtblattseiten bei der Indexerstellung maximal 10% Speicherbereich frei bleibt.

Die Zahl der Nichtblattseiten kann mit folgender Formel abgeschätzt werden:

```
if L > 1 then {P++; Z++}
While (Y > 1)
{
   P = P + Y
   Y = Y / N
  Z++
}
```

Dabei gilt folgendes:
- P ist die Anzahl von Seiten(zunächst 0).
- L ist die Anzahl der Blattseiten.
- N ist die Anzahl von Schlüsseln pro Nichtblattseite.
- Y = L / N
- Z ist die Anzahl der Ebenen in der Indexbaumstruktur (zunächst 1).

Als Gesamtzahl der Seiten ergibt sich:

```
T = (L + P + 2) * 1,0002
```

Die zusätzlichen 0,02% sind für Systemaufwand (einschließlich Speicherabbildseiten).

Der für die Indexerstellung erforderliche Speicherbereich lässt sich folgendermaßen abschätzen:

```
T * Seitengröße
```

Zusätzlicher Platzbedarf

Folgende zusätzliche Speicherbereiche sind erforderlich:
- „Speicherbereich für die Protokolldatei"
- „Temporärer Arbeitsbereich" auf Seite 116

Speicherbereich für die Protokolldatei

Der Speicherbereich (in Byte), der für Protokolldateien erforderlich ist, reicht von:

```
( logprimary * (logfilsiz + 2 ) * 4096 ) + 8192
```

bis:

```
( (logprimary + logsecond) * (logfilsiz + 2 ) * 4096 ) + 8192
```

Dabei gilt folgendes:

- *logprimary* ist die Anzahl der primären Protokolldateien, die in der Konfigurationsdatei der Datenbank definiert sind.
- *logsecond* ist die Anzahl der sekundären Protokolldateien, die in der Konfigurationsdatei der Datenbank definiert sind.
- *logfilsiz* ist die Anzahl der Seiten in jeder Protokolldatei, die in der Konfigurationsdatei der Datenbank definiert sind.
- 2 ist die Anzahl der Kopfseiten, die für jede Protokolldatei erforderlich ist.
- 4096 ist die Anzahl der Byte einer Seite.
- 8192 ist die Größe (in Byte) der Protokollsteuerdatei.

Weitere Informationen zu den Konfigurationsparametern *logprimary*, *logsecond* und *logfilsiz* finden Sie im Handbuch *Systemverwaltung: Optimierung*.

Anmerkung: Der Gesamtspeicherbereich für die aktive Protokolldatei kann eine Größe von 32 GB nicht überschreiten.

Der obere Grenzwert für den Speicherbereich für Protokolldateien ist abhängig von der tatsächlichen Anzahl der sekundären Protokolldateien, die der Datenbankmanager zur Laufzeit benötigt. Dieser obere Grenzwert wird in der Regel nicht benötigt bzw. lediglich zu gelegentlich auftretenden Zeiten hoher Aktivität benötigt.

Wenn für die Datenbank die aktualisierende Wiederherstellung aktiviert ist, sollten spezielle Speicheranforderungen für Protokolle berücksichtigt werden:

- Bei aktiviertem Konfigurationsparameter *logretain* werden die Protokolldateien im Protokollpfadverzeichnis archiviert. Der Online-Plattenspeicherplatz wird schließlich belegt, sofern Sie die Protokolldateien nicht an eine andere Speicherposition verschieben.
- Bei aktiviertem Konfigurationsparameter *userexit* verschiebt ein Benutzerausgangsprogramm die archivierten Protokolldateien an eine andere Speicherposition. Zusätzlicher Speicherbereich ist weiterhin erforderlich für folgende Dateien:
  - Online archivierte Protokolle, die darauf warten, vom Benutzerausgangsprogramm verschoben zu werden
  - Neue Protokolle, die für künftige Zwecke formatiert werden

Temporärer Arbeitsbereich

Einige SQL-Anweisungen benötigen für die Verarbeitung temporäre Tabellen (z. B. eine Arbeitsdatei für Sortieroperationen, die nicht im Speicher vorgenommen werden können). Diese temporären Tabellen benötigen Plattenspeicherplatz. Die Größe des erforderlichen Speichers hängt von den Abfragen sowie von der Größe der Ergebnistabellen ab und kann nicht abgeschätzt werden.

Mit Hilfe des Datenbanksystemmonitors und der APIs zum Abfragen des Tabellenbereichs können Sie verfolgen, wie groß der verwendete Arbeitsbereich während des normalen Betriebsablaufs ist.

Entwerfen von Knotengruppen

Eine *Knotengruppe* ist eine benannte Gruppe aus einem oder mehreren Knoten, die als zu einer Datenbank gehörig definiert sind. Jede Datenbankpartition, die Teil einer Datenbanksystemkonfiguration ist, muss bereits in einer *Partitionskonfigurationsdatei* namens `db2nodes.cfg` definiert sein. Eine Knotengruppe kann eine Datenbankpartition, mehrere Datenbankpartitionen und auch alle für das Datenbanksystem definierten Datenbankpartitionen enthalten.

Eine neue Knotengruppe wird mit Hilfe der Anweisung CREATE NODEGROUP erstellt und kann mit Hilfe der Anweisung ALTER NODEGROUP geändert werden. Es können eine oder mehrere Datenbankpartitionen einer Knotengruppe hinzugefügt oder aus ihr entfernt werden. Die Datenbankpartitionen müssen in der Datei `db2nodes.cfg` definiert sein, bevor die Knotengruppe geändert wird. Tabellenbereiche befinden sich innerhalb von Knotengruppen. Tabellen befinden sich innerhalb von Tabellenbereichen.

Wenn eine Knotengruppe erstellt oder geändert wird, wird ihr eine *Partitionierungszuordnung* zugeordnet. Die Partitionierungszuordnung wird vom Datenbankmanager in Verbindung mit einem *Partitionierungsschlüssel* und einem Hash-Algorithmus verwendet, um festzulegen, in welcher Datenbankpartition der Knotengruppe eine bestimmte Datenzeile gespeichert wird. Weitere Informationen zu Partitionierungszuordnungen finden Sie im Abschnitt „Partitionierungszuordnungen" auf Seite 119. Weitere Informationen zu Partitionierungsschlüsseln enthält der Abschnitt „Partitionierungsschlüssel" auf Seite 120.

in einer nichtpartitionierten Datenbank ist kein Partitionierungsschlüssel und keine Partitionierungszuordnung erforderlich. Wenn Sie eine nichtpartitionierte Datenbank verwenden, spielen Überlegungen zu Knotengruppen keine Rolle. Eine *Datenbankpartition* ist ein Teil einer Datenbank, mit eigenen Benutzerdaten, Indizes, Konfigurationsdateien und Transaktionsprotokollen. Vom Datenbankmanager werden Standardknotengruppen verwendet, die

erstellt werden, wenn die Datenbank erstellt wird. IBMCATGROUP ist die Standardknotengruppe für den Tabellenbereich, der die Systemkatalogtabellen enthält. IBMTEMPGROUP ist die Standardknotengruppe für die Tabellenbereiche mit den temporären Systemtabellen. IBMDEFAULTGROUP ist die Standardknotengruppe für die Tabellenbereiche, die die benutzerdefinierten Tabellen enthalten, die Sie dort speichern. Ein temporärer Benutzertabellenbereich für deklarierte temporäre Benutzertabellen kann in IBMDEFAULTGROUP bzw. in einer beliebigen benutzererstellten Knotengruppe, jedoch nicht in IBMTEMPGROUP erstellt werden.

Wenn Sie eine Knotengruppe mit mehreren Partitionen verwenden, beachten Sie die folgenden Gesichtspunkte:

- In einer Knotengruppe mit mehreren Partitionen können Sie nur dann einen eindeutigen Index erstellen, wenn er eine Obermenge des Partitionierungsschlüssels ist.
- Abhängig von der Anzahl der Datenbankpartitionen in der Datenbank können eine oder mehrere Knotengruppen mit einer Partition und eine oder mehrere Knotengruppen mit mehreren Partitionen vorhanden sein.
- Jeder Datenbankpartition muss eine eindeutige Partitionsnummer zugeordnet sein. Dieselbe Datenbankpartition kann in einer oder mehreren Knotengruppen enthalten sein.
- Zur Gewährleistung einer schnellen Wiederherstellung der Datenbankpartition, die die Systemkatalogtabellen enthält, sollten Sie keine Benutzertabellen in derselben Datenbankpartition anlegen. Dies wird erreicht, wenn die Benutzertabellen in Knotengruppen untergebracht werden, die nicht die Datenbankpartition der Knotengruppe IBMCATGROUP enthalten.

Kleine Tabellen sollten in Knotengruppen mit einer Einzelpartition angelegt werde, außer wenn Sie Nutzen aus der *Kollokation* mit einer größeren Tabelle ziehen wollen. Unter Kollokation (Zusammenfassung) ist die Platzierung von Zeilen aus verschiedenen Tabellen, die zusammengehörige Daten enthalten, in die gleiche Datenbankpartition zu verstehen. Zusammengefasste Tabellen ermöglichen DB2 den Einsatz effizienterer Verknüpfungsstrategien. Durch Kollokation zusammengefasste Tabellen können sich in einer Knotengruppe mit einer Einzelpartition befinden. Tabellen werden als durch Kollokation zusammengefasst betrachtet, wenn sie sich in einer Knotengruppe mit mehreren Partitionen befinden, dieselbe Anzahl von Spalten im Partitionierungsschlüssel haben und die Datentypen der sich entsprechenden Spalten partitionskompatibel sind. Zeilen in zusammengefassten Tabellen mit demselben Wert im Partitionierungsschlüssel werden in derselben Datenbankpartition gespeichert. Tabellen können sich in separaten Tabellenbereichen in derselben Knotengruppe befinden und trotzdem als durch Kollokation zusammengefasst betrachtet werden.

Mittelgroße Tabellen sollten nicht über zu viele Datenbankpartitionen verteilt werden. Zum Beispiel ist es möglich, dass eine 100-MB-Tabelle in einer Knotengruppe mit 16 Partitionen zu besseren Leistungen führt als in einer Knotengruppe mit 32 Partitionen.

Sie können Knotengruppen dazu verwenden, OLTP-Tabellen (OLTP - Online-Transaktionsverarbeitung) von Entscheidungshilfetabellen (DSS-Tabellen - Decision Support System) zu trennen, um sicherzustellen, dass die Leistung von OLTP-Transaktionen nicht beeinträchtigt wird.

Überlegungen zum Aufbau von Knotengruppen

Der logische Datenbankentwurf und die Menge an Daten, die verarbeitet werden muss, legen eine Antwort auf die Frage nahe, ob eine Datenbank partitioniert werden muss. Dieser Abschnitt behandelt die folgenden Themen zur Datenbankpartitionierung:

- „Datenpartitionierung"
- „Partitionierungszuordnungen" auf Seite 119
- „Partitionierungsschlüssel" auf Seite 120
- „Tabellenkollokation" auf Seite 123
- „Partitionskompatibilität" auf Seite 123
- „Replizierte Übersichtstabellen" auf Seite 124

Datenpartitionierung

DB2 unterstützt ein partitioniertes Speichermodell, das Ihnen ermöglicht, Daten über verschiedene Datenbankpartitionen in der Datenbank verteilt zu speichern. Dies bedeutet, dass die Daten zwar physisch über mehr als eine Datenbankpartition verteilt werden, jedoch so auf sie zugegriffen werden kann, als ob sie sich an derselben Speicherposition befänden. Anwendungen und Benutzer, die auf Daten in einer partitionierten Datenbank zugreifen, brauchen die physische Speicherposition der Daten nicht zu kennen.

Die Daten werden trotz der physischen Trennung als logische Einheit verwendet und verwaltet. Benutzer können wählen, wie Ihre Daten partitioniert werden, indem Sie Partitionierungsschlüssel festlegen. Benutzer können darüber hinaus bestimmen, auf welche und auf wie viele Datenbankpartitionen Ihre Tabellendaten verteilt werden können, indem sie den Tabellenbereich und die Knotengruppe auswählen, in der die Daten gespeichert werden sollen. Weiterhin wird eine aktualisierbare Partitionierungszuordnung zusammen mit einem Hash-Algorithmus verwendet, um die Zuordnung von Werten der Partitionierungsschlüssel zu Datenbankpartitionen anzugeben, wodurch das Platzieren und Abrufen der einzelnen Datenzeilen festgelegt wird. Dadurch können Sie die Auslastung für große Tabellen über eine partitionierte Datenbank hinweg verteilen, während kleinere Tabellen in einer oder mehreren Datenbankpartitionen gespeichert werden können. Jede Datenbankpartition verfügt über

lokale Indizes für die in ihr gespeicherten Daten, so dass für den lokalen Datenzugriff eine erhöhte Leistung erzielt wird.

Sie sind nicht darauf beschränkt, alle Tabelle über alle Datenbankpartitionen in der Datenbank verteilt zu speichern. DB2 unterstützt eine *Teilentclusterung*, das heißt, Sie können die Tabellen und die zugehörigen Tabellenbereiche über eine Teilgruppe der Datenbankpartitionen im System (d. h. eine Knotengruppe) verteilen.

Als alternative Methode, wenn Sie Tabellen in den einzelnen Datenbankpartitionen angelegt haben wollen, sollten Sie die Verwendung von Übersichtstabellen in Erwägung ziehen, die anschließend repliziert werden. Sie können eine Übersichtstabelle mit den von Ihnen benötigten Informationen erstellen und diese dann auf jeden Knoten replizieren. Weitere Informationen finden Sie in „Replizierte Übersichtstabellen" auf Seite 124.

Partitionierungszuordnungen

In einer Umgebung mit einer partitionierten Datenbank muss der Datenbankmanager eine Möglichkeit besitzen, zu wissen, welche Tabellenzeilen in welcher Datenbankpartition gespeichert werden. Der Datenbankmanager muss wissen, wo er die benötigten Daten findet. Zum Auffinden der Daten verwendet er ein Zuordnungsverzeichnis, das als *Partitionierungszuordnung* bezeichnet wird.

Eine Partitionierungszuordnung ist eine intern generierte Tabelle von entweder 4 096 Einträgen für Knotengruppen mit mehreren Partitionen oder einem einzigen Eintrag für Knotengruppen mit einer Einzelpartition. Für eine Knotengruppe mit nur einer Partition enthält die Partitionierungszuordnung nur einen Eintrag mit der Partitionsnummer der Datenbankpartition, in der alle Zeilen einer Datenbanktabelle gespeichert werden. Für Knotengruppen mit mehreren Partitionen werden die Partitionsnummern der Knotengruppe immer wiederholt reihum angegeben. Vergleichbar mit der Verwendung einer Stadtkarte, die durch Gitterlinien in Sektoren unterteilt ist, verwendet der Datenbankmanager einen *Partitionierungsschlüssel*, um die Speicherposition (die Datenbankpartition) zu bestimmen, in der die Daten gespeichert werden.

Nehmen Sie zum Beispiel an, dass Sie eine Datenbank in vier Datenbankpartitionen (mit den Nummern 0–3) erstellt haben. Die Partitionierungszuordnung für die Knotengruppe IBMDEFAULTGROUP dieser Datenbank wäre wie folgt:

```
0 1 2 3 0 1 2 ...
```

Wenn eine Knotengruppe in der Datenbank mit den Datenbankpartitionen 1 und 2 erstellt würde, wäre die Partitionierungszuordnung für diese Knotengruppe wie folgt:

```
1 2 1 2 1 2 1 ...
```

Wenn der Partitionierungsschlüssel für eine in die Datenbank zu ladende Tabelle eine ganze Zahl (Integer) mit möglichen Werten zwischen 1 und 500 000 ist, wird der Partitionierungsschlüssel mit einem Hash-Verfahren auf eine Partitionsnummer zwischen 0 und 4 095 abgebildet. Diese Nummer wird als Index für die Partitionierungszuordnung verwendet, um die Datenbankpartition für die betreffende Zeile auszuwählen.

Abb. 27 zeigt, wie der Zeile mit dem Partitionierungsschlüsselwert (c1, c2, c3) die Partitionsnummer 2 zugeordnet wird, die ihrerseits auf Datenbankpartition n5 verweist.

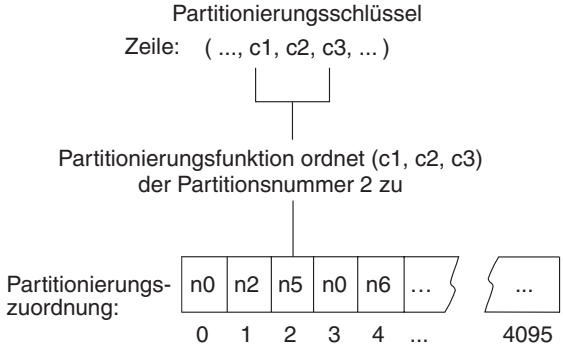

Abbildung 27. Datenverteilung mit einer Partitionierungszuordnung

Eine solche Partitionierungszuordnung ist eine flexible Steuermethode für die Speicherung von Daten in einer partitionierten Datenbank. Wenn zu einem zukünftigen Zeitpunkt die Notwendigkeit eintritt, die Datenverteilung auf die Datenbankpartitionen Ihrer Datenbank zu ändern, können Sie dazu das Dienstprogramm zur Datenumverteilung verwenden. Dieses Dienstprogramm ermöglicht Ihnen, die Datenverteilung neu auszugleichen oder bewusst ungleichmäßig zu gestalten. Weitere Informationen zu diesem Dienstprogramm finden Sie im Abschnitt zum Umverteilen von Daten über Datenbankpartitionen des Handbuchs *Systemverwaltung: Optimierung*.

Mit Hilfe der API **sqlugtpi** zum Abrufen von Informationen zur Tabellenpartitionierung können Sie eine Kopie der Partitionierungszuordnung abrufen, die Sie anzeigen können. Weitere Informationen zu dieser API finden Sie im Handbuch *Administrative API Reference*.

Partitionierungsschlüssel

Ein *Partitionierungsschlüssel* ist eine Spalte (oder Spaltengruppe), die dazu verwendet wird, die Partition zu bestimmen, in der eine bestimmte Datenzeile gespeichert ist. Ein Partitionierungsschlüssel wird in einer Tabelle mit Hilfe der Anweisung CREATE TABLE definiert. Wenn kein Partitionierungsschlüssel für eine Tabelle in einem Tabellenbereich, der über mehr als eine

Datenbankpartition in einer Knotengruppe verteilt ist, definiert wird, wird ein Partitionierungsschlüssel standardmäßig aus der ersten Spalte des Primärschlüssels erstellt. Wenn kein Primärschlüssel angegeben ist, wird standardmäßig die erste Spalte der Tabelle, die keine langen Daten enthält, als Partitionierungsschlüssel angenommen. (*Lang* beinhaltet hier alle langen Datentypen (LONG) und alle LOB-Datentypen). Wenn Sie eine Tabelle in einem Tabellenbereich erstellen, der zu einer Knotengruppe mit nur einer Partition gehört, und einen Partitionierungsschlüssel erstellen wollen, müssen Sie den Partitionierungsschlüssel explizit definieren. In diesem Fall wird kein Standardschlüssel erstellt.

Wenn keine Spalte die Voraussetzungen für einen standardmäßig erstellten Partitionierungsschlüssel erfüllt, wird die Tabelle ohne Partitionierungsschlüssel erstellt. Tabellen ohne Partitionierungsschlüssel sind nur in Knotengruppen mit einer Partition zulässig. Sie können Partitionierungsschlüssel auch später noch mit der Anweisung ALTER TABLE hinzufügen oder entfernen. Das Ändern des Partitionierungsschlüssels ist nur möglich in einer Tabelle, deren Tabellenbereich zu einer Knotengruppe mit einer Einzelpartition gehört.

Die Auswahl eines guten Partitionierungsschlüssels ist von großer Bedeutung. Sie sollten sich über Folgendes im Klaren sein:

- Wie der Zugriff auf Tabellen erfolgen soll
- Die Art der Abfrageauslastung
- Die vom Datenbanksystem angewendeten Verknüpfungsstrategien

Wenn Kollokation kein Hauptgesichtspunkt ist, dann zeichnet sich ein guter Partitionierungsschlüssel für eine Tabelle dadurch aus, dass er die Daten gleichmäßig über alle Datenbankpartitionen in der Knotengruppe verteilt. Der Partitionierungsschlüssel jeder Tabelle in einem Tabellenbereich, der einer Knotengruppe zugeordnet ist, bestimmt, ob die Tabellen durch Kollokation zusammengefasst werden. Tabellen werden als zusammengefasst betrachtet, wenn folgende Bedingungen gelten:

- Die Tabellen liegen in Tabellenbereichen, die in derselben Knotengruppe sind.
- Die Partitionierungsschlüssel in jeder Tabelle haben dieselbe Anzahl von Spalten.
- Die Datentypen der entsprechenden Spalten sind partitionskompatibel.

Diese Merkmale stellen sicher, dass Zeilen zusammengefasster Tabellen mit denselben Werten für ihre Partitionierungsschlüssel in derselben Partition gespeichert werden. Weitere Informationen zur Partitionskompatibilität finden Sie in „Partitionskompatibilität" auf Seite 123. Weitere Informationen zur Tabellenkollokation finden Sie in „Tabellenkollokation" auf Seite 123.

Ein ungeeigneter Partitionierungsschlüssel kann zu einer ungleichmäßigen Datenverteilung führen. Spalten mit ungleichmäßig verteilten Daten und Spalten mit einer kleinen Anzahl unterschiedlicher Werte sollten nicht als Partitionierungsschlüssel ausgewählt werden. Die Anzahl der unterschiedlichen Werte muss ausreichend groß sein, um eine gleichmäßige Verteilung der Zeilen auf alle Datenbankpartitionen in der Knotengruppe sicherzustellen. Der Aufwand für die Anwendung des Hash-Algorithmus zur Partitionierung ist proportional zur Größe des Partitionierungsschlüssels. Der Partitionierungsschlüssel kann nicht mehr als 16 Spalten enthalten. Jedoch wird bei weniger Spalten eine bessere Leistung erzielt. Nicht benötigte Spalten sollten daher nicht in den Partitionierungsschlüssel aufgenommen werden.

Die folgenden Punkte sollten bei der Definition von Partitionierungsschlüsseln beachtet werden:

- Die Erstellung einer Mehrpartitionstabelle, die ausschließlich lange Datentypen (LONG VARCHAR, LONG VARGRAPHIC, BLOB, CLOB oder DBCLOB) enthält, wird wird nicht unterstützt.
- Die Definition des Partitionierungsschlüssels kann nicht geändert werden.
- Der Partitionierungsschlüssel sollte die am häufigsten an Verknüpfungen beteiligten Spalten enthalten.
- Der Partitionierungsschlüssel sollte außerdem aus Spalten bestehen, die häufig von Klauseln GROUP BY betroffen sind.
- Jeder eindeutige Schlüssel oder Primärschlüssel muss alle Spalten des Partitionierungsschlüssels enthalten.
- In einer OLTP-Umgebung (Online-Transaktionsverarbeitung) sollten alle Spalten des Partitionierungsschlüssels an der Transaktion über Gleichheitsvergleichselemente (=) mit Konstanten oder Host-Variablen beteiligt sein. Nehmen Sie zum Beispiel an, Sie haben eine Personalnummer *emp_no*, die in Transaktionen häufig wie folgt oder ähnlich verwendet wird:

  ```
  UPDATE emp_table SET ... WHERE
  emp_no = host-variable
  ```

  In diesem Fall wäre die Spalte EMP_NO ein guter einspaltiger Partitionierungsschlüssel für die Tabelle EMP_TABLE.

Wenn die Variable DB2_UPDATE_PART_KEY der Registrierungsdatenbank auf den Wert NO gesetzt ist, kann der Wert einer Spalte des Partitionierungsschlüssels für eine Zeile der Tabelle nicht aktualisiert werden. Werte von Partitionierungsschlüsselspalten können nur gelöscht oder eingefügt werden.

Hash-Partitionierung heißt die Methode, mit der die Speicherposition jeder Zeile in der partitionierten Tabelle ermittelt wird. Diese Methode funktioniert folgendermaßen:

1. Der Hash-Algorithmus wird auf den Wert des Partitionierungsschlüssels angewendet und generiert eine Partitionsnummer zwischen 0 und 4095.
2. Die Partitionierungszuordnung wird bei der Erstellung einer Knotengruppe erstellt. Die Partitionsnummern werden immer wieder in derselben Reihenfolge nacheinander wiederholt, bis die Partitionierungszuordnung gefüllt ist. Weitere Informationen zu Partitionierungszuordnungen finden Sie im Abschnitt „Partitionierungszuordnungen" auf Seite 119.
3. Die Partitionsnummer wird als Index für die Position in der Partitionierungszuordnung verwendet. Die Nummer an dieser Position in der Partitionierungszuordnung ist die Nummer der Datenbankpartition, in der die Zeile gespeichert wird.

Tabellenkollokation

Sie stellen möglicherweise fest, dass zwei oder mehr Tabellen häufig zusammen Daten als Antwort für bestimmte Abfragen liefern. In diesem Fall ist es sinnvoll, zusammengehörige Daten aus solchen Tabellen möglichst nah beieinander zu speichern. In einer Umgebung, in der die Datenbank physisch auf zwei oder mehr Datenbankpartitionen verteilt ist, muss es eine Möglichkeit geben, zusammengehörige Teile der verteilten Tabellen so nah wie möglich beieinander zu halten. Diese Möglichkeit wird als *Tabellenkollokation* bezeichnet.

Tabellen sind in einer Kollokation zusammengefasst, wenn sie in derselben Knotengruppe gespeichert sind und ihre Partitionierungsschlüssel kompatibel sind. Durch Speichern beider Tabellen in derselben Knotengruppe wird sichergestellt, dass sie eine gemeinsame Partitionierungszuordnung haben. Die Tabellen können sich in verschiedenen Tabellenbereichen befinden, jedoch müssen die Tabellenbereiche derselben Knotengruppe zugeordnet sein. Die Datentypen der entsprechenden Spalten in den jeweiligen Partitionierungsschlüsseln müssen *partitionskompatibel* sein. Informationen zur Partitionskompatibilität finden Sie in „Partitionskompatibilität".

DB2 kann beim Zugriff auf mehr als eine Tabelle bei einer Verknüpfung oder einer Unterabfrage erkennen, dass sich die zu verknüpfenden Daten in derselben Datenbankpartition befinden. Wenn dies geschieht, kann DB2 entscheiden, die Verknüpfung oder Unterabfrage in der Datenbankpartition auszuführen, in der die Daten gespeichert sind, anstatt die Daten zwischen Datenbankpartitionen auszutauschen. Diese Möglichkeit, Verknüpfungen oder Unterabfragen in der Datenbankpartition auszuführen, bietet wesentliche Leistungsvorteile. Weitere Informationen finden Sie unter „Zusammengefasste Verknüpfungen" im Handbuch *Systemverwaltung: Optimierung*.

Partitionskompatibilität

Die Basisdatentypen entsprechender Spalten von Partitionierungsschlüsseln werden verglichen und können als *partitionskompatibel* deklariert werden. Partitionskompatible Datentypen haben die Eigenschaft, dass ein bestimmter

Partitionierungsalgorithmus zwei Variablen mit jeweils einem dieser Datentypen dieselbe Partitionsnummer zuordnet, wenn sie denselben Wert haben.

Partitionskompatibilität hat folgende Merkmale:

- Ein Wert des Basisdatentyps ist mit einem anderen desselben Basistyps kompatibel.
- Interne Formate werden für die Datentypen DATE (Datum), TIME (Uhrzeit) und TIMESTAMP (Zeitmarke) verwendet. Sie sind untereinander nicht kompatibel, und keiner dieser Typen ist mit dem Zeichendatentyp (CHAR) kompatibel.
- Die Partitionskompatibilität wird von Spalten mit den Definitionen NOT NULL oder FOR BIT DATA nicht berührt.
- Nullwerte (NULL) kompatibler Datentypen werden auf identische Weise behandelt, Nullwerte nicht kompatibler Datentypen möglicherweise nicht.
- Die Basisdatentypen eines benutzerdefinierten Datentyps werden zur Analyse der Partitionskompatibilität verwendet.
- Dezimalzahlen desselben Werts im Partitionierungsschlüssel werden identisch behandelt, auch wenn ihre Anzahl an Kommastellen und ihre Genauigkeit unterschiedlich sind.
- Folgende Leerzeichen in Zeichenfolgen (CHAR, VARCHAR, GRAPHIC oder VARGRAPHIC) werden vom Hash-Algorithmus ignoriert.
- BIGINT, SMALLINT und INTEGER sind kompatible Datentypen.
- REAL und FLOAT sind kompatible Datentypen.
- CHAR und VARCHAR verschiedener Längen sind kompatible Datentypen.
- GRAPHIC und VARGRAPHIC sind kompatible Datentypen.
- Die Partitionskompatibilität gilt nicht für die Datentypen LONG VARCHAR, LONG VARGRAPHIC, CLOB, DBCLOB und BLOB, da sie in Partitionierungsschlüsseln nicht unterstützt werden.

Replizierte Übersichtstabellen

Eine *Übersichtstabelle* ist eine Tabelle, die durch eine Abfrage definiert ist, mit der auch die Daten für die Tabelle festgelegt werden. Durch Übersichtstabellen kann die Leistungsfähigkeit von Abfragen erhöht werden. Wenn DB2 erkennt, dass ein Teil einer Abfrage durch eine Übersichtstabelle aufgelöst werden könnte, kann der Datenbankmanager die Abfrage so abwandeln, dass die entsprechende Übersichtstabelle verwendet wird.

In einer partitionierten Datenbankumgebung können Sie Übersichtstabellen replizieren. Durch die Verwendung *replizierter Übersichtstabellen* können Sie die Leistung von Abfragen erhöhen. Eine replizierte Übersichtstabelle basiert auf einer Tabelle, die möglicherweise in einer Knotengruppe mit einer Einzelpartition erstellt wurde, die Sie jedoch über alle Datenbankpartitionen in der

Knotengruppe replizieren wollen. Die replizierte Übersichtstabelle wird durch Ausführen der Anweisung CREATE TABLE mit dem Schlüsselwort REPLICATED erstellt.

Weitere Informationen zu Übersichtstabellen finden Sie in "Erstellen einer Übersichtstabelle" im Handbuch *Systemverwaltung: Implementierung*.

Mit Hilfe von replizierten Übersichtstabellen können Sie Kollokationen zwischen Tabellen herstellen, die normalerweise nicht kollokiert sind. Replizierte Übersichtstabellen sind besonders nützlich für Verknüpfungen großer Fakttabellen mit kleinen Dimensionstabellen. Zur Minimierung des erforderlichen zusätzlichen Speicherbedarfs sowie des Aufwands zum Aktualisieren aller Replikate, sollten zu replizierende Tabellen klein sein und nicht häufig aktualisiert werden.

Anmerkung: Sie sollten auch in Erwägung ziehen, umfangreiche Tabellen, die selten aktualisiert werden, zu replizieren: Der Zusatzaufwand für eine Replikation durch die mit Hilfe der Kollokation erzielten Leistungsvorteile wettgemacht.

Durch Angeben eines geeigneten Vergleichselements in der Unterauswahlklausel, die zum Definieren der replizierten Tabelle verwendet wird, können Sie ausgewählte Spalten, ausgewählte Zeilen (oder beides) replizieren.

Weitere Informationen zu replizierten Übersichtstabellen finden Sie im Abschnitt zur Anweisung CREATE TABLE im Handbuch *SQL Reference*. Weitere Informationen zu zusammengefassten Verknüpfungen finden Sie unter „Zusammengefasste Verknüpfungen" im Handbuch *Systemverwaltung: Implementierung*.

Entwerfen und Auswählen von Tabellenbereichen

Ein Tabellenbereich ist ein Speichermodell, das eine Zwischenstufe zwischen einer Datenbank und den in ihr gespeicherten Tabellen zur Verfügung stellt. Tabellenbereiche befinden sich in Knotengruppen (Nodegroups). Mit ihrer Hilfe können Sie die Speicherposition der Datenbank- und Tabellendaten direkt bestimmten Behältern zuweisen. (Ein Behälter kann ein Verzeichnisname, Einheitenname oder Dateiname sein.) Die möglichen Vorzüge zeigen sich in einer besseren Leistung, einer flexibleren Konfiguration und einer höheren Integrität.

Informationen zum Erstellen oder Ändern eines Tabellenbereichs finden Sie in "Erstellen eines Tabellenbereichs" bzw. "Ändern eines Tabellenbereichs" im Handbuch *Systemverwaltung: Implementierung*.

Da sich Tabellenbereiche in Knotengruppen befinden, bestimmt der zur Speicherung einer Tabelle ausgewählte Tabellenbereich, wie die Daten dieser Tabelle auf die Datenbankpartitionen in einer Knotengruppe verteilt werden. Ein einzelner Tabellenbereich kann sich über mehrere Behälter erstrecken. Mehrere Behälter (eines oder mehrerer Tabellenbereiche) können auf derselben physischen Platte (bzw. auf demselben Laufwerk) erstellt werden. Zur Erzielung einer optimalen Leistung sollte jeder Behälter eine andere Platte verwenden. Abb. 28 zeigt ein Beispiel für die Anordnungsbeziehung zwischen Tabellen und Tabellenbereichen innerhalb einer Datenbank und den Behältern, die dieser Datenbank zugeordnet sind.

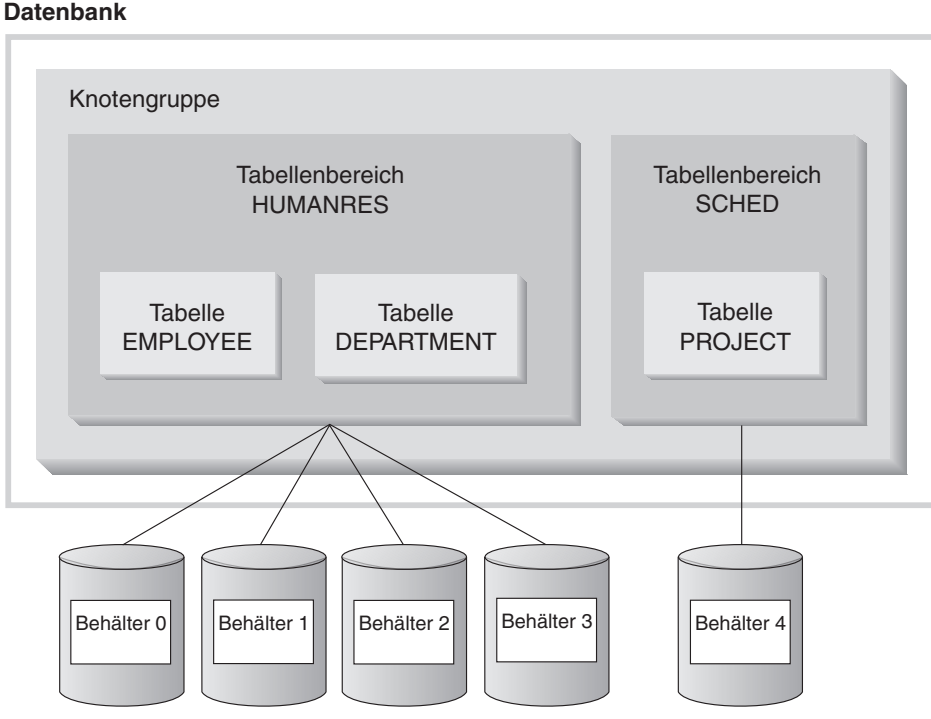

Abbildung 28. Tabellenbereiche und Tabellen in einer Datenbank

Die Tabellen EMPLOYEE und DEPARTMENT befinden Sie im Tabellenbereich HUMANRES, der sich über die Behälter 0, 1, 2 und 3 erstreckt. Die Tabelle PROJECT befindet sich im Tabellenbereich SCHED im Behälter 4. Dieses Beispiel zeigt jeden Behälter auf einer getrennten Platte.

Der Datenbankmanager versucht, die Datenmenge möglichst gleichmäßig über die Behälter zu verteilen. Das heißt, es werden alle Behälter zum Speichern der Daten verwendet. Die Anzahl der Seiten, die der Datenbankmanager in einen Behälter schreibt, bevor er einen anderen Behälter verwen-

det, wird mit dem Parameter *EXTENTSIZE* definiert. Der Datenbankmanager beginnt beim Speichern der Tabellendaten nicht immer mit dem ersten Behälter.

Abb. 29 zeigt den Tabellenbereich HUMANRES mit einem Wert von zwei 4-KB-Seiten für EXTENTSIZE und vier Behältern mit einer kleinen Anzahl zugeordneter Speicherbereiche. Die Tabellen DEPARTMENT und EMPLOYEE haben jeweils sieben Seiten und verteilen sich über alle vier Behälter.

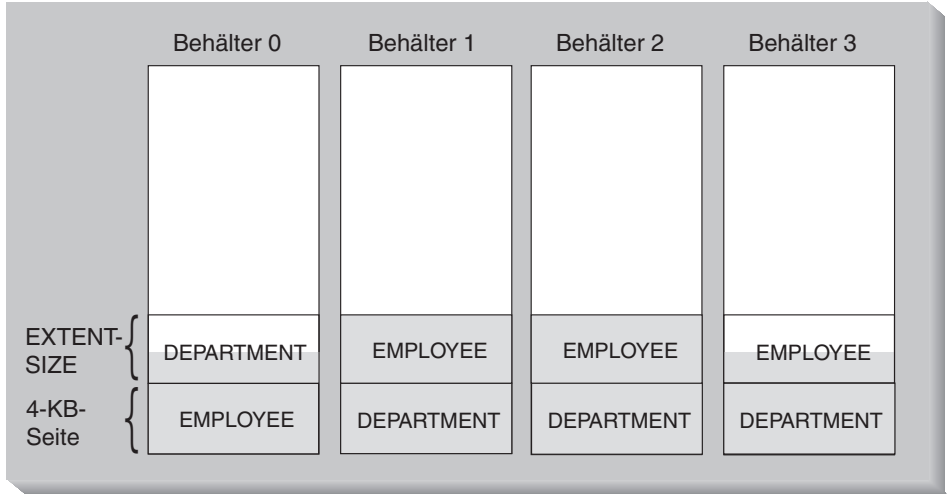

Abbildung 29. Behälter und EXTENTSIZE-Bereiche

Eine Datenbank muss mindestens drei Tabellenbereiche enthalten:

- Einen *Katalogtabellenbereich*, der alle Systemkatalogtabellen für die Datenbank enthält. Dieser Tabellenbereich heißt SYSCATSPACE und kann nicht gelöscht werden. IBMCATGROUP ist die Standardknotengruppe für diesen Tabellenbereich.
- Einen oder mehrere *Benutzertabellenbereiche*, die alle benutzerdefinierten Tabellen enthalten. Standardmäßig wird ein Tabellenbereich namens USERSPACE1 erstellt. IBMDEFAULTGROUP ist die Standardknotengruppe für diesen Tabellenbereich.

  Beim Erstellen einer Tabelle sollten Sie einen Tabellenbereichsnamen angeben, andernfalls kann es zu unerwünschten Ergebnissen kommen. Wenn Sie keinen Tabellenbereichsnamen angeben, wird die Tabelle nach folgenden Richtlinien gespeichert: Wenn benutzererstellte Tabellenbereiche existieren, wird derjenige mit der kleinsten Seitengröße, die für diese Tabelle ausreicht, ausgewählt. Ansonsten wird der Tabellenbereich USERSPACE1 verwendet,

falls er über eine für die Tabelle ausreichende Seitengröße verfügt. Wenn keine Tabellenbereiche mit ausreichender Seitengröße vorhanden sind, wird die Tabelle nicht erstellt.

Die Seitengröße für eine Tabelle wird entweder durch die Zeilengröße oder durch die Anzahl von Spalten bestimmt. Die für eine Zeile maximal zulässige Länge hängt von der Seitengröße des Tabellenbereichs ab, in dem die Tabelle erstellt wird. Gültige Werte für die Seitengröße sind 4 KB (Standardwert), 8 KB, 16 KB und 32 KB. Sie können einen Tabellenbereich mit einer Seitengröße für die Basistabelle und einen weiteren Tabellenbereich mit einer anderen Seitengröße für lange Daten (LONG) oder LOB-Daten verwenden. (Hier ist wiederum zu beachten, dass SMS keine Tabellen unterstützt, die sich über mehrere Tabellenbereiche erstrecken, im Gegensatz zu DMS.) Wenn die Anzahl der Spalten oder die Zeilengröße die Grenze für die Seitengröße eines Tabellenbereichs überschreitet, wird ein Fehler zurückgegeben (SQLSTATE 42997).

- Einen oder mehrere *temporäre Tabellenbereiche*, die temporäre Tabellen enthalten. Temporäre Tabellenbereiche können *temporäre Systemtabellenbereiche* oder *temporäre Benutzertabellenbereiche* sein. Eine Datenbank muss mindestens über einen temporären Tabellenbereich verfügen. Standardmäßig wird ein temporärer Systemtabellenbereich namens TEMPSPACE1 bei der Erstellung der Datenbank erstellt. IBMTEMPGROUP ist die Standardknotengruppe für diesen Tabellenbereich. Temporäre Benutzertabellenbereiche werden *nicht* standardmäßig bei der Erstellung einer Datenbank erstellt.

  Wenn eine Datenbank mehr als einen temporären Tabellenbereich verwendet und ein neues temporäres Objekt benötigt wird, wählt das Optimierungsprogramm eine geeignete Seitengröße für dieses Objekt aus. Anschließend wird dieses Objekt dem temporären Tabellenbereich mit der entsprechenden Seitengröße zugeordnet. Wenn mehr als ein temporärer Tabellenbereich mit dieser Seitengröße vorhanden ist, werden die Tabellenbereiche reihum ausgewählt.

  Wenn Abfragen auf Tabellen in Tabellenbereichen ausgeführt werden, die mit einer größeren Seitengröße als dem Standardwert von 4 KB definiert sind (z. B. eine Klausel ORDER BY auf 1012 Spalten), schlagen manche dieser Abfragen möglicherweise fehl. Dies geschieht, wenn keine temporären Tabellenbereiche mit einer größeren Seitengröße definiert sind. Möglicherweise müssen Sie einen temporären Tabellenbereich mit einer größeren Seitengröße (8 KB, 16 KB oder 32 KB) erstellen. Jede DML-Anweisung (Datenbearbeitungssprache) könnte fehlschlagen, sofern kein temporärer Tabellenbereich mit derselben Seitengröße wie die größte in dem Benutzertabellenbereich verwendete Seitengröße vorhanden ist.

  Sie sollten einen einzelnen temporären SMS-Tabellenbereich definieren, dessen Seitengröße der Seitengröße entspricht, die in den meisten der Benutzertabellenbereiche verwendet wird. Dies sollte für typische Umgebungen und Auslastungen angemessen sein. Siehe auch „Empfehlungen für temporäre Tabellenbereiche” auf Seite 143.

In einer Umgebung mit partitionierten Datenbanken verfügt der Katalogknoten über alle drei Tabellenbereiche, während die anderen Datenbankpartitionen nur die Tabellenbereiche TEMPSPACE1 und USERSPACE1 enthalten.

Es gibt zwei Typen von Tabellenbereich, die beide in einer einzelnen Datenbank verwendet werden können:

- „SMS-Tabellenbereich": Der Dateimanager des Betriebssystems steuert den Speicherbereich.
- „DMS-Tabellenbereich" auf Seite 134: Der Datenbankmanager steuert den Speicherbereich.

SMS-Tabellenbereich

In einem vom Betriebssystem verwalteten Tabellenbereich (SMS - System Managed Space) ordnet der Dateisystemmanager des Betriebssystems den Speicherbereich zu, in dem die Tabelle gespeichert werden soll, und verwaltet diesen Bereich. Das Speichermodell enthält in der Regel viele Dateien, die Tabellenobjekte darstellen, die im Speicherbereich des Dateisystems gespeichert sind. Der Benutzer entscheidet über die Speicherposition der Dateien, DB2 verwaltet ihre Namen, und das Dateisystem ist für die Verwaltung auf dem System zuständig. Durch Steuern der Datenmenge, die in jede Datei geschrieben wird, verteilt der Datenbankmanager die Daten gleichmäßig auf die Behälter der Tabellenbereiche. Ein SMS-Tabellenbereich ist der Standardtabellenbereich.

Jeder Tabelle ist mindestens eine physische SMS-Datei zugeordnet. Eine Liste dieser Dateien sowie eine Beschreibung des Inhalts finden Sie in „Physische SMS-Dateien" auf Seite 132.

In einem SMS-Tabellenbereich wird eine Datei mit dem Anwachsen des Objekts um jeweils eine Seite erweitert. Wenn Sie eine bessere Leistung benötigen, können Sie die Aktivierung der mehrseitigen Dateizuordnung in Betracht ziehen. Dadurch kann das System der Datei mehr als eine Seite gleichzeitig zuordnen. Zur Aktivierung der mehrseitigen Dateizuordnung müssen Sie das Dienstprogramm **db2empfa** ausführen. In einer Umgebung mit partitionierten Datenbanken muss dieses Dienstprogramm in jeder Datenbankpartition ausgeführt werden. Wenn die mehrseitige Dateizuordnung aktiviert ist, kann sie nicht inaktiviert werden. Weitere Informationen zum Dienstprogramm **db2empfa** finden Sie im Handbuch *Command Reference*.

Sie sollten SMS-Tabellenbereiche explizit mit der Option MANAGED BY SYSTEM im Befehl CREATE DATABASE oder in der Anweisung CREATE TABLESPACE definieren. Dabei müssen Sie zwei Schlüsselfaktoren beim Entwerfen Ihrer SMS-Tabellenbereiche beachten:

- Behälter für den Tabellenbereich

Sie müssen die Anzahl der Behälter angeben, die Sie für Ihren Tabellenbereich verwenden wollen. Es ist sehr wichtig, alle gewünschten Behälter zu ermitteln, weil Sie nach dem Erstellen des Tabellenbereichs keine Behälter löschen oder hinzufügen können. In einer partitionierten Datenbankumgebung kann die Anweisung ALTER TABLESPACE verwendet werden, um beim Hinzufügen einer neuen Partition zu einer Knotengruppe für einen SMS-Tabellenbereich Behälter für die neue Partition hinzuzufügen.

Jeder Behälter, der für einen SMS-Tabellenbereich verwendet wird, identifiziert einen absoluten oder relativen Verzeichnisnamen. Jedes dieser Verzeichnisse kann sich auf einem anderen Dateisystem (oder einer anderen physischen Platte) befinden. Die Maximalgröße des Tabellenbereichs kann wie folgt abgeschätzt werden:

```
Anzahl von Behältern * (maximale Dateisystemgröße, die vom
   Betriebssystem unterstützt wird)
```

Diese Formel setzt voraus, dass jedem Behälter ein bestimmtes Dateisystem zugeordnet wird und dass für jedes Dateisystem der maximale Speicherbereich verfügbar ist. In der Praxis ist dies möglicherweise nicht der Fall, und die Maximalgröße des Tabellenbereichs kann sehr viel kleiner sein.

Anmerkung: Gehen Sie beim Definieren der Behälter mit besonderer Sorgfalt vor. Wenn die Behälter bereits Dateien oder Verzeichnisse enthalten, wird eine Fehlernachricht (SQL0298N) zurückgegeben.

- Parameter EXTENTSIZE für den Tabellenbereich

  Der Wert für den Parameter EXTENTSIZE kann nur beim Erstellen des Tabellenbereichs angegeben werden. Da spätere Änderungen daran nicht möglich sind, ist es wichtig, einen geeigneten Wert für EXTENTSIZE anzugeben. Weitere Informationen finden Sie in „Auswählen eines Werts für EXTENTSIZE" auf Seite 142.

  Wenn Sie beim Erstellen eines Tabellenbereichs für den Parameter EXTENTSIZE keinen Wert angeben, erstellt der Datenbankmanager den Tabellenbereich mit dem Standardwert, der durch den Konfigurationsparameter *dft_extent_sz* der Datenbank definiert ist (Das Handbuch *Systemverwaltung: Optimierung* enthält weitere Informationen zu diesem Parameter). Dieser Konfigurationsparameter wird anfangs auf der Grundlage der Informationen gesetzt, die beim Erstellen der Datenbank angegeben werden. Wird der Parameter *dft_extent_sz* nicht mit dem Befehl CREATE DATABASE angegeben, wird der Standardwert auf 32 gesetzt.

Um geeignete Werte für die Anzahl der Behälter und für den Parameter EXTENTSIZE für den Tabellenbereich festlegen zu können, benötigen Sie folgende Kenntnisse:

- Den oberen Grenzwert, den Ihr Betriebssystem für die Größe eines logischen Dateisystems festlegt

  Zum Beispiel haben einige Betriebssysteme eine obere Begrenzung von 2 GB. Wenn Sie also ein Tabellenobjekt mit einer Größe von 64 GB erstellen möchten, benötigen Sie auf dieser Art System mindestens 32 Behälter.

  Wenn Sie einen Tabellenbereich erstellen, können Sie Behälter angeben, die sich auf verschiedenen Dateisystemen befinden, und dadurch die Menge der Daten erhöhen, die in der Datenbank gespeichert werden können.

- Die Art und Weise, wie der Datenbankmanager die einem Tabellenbereich zugeordneten Datendateien und Behälter verwaltet

  Die erste Datei mit Tabellendaten (`SQL00001.DAT`) wird im ersten für den Tabellenbereich angegebenen Behälter erstellt. Diese Datei darf so weit anwachsen, bis sie die durch den Wert für EXTENTSIZE festgelegte Größe erreicht. Nach Erreichen dieser Größe schreibt der Datenbankmanager Daten in die Datei `SQL00001.DAT` im nächsten Behälter. Dieser Prozess wird fortgesetzt, bis alle Behälter Dateien des Namens `SQL00001.DAT` enthalten. Wenn dies eintritt, kehrt der Datenbankmanager zum ersten Behälter zurück. Dieser Prozess (der auch als *Striping - einheitenübergreifendes Lesen und Schreiben von Daten* bezeichnet wird) wird über die Behälterverzeichnisse fortgesetzt, bis ein Behälter voll ist (SQL0289N) oder vom Betriebssystem kein weiterer Speicherbereich mehr zugeordnet werden kann (Fehlernachricht: Datenträger voll). Das Striping-Verfahren wird auch für Index- (`SQL`*`nnnnn`*`.INX`), Langfeld- (`SQL`*`nnnnn`*`.LF`) und LOB-Dateien (`SQL`*`nnnnn`*`.LB` und `SQL`*`nnnnn`*`.LBA`) verwendet.

  Anmerkung: Der SMS-Tabellenbereich ist voll, sobald irgendeiner seiner Behälter voll ist. Daher ist es wichtig, jedem Behälter dieselbe Menge an Speicherbereich zuzuordnen.

  Um die Daten gleichmäßiger über die Behälter zu verteilen, bestimmt der Datenbankmanager den Behälter, der zuerst verwendet werden soll, indem er den Wert der Tabellen-ID (1 im Beispiel oben) Modulo die Anzahl der Behälter ermittelt. Die Behälter werden beginnend mit dem Wert 0 durchnummeriert.

  Weitere Informationen über die Dateien, die in einem SMS-Tabellenbereich verwendet werden, finden Sie in „Physische SMS-Dateien” auf Seite 132.

Physische SMS-Dateien

Folgende Dateien befinden sich im Verzeichnisbehälter eines SMS-Tabellenbereichs:

Dateiname **Beschreibung**

SQLTAG.NAM
In jedem Behälterunterverzeichnis befindet sich jeweils eine dieser Dateien. Der Datenbankmanager verwendet sie, wenn Sie eine Verbindung zur Datenbank herstellen, um zu prüfen, ob die Datenbank vollständig und konsistent ist.

SQLxxxxx.DAT
Dies ist eine Tabellendatei. Alle Tabellenzeilen werden hier gespeichert, ausgenommen Daten der Typen LONG VARCHAR, LONG VARGRAPHIC, BLOB, CLOB und DBCLOB.

SQLxxxxx.LF Eine Datei, die Daten der Typen LONG VARCHAR oder LONG VARGRAPHIC (auch als „Langfelddaten" bezeichnet) enthält. Die Datei wird nur erstellt, wenn die Tabelle Spalten der Datentypen LONG VARCHAR oder LONG VARGRAPHIC enthält.

SQLxxxxx.LB Eine Datei, die Daten der Typen BLOB, CLOB oder DBCLOB (auch als „LOB-Daten" bezeichnet) enthält. Die Datei wird nur erstellt, wenn die Tabelle Spalten der Datentypen BLOB, CLOB oder DBCLOB enthält.

SQLxxxxx.LBA
Eine Datei, die Informationen zur Zuordnung und zu freien Speicherbereichen der Dateien SQL*xxxxx*.LB enthält.

SQLxxxxx.INX
Eine Indexdatei für eine Tabelle. Alle Indizes für die entsprechende Tabelle werden in dieser einen Datei gespeichert. Die Datei wird nur erstellt, wenn Indizes definiert wurden.

Anmerkung: Beim Löschen von Indizes wird der von der Indexdatei (.INX) belegte Speicherbereich erst dann physisch freigegeben, wenn die Indexdatei gelöscht wird. Die Indexdatei wird gelöscht, wenn alle Indizes der Tabelle entfernt (und festgeschrieben) werden oder wenn die Tabelle reorganisiert wird. Wenn die Indexdatei nicht gelöscht wird, wird der Speicherbereich als frei markiert, sobald der Löschvorgang (DROP) festgeschrieben ist. Im Anschluss daran kann der Speicherbereich für künftige Indexerstellungs- und Indexverwaltungsoperationen erneut verwendet werden.

SQLxxxxx.DTR

Eine temporäre Datendatei für das Reorganisieren einer DAT-Datei. Beim Reorganisieren einer Tabelle erstellt das REORG-Dienstprogramm (über den Befehl REORG TABLE) eine Tabelle in einem der temporären Systemtabellenbereiche. Diese temporären Tabellenbereiche können definiert werden, um andere Behälter als die für die benutzerdefinierten Tabellen angegebenen Behälter zu verwenden.

SQLxxxxx.LFR

Eine temporäre Datendatei für das Reorganisieren einer LF-Datei. Beim Reorganisieren einer Tabelle erstellt das REORG-Dienstprogramm (über den Befehl REORG TABLE) eine Tabelle in einem der temporären Systemtabellenbereiche. Diese temporären Tabellenbereiche können definiert werden, um andere Behälter als die für die benutzerdefinierten Tabellen angegebenen Behälter zu verwenden.

SQLxxxxx.RLB

Eine temporäre Datendatei für das Reorganisieren einer LB-Datei. Beim Reorganisieren einer Tabelle erstellt das REORG-Dienstprogramm (über den Befehl REORG TABLE) eine Tabelle in einem der temporären Systemtabellenbereiche. Diese temporären Tabellenbereiche können definiert werden, um andere Behälter als die für die benutzerdefinierten Tabellen angegebenen Behälter zu verwenden.

SQLxxxxx.RBA

Eine temporäre Datendatei für das Reorganisieren einer LBA-Datei. Beim Reorganisieren einer Tabelle erstellt das REORG-Dienstprogramm (über den Befehl REORG TABLE) eine Tabelle in einem der temporären Systemtabellenbereiche. Diese temporären Tabellenbereiche können definiert werden, um andere Behälter als die für die benutzerdefinierten Tabellen angegebenen Behälter zu verwenden.

Anmerkungen:

1. Nehmen Sie *auf keinen Fall* direkte Änderungen an diesen Dateien vor. Der Zugriff auf diese Dateien kann nur indirekt über die dokumentierten APIs und mit Tools erfolgen, die diese APIs implementieren (einschließlich des Befehlszeilenprozessors und der Steuerzentrale).
2. Verschieben Sie diese Dateien nicht.
3. Löschen Sie diese Dateien nicht.
4. Das Sichern einer Datenbank oder eines Tabellenbereichs über die API **sqlubkp** (Backup Database), einschließlich der Implementierungen dieser API durch den Befehlszeilenprozessor und die Steuerzentrale, ist die einzige Sicherungsmethode, die unterstützt wird.

DMS-Tabellenbereich

In einem DMS-Tabellenbereich (DMS - Database Managed Space) steuert der Datenbankmanager den Speicherbereich. Das Speichermodell besteht aus einer begrenzten Anzahl von Einheiten, deren Speicherbereich von DB2 verwaltet wird. Der Administrator entscheidet, welche Einheiten zu verwenden sind, und DB2 verwaltet den Speicherbereich auf diesen Einheiten. Der Tabellenbereich ist im Wesentlichen eine Implementierung eines Dateisystems, das einem bestimmten Zweck dient und entworfen wurde, um die Anforderungen des Datenbankmanagers optimal zu erfüllen. Zur Tabellenbereichsdefinition gehört eine Liste der Einheiten oder Dateien, die zum Tabellenbereich gehören und in denen Daten gespeichert werden können.

Ein DMS-Tabellenbereich, der benutzerdefinierte Tabellen und Daten enthält, kann wie folgt definiert werden:

- Als *regulärer* Tabellenbereich zum Speichern normaler Tabellen- und Indexdaten
- Als *langer* Tabellenbereich (LONG) zum Speichern von Langfeld- oder LOB-Daten

Beachten Sie beim Entwerfen Ihrer DMS-Tabellenbereiche und Behälter folgendes:

- Der Datenbankmanager arbeitet mit einheitenübergreifendem Lesen und Schreiben von Daten (Striping), um eine gleichmäßige Verteilung von Daten auf alle Behälter sicherzustellen.
- Die Maximalgröße regulärer Tabellenbereiche beträgt 64 GB für 4-KB-Seiten, 128 GB für 8-KB-Seiten, 256 GB für 16-KB-Seiten und 512 GB für 32-KB-Seiten. Die Maximalgröße von Tabellenbereichen für lange Objektdaten beträgt 2 TB.
- Im Unterschied zu SMS-Tabellenbereichen müssen die Behälter, die einen DMS-Tabellenbereich bilden, nicht die gleiche Größe haben. Dies wird im Normalfall jedoch nicht empfohlen, da es zu ungleichmäßiger Verteilung (Striping) auf die Behälter und nicht optimaler Leistung führt. Wenn ein Behälter voll ist, wird in DMS-Tabellenbereichen jeder verfügbare freie Speicherbereich anderer Behälter genutzt.
- Da der Speicherbereich vorab zugeordnet wird, muss er zur Verfügung stehen, bevor der Tabellenbereich erstellt werden kann. Bei Verwendung von Einheitenbehältern muss die Einheit ebenfalls mit genügend Speicherbereich für die Definition des Behälters verfügbar sein. Auf jeder Einheit kann nur ein Behälter definiert werden. Um eine Verschwendung von Speicherbereich zu vermeiden, sollten die Größe der Einheit und die Größe des Behälters äquivalent sein. Wenn z. B. die Einheit mit 5 000 Seiten zugeordnet ist, und der Einheitenbehälter zum Zuordnen von 3 000 Seiten definiert ist, sind 2 000 Seiten der Einheit nicht verwendbar.

- Eine Seite in jedem Behälter ist für den Systemaufwand reserviert, und von den verbleibenden Seiten wird jeweils ein durch EXTENTSIZE definierter Speicherbereich gleichzeitig verwendet werden. Nur ganze, durch EXTENTSIZE definierte Speicherbereiche werden verwendet. Für eine optimale Speicherverwaltung können Sie daher beim Zuordnen eines Behälters eine geeignete Größe anhand der folgenden Formel bestimmen:

  ```
  (extent_size * n) + 1
  ```

  Dabei ist *extent_size* die Größe jedes durch EXTENTSIZE definierten Speicherbereichs im Tabellenbereich, und *n* ist die Anzahl dieser Speicherbereiche, die Sie in dem Behälter speichern wollen.
- Drei EXTENTSIZE große Speicherbereiche im Tabellenbereich sind für den Systemaufwand reserviert.
- Mindestens zwei EXTENTSIZE große Speicherbereiche sind erforderlich, um Tabellendaten des Benutzers zu speichern. (Diese zwei Speicherbereiche sind für die regulären Daten einer Tabelle vorgesehen, und nicht für Index-, Langfeld- oder LOB-Daten, die eigene Speicherbereiche benötigen.)
- Einheitenbehälter müssen logische Datenträger mit einer „zeichenspezifischen Schnittstelle" (d. h. keine physischen Datenträger) verwenden.
- Für DMS-Tabellenbereiche können auch Dateien anstelle von Einheiten (devices) verwendet werden. Zwischen einer Datei und einer Einheit gibt es keine betriebsbedingten Unterschiede. Jedoch kann eine Datei wegen des zur Laufzeit mit dem Dateisystem verbundenen Systemaufwands weniger effizient sein. Dateien sind in folgenden Situationen nützlich:
  - Wenn Einheiten nicht direkt unterstützt werden.
  - Wenn eine Einheit nicht verfügbar ist.
  - Wenn die Maximalleistung nicht erforderlich ist.
  - Wenn Sie keine Einheiten einrichten wollen.
- Wenn LOB- oder LONG VARCHAR-Daten verarbeitet werden, kann die Verwendung des Dateisystem-Cache Leistungsvorteile erbringen. Beachten Sie, dass LOB- und LONG VARCHAR-Daten nicht vom DB2-Pufferpool zwischengespeichert (gepuffert) werden.
- Einige Betriebssysteme erlauben den Betrieb physischer Einheiten, die größer als 2 GB sind. Sie sollten in Betracht ziehen, die physische Einheit in logische Einheiten zu partitionieren, damit kein Behälter größer als die vom Betriebssystem zugelassene Größe ist.

Hinzufügen von Behältern in DMS-Tabellenbereichen

Mit der Anweisung ALTER TABLESPACE können Sie einem bereits vorhandenen Tabellenbereich einen Behälter hinzufügen, um seine Speicherkapazität zu erhöhen. Der Inhalt des Tabellenbereichs wird dann über alle Behälter neu verteilt. Während dieser Neuverteilung wird der Zugriff auf den Tabellenbereich nicht eingeschränkt. Wenn mehr als ein Behälter hinzugefügt werden

muss, sollten diese Behälter gleichzeitig in einer Anweisung ALTER TABLESPACE bzw. innerhalb derselben Transaktion hinzugefügt werden, um zu vermeiden, dass der Datenbankmanager den Inhalt der Behälter mehr als einmal neu verteilen muss.

Sie sollten die Auslastung der Behälter für einen Tabellenbereich mit Hilfe des Befehl LIST TABLESPACE CONTAINERS oder LIST TABLESPACES überprüfen. Neue Behälter sollten hinzugefügt werden, bevor die vorhandenen Behälter fast oder ganz voll sind. Der neue Speicherbereich aller Behälter wird erst nach der Neuverteilung der Daten verfügbar.

Durch das Hinzufügen eines Behälters, der kleiner als vorhandene Behälter ist, wird eine ungleichmäßige Verteilung der Daten verursacht. Dies kann dazu führen, dass parallele Operationen wie das Vorablesen von Daten weniger effizient arbeiten, als sie es auf Behältern gleicher Größe könnten.

Überlegungen zum Tabellenbereichsentwurf

Dieser Abschnitt behandelt die folgenden Themen:

Überlegungen zur Ein-/Ausgabe (E/A) von Tabellenbereichen

Die Art und der Aufbau Ihres Tabellenbereichs bestimmen die Effizienz der Ein-/Ausgabeoperationen, die mit diesem Tabellenbereich erzielt werden kann. Es folgen einige Begriffe, mit denen Sie vertraut sein sollten, bevor Sie mit den weiteren Themen über Aufbau und Verwendung von Tabellenbereichen fortfahren:

Lesen großer Blöcke (Big Blocks)
Eine Leseoperation, bei der mehrere Seiten (in der Regel ein durch EXTENTSIZE definierter Bereich) in einer einzigen Anforderung abgerufen werden. Das Lesen mehrerer Seiten in einem Vorgang ist effizienter als das Lesen jeder Seite in getrennten Vorgängen.

Vorablesezugriff
Das Vorablesen der Seiten, auf die in einer Abfrage zugegriffen wird. Der Hauptzweck des Vorablesens ist die Verringerung von Antwortzeiten. Dies kann erreicht werden, wenn das Vorablesen von Seiten asynchron zur Ausführung der Abfrage stattfinden kann. Die besten Antwortzeiten werden erzielt, wenn entweder die CPU(s) oder das E/A-Subsystem mit maximaler Kapazität arbeiten.

Seitenlöschfunktion
Durch das Lesen und Ändern von Seiten sammeln diese sich im Pufferpool der Datenbank an. Wenn eine Datenseite eingelesen wird, wird sie in eine Seite des Pufferpools eingelesen. Wenn der Pufferpool ganz mit geänderten Seiten gefüllt ist, muss eine dieser geänderten Seiten auf die Platte geschrieben werden, bevor die neue Seite eingelesen werden kann. Bevor nun der Pufferpool gänzlich gefüllt wird, treten Seitenlöschagenten in Aktion, die geänderte Seiten auf die Platte schreiben und im Pufferpool löschen, um die Verfügbarkeit von Pufferpoolseiten für zukünftige Leseanforderungen sicherzustellen.

Immer wenn DB2 das Lesen großer Blöcke (Big Blocks) als vorteilhaft erkennt, werden große Blöcke gelesen. Dies tritt in der Regel beim Abrufen von Daten, die sequenzieller oder teilweise sequenzieller Art sind. Die Menge der Daten, die in einer Leseoperation gelesen werden, hängt von der Größe des mit EXTENTSIZE definierten Bereichs ab. Je größer der Wert für EXTENTSIZE ist, desto mehr Seiten können in einem Vorgang gelesen werden.

Die Art, wie der Bereich auf der Platte gespeichert ist, hat Einfluss auf die E/A-Effizienz. In einem DMS-Tabellenbereich mit Einheitenbehältern werden die Daten eher zusammenhängend auf der Platte gespeichert und können mit minimaler Suchzeit und Plattenlatenzzeit gelesen werden. Wenn jedoch Dateien verwendet werden, können die Daten vom Dateisystem in Teile getrennt an mehr als einer Position auf der Platte gespeichert werden. Dies geschieht häufiger bei Verwendung von SMS-Tabellenbereichen, bei denen Dateien um jeweils eine Seite erweitert werden, wodurch die Fragmentierung wahrscheinlicher wird. Eine große Datei, die zur Verwendung durch einen DMS-Tabellenbereich vorab zugeordnet wurde, führt eher dazu, dass die Datei auf der Platte zusammenhängend gespeichert wird, insbesondere wenn die Datei in einem noch ungenutzten Speicherbereich zugeordnet wurde.

Sie können den Grad des Vorablesens durch Optimieren des Parameters PREFETCHSIZE in der Anweisung CREATE TABLESPACE steuern. (Der Standardwert für alle Tabellenbereiche in der Datenbank wird durch den Konfigurationsparameter *dft_prefetch_sz* der Datenbank festgelegt.) Der Parameter PREFETCHSIZE gibt DB2 an, wie viele Seiten zu lesen sind, wenn ein

Vorablesezugriff ausgelöst wird. Wenn der Wert des Parameters PREFETCHSIZE auf ein Vielfaches des Parameters EXTENTSIZE in der Anweisung CREATE TABLESPACE gesetzt wird, können mehrere durch EXTENTSIZE definierte Bereiche parallel gelesen werden. (Der Standardwert für alle Tabellenbereiche in der Datenbank wird durch den Konfigurationsparameter *dft_extent_sz* der Datenbank festgelegt.) Der Parameter EXTENTSIZE gibt die Anzahl der 4-KB-Seiten an, die in einen Behälter geschrieben werden, bevor zum nächsten Behälter übergegangen wird.

Nehmen Sie zum Beispiel an, Sie hätten einen Tabellenbereich, der drei Einheiten verwendet. Wenn Sie den Wert für PREFETCHSIZE auf das Dreifache des Werts für EXTENTSIZE setzen, kann DB2 einen Lesezugriff in großen Blöcken von jeder Einheit parallel durchführen, wodurch der E/A-Durchsatz erheblich erhöht wird. Voraussetzungen sind, dass jede Einheit eine getrennte physische Einheit ist und dass der Controller über eine ausreichende Bandbreite verfügt, um den Datenstrom von jeder Einheit zu verarbeiten. Beachten Sie, dass DB2 eventuell die Parameter für den Vorablesezugriff zur Laufzeit aufgrund der Abfragegeschwindigkeit, der Pufferpoolauslastung und anderer Faktoren dynamisch anpassen muss.

Einige Dateisysteme (z. B. Journaled File System unter AIX) verfügen über eine eigene Vorablesemethode. In einigen Fällen kann der Vorablesezugriff des Dateisystems auf größere Datenmengen eingestellt sein als der Vorablesezugriff von DB2. Dies kann dazu führen, dass der Vorablesezugriff für SMS- und DMS-Tabellenbereiche mit Dateibehältern scheinbar eine bessere Leistung zeigt als der Vorablesezugriff für DMS-Tabellenbereiche mit Einheitenbehältern. Dies ist jedoch irreführend, da es sich wahrscheinlich um das Ergebnis einer zusätzlichen Ebene des Vorablesezugriffs handelt, die innerhalb des Dateisystems wirksam ist. Normalerweise sollten DMS-Tabellenbereiche jeder äquivalenten Konfiguration im Hinblick auf die Leistung überlegen sein.

Für ein effizientes Vorablesen (oder auch nur Lesen) muss eine ausreichende Anzahl verfügbarer Pufferpoolseiten vorhanden sein. Zum Beispiel könnte es eine Anforderung zum parallelen Vorablesezugriff geben, mit dem drei (durch EXTENTSIZE bestimmte) Bereiche aus einem Tabellenbereich gelesen werden und durch den für jede einzulesende Seite eine geänderte Seite aus dem Pufferpool herausgeschrieben wird. Die Anforderung zum Vorablesezugriff könnte so weit verlangsamt werden, dass sie mit der Abfrage nicht Schritt halten kann. Daher sollten Seitenlöschfunktionen in ausreichender Anzahl konfiguriert werden, um die Anforderungen von Vorablesezugriffen erfüllen zu können. Für jede reale Platte, die von der Datenbank verwendet wird, sollte mindestens eine Seitenlöschfunktion definiert werden. Weitere Informationen zu diesen Themen finden Sie im Handbuch *Systemverwaltung: Optimierung*.

Zuordnen von Tabellenbereichen zu Pufferpools

Jeder Tabellenbereich wird einem bestimmten Pufferpool zugeordnet. Der Standardpufferpool ist IBMDEFAULTBP. Wenn ein anderer Pufferpool einem Tabellenbereich zugeordnet werden soll, muss der Pufferpool existieren (er wird mit der Anweisung CREATE BUFFERPOOL definiert). Die Zuordnung wird bei der Erstellung des Tabellenbereichs (mit der Anweisung CREATE TABLESPACE) definiert. Die Zuordnung zwischen dem Tabellenbereich und dem Pufferpool kann mit der Anweisung ALTER TABLESPACE geändert werden.

Durch die Verwendung mehr als eines Pufferpools haben Sie die Möglichkeit, die Verwendung von Hauptspeicher durch die Datenbank zu konfigurieren, um die allgemeine Leistung zu verbessern. Bei Tabellenbereichen mit einer oder mehreren umfangreichen Tabellen, auf die die Benutzer wahlfrei zugreifen, kann die Größe des Pufferpools begrenzt werden, da ein Zwischenspeichern (Caching) der Datenseiten vielleicht keine Vorteile bietet. Dem Tabellenbereich für eine Online-Transaktionsanwendung kann ein größerer Pufferpool zugeordnet werden, so dass die Datenseiten, die von der Anwendung genutzt werden, länger im Cache zwischengespeichert und so schnellere Antwortzeiten erzielt werden können. Vorsicht ist aber bei der Konfiguration neuer Pufferpools geboten. Weitere Informationen zu diesem Thema finden Sie im Abschnitt zum Verwalten des Datenbankpufferpools im Handbuch *Systemverwaltung: Optimierung*.

Anmerkung: Wenn Sie ermittelt haben, dass für Ihre Datenbank eine Seitengröße von 8 KB, 16 KB oder 32 KB erforderlich ist, muss jeder Tabellenbereich mit einer dieser Seitengrößen einem Pufferpool mit derselben Seitengröße zugeordnet werden.

Der Speicher, der für alle Pufferpools benötigt wird, muss dem Datenbankmanager bereits beim Starten der Datenbank zur Verfügung stehen. Wenn DB2 den erforderlichen Speicherbereich nicht erhalten kann, startet der Datenbankmanager mit den Standardpufferpools (je einer mit 4-KB-Seiten, 8-KB-Seiten, 16-KB-Seiten und 32-KB-Seiten) und gibt eine Warnung aus.

In einer Umgebung mit partitionierten Datenbanken können Sie einen Pufferpool mit derselben Größe für alle Partitionen in der Datenbank erstellen. Sie können auch Pufferpools verschiedener Größen in verschiedenen Partitionen erstellen. Weitere Informationen zur Anweisung CREATE BUFFERPOOL finden Sie im Handbuch *SQL Reference*.

Zuordnen von Tabellenbereichen zu Knotengruppen

In einer Umgebung mit partitionierten Datenbanken wird jeder Tabellenbereich einer bestimmten Knotengruppe zugeordnet. Dadurch können die Merkmale des Tabellenbereichs auf jeden Knoten in der Knotengruppe übertragen werden. Die Knotengruppe muss vorhanden sein (sie wird mit der

Anweisung CREATE NODEGROUP definiert). Die Zuordnung zwischen dem Tabellenbereich und der Knotengruppe wird bei der Erstellung des Tabellenbereichs mit der Anweisung CREATE TABLESPACE definiert.

Die Zuordnung zwischen dem Tabellenbereich und der Knotengruppe kann mit der Anweisung ALTER TABLESPACE nicht geändert werden. Sie können lediglich die Spezifikationen des Tabellenbereichs für einzelne Partitionen innerhalb der Knotengruppe ändern. In einer Umgebung mit einer Einzelpartition wird jeder Tabellenbereich der Standardknotengruppe zugeordnet. Die Standardknotengruppe bei der Definition eines Tabellenbereichs ist IBMDEFAULTGROUP, sofern kein temporärer Systemtabellenbereich definiert wird. In diesem Fall wird die Standardknotengruppe IBMTEMPGROUP verwendet. Weitere Informationen zur Anweisung CREATE NODEGROUP finden Sie im Handbuch *SQL Reference*. Weitere Informationen zu Knotengruppen und zum physischen Datenbankentwurf finden Sie in „Entwerfen von Knotengruppen" auf Seite 116.

Zuordnen von Tabellen zu Tabellenbereichen

Bei der Festlegung, wie Tabellen Tabellenbereichen zugeordnet werden sollen, sollten Sie Folgendes berücksichtigen:

- Die Partitionierung Ihrer Tabellen

  Sie sollten mindestens sicherstellen, dass der Tabellenbereich, den sie auswählen, in einer Knotengruppe mit der gewünschten Partitionierung ist.

- Menge der Daten in der Tabelle

  Wenn Sie planen, viele kleine Tabellen in einem Tabellenbereich zu speichern, sollten Sie dazu die Verwendung eines SMS-Tabellenbereichs in Betracht ziehen. Die Vorteile von DMS-Tabellenbereichen im Hinblick auf die Effizienz bei E/A-Operationen und Speicherbereichsverwaltung sind bei kleinen Tabellen nicht so bedeutsam. Die Vorteile von SMS-Tabellenbereichen hinsichtlich der Speicherzuordnung von einer Seite gleichzeitig und nur bei Bedarf sind bei kleinen Tabellen attraktiver. Wenn eine Ihrer Tabellen größer ist oder Sie einen schnelleren Zugriff auf die Daten in den Tabellen benötigen, sollte ein DMS-Tabellenbereich mit einem kleinen Wert für EXTENTSIZE in Betracht gezogen werden.

  Eventuell empfiehlt es sich, einen separaten Tabellenbereich für jede umfangreiche Tabelle zu verwenden und kleine Tabellen gemeinsam in einem einzigen Tabellenbereich anzulegen. Diese Trennung ermöglicht Ihnen, anhand der Auslastung des Tabellenbereichs einen geeigneten Wert für den Parameter EXTENTSIZE auszuwählen. (Weitere Informationen hierzu finden Sie in „Auswählen eines Werts für EXTENTSIZE" auf Seite 142.)

- Typ der Daten in der Tabelle

  Sie könnten beispielsweise über Tabellen mit alten Daten verfügen, die relativ selten verwendet werden und bei denen der Endbenutzer eine längere

Antwortzeit für Abfragen, die diese Daten betreffen, vielleicht akzeptiert. In diesem Fall könnten Sie für diese „historischen" Tabellen einen anderen Tabellenbereich verwenden und diesem Tabellenbereich kostensparendere physische Einheiten mit einer langsameren Zugriffsgeschwindigkeit zuordnen.

Auf der anderen Seite können Sie einige wichtige Tabellen identifizieren, für die eine schnelle Verfügbarkeit und schnelle Antwortzeiten erforderlich sind. Diese Tabellen könnten Sie in einen Tabellenbereich stellen, der einer schnellen physischen Einheit zugeordnet ist, die die Anforderungen für diese wichtigen Daten besser erfüllt.

Wenn Sie mit DMS-Tabellenbereichen arbeiten, können Sie Ihre Tabellendaten auf drei verschiedene Tabellenbereiche verteilen: einen für Indexdaten, einen für LOB- und Langfelddaten sowie einen für reguläre Tabellendaten. Auf diese Weise können Sie die Tabellenbereichsmerkmale und die physischen Einheiten der Tabellenbereiche auswählen, die für die Daten am besten geeignet sind. Sie könnten z. B. die Indexdaten auf die schnellste verfügbare Einheit stellen und dadurch eine erhebliche Verbesserung der Leistung erzielen. Wenn Sie eine Tabelle auf DMS-Tabellenbereiche aufteilen, sollten Sie in Betracht ziehen, diese Tabellenbereiche zusammen zu sichern und wiederherzustellen, wenn die aktualisierende Wiederherstellung (Rollforward) aktiviert ist. SMS-Tabellenbereiche unterstützen diese Art der Datenverteilung auf Tabellenbereiche nicht.

- Verwaltungserfordernisse

  Einige Verwaltungsfunktionen können auf Tabellenbereichsebene anstatt auf Datenbank- oder Tabellenebene ausgeführt werden. Wenn Sie beispielsweise eine Sicherungskopie eines Tabellenbereichs anstelle der Sicherungskopie einer Datenbank erstellen, können Sie Ihre Zeit und Ressourcen besser ausnutzen. Diese Vorgehensweise ermöglicht Ihnen, Tabellenbereiche mit umfangreichen Änderungen häufig zu sichern und von den Tabellenbereichen, die wenig geändert werden, nur gelegentlich neue Sicherungskopien anzulegen.

  Sie können eine Datenbank oder einen Tabellenbereich wiederherstellen. Wenn Tabellen, die sich nicht aufeinander beziehen, keine gemeinsamen Tabellenbereiche benutzen, haben Sie die Option, kleinere Teile Ihrer Datenbank wiederherzustellen und den Aufwand verringern.

  Ein gutes Verfahren besteht darin, Tabellen, die voneinander abhängig sind, in einer Gruppe von Tabellenbereichen zusammenzufassen. Diese Tabellen könnten über referenzielle Integritätsbedingungen oder andere definierte Geschäftsregeln voneinander abhängig sein.

  Falls Sie eine bestimmte Tabelle häufig löschen und erneut definieren müssen, können Sie die Tabelle in einem eigenen Tabellenbereich definieren, weil das Löschen eines DMS-Tabellenbereichs effizienter ist als das Löschen einer Tabelle.

Auswählen eines Werts für EXTENTSIZE

Der Wert des Parameters EXTENTSIZE für einen Tabellenbereich ist die Anzahl der Seiten mit Tabellendaten, die in einen Behälter geschrieben werden, bevor Daten in den nächsten Behälter geschrieben werden. Beim Auswählen eines Werts für EXTENTSIZE ist Folgendes zu beachten:

- Größe und Art der Tabellen im Tabellenbereich

  In DMS-Tabellenbereichen wird einer Tabelle gleichzeitig jeweils ein durch EXTENTSIZE definierter Speicherbereich zugeordnet. Wenn beim Füllen der Tabelle mit Daten ein durch EXTENTSIZE definierter Bereich voll ist, wird ein neuer Bereich dieser Größe zugeordnet.

  Eine Tabelle besteht aus folgenden separaten Tabellenobjekten:

  – Ein Datenobjekt. Hier werden die regulären Spaltendaten gespeichert.
  – Ein Indexobjekt. Hier werden alle für die Tabelle definierten Indizes gespeichert.
  – Ein Langfeldobjekt. Hier werden Langfelddaten gespeichert, wenn die Tabelle eine oder mehrere Spalten mit LONG-Datentypen besitzt.
  – Zwei LOB-Objekte. Wenn die Tabelle eine oder mehrere LOB-Spalten hat, werden diese in folgenden beiden Tabellenobjekten gespeichert:
    - Ein Tabellenobjekt für die LOB-Daten
    - Ein zweites Tabellenobjekt für Metadaten, die die LOB-Daten beschreiben

  Jedes Tabellenobjekt wird getrennt gespeichert und ordnet nach Bedarf neue (durch EXTENTSIZE definierte) Bereiche zu. Jedes Tabellenobjekt wird außerdem mit einem Metadatenobjekt verbunden, das als *Speicherbereichsmaske* bezeichnet wird und alle durch EXTENTSIZE definierten Bereiche im Tabellenbereich beschreibt, die zu dem Tabellenobjekt gehören. Der Speicherbereich für Speicherbereichsmasken wird ebenfalls jeweils in der Größe von EXTENTSIZE zugeordnet.

  Die Erstzuordnung von Speicherbereich für eine Tabelle umfasst daher zwei EXTENTSIZE-Bereiche für jedes Tabellenobjekt. Wenn Sie viele kleine Tabellen in einem Tabellenbereich haben, ist es möglich, dass eine relativ große Menge an Speicher für eine relativ kleine Menge von Daten zugeordnet ist. In einem solchen Fall sollten Sie einen kleinen Wert für EXTENTSIZE definieren oder einen SMS-Tabellenbereich verwenden, in dem jeweils eine Seite gleichzeitig zugeordnet wird.

  Wenn Sie andererseits über eine umfangreiche Tabelle mit einer hohen Zuwachsrate verfügen und einen DMS-Tabellenbereich mit einem niedrigen Wert für EXTENTSIZE verwenden, könnte durch häufiges Zuordnen zusätzlichen Speicherbereichs unnötiger Systemaufwand entstehen.

- Art des Zugriffs auf die Tabellen

  Wenn auf die Tabellen durch zahlreiche Abfragen oder Transaktionen zugegriffen wird, die große Datenmengen verarbeiten, kann das Vorablesen von Daten aus den Tabellen eine wesentliche Leistungsverbesserung bewirken. (Das Handbuch *Systemverwaltung: Optimierung* enthält Informationen zum Vorablesen von Daten sowie zur Beziehung zwischen dem Vorablesezugriff und dem Wert für EXTENTSIZE.)
- Minimal erforderliche Anzahl mit EXTENTSIZE definierter Speicherbereiche

  Falls nicht genügend Platz für fünf durch EXTENTSIZE definierte Speicherbereiche des Tabellenbereichs in den Behältern vorhanden ist, wird der Tabellenbereich nicht erstellt.

Empfehlungen für temporäre Tabellenbereiche

Es wird empfohlen, einen einzelnen temporären SMS-Tabellenbereich zu definieren, dessen Seitengröße der Seitengröße entspricht, die in den meisten regulären Tabellenbereichen verwendet wird. Dies sollte für typische Umgebungen und Auslastungen geeignet sein. Es kann jedoch vorteilhaft sein, mit unterschiedlichen Konfigurationen für temporäre Tabellenbereiche und Auslastungen zu experimentieren. Sie sollten die folgenden Punkte beachten:

- Auf temporäre Tabellen wird meist gruppenweise und sequenziell zugegriffen. Das heißt, eine Gruppe von Zeilen wird eingefügt oder eine Gruppe sequenzieller Zeilen wird abgerufen. Daher führt eine größere Seitengröße in der Regel zu einer besseren Leistung, weil weniger E/A-Anforderungen logischer oder physischer Seiten erforderlich sind, um eine bestimmte Datenmenge einzulesen. Dies ist nicht immer der Fall, wenn die durchschnittliche Zeilengröße einer temporären Tabelle kleiner ist als die Seitengröße dividiert durch 255. Ungeachtet der Seitengröße können maximal 255 Zeilen auf einer Seite vorhanden sein. Eine Abfrage, die z. B. eine temporäre Tabelle mit 15-Byte-Zeilen erfordert, könnte von einem temporären Tabellenbereich mit einer Seitengröße von 4 KB besser bedient werden, da alle 255 solcher Zeilen in eine 4-KB-Seite aufgenommen werden können. Eine 8-KB-Seite (oder größere Seite) ergäbe mindestens 4 KB (oder mehr) Byte ungenutzten Speicherbereich auf jeder Seite der temporären Tabelle und würde die Anzahl der erforderlichen E/A-Anforderungen nicht reduzieren.
- Wenn über fünfzig Prozent der regulären Tabellenbereiche in der Datenbank dieselbe Seitengröße verwenden, kann es vorteilhaft sein, die temporären Tabellenbereiche mit derselben Seitengröße zu definieren. Der Vorteil liegt darin, dass der temporäre Tabellenbereich mit dieser Konfiguration denselben Pufferpoolspeicherbereich mit den meisten oder allen regulären Tabellenbereichen gemeinsam benutzen kann. Dadurch wird die Optimierung des Pufferpools wiederum vereinfacht.
- Wenn Sie mit Hilfe eines temporären Tabellenbereichs eine Tabelle reorganisieren, muss die Seitengröße des temporären Tabellenbereichs mit der Seitengröße der Tabelle übereinstimmen. Deshalb sollten Sie sicherstellen,

dass temporäre Tabellenbereiche vorhanden sind, die für jede Seitengröße definiert sind, die von vorhandenen Tabellen verwendet wird, die Sie vielleicht mit Hilfe eines temporären Tabellenbereichs reorganisieren.

Sie können eine Reorganisierung auch ohne temporären Tabellenbereich ausführen, indem Sie die Tabelle „am Ort" reorganisieren, das heißt, direkt im Zieltabellenbereich. Natürlich setzt eine Reorganisierung „am Ort" voraus, dass im Zieltabellenbereich zusätzlicher Speicherbereich für den Reorganisierungsprozess vorhanden ist. Weitere Informationen zur Reorganisierung von Tabellen finden Sie im Handbuch *Systemverwaltung: Optimierung*.

- Im allgemeinen, wenn temporäre Tabellenbereiche mit unterschiedlichen Seitengrößen vorhanden sind, wählt das Optimierungsprogramm am häufigsten den temporären Tabellenbereich mit dem größten Pufferpool aus. In solchen Fällen empfiehlt es sich, einem der temporären Tabellenbereiche einen großen Pufferpool und den übrigen einen kleineren Pufferpool zuzuordnen. Eine solche Pufferpoolzuordnung hilft bei der Sicherstellung einer effizienten Auslastung des Hauptspeichers. Wenn z. B. Ihr Katalogtabellenbereich 4-KB-Seiten verwendet und die übrigen Tabellenbereiche 8-KB-Seiten, kann sich folgende Konfiguration für temporäre Tabellenbereiche am besten eignen: ein einzelner temporärer 8-KB-Tabellenbereich mit einem großen Pufferpool und ein einzelner 4-KB-Tabellenbereich mit einem kleinen Pufferpool.

  Anmerkung: Katalogtabellenbereiche sind auf die Verwendung von 4-KB-Seiten beschränkt. Daher erzwingt der Datenbankmanager immer das Vorhandensein eines temporären 4-KB-Tabellenbereichs, um die Reorganisierung von Katalogtabellen zu ermöglichen.
- Es bringt im Allgemeinen keinen Vorteil, mehr als einen temporären Tabellenbereich von jeder Seitengröße zu definieren.
- Für temporäre Tabellenbereiche sind SMS-Bereiche aus den folgenden Gründen DMS-Bereichen meist vorzuziehen:
  - Plattenspeicherplatz wird im SMS nach Bedarf zugeordnet, wohingegen er im DMS zuvor zugeordnet werden muss. Eine vorherige Zuordnung kann ein Problem darstellen: Temporäre Tabellenbereiche enthalten Übergangsdaten, die einen extremen Höchstspeicherbedarf und einen wesentlich geringeren durchschnittlichen Speicherbedarf besitzen können. Bei DMS muss der Höchstspeicherbedarf vorab zugeordnet werden, wohingegen bei SMS der zusätzliche Plattenspeicherplatz außerhalb der Spitzenlastzeiten für andere Zwecke verwendet werden kann.
  - Der Datenbankmanager versucht, temporäre Tabellenseiten im Speicher zu behalten und sie nicht auf die Platte auszulagern. Im Endeffekt sind die Leistungsvorteile von DMS weniger signifikant.
  - SMS-Behälter können das Zwischenspeichern von Dateisystemen ausnutzen. DMS-Behälter können dies nicht.

Empfehlungen für Katalogtabellenbereiche

Ein SMS-Tabellenbereich empfiehlt sich aus folgenden Gründen für Datenbankkataloge:

- Der Datenbankkatalog besteht aus zahlreichen Tabellen unterschiedlicher Größen. Bei der Verwendung eines DMS-Tabellenbereichs wird mindestens ein Speicherbereich in der Größe von zwei EXTENTSIZE-Bereichen für jedes Tabellenobjekt zugeordnet. Je nachdem, welcher Wert für EXTENTSIZE ausgewählt wurde, kann dies zu einer erheblichen Menge an zugeordnetem, aber ungenutztem Speicher führen. Wenn ein DMS-Tabellenbereich verwendet wird, sollte ein kleiner Wert für EXTENTSIZE (zwei bis vier Seiten) ausgewählt werden. Ansonsten sollte ein SMS-Tabellenbereich verwendet werden.
- In den Katalogtabellen gibt es Spalten mit großen Objekten (LOB-Spalten). LOB-Daten werden nicht mit anderen Daten im Pufferpool behalten, sondern jedes Mal, wenn sie benötigt werden, von der Platte gelesen. Das Lesen von LOB-Daten von der Platte verlangsamt die Leistung. Da ein Dateisystem in der Regel über eigene Mechanismen zum Zwischenspeichern (Caching) von Daten verfügt, kann die Verwendung eines SMS-Tabellenbereichs oder eines DMS-Tabellenbereichs, der auf Dateibehältern basiert, die E/A-Operationen möglicherweise umgehen, wenn auf die LOB-Daten zuvor bereits zugegriffen wurde.

Unter diesen Gesichtspunkten erweist sich ein SMS-Tabellenbereich als die etwas günstigere Wahl für die Katalogtabellen.

Ein anderer Faktor, der zu berücksichtigen wäre, ist die Frage, ob der Katalogtabellenbereich in der Zukunft erweitert werden müsste. Obwohl einige Plattformen die Erweiterung des zugrundeliegenden Speichers für SMS-Behälter unterstützen und die Möglichkeit einer umgeleiteten Wiederherstellung zur Vergrößerung eines SMS-Tabellenbereichs gegeben ist, macht ein DMS-Tabellenbereich das Hinzufügen neuer Behälter einfacher.

Überlegungen zur Art der Auslastung

Der primäre Typ der Auslastung, die von DB2 in Ihrer Umgebung verwaltet wird, kann Ihre Wahl beeinflussen, welcher Typ von Tabellenbereich zu verwenden und welche Seitengröße anzugeben ist. Eine OLTP-Auslastung (Online-Transaktionsverarbeitung) ist durch Transaktionen charakterisiert, die einen wahlfreien Zugriff auf Daten benötigen und die in der Regel nur kleine Datenmengen zurückliefern. In Anbetracht dessen, dass der Zugriff wahlfrei nur auf eine bzw. wenige Seiten erfolgt, ist ein Vorablesezugriff nicht möglich.

In diesem Fall zeigen DMS-Tabellenbereiche mit Einheitenbehältern die beste Leistung. DMS-Tabellenbereiche mit Dateibehältern oder SMS-Tabellenbereiche sind ebenfalls sinnvolle Möglichkeiten für eine OLTP-Auslastung, wenn es nicht auf maximale Leistung ankommt. Wenn nur wenig oder gar keine sequenziellen E/A-Operationen zu erwarten sind, spielen die Werte der Para-

meter EXTENTSIZE und PREFETCHSIZE in der Anweisung CREATE TABLESPACE keine wichtige Rolle für die E/A-Effizienz.

Eine Abfrageauslastung wird durch Transaktionen charakterisiert, die einen sequenziellen oder teilweise sequenziellen auf Daten benötigen und in der Regel große Datenmengen zurückliefern. Ein DMS-Tabellenbereich mit mehreren Einheitenbehältern (wobei sich jeder Behälter auf einer separaten Platte befindet) bietet das größte Potenzial für einen effizienten parallelen Vorablesezugriff. Der Wert des Parameters PREFETCHSIZE in der Anweisung CREATE TABLESPACE sollte das Produkt aus dem Wert des Parameters EXTENTSIZE multipliziert mit der Anzahl der Einheitenbehälter sein. Dadurch kann DB2 von allen Behältern parallel vorablesen.

Eine sinnvolle Alternative für eine Abfrageauslastung ist die Verwendung von Dateien, wenn das Dateisystem über eine eigene Vorablesefunktion verfügt. Die Dateien können entweder zu DMS-Tabellenbereichen mit Dateibehältern gehören oder Dateien für SMS-Tabellenbereiche sein. Beachten Sie, dass Sie bei Verwendung von SMS sicherstellen müssen, dass die Verzeichnisbehälter getrennten physischen Datenträgern zugeordnet sind, um E/A-Parallelität zu erreichen.

Das Ziel für eine gemischte Auslastung besteht darin, einzelne E/A-Anforderungen so effizient wie möglich für OLTP-Auslastungen zu machen und die Effizienz paralleler E/A-Operationen für Abfrageauslastungen zu maximieren.

Beim Ermitteln der Seitengröße für einen Tabellenbereich sind folgende Überlegungen zu berücksichtigen:

- Für OLTP-Anwendungen, die wahlfreie Lese- und Schreiboperationen durchführen, ist eine geringere Seitengröße empfehlenswert, weil dabei weniger Pufferspeicher durch unerwünschte Zeilen belegt wird.
- Für DSS-Anwendungen (Decision Support System), die jeweils auf eine große Anzahl aufeinander folgender Zeilen zugreifen, sind größere Seiten empfehlenswert, weil dadurch weniger E/A-Anforderungen erforderlich sind, um eine bestimmte Anzahl Zeilen zu lesen. Es gibt jedoch eine Ausnahme von dieser Regel. Diese betrifft den Fall, dass die Zeilengröße kleiner als folgender Wert ist:

  ```
  seitengröße / 255
  ```

  Bei dieser Größe bleibt auf jeder Seite Speicherbereich ungenutzt (weil jede Seite maximal 255 Zeilen enthalten kann). In diesem Fall ist eine geringere Seitengröße zu empfehlen.
- Durch größere Seiten können Sie möglicherweise die Zahl der Indexstufen reduzieren.
- Größere Seiten unterstützen längere Zeilen.

- Auf 4-KB-Standardseiten sind Tabellen auf 500 Spalten begrenzt, während auf größeren Seiten (8 KB, 16 KB und 32 KB) 1012 Spalten unterstützt werden.
- Die Maximalgröße des Tabellenbereichs ist proportional zur Seitengröße des Tabellenbereichs. Die Begrenzungen sind im Handbuch *SQL Reference* dokumentiert.

Auswählen eines SMS- oder DMS-Tabellenbereichs

Bei der Entscheidung, welcher Tabellenbereichstyp zum Speichern der Daten verwendet werden soll, sollten Sie das Pro und Kontra gut abwägen.

Vorteile eines SMS-Tabellenbereichs:

- Der Speicher wird vom System erst dann zugeordnet, wenn er benötigt wird.
- Beim Erstellen einer Datenbank sind weniger vorbereitende Arbeiten erforderlich, weil Sie keine Behälter vordefinieren müssen.

Vorteile eines DMS-Tabellenbereichs:

- Die Größe eines Tabellenbereichs kann durch Hinzufügen von Behältern erweitert werden (Anweisung ALTER TABLESPACE). Vorhandene Daten werden automatisch auf die neue Gruppe von Behältern verteilt, um die optimale E/A-Effizienz zu erhalten.
- Eine Tabelle kann nach dem Typ der zu speichernden Daten auf mehrere Tabellenbereiche verteilt werden:
  - Langfeld- und LOB-Daten
  - Indizes
  - Reguläre Tabellendaten

  Vielleicht sollen die Tabellendaten zur Erhöhung der Leistung oder zur Vergrößerung der für eine Tabelle gespeicherten Datenmenge getrennt gehalten werden. Sie könnten beispielsweise eine Tabelle mit 64 GB für reguläre Tabellendaten, 64 GB für Indexdaten und 2 TB für Langfelddaten anlegen. Bei Verwendung von 8-KB-Seiten können die Tabellendaten und die Indexdaten bis zu 128 GB umfassen. Bei Verwendung von 16-KB-Seiten können sie bis zu 256 GB umfassen. Bei Verwendung von 32-KB-Seiten können die Tabellendaten und die Indexdaten bis zu 512 GB umfassen.
- Es ist möglich, die Speicherposition der Daten auf der Platte zu steuern, sofern das Betriebssystem dies zulässt.
- Wenn sich alle Tabellendaten in einem einzigen Tabellenbereich befinden, ist der Systemaufwand beim Löschen und erneuten Definieren eines Tabellenbereichs geringer, als dies beim Löschen und erneuten Definieren einer Tabelle der Fall wäre.

- Im allgemeinen gilt, dass sich mit einer sinnvoll organisierten Gruppe von DMS-Tabellenbereichen eine bessere Leistung erzielen lässt als mit SMS-Tabellenbereichen.

Anmerkung: Bei Solaris und PTX (IBM NUMA-Q) wird die Verwendung von DMS-Tabellenbereichen mit unformatierten Einheiten für leistungskritische Auslastungen dringend empfohlen.

Generell lassen sich kleine Datenbanken, die von einem kleinen Personenkreis genutzt werden, am einfachsten mit SMS-Tabellenbereichen verwalten. Andererseits sollten Sie für umfangreiche, wachsende Datenbanken SMS-Tabellenbereiche nur für temporäre Tabellenbereiche und den Katalogtabellenbereich sowie für jede Tabelle getrennte DMS-Tabellenbereiche mit mehreren Behältern verwenden. Außerdem ist es wahrscheinlich sinnvoll, Langfelddaten und Indizes in eigenen Tabellenbereichen zu speichern.

Wenn Sie sich entschließen, DMS-Tabellenbereiche mit Einheitenbehältern zu verwenden, müssen Sie bereit sein, Ihre Umgebung zu optimieren und zu verwalten. Weitere Informationen finden Sie im Abschnitt über die Leistung von DMS-Einheiten im Handbuch *Systemverwaltung: Optimierung*.

Optimieren von Leistung, wenn Daten auf RAID-Einheiten platziert werden

In diesem Abschnitt wird beschrieben, wie Sie die Leistung optimieren können, wenn Daten auf RAID-Einheiten platziert werden. In allgemeinen sollten Sie folgende Maßnahmen für jeden Tabellenbereich ergreifen, der eine RAID-Einheit verwendet:

- Definieren Sie einen einzelnen Behälter für den Tabellenbereich (der eine RAID-Einheit verwendet).
- Setzen Sie den Wert für EXTENTSIZE des Tabellenbereichs so, dass er der Größe des einheitenübergreifend gespeicherten RAID-Datenblocks (Stripe Size) oder einem Vielfachen der Größe entspricht.
- Stellen Sie sicher, dass der Wert für PREFETCHSIZE des Tabellenbereichs die folgenden Bedingungen erfüllt:
  - Er entspricht der Größe des einheitenübergreifend gespeicherten RAID-Datenblocks (Stripe Size) multipliziert mit der Anzahl paralleler RAID-Einheiten (oder einem ganzen Vielfachen dieses Produkts).
  - Er ist ein Vielfaches des Werts für EXTENTSIZE.
- Verwenden Sie die Registrierungsvariable DB2_PARALLEL_IO (siehe „DB2_PARALLEL_IO" auf Seite 149), um die parallele Ein-/Ausgabe für den Tabellenbereich zu aktivieren.
- Verwenden Sie die Registrierungsvariable DB2_STRIPED_CONTAINERS (siehe „DB2_STRIPED_CONTAINERS" auf Seite 149), um sicherzustellen, dass die Grenzen der durch EXTENTSIZE definierten Bereiche im Tabellenbereich ausgerichtet werden.

DB2_PARALLEL_IO

Wenn Daten aus Tabellenbereichsbehältern gelesen werden oder in diese geschrieben werden, kann DB2 die parallele Ein-/Ausgabe verwenden, falls die Anzahl der Behälter in der Datenbank größer als 1 ist. Es gibt jedoch Fälle, in denen es vorteilhaft wäre, die parallele Ein-/Ausgabe für einzelne Tabellenbereichsbehälter zu aktivieren. Wenn z. B. der Behälter auf einer einzelnen RAID-Einheit erstellt wird, die aus mehr als einer physischen Platte besteht, ist es vielleicht sinnvoll, Aufrufe zum parallelen Lesen und Schreiben absetzen zu können.

Sie können die Registrierungsvariable DB2_PARALLEL_IO verwenden, um die parallele Ein-/Ausgabe für einen Tabellenbereich zu erzwingen, der einen einzelnen Behälter besitzt. Diese Variable kann auf "*" (Stern) gesetzt werden, d. h. sich auf jeden Tabellenbereich beziehen, oder sie kann auf eine Liste von Tabellenbereichs-IDs gesetzt werden, die durch Komma voneinander getrennt sind. Zum Beispiel:

```
db2set DB2_PARALLEL_IO=*        {parallele E/A für alle
                                 Tabellenbereiche aktivieren}
db2set DB2_PARALLEL_IO=1,2,4,8  {parallele E/A für die Tabellenbereiche 1, 2,
                                 4 und 8 aktivieren}
```

Nach dem Setzen der Registrierungsvariablen muss DB2 gestoppt (**db2stop**) und anschließend erneut gestartet (**db2start**) werden, damit die Änderungen wirksam werden.

DB2_STRIPED_CONTAINERS

Zur Zeit wird beim Erstellen eines DMS-Tabellenbereichsbehälters (Einheit oder Datei) eine Kennung, die eine Seite lang ist, am Anfang des Behälters gespeichert. Die übrigen Seiten sind für die Datenspeicherung durch DB2 verfügbar und werden in Blöcke von der Größe des Werts für EXTENTSIZE gruppiert.

Wenn RAID-Einheiten für Tabellenbereichsbehälter verwendet werden, wird empfohlen, dass der Tabellenbereich mit einem Wert für EXTENTSIZE erstellt wird, der der Größe des einheitenübergreifend gespeicherten RAID-Datenblocks (Stripe Size) oder einem Vielfachen dieser Größe entspricht. Jedoch können die Speicherbereiche aufgrund der Behälterkennung, die eine Seite lang ist, nicht mit den einheitenübergreifend gespeicherten RAID-Datenblöcke ausgerichtet werden, und es kann während einer E/A-Anforderung notwendig werden, auf mehr physische Platten zuzugreifen als optimal wäre.

DMS-Tabellenbereichsbehälter können so erstellt werden, dass sich die Kennung in ihrem eigenen separaten Speicherbereich (in der vollen EXTENTSIZE-Größe) befindet. Dadurch wird das oben beschriebene Problem umgangen, aber es ist ein zusätzlicher Speicherbereich in Größe von EXTENTSIZE innerhalb des Behälters für den Systemaufwand erforderlich. Sie müssen die DB2-Registrierungsvariable DB2_STRIPED_CONTAINERS auf "ON" setzen. Anschließend stoppen Sie das Exemplar und starten es erneut, um Behälter auf diese Weise zu erstellen:

```
db2set DB2_STRIPED_CONTAINERS=ON
db2stop
db2start
```

Jeder DMS-Behälter, der mit der Anweisung CREATE TABLESPACE oder ALTER TABLESPACE erstellt wurde, enthält Kennungen, die einen Speicherbereich einnehmen, der einen ganzen EXTENTSIZE-Wert groß ist. Vorhandene Behälter bleiben unverändert.

Zum Stoppen der Erstellung von Behältern mit diesem Attribut setzen Sie die Variable zurück (der Standardwert ist NULL). Anschließend stoppen Sie Ihr Exemplar und starten es erneut:

```
db2set DB2_STRIPED_CONTAINERS=
db2stop
db2start
```

Die Steuerzentrale und der Befehl LIST TABLESPACE CONTAINERS zeigen nicht an, ob ein Behälter als einheitenübergreifend (Striped) erstellt wurde. Sie verwenden die Bezeichnung "Datei" oder "Einheit", abhängig davon, wie der Behälter erstellt wurde. Zum Prüfen, ob ein Behälter als einheitenübergreifend erstellt wurde, können Sie die Option /DTSF von DB2DART verwenden, um Informationen zu Tabellenbereichen und Behältern auszulesen, und sich dann das Typfeld für den fraglichen Behälter ansehen. Sie können mit Hilfe der APIs zum Abfragen von Behältern (**sqlbftcq** und **sqlbtcq**) eine einfache Anwendung erstellen, die den Typ anzeigt.

Überlegungen zum Entwerfen einer zusammengeschlossenen Datenbank

Beim Entwerfen einer zusammengeschlossenen Datenbank müssen Sie die folgenden Aspekte berücksichtigen:

- Platzbedarf
- Prioritätensetzung im Netzwerk

Die Daten, auf die von einer zusammengeschlossenen Datenbank zugegriffen werden kann, sind in der Regel nicht in der Datenbank gespeichert. Verweise auf Datenquellentabellen und -sichten sind innerhalb des Systemkatalogs gespeichert, aber die tatsächlichen Daten befinden sich an der Datenquelle. Als solche kann eine zusammengeschlossene Datenbank eventuell weniger Speicherplatz als eine herkömmliche Datenbank erfordern. Diese allgemeine Regel ist möglicherweise nicht anwendbar, wenn die Abfragen aufgrund von Systemunterschieden bei Sortierfolgen oder fehlenden Funktionen an einer Datenquelle lokal ausgeführt werden müssen. In diesem Fall werden Tabellen für die Verarbeitung in DB2 erstellt.

Da sich die meisten Daten von Systemen zusammengeschlossener Datenbanken in der Regel an einer oder mehreren Datenquellen befinden, die in einem Netzwerk verteilt sind, ziehen Sie in Betracht, die Ressourcen zu ändern, die DB2 und Ihrem Netzwerksystem zugeordnet sind. Sie erzielen möglicherweise Leistungsverbesserungen, wenn Sie für das Netzwerk mit dem DB2-System mehr Ressourcen zuordnen als für den Datenbankmanager selbst.

Kapitel 9. Entwerfen verteilter Datenbanken

Eine Transaktion wird in DB2 allgemein als *Arbeitseinheit* bezeichnet. Eine Arbeitseinheit ist eine wiederherstellbare Folge von Operationen innerhalb eines Anwendungsprozesses. Der Datenbankmanager kann mit ihrer Hilfe sicherstellen, dass sich eine Datenbank in einem konsistenten Status befindet. Jeder Lese- oder Schreibvorgang in der Datenbank erfolgt innerhalb einer Arbeitseinheit.

Eine Banktransaktion könnte beispielsweise darin bestehen, dass eine Geldsumme von einem Sparkonto auf ein Girokonto überwiesen wird. Nachdem die Anwendung einen Betrag vom Sparkonto abgezogen hat, sind die beiden Konten inkonsistent und bleiben es, bis der Betrag auf dem Girokonto gutgeschrieben ist. Wenn *beide* Schritte zu Ende geführt sind, ist ein Konsistenzzustand erreicht. Die Änderungen können festgeschrieben und anderen Anwendungen zur Verfügung gestellt werden.

Eine Arbeitseinheit beginnt, wenn die erste SQL-Anweisung für die Datenbank ausgeführt wird. Die Anwendung muss die Arbeitseinheit beenden, indem sie die Anweisung COMMIT oder ROLLBACK absetzt. Die Anweisung COMMIT schreibt alle Änderungen, die innerhalb der Arbeitseinheit vorgenommen wurden, permanent fest. Die Anweisung ROLLBACK löscht diese Änderungen aus der Datenbank. Wenn bei einer normalen Beendigung der Anwendung keine dieser Anweisungen abgesetzt wird, wird die Arbeitseinheit automatisch festgeschrieben. Wenn die Anwendung inmitten einer Arbeitseinheit abnormal beendet wird, wird die Arbeitseinheit automatisch rückgängig gemacht. Sobald eine Anweisung COMMIT oder ROLLBACK abgesetzt wurde, kann sie nicht mehr gestoppt werden. Bei einigen Multithread-Anwendungen oder Betriebssystemen (z. B. Windows) wird die Arbeitseinheit, wenn eine Anwendung normal, jedoch ohne explizite Ausführung einer dieser Anweisungen beendet wird, automatisch rückgängig (ROLLBACK) gemacht. Es wird daher empfohlen, dafür zu sorgen, dass Anwendungen vollständige Arbeitseinheiten immer explizit mit einer Anweisung COMMIT festschreiben oder mit einer Anweisung ROLLBACK rückgängig machen. Wenn ein Teil einer Arbeitseinheit nicht erfolgreich beendet wird, werden die Aktualisierungen rückgängig gemacht, so dass die beteiligten Tabellen in dem Zustand verbleiben, in dem sie vor Beginn der Transaktion waren. Auf diese Weise wird sichergestellt, dass Anforderungen weder verloren gehen noch mehrfach ausgeführt werden.

Unter den folgenden Themen finden Sie weitere Informationen:

- „Verwenden einer einzelnen Datenbank in einer Transaktion"
- „Verwenden mehrerer Datenbanken in einer Transaktion" auf Seite 155
- „Weitere Überlegungen zur Konfiguration" auf Seite 161
- „Prozess der zweiphasigen Festschreibung" auf Seite 164
- „Beheben von Problemen bei der zweiphasigen Festschreibung" auf Seite 167.

Informationen zur Erstellung von Anwendungen für verteilte Datenbanken finden Sie in den Handbüchern *Application Development Guide* und *CLI Guide and Reference*.

Verwenden einer einzelnen Datenbank in einer Transaktion

Die einfachste Form der Transaktion besteht darin, innerhalb einer Arbeitseinheit lediglich für eine Datenbank Lese- und Schreibvorgänge auszuführen. Diese Art des Datenbankzugriffs wird als *ferne Arbeitseinheit* bezeichnet.

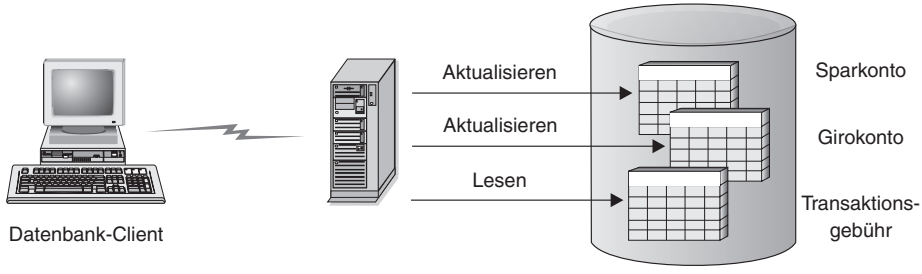

Abbildung 30. Verwenden einer einzelnen Datenbank in einer Transaktion

Abb. 30 zeigt einen Datenbank-Client, der eine Geldtransaktionsanwendung ausführt. Diese Anwendung greift auf eine Datenbank zu, die Tabellen für Giro- und Sparkonten sowie einen Plan für die Transaktionsgebühren enthält. Die Anwendung muss folgende Schritte ausführen:

- Akzeptieren des Betrags, der von der Benutzerschnittstelle überwiesen werden soll
- Subtrahieren des Betrags vom Sparkonto und Ermitteln des neuen Kontostands
- Lesen des Gebührenplans, um die Transaktionsgebühr für ein Sparkonto mit dem angegebenen Kontostand zu bestimmen

- Subtrahieren der Transaktionsgebühr vom Sparkonto
- Gutschreiben des Überweisungsbetrags auf dem Girokonto
- Festschreiben der Transaktion (Arbeitseinheit)

Zur Einrichtung einer solchen Anwendung sind folgende Schritte auszuführen:

1. Erstellen der Tabellen für das Sparkonto, das Girokonto und den Gebührenplan für die Transaktion innerhalb derselben Datenbank (siehe "Implementieren des Datenbankentwurfs" im Handbuch *Systemverwaltung: Implementierung*)
2. Wenn physisch fern, Einrichten des Datenbankservers für die Verwendung des geeigneten Übertragungsprotokolls (siehe die Beschreibung im Handbuch *Installation und Konfiguration Ergänzung*
3. Wenn physisch fern, Katalogisieren des Knotens und der Datenbank, um die Datenbank auf dem Datenbankserver zu identifizieren (siehe die Beschreibung in den Handbüchern *Einstieg*)
4. Vorkompilieren Ihres Anwendungsprogramms, um eine Verbindung des Typs 1 anzugeben, d. h. Angeben von CONNECT 1 (Standardwert) im Befehl PRECOMPILE PROGRAM (siehe die Beschreibung im Handbuch *Application Development Guide*

Verwenden mehrerer Datenbanken in einer Transaktion

Bei Verwendung mehrerer Datenbanken in einer einzelnen Transaktion gelten andere Anforderungen für das Einrichten und Verwalten Ihrer Umgebung, je nachdem, wie viele Datenbanken in der Transaktion aktualisiert werden.

Aktualisieren einer einzelnen Datenbank

Wenn Ihre Daten über mehrere Datenbanken verteilt sind, möchten Sie eventuell eine Datenbank aktualisieren, während Sie für eine oder mehrere andere Datenbanken Lesevorgänge ausführen. Diese Art des Zugriffs kann innerhalb einer einzigen Arbeitseinheit (Transaktion) erfolgen.

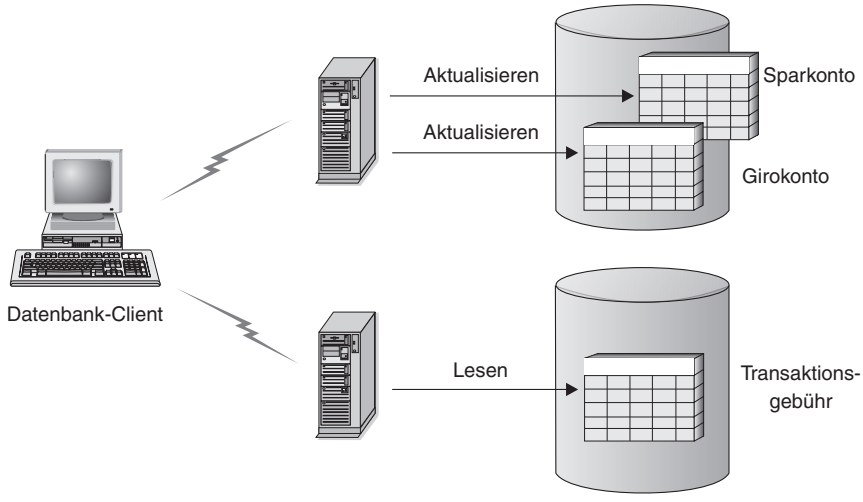

Abbildung 31. Verwenden mehrerer Datenbanken in einer Transaktion

Abb. 31 zeigt einen Datenbank-Client, der eine Geldtransaktionsanwendung ausführt, die auf zwei Datenbankserver zugreift: einer enthält das Sparkonto und das Girokonto, der andere den Plan für die Transaktionsgebühren. Dieses Beispiel ähnelt dem in Abb. 30 auf Seite 154 gezeigten. Der Unterschied besteht in der Anzahl der Datenbanken und der Speicherposition der Tabellen. Zur Einrichtung einer Geldtransaktionsanwendung für diese Umgebung sind folgende Schritte auszuführen:

1. Erstellen der erforderlichen Tabellen in den entsprechenden Datenbanken (siehe "Implementieren des Datenbankentwurfs" im Handbuch *Systemverwaltung: Implementierung*)
2. Wenn physisch fern, Einrichten der Datenbankserver für die Verwendung der geeigneten Übertragungsprotokolle (siehe die Beschreibung im Handbuch *Installation und Konfiguration Ergänzung*
3. Wenn physisch fern, Katalogisieren der Knoten und der Datenbanken, um die Datenbanken auf den Datenbankservern zu identifizieren (siehe die Beschreibung in den Handbüchern *Einstieg*)
4. Vorkompilieren Ihres Anwendungsprogramms, um eine Verbindung des Typs 2, d. h. CONNECT 2 im Befehl PRECOMPILE PROGRAM, sowie das einphasige Festschreiben, d. h. SYNCPOINT ONEPHASE im Befehl PRECOMPILE PROGRAM, anzugeben (siehe die Beschreibung im Handbuch *Application Development Guide*

Wenn sich Datenbanken auf einem Host oder einem AS/400-Datenbankserver befinden, wird das Produkt DB2 Connect zur Herstellung der Konnektivität zu diesen Servern benötigt. Informationen zur Einrichtung von DB2 Connect finden Sie in einem der Handbücher DB2 Connect *Einstieg*. Informationen zur Verwendung von DB2 Connect enthält das *DB2 Connect Benutzerhandbuch*.

Aktualisieren mehrerer Datenbanken

Wenn Ihre Daten über mehrere Datenbanken verteilt sind, wollen Sie vielleicht mehrere Datenbanken in einer einzelnen Transaktion lesen und aktualisieren. Diese Art des Datenbankzugriffs wird als *Aktualisierung auf mehreren Systemen* bezeichnet.

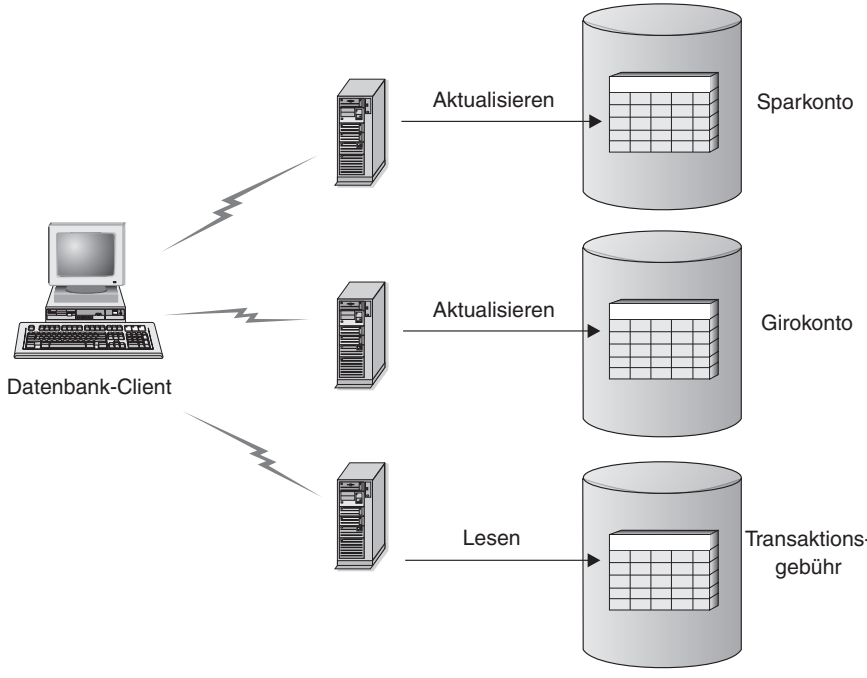

Abbildung 32. Aktualisieren mehrerer Datenbanken in einer Transaktion

Abb. 32 zeigt einen Datenbank-Client, der eine Geldtransaktionsanwendung ausführt, die auf drei Datenbankserver zugreift: einer enthält das Girokonto, ein anderer das Sparkonto und der dritte den Plan für die Transaktionsgebühren.

Zur Einrichtung einer Geldtransaktionsanwendung für diese Umgebung stehen zwei Optionen zur Auswahl:

1. Unter Verwendung eines Transaktionsmanagers (TM):
   a. Erstellen der erforderlichen Tabellen in den entsprechenden Datenbanken (siehe "Implementieren des Datenbankentwurfs" im Handbuch *Systemverwaltung: Implementierung*)

b. Wenn physisch fern, Einrichten der Datenbankserver für die Verwendung der geeigneten Übertragungsprotokolle (siehe die Beschreibung im Handbuch *Installation und Konfiguration Ergänzung*)
c. Wenn physisch fern, Katalogisieren der Knoten und der Datenbanken, um die Datenbanken auf den Datenbankservern zu identifizieren (siehe die Beschreibung in den Handbüchern *Einstieg*)
d. Vorkompilieren Ihres Anwendungsprogramms, um eine Verbindung des Typs 2, d. h. CONNECT 2 im Befehl PRECOMPILE PROGRAM, sowie das zweiphasige Festschreiben, d. h. SYNCPOINT TWOPHASE im Befehl PRECOMPILE PROGRAM, anzugeben (siehe die Beschreibung im Handbuch *Application Development Guide*)
e. Konfigurieren des DB2-Transaktionsmanagers (TM) (siehe die Beschreibung im Abschnitt „Verwenden des DB2-Transaktionsmanagers")

2. Ohne Transaktionsmanager:
   a. Erstellen der erforderlichen Tabellen in den entsprechenden Datenbanken (siehe "Implementieren des Datenbankentwurfs" im Handbuch *Systemverwaltung: Implementierung*)
   b. Wenn physisch fern, Einrichten der Datenbankserver für die Verwendung der geeigneten Übertragungsprotokolle (siehe die Beschreibung im Handbuch *Installation und Konfiguration Ergänzung*)
   c. Wenn physisch fern, Katalogisieren der Knoten und der Datenbanken, um die Datenbanken auf den Datenbankservern zu identifizieren (siehe die Beschreibung in den Handbüchern *Einstieg*)
   d. Vorkompilieren Ihres Anwendungsprogramms, um eine Verbindung des Typs 2, d. h. CONNECT 2 im Befehl PRECOMPILE PROGRAM, sowie das einphasige Festschreiben, d. h. SYNCPOINT ONEPHASE im Befehl PRECOMPILE PROGRAM, anzugeben (siehe die Beschreibung im Handbuch *Application Development Guide*)

Verwenden des DB2-Transaktionsmanagers

Der Datenbankmanager stellt Transaktionsmanagerfunktionen bereit, mit deren Hilfe das Aktualisieren mehrerer Datenbanken innerhalb einer einzelnen Arbeitseinheit koordiniert werden kann. Der Datenbank-Client koordiniert automatisch die Arbeitseinheit und verwendet eine *Transaktionsmanagerdatenbank*, um jede Transaktion zu registrieren und ihren Fertigstellungsstatus zu protokollieren.

Wenn Sie einen XA-konformen Transaktionsmanager wie IBM TXSeries, BEA Tuxedo oder Microsoft Transaction Server verwenden, finden Sie Anweisungen zur Integration in „Kapitel 10. Entwerfen für Transaktionsmanager" auf Seite 171.

Wenn Sie zur Koordinierung Ihrer Transaktionen DB2 UDB für auf UNIX basierende Systeme, Windows-Betriebssysteme oder OS/2 verwenden, müssen

Sie bestimmte Konfigurationsanforderungen erfüllen. Wenn Sie ausschließlich TCP/IP zur Kommunikation verwenden, und DB2 UDB und DB2 für OS/390 die einzigen Datenbankserver sind, die an Ihren Transaktionen beteiligt sind, ist die Konfiguration recht einfach.

DB2 UDB und DB2 für OS/390 mit TCP/IP-Konnektivität: Wenn jeder der folgenden Punkte auf Ihre Umgebung zutrifft, gestalten sich die Konfigurationsschritte für eine Aktualisierung auf mehreren Systemen relativ einfach.

- Für die gesamte Kommunikation mit fernen Datenbankservern (einschließlich DB2 UDB für OS/390) wird ausschließlich TCP/IP verwendet.
- DB2 UDB für auf UNIX basierende Systeme, Windows-Betriebssysteme, OS/2 oder OS/390 sind die einzigen, an der Transaktion beteiligten Datenbankserver.
- Der Synchronisationspunktmanager (SPM) von DB2 Connect ist nicht konfiguriert.

  Der DB2 Synchronisationspunktmanager wird automatisch bei der Erstellung eines DB2-Exemplars konfiguriert und ist erforderlich, wenn:

  – SNA-Konnektivität mit Host- oder AS/400-Datenbankservern für Aktualisierungen auf mehreren Systemen verwendet wird.
  – Ein XA-konformer Transaktionsmanager (wie ein IBM TXSeries CICS) die zweiphasige Festschreibung koordiniert.

    Dies gilt sowohl für die SNA- als auch für die TCP/IP-Konnektivität mit den Host- oder AS/400-Datenbankservern. Weitere Informationen finden Sie in „Kapitel 10. Entwerfen für Transaktionsmanager" auf Seite 171. Wenn in Ihrer Umgebung der DB2 Connect-Synchronisationspunktmanager nicht erforderlich ist, können Sie ihn durch folgenden Befehl, der auf dem DB2 Connect-Server eingegeben wird, ausschalten: `db2 update dbm cfg using spm_name NULL` Anschließend muss DB2 gestoppt und erneut gestartet werden.

Die Datenbank, die als Transaktionsmanagerdatenbank fungieren soll, wird auf dem Datenbank-Client durch den Konfigurationsparameter *tm_database* des Datenbankmanagers bestimmt. Weitere Informationen zu diesem Konfigurationsparameter finden Sie in "Konfigurieren von DB2" im Handbuch *Systemverwaltung: Optimierung*. Bei der Einstellung dieses Konfigurationsparameters sind folgende Faktoren zu beachten:

- Die Transaktionsmanagerdatenbank kann sein:
  – Eine Datenbank unter DB2 UDB für auf UNIX basierende Systeme, Windows-Betriebssysteme oder OS/2
  – Eine Datenbank unter DB2 für OS/390 Version 5 oder höher

    Dies ist der empfohlene Datenbankserver, der als Transaktionsmanagerdatenbank verwendet werden sollte. OS/390-Systeme sind in der Regel

sicherer als Workstation-Server, da sie die Möglichkeit versehentlichen Ausschaltens, Neustartens usw. verringern. Aus diesem Grund sind die Wiederherstellungsprotokolle, die im Fall einer Resynchronisation verwendet werden, sicherer.

- Wenn der Wert 1ST_CONN für den Konfigurationsparameter *tm_database* angegeben ist, wird die erste Datenbank, zu der eine Anwendung eine Verbindung herstellt, als Transaktionsmanagerdatenbank verwendet.

  Bei der Verwendung von 1ST_CONN ist besondere Sorgfalt geboten. Sie sollten diese Konfiguration nur verwenden, wenn es einfach ist, sicherzustellen, dass alle betroffenen Datenbanken korrekt katalogisiert werden, d. h. wenn folgende Bedingungen zutreffen:

  – Der Datenbank-Client, der die Transaktion einleitet, befindet sich in demselben Exemplar, das die beteiligten Datenbanken, einschließlich der Transaktionsmanagerdatenbank, enthält.
  – Sie verwenden die DCE Verzeichnisservices zum Katalogisieren und Verwalten des Zugriffs auf Ihre Datenbanken.

  Falls Ihre Anwendung versucht, die Verbindung zu der Datenbank, die als Transaktionsmanagerdatenbank fungiert, zu trennen, wird eine Fehlernachricht ausgegeben. Die Verbindung bleibt in diesem Fall erhalten, bis die Arbeitseinheit festgeschrieben ist.

Andere Umgebungen: Wenn in Ihrer Umgebung Folgendes zutrifft:

- Es wird nicht nur TCP/IP für die Kommunikation mit fernen Datenbankservern verwendet (es wird z. B. NETBIOS verwendet).
- Es wird auf DB2 für MVS Version 3 oder Version 4, DB2 für AS/400 oder DB2 für VM&VSE zugegriffen.
- Es wird auf DB2 für OS/390 über SNA zugegriffen.
- Der DB2 Connect-Synchronisationspunktmanager wird für den Zugriff auf Host- oder AS/400-Datenbankserver verwendet.

In diesen Fällen sind die Konfigurationsschritte für die Aktualisierung auf mehreren Systemen etwas komplizierter.

Die Datenbank, die als Transaktionsmanagerdatenbank fungieren soll, wird auf dem Datenbank-Client durch den Konfigurationsparameter *tm_database* des Datenbankmanagers bestimmt. Weitere Informationen zu diesem Konfigurationsparameter finden Sie in ″Konfigurieren von DB2″ im Handbuch *Systemverwaltung: Optimierung*. Bei der Einstellung dieses Konfigurationsparameters sind folgende Faktoren zu beachten:

- Die Transaktionsmanagerdatenbank kann eine Datenbank unter DB2 UDB für auf UNIX basierende Systeme, Windows-Betriebssysteme oder OS/2 sein.

- Wenn der Wert 1ST_CONN für den Konfigurationsparameter *tm_database* angegeben ist, wird die erste Datenbank, zu der eine Anwendung eine Verbindung herstellt, als Transaktionsmanagerdatenbank verwendet.

  Bei der Verwendung von 1ST_CONN ist besondere Sorgfalt geboten. Sie sollten diese Konfiguration nur verwenden, wenn es einfach ist, sicherzustellen, dass alle betroffenen Datenbanken korrekt katalogisiert werden, d. h. wenn folgende Bedingungen zutreffen:

  – Der Datenbank-Client, der die Transaktion einleitet, befindet sich in demselben Exemplar, das die beteiligten Datenbanken, einschließlich der Transaktionsmanagerdatenbank, enthält.
  – Sie verwenden die DCE Verzeichnisservices zum Katalogisieren und Verwalten des Zugriffs auf Ihre Datenbanken.

  Falls Ihre Anwendung versucht, die Verbindung zu der Datenbank, die als Transaktionsmanagerdatenbank fungiert, zu trennen, wird eine Fehlernachricht ausgegeben. Die Verbindung bleibt in diesem Fall erhalten, bis die Arbeitseinheit festgeschrieben ist.

Weitere Überlegungen zur Konfiguration

Bei der Einrichtung der Umgebung sind die folgenden Konfigurationsparameter zu berücksichtigen: Weitere Informationen zur Einstellung dieser Parameter finden Sie im *DB2 Connect Benutzerhandbuch*.

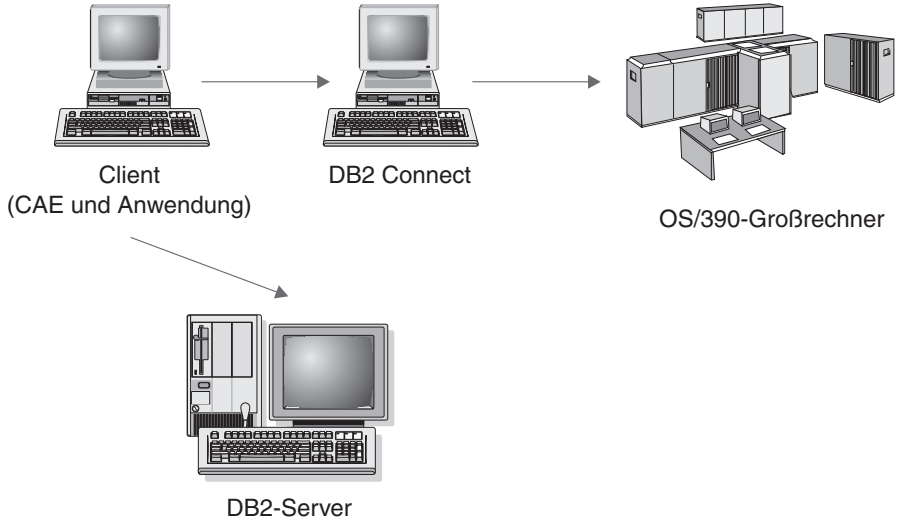

Abbildung 33. Überlegungen zur Konfiguration

Konfigurationsparameter des Datenbankmanagers

- *tm_database*

  Mit diesem Parameter wird der Name der Transaktionsmanagerdatenbank für das jeweilige DB2-Exemplar definiert.

- *spm_name*

  Mit diesem Parameter wird der Name des Exemplars mit dem DB2 Connect-Synchronisationspunktmanager für den Datenbankmanager definiert. Für eine erfolgreiche Resynchronisation muss der Name innerhalb Ihres gesamten Netzwerks eindeutig sein.

- *resync_interval*

  Mit diesem Parameter wird das Zeitintervall (in Sekunden) angegeben, nach dem der DB2-Transaktionsmanager, der DB2-Server-Datenbankmanager und der DB2 Connect-Synchronisationspunktmanager den Versuch wiederholen sollen, alle ausstehenden unbestätigten Transaktionen wiederherzustellen.

- *spm_log_file_sz*

  Mit diesem Parameter wird die Größe (in 4-KB-Seiten) der SPM-Protokolldatei angegeben.

- *spm_max_resync*

  Mit diesem Parameter wird die Anzahl der Agenten angegeben, die gleichzeitig Resynchronisationsoperationen durchführen können.

- *spm_log_path*

  Mit diesem Parameter wird der Protokollpfad für die SPM-Protokolldateien angegeben.

Konfigurationsparameter der Datenbank

- *maxappls*

  Mit diesem Parameter wird die zulässige maximale Anzahl aktiver Anwendungen angegeben. Der Wert dieses Parameters muss gleich oder größer sein als die Summe der verbundenen Anwendungen plus die Anzahl dieser Anwendungen, die sich gleichzeitig im Prozess einer zweiphasigen Festschreibung bzw. einer ROLLBACK-Operation befinden können, plus die angenommene Anzahl unbestätigter Transaktionen, die zu einem beliebigen Zeitpunkt gleichzeitig vorhanden sein können. Weitere Informationen zu unbestätigten Transaktionen finden Sie in „Beheben von Problemen bei der zweiphasigen Festschreibung” auf Seite 167.

- *autorestart*

  Mit diesem Konfigurationsparameter der Datenbank wird festgelegt, ob die Routine RESTART DATABASE bei Bedarf automatisch aufgerufen wird. Der Standardwert ist `YES` (d. h. aktiviert). Für eine Datenbank, die unbestätigte Transaktionen enthält, ist eine RESTART DATABASE-Operation für den Neustart erforderlich. Wenn *autorestart* nicht aktiviert ist und die letzte Verbindung zur Datenbank getrennt wurde, schlägt die nächste Verbindungs-

anforderung fehl, so dass ein explizites Aufrufen des Befehls RESTART DATABASE erforderlich wird. Diese Bedingung bleibt solange bestehen, bis die unbestätigten Transaktionen entweder durch die Resynchronisationsoperation des Transaktionsmanagers oder durch eine vom Administrator eingeleitete heuristische Operation beseitigt wird. Wenn der Befehl RESTART DATABASE abgesetzt wird, wird eine Nachricht zurückgegeben, wenn unbestätigte Transaktionen in der Datenbank vorhanden sind. Der Administrator kann dann mit Hilfe des Befehls LIST INDOUBT TRANSACTIONS und anderer Befehle des Befehlszeilenprozessors Informationen über diese unbestätigten Transaktionen erhalten.

Weitere Informationen zu diesen Konfigurationsparametern finden Sie im Handbuch *Systemverwaltung: Optimierung*.

Host- oder IBM AS/400-Anwendungen, die auf einen LAN-basierten DB2 Universal Database-Server zur Aktualisierung auf mehreren Systemen zugreifen

DB2 Universal Database unterstützt keine Aktualisierung auf mehreren Systemen von Host- oder von IBM AS/400-Datenbank-Clients mit Hilfe der TCP/IP-Konnektivität. In diesem Fall wird nur die SNA-Konnektivität (Systems Network Architecture) unterstützt. Der DB2-Synchronisationspunktmanager wird für Aktualisierungen auf mehreren Systemen benötigt. DB2 Connect wird in diesem Szenario nicht eingesetzt. Der Datenbankserver, auf den vom Host- oder vom IBM AS/400-Datenbank-Client zugegriffen wird, muss nicht lokal auf der Workstation vorhanden sein, auf der sich der DB2-Synchronisationspunktmanager befindet. Der Host- oder IBM AS/400-Datenbank-Client könnte eine Verbindung zu einem DB2 UDB-Server herstellen, indem er die Workstation mit dem DB2-Synchronisationspunktmanager als Übergangs-Gateway verwendet. Dadurch können Sie die Workstation mit dem DB2-Synchronisationspunktmanager in einer sicheren Umgebung isolieren, während die tatsächlichen DB2 UDB-Server in Ihrer Organisation ferne Server sind. Dadurch kann auch eine Datenbank auf der DB2 Server-Plattform der Version 2 an Aktualisierungen auf mehreren Systemen beteiligt sein, die von Host- oder IBM AS/400-Datenbank-Clients gestartet wurden.

Folgende Schritte sind auszuführen:

- Auf der Workstation, auf die durch die Host- oder AS/400-Anwendung direkt zugegriffen werden soll:
  1. Installieren der DB2 Universal Database Enterprise Edition oder Enterprise - Extended Edition, um die Aktualisierung auf mehreren Systemen mit Host- oder IBM AS/400-Datenbank-Clients bereitzustellen.
  2. Erstellen eines Datenbankexemplars auf demselben System. Sie können z. B. das Standardexemplar `DB2` verwenden oder mit dem folgenden Befehl ein neues Exemplar erstellen:

     ```
     db2icrt meinexemplar
     ```

3. Eingeben der erforderlichen Lizenzinformationen.
4. Sicherstellen, dass der Registrierungswert DB2COMM den Wert APPC enthält.
5. Konfigurieren der erforderlichen SNA-Kommunikation. Wenn die unterstützten IBM SNA-Produkte eingesetzt werden, werden die für den DB2-Synchronisationspunktmanager erforderlichen SNA-Profile auf der Grundlage des für den Konfigurationsparameter *spm_name* des Datenbankmanagers definierten Werts automatisch erstellt. Jeder andere unterstützte SNA-Stapel (Stack) macht eine manuelle Konfiguration erforderlich. Genauere Informationen enthält das Handbuch *Installation und Konfiguration Ergänzung*.
6. Bestimmen des Werts, der für den Konfigurationsparameter *spm_name* des Datenbankmanagers anzugeben ist. Dieser Parameter wird bei der Erstellung eines DB2-Exemplars mit einem aus dem TCP/IP-Host-Namen für die Maschine abgeleiteten Wert vorkonfiguriert. Falls dieser Wert annehmbar und in der Umgebung eindeutig ist, sollte er nicht geändert werden.
7. Falls erforderlich, Aktualisieren des Werts für *spm_name* auf dem DB2 Universal Database-Server mit Hilfe des Befehls UPDATE DATABASE MANAGER CONFIGURATION.
8. Konfigurieren der für diese DB2-Workstation erforderlichen Kommunikation zur Verbindung zu fernen DB2 UDB-Servern, falls nötig.
9. Konfigurieren der für ferne DB2 UDB-Server erforderlichen Kommunikation zur Verbindung mit diesem DB2-Server.
10. Stoppen und erneutes Starten des Datenbankmanagers auf dem DB2 Universal Database-Server, um den SPM das erste Mal zu starten.

    Sie sollten in der Lage sein, eine Verbindung von dieser Workstation aus zu fernen DB2 UDB-Servern herzustellen.

- Auf jedem fernen DB2 UDB-Server, auf den durch den Host- oder den IBM AS/400-Datenbank-Client zugegriffen wird:
  1. Konfigurieren der für die ferne DB2 UDB-Workstation mit dem DB2-Synchronisationspunktmanager erforderlichen Kommunikation zur Verbindung mit diesem DB2 UDB-Server.
  2. Stoppen und erneutes Starten des Datenbankmanagers.

Prozess der zweiphasigen Festschreibung

Abb. 34 auf Seite 165 veranschaulicht die Schritte, die zu einer Aktualisierung auf mehreren Systemen gehören. Kenntnisse über die Art und Weise, wie eine Transaktion verwaltet wird, erweisen sich bei der Problemlösung als hilfreich, wenn während des Prozesses der zweiphasigen Festschreibung ein Fehler auftritt.

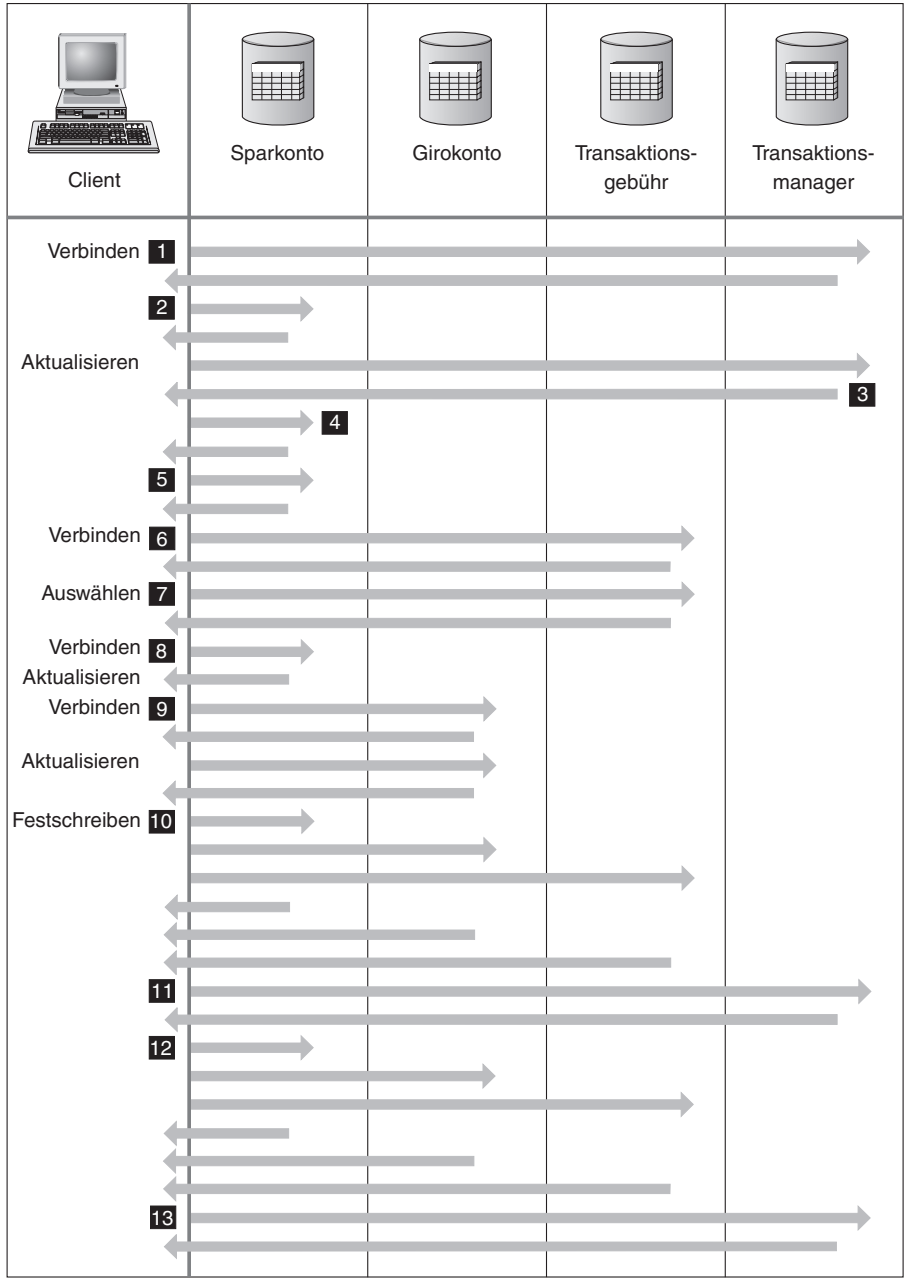

Abbildung 34. Aktualisieren mehrerer Datenbanken

0 Die Anwendung ist für die zweiphasige Festschreibung vorbereitet. Dies kann durch Vorkompilierungsoptionen erreicht werden (Einzelangaben finden Sie im Handbuch *Application Development Guide*). Dies

kann auch durch eine Konfiguration von DB2 CLI (Call Level Interface) erreicht werden (Einzelangaben finden Sie im Handbuch *CLI Guide and Reference*).

1 Wenn der Datenbank-Client die Verbindung zur Datenbank SPARKO_DB herstellen will, stellt er zunächst intern die Verbindung zur Transaktionsmanagerdatenbank (TMD) her. Die TMD gibt eine Bestätigung an den Datenbank-Client zurück. Falls der Konfigurationsparameter *tm_database* des Datenbankmanagers auf den Wert `1ST_CONN` gesetzt ist, wird die Datenbank SPARKO_DB für die Dauer dieses Anwendungsexemplars zur Transaktionsmanagerdatenbank.

2 Die Verbindung zur Datenbank SPARKO_DB wird hergestellt und bestätigt.

3 Der Datenbank-Client beginnt mit der Aktualisierung der Tabelle SPAR_KONTO. Dies ist der Anfang der Arbeitseinheit. Die TMD reagiert auf den Datenbank-Client, indem sie eine Transaktions-ID für die Arbeitseinheit bereitstellt. Beachten Sie, dass die Registrierung einer Arbeitseinheit erfolgt, wenn die erste SQL-Anweisung in der Arbeitseinheit ausgeführt wird, und nicht während der Herstellung der Verbindung.

4 Nach dem Empfang der Transaktions-ID registriert der Datenbank-Client die Arbeitseinheit bei der Datenbank, die die Tabelle SPAR_KONTO enthält. Der Client erhält eine Antwort, die besagt, dass die Arbeitseinheit erfolgreich registriert wurde.

5 SQL-Anweisungen, die an der Datenbank SPARKO_DB ausgeführt werden, werden auf normale Weise verarbeitet. Die Anwort auf jede Anweisung wird im SQL-Kommunikationsbereich (SQLCA) zurückgegeben, wenn mit SQL-Anweisungen gearbeitet wird, die in ein Programm eingebettet sind. (Eine Beschreibung des SQL-Kommunikationsbereichs (SQLCA) finden Sie in den Handbüchern *Application Development Guide* und *SQL Reference*.)

6 Die Transaktions-ID wird in der Datenbank GEB_DB, die die Tabelle TRANSAKTION_GEB enthält, beim ersten Zugriff auf diese Datenbank innerhalb der Arbeitseinheit registriert.

7 SQL-Anweisungen, die für die Datenbank GEB_DB abgesetzt werden, werden auf normale Weise bearbeitet.

8 Zusätzliche SQL-Anweisungen können für die Datenbank SPARKO_DB ausgeführt werden, indem die Verbindung entsprechend eingerichtet wird. Da die Arbeitseinheit bei der Datenbank SPARKO_DB bereits registriert (**4**) wurde, muss der Datenbank-Client den Registrierungsschritt nicht wiederholen.

9 Für die Verbindung mit der Datenbank GIROKO_DB und ihre Verwendung gelten die gleichen Regeln, die unter **6** und **7** beschrieben sind.

10 Wenn der Datenbank-Client das Festschreiben der Arbeitseinheit anfordert, wird eine Nachricht *prepare* an alle Datenbanken gesendet, die an der Arbeitseinheit mitwirken. Jede Datenbank schreibt einen Satz "PREPARED" in ihre Protokolldatei und antwortet dem Datenbank-Client.

11 Nachdem der Datenbank-Client von allen Datenbanken eine positive Antwort erhalten hat, sendet er eine Nachricht an die Transaktionsmanagerdatenbank, die besagt, dass die Arbeitseinheit für das Festschreiben bereit (PREPARED) ist. Die Transaktionsmanagerdatenbank schreibt den Satz "PREPARED" in ihre Protokolldatei und teilt dem Client in einer Antwort mit, dass die zweite Phase der Festschreibung begonnen werden kann.

12 Während der zweiten Phase des Festschreibungsprozesses fordert der Datenbank-Client alle beteiligten Datenbanken in einer Nachricht zum Festschreiben auf. Jede Datenbank schreibt den Satz "COMMITTED" in ihre Protokolldatei und gibt die Sperren frei, die für diese Arbeitseinheit aktiv waren. Wenn die Datenbank das Festschreiben der Änderungen beendet hat, sendet sie eine Antwort an den Client.

13 Nachdem der Datenbank-Client von allen beteiligten Datenbanken eine positive Antwort erhalten hat, sendet er eine Anforderung an die Transaktionsmanagerdatenbank, um ihr mitzuteilen, dass die Arbeitseinheit beendet wurde. Daraufhin schreibt die Transaktionsmanagerdatenbank einen Satz "COMMITTED" in ihre Protokolldatei, der anzeigt, dass die Arbeitseinheit beendet ist, und sendet eine Antwort an den Client, dass der Prozess beendet wurde.

Beheben von Problemen bei der zweiphasigen Festschreibung

Das Beheben von Fehlerbedingungen ist eine normale Aufgabe bei der Anwendungsprogrammierung, Systemverwaltung, Datenbankverwaltung sowie beim Systembetrieb. Durch die Verteilung von Datenbanken über mehrere ferne Server erhöht sich das Potenzial von Fehlerbedingungen, die aus Netzwerk- oder Übertragungsstörungen resultieren können. Um die Datenintegrität zu gewährleisten, stellt der Datenbankmanager den Prozess der zweiphasigen Festschreibung zur Verfügung. Eine Darstellung dieses Prozesses finden Sie in „Prozess der zweiphasigen Festschreibung" auf Seite 164 (siehe Punkte **10**, **11** und **12**). Im folgenden wird erläutert, wie Datenbankmanager Fehler während des Prozesses der zweiphasigen Festschreibung behandelt:

- **Fehler in der ersten Phase**

  Wenn eine Datenbank mitteilt, dass sie die Festschreibung der Arbeitseinheit nicht vorbereiten (und kein ″PREPARED″ in die Protokolldatei eintragen) konnte, macht der Datenbank-Client die Arbeitseinheit in der zweiten Phase des Festschreibungsprozesses rückgängig. In diesem Fall wird *keine* PREPARE-Nachricht an die Transaktionsmanagerdatenbank gesendet.

  Während der zweiten Phase der Festschreibung weist der Client alle beteiligten Datenbanken, die während der ersten Phase die Festschreibung erfolgreich vorbereitet haben, an, die Arbeitseinheit rückgängig (ROLLBACK) zu machen. Jede Datenbank schreibt daraufhin einen Satz ″ABORT″ in ihre Protokolldatei und gibt die Sperren frei, die für diese Arbeitseinheit aktiv waren.

- **Fehler in der zweiten Phase**

  Die Fehlerbehandlung ist in diesem Fall davon abhängig, ob die Transaktion in der zweiten Phase festgeschrieben oder rückgängig gemacht wird. Die zweite Phase macht die Transaktion nur dann rückgängig, wenn in der ersten Phase ein Fehler auftrat.

  Wenn eine der beteiligten Datenbanken die Arbeitseinheit nicht festschreiben kann (u. U. aufgrund eines Übertragungsfehlers), versucht die Transaktionsmanagerdatenbank erneut, für die fehlgeschlagene Datenbank eine Festschreibung auszuführen. Die Anwendung wird jedoch über den SQL-Kommunikationsbereich (SQLCA) darüber informiert, dass die Festschreibung erfolgreich war. DB2 stellt sicher, dass die nicht festgeschriebene Transaktion im Datenbankserver festgeschrieben wird. Mit Hilfe des Konfigurationsparameters *resync_interval* des Datenbankmanagers (siehe ″Konfigurieren von DB2″ im Handbuch *Systemverwaltung: Optimierung*) wird das Intervall festgelegt, das die Transaktionsmanagerdatenbank zwischen den einzelnen Versuchen, die Arbeitseinheit festzuschreiben, verstreichen lässt. Alle Sperren bleiben auf dem Datenbankserver aktiv, bis die Arbeitseinheit festgeschrieben wird.

  Fällt Transaktionsmanagerdatenbank aus, resynchronisiert sie die Arbeitseinheit, wenn sie erneut gestartet wird. Der Resynchronisationsprozess versucht, alle *unbestätigten Transaktionen* zu beenden, d. h. die Transaktionen, die die erste Phase der Festschreibung beendet haben, deren zweite Phase jedoch nicht abgeschlossen wurde. Der Datenbankmanager, unter dem die Transaktionsmanagerdatenbank aktiv ist, führt zur Resynchronisation folgende Schritte aus:

1. Herstellen einer Verbindung zu den Datenbanken, die während der ersten Phase des Festschreibungsprozesses mitteilten, dass sie zur Festschreibung bereit ("PREPARED") waren
2. Versuch einer Festschreibung der unbestätigten Transaktionen in diesen Datenbanken. (Können die unbestätigten Transaktionen nicht gefunden werden, nimmt der Datenbankmanager an, dass die Datenbank die Transaktionen in der zweiten Phase des Festschreibungsprozesses erfolgreich festgeschrieben hat.)
3. Festschreiben der unbestätigten Transaktionen in der Transaktionsmanagerdatenbank, nachdem alle unbestätigten Transaktionen in den beteiligten Datenbanken festgeschrieben wurden

Wenn bei einer der beteiligten Datenbanken ein Fehler auftritt und die Datenbank erneut gestartet wird, fragt der Datenbankmanager für diese Datenbank die Transaktionsmanagerdatenbank nach dem Status dieser Transaktion ab, um festzustellen, ob die Transaktion rückgängig gemacht werden sollte. Wenn die Transaktion im Protokoll nicht gefunden wird, nimmt der Datenbankmanager an, dass die betreffende Transaktion rückgängig gemacht wurde, und macht die unbestätigte Transaktion in dieser Datenbank rückgängig. Andernfalls wartet die Datenbank auf eine Festschreibungsanforderung der Transaktionsmanagerdatenbank.

Wenn die Transaktion von einem Transaktionsverarbeitungsmonitor (XA-konformem Transaktionsmanager) koordiniert wurde, ist die Datenbank immer davon abhängig, dass die Resynchronisation vom TP-Monitor eingeleitet wird.

Wenn Sie aus einem bestimmten Grund nicht darauf warten können, dass der Transaktionsmanager unbestätigte Transaktionen automatisch auflöst, haben Sie die Möglichkeit, Maßnahmen zur manuellen Auflösung unbestätigter Transaktionen zu ergreifen. Dieser manuelle Prozess wird manchmal als das „Treffen einer heuristischen Entscheidung" bezeichnet. Weitere Informationen zur manuellen Wiederherstellung unbestätigter Transaktionen finden Sie in „Treffen einer heuristischen Entscheidung" auf Seite 185.

Resynchronisieren unbestätigter Transaktionen bei AUTORESTART=OFF

Hinweise zur Konfiguration in einer Umgebung von DB2 Universal Database mit zweiphasiger Festschreibung finden Sie in „Weitere Überlegungen zur Konfiguration" auf Seite 161.

Insbesondere ist zu beachten, dass, wenn der Konfigurationsparameter *autorestart* der Datenbank den Wert `OFF` hat und unbestätigte Transaktionen in der Transaktionsmanagerdatenbank oder den Ressourcenmanagerdatenbanken vorhanden sind, der Befehl RESTART DATABASE ausgeführt werden muss, um den Resynchronisationsprozess zu starten. Wenn der Befehl RESTART DATABASE über den Befehlszeilenprozessor ausgeführt wird, verwenden Sie verschiedene Sitzungen. Wenn Sie eine andere Datenbank aus derselben Sitzung erneut starten, wird die Verbindung, die durch die vorherige Ausführung des Befehls RESTART DATABASE hergestellt wurde, beendet und muss erneut gestartet werden. Setzen Sie den Befehl TERMINATE ab, um die Verbindung zu beenden, wenn der Befehl LIST INDOUBT TRANSACTIONS keine weiteren unbestätigten Transaktionen mehr liefert.

Kapitel 10. Entwerfen für Transaktionsmanager

Wenn Sie neben den DB2-Datenbanken über weitere Ressourcen verfügen, die an einer zweiphasigen Festschreibungstransaktion beteiligt sein sollen, kann es sinnvoll sein, einen dem XA-Standard entsprechenden Transaktionsmanager zu verwenden. Wenn Ihre Transaktionen nur auf DB2-Datenbanken zugreifen, sollten Sie den DB2-Transaktionsmanager verwenden, der in „Aktualisieren mehrerer Datenbanken" auf Seite 157 beschrieben wird.

Die folgenden Themen enthalten Informationen zur Verwendung des Datenbankmanagers mit einem XA-konformen Transaktionsmanager wie IBM TXSeries CICS, IBM TXSeries Encina, BEA Tuxedo oder Microsoft Transaction Server:

Wenn Sie einen dem XA-Standard entsprechenden Transaktionsmanager verwenden oder implementieren, stehen weitere Informationen auf unserer Web-Adresse für technische Unterstützung zur Verfügung:

`http://www.ibm.com/software/data/db2/library/`

Dort wählen Sie ″DB2 Universal Database″ aus, und durchsuchen anschließend die Web-Site mit dem Schlüsselwort ″XA″ nach den neuesten verfügbaren Informationen zu Transaktionsmanagern, die dem XA-Standard entsprechen.

X/Open-Modell für die verteilte Transaktionsverarbeitung

Das X/Open-Modell für die verteilte Transaktionsverarbeitung (DTP - Distributed Transaction Processing) umfasst drei in Wechselbeziehung zueinander stehende Komponenten:

- „Anwendungsprogramm (AP)″
- „Transaktionsmanager (TM)″ auf Seite 174
- „Ressourcenmanager (RM)″ auf Seite 176.

Abb. 35 veranschaulicht dieses Modell und zeigt die Beziehungen dieser Komponenten untereinander.

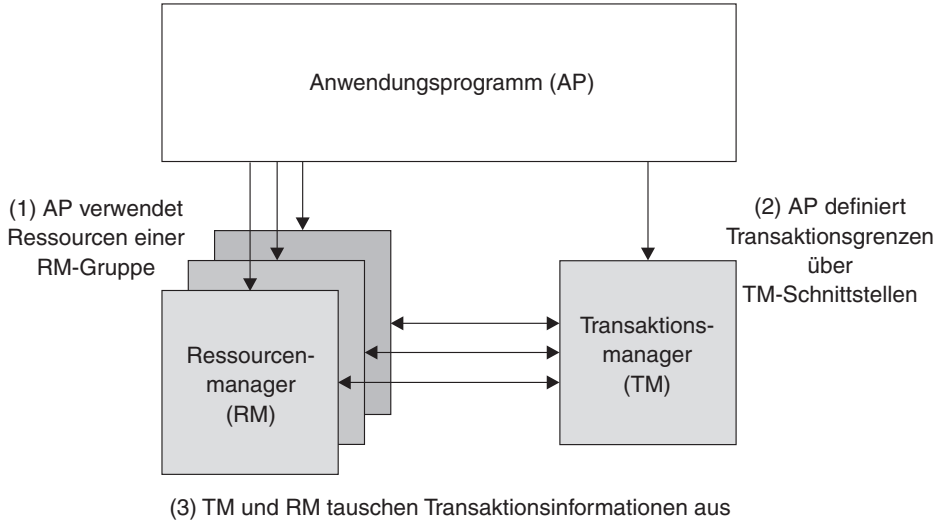

Abbildung 35. X/Open-Modell für die verteilte Transaktionsverarbeitung (DTP)

Anwendungsprogramm (AP)

Das Anwendungsprogramm (AP) definiert die Transaktionsgrenzen und gibt die anwendungsspezifischen Aktionen an, aus denen die Transaktion besteht.

Ein CICS*-Anwendungsprogramm möchte zum Beispiel auf Ressourcenmanager (RMs) wie eine Datenbank und eine CICS-Warteschlange mit Übergangsdaten (CICS Transient Data Queue) zugreifen und eine Programmlogik verwenden, um die Daten zu ändern. Jede Zugriffsanforderung wird an die entsprechenden Ressourcenmanager über für diesen Ressourcenmanager spe-

zifische Funktionsaufrufe übergeben. Im Fall von DB2 können diese Funktionsaufrufe vom DB2-Precompiler für jede SQL-Anweisung generiert worden sein, oder es kann sich um Datenbankaufrufe handeln, die direkt vom Programmierer mit Hilfe der APIs codiert wurden.

Zum Transaktionsmanagerprodukt (TM-Produkt) gehört in der Regel ein Transaktionsverarbeitungsmonitor (TP-Monitor) zur Ausführung der Benutzeranwendung. Der TP-Monitor stellt APIs bereit, die einer Anwendung die Möglichkeit geben, eine Transaktion zu starten und zu beenden, und die Funktionen zur Terminierung der zeitlichen Abläufe von Anwendungen sowie zum Lastausgleich unter den vielen Benutzern, die die Anwendung ausführen möchten, zur Verfügung stellen. Das Anwendungsprogramm (AP) in einer DTP-Umgebung ist im Grunde eine Kombination aus der Benutzeranwendung und dem TP-Monitor.

Um eine effiziente Online-Transaktionsverarbeitung (OLTP, Online Transaction Processing) zu ermöglichen, ordnet der TP-Monitor eine Reihe von Server-Prozessen beim Start im Voraus zu und verwaltet und verwendet diese anschließend für die zahlreichen Benutzertransaktionen. Auf diese Weise werden Systemressourcen eingespart, da mehr Benutzer mit Hilfe einer geringeren Anzahl von Server-Prozessen und ihrer zugehörigen RM-Prozesse unterstützt werden. Durch die mehrfache Verwendung dieser Prozesse wird auch der Systemaufwand vermieden, der mit dem Starten eines Prozesses im Transaktionsmanager oder den Ressourcenmanagern (RM) für jede Benutzertransaktion bzw. jedes Benutzerprogramm verbunden ist. (Ein Programm ruft ein oder mehrere Transaktionen auf.) Dies bedeutet auch, dass die Server-Prozesse gegenüber dem Transaktionsmanager und den Ressourcenmanagern als die wirklichen „Benutzerprozesse" auftreten. Hieraus ergeben sich bestimmte Konsequenzen für die Sicherheitsverwaltung und die Anwendungsprogrammierung. Nähere Informationen finden Sie in „Überlegungen zur Sicherheit" auf Seite 188.

Die folgenden Arten von Transaktionen sind von einem TP-Monitor aus möglich:

- Nicht-XA-Transaktionen

  Diese Transaktionen betreffen Ressourcenmanager, die für den Transaktionsmanager nicht definiert sind und daher nicht unter dem Protokoll für die zweiphasige Festschreibung des Transaktionsmanagers koordiniert werden. Diese Transaktionen können erforderlich werden, wenn eine Anwendung auf einen Ressourcenmanager zugreifen muss, der die XA-Schnittstelle nicht unterstützt. Der TP-Monitor stellt einfach effiziente Einrichtungen zur Zeitplanung und zum Lastausgleich bereit. Da der Transaktionmanager den Ressourcenmanager nicht explizit für eine XA-Verarbeitung „öffnet", behandelt der Ressourcenmanager diese Anwendung wie jede andere Anwendung, die in einer Nicht-DTP-Umgebung ausgeführt wird.

- Globale Transaktionen

  Diese Transaktionen arbeiten mit Ressourcenmanagern, die gegenüber dem Transaktionsmanager definiert sind und der Steuerung der zweiphasigen Festschreibung des Transaktionsmanagers unterliegen. Eine globale Transaktion stellt eine Arbeitseinheit dar, die auf einen oder mehrere Ressourcenmanager zugreift. Eine *Transaktionsverzweigung* ist der Teil der Arbeit zwischen einem Transaktionsmanager und einem Ressourcenmanager, der die globale Transaktion unterstützt. Eine globale Transaktion kann mehrere Transaktionsverzweigungen umfassen, wenn auf mehrere Ressourcenmanager über einen oder mehrere vom Transaktionsmanager koordinierte Anwendungsprozesse zugegriffen wird.

  Lose gekoppelte globale Transaktionen sind vorhanden, wenn jeder Prozess einer Anzahl von Anwendungsprozessen auf die Ressourcenmanager so zugreift, als wäre er in einer getrennten globalen Transaktion, die Anwendungen jedoch alle der Koordination durch den Transaktionsmanager unterliegen. Jeder Anwendungsprozess verfügt über eine eigene Transaktionsverzweigung innerhalb eines Ressourcenmanagers. Wenn eine COMMIT- oder ROLLBACK-Operation von einem der Anwendungsprogramme, dem Transaktionsmanager oder den Ressourcenmanagern angefordert wird, werden alle Transaktionsverzweigungen zusammen abgeschlossen. Es obliegt den Anwendungen, gegenseitige Sperren von Ressourcen unter den Transaktionsverzweigungen auszuschließen. (Beachten Sie, dass die Koordination der Transaktionen, die durch den DB2-Transaktionsmanager für Anwendungen implementiert wird, die mit der Option SYNCPOINT(TWOPHASE) vorbereitet (PREPARE) wurden, diesen lose gekoppelten globalen Transaktionen ungefähr entspricht. Siehe „Aktualisieren mehrerer Datenbanken" auf Seite 157.)

  Fest gekoppelte globale Transaktionen sind vorhanden, wenn mehrere Anwendungsprozesse abwechselnd dieselbe Transaktionsverzweigung in einem Ressourcenmanager verwenden, um Operationen auszuführen. Dem Ressourcenmanager gegenüber bilden die beiden Anwendungsprozesse eine Einheit. Der Ressourcenmanager muss dafür sorgen, dass keine gegenseitigen Ressourcensperren innerhalb der Transaktionsverzweigung auftreten.

Transaktionsmanager (TM)

Der Transaktionsmanager (TM) ordnet Transaktionen Kennungen zu, überwacht die Verarbeitung von Transaktionen und ist zuständig für die Beendigung bzw. für Fehler der Transaktionen. Die Kennungen für Transaktionsverzweigungen (XIDs) werden vom Transaktionsmanager zugeordnet, um die globale Transaktion und die spezielle Verzweigung innerhalb eines Ressourcenmanagers (RM) zu identifizieren. Diese Kennung ist das Korrelations-Token zwischen dem Protokoll in einem Transaktionsmanager (TM) und dem Protokoll in einem Ressourcenmanager (RM). Die XID wird für eine zweiphasige Festschreibung bzw. für eine ROLLBACK-Operation benötigt, um die *Resynchronisation* (abgekürzt *resync*) bei einem Systemstart durchzuführen

oder um dem Administrator die Möglichkeit zu geben, bei Bedarf eine *heuristische* Operation (auch als *manueller Eingriff* bezeichnet) vorzunehmen.

Nach dem Start fordert ein TP-Monitor den Transaktionmanager auf, alle Ressourcenmanager (RMs) zu öffnen, die in einer Gruppe von Anwendungs-Servern definiert sind. Der Transaktionsmanager übergibt die **xa_open**-Aufrufe an die Ressourcenmanager, damit diese für die DTP-Verarbeitung initialisiert werden können. Im Rahmen dieser Startprozedur führt der Transaktionsmanager eine Resynchronisation (resync) durch, um alle *unbestätigten Transaktionen* aufzulösen. Eine unbestätigte Transaktion ist eine globale Transaktion, die nicht vollständig zu Ende geführt wurde. Dies geschieht, wenn auf den Transaktionsmanager (oder mindestens auf einen Ressourcenmanager) nach einer erfolgreich abgeschlossenen ersten Phase (Vorbereitunsphase) des Protokolls für zweiphasige Festschreibung nicht mehr zugegriffen werden kann. Der Ressourcenmanager weiß solange nicht, ob seine Verzweigung der Transaktion festzuschreiben oder rückgegängig zu machen ist, bis der Transaktionsmanager sein eigenes Protokoll mit den Protokollen der Ressourcenmanager abgleichen kann, wenn sie wieder verfügbar werden. Zur Durchführung der Resynchronisation setzt der Transaktionsmanager einen **xa_recover**-Aufruf an jeden Ressourcenmanager mindestens einmal ab, um alle unbestätigten Transaktionen zu identifizieren. Der Transaktionsmanager vergleicht die Antworten mit den Informationen im eigenen Protokoll, um zu bestimmen, ob die Ressourcenmanager angewiesen werden sollen, eine **xa_commit**-Operation oder eine **xa_rollback**-Operation für diese Transaktionen durchzuführen. Wenn ein Ressourcenmanager seine Verzweigung einer unbestätigten Transaktion aufgrund einer heuristischen Operation durch den Administrator bereits festgeschrieben oder rückgängig gemacht hat, setzt der Transaktionmanager einen Aufruf **xa_forget** an diesen Ressourcenmanager ab, um die Resynchronisation abzuschließen.

Wenn eine Benutzeranwendung eine COMMIT- oder ROLLBACK-Operation anfordert, muss sie die vom TP-Monitor oder Transaktionmanager bereitgestellte API verwenden, so dass der Transaktionmanager die COMMIT- und ROLLBACK-Operation unter allen beteiligten Ressourcenmanagern koordinieren kann. Wenn zum Beispiel eine CICS-Anwendung die CICS-Anforderung SYNCPOINT absetzt, um eine Transaktion festzuschreiben, setzt der Transaktionsmanager CICS XA TM (im Encina-Server implementiert) seinerseits die entsprechenden XA-Aufrufe wie **xa_end**, **xa_prepare**, **xa_commit** oder **xa_rollback** ab, um den Ressourcenmanager anzuweisen, die Transaktion festzuschreiben oder rückgängig zu machen. Der Transaktionsmanager kann eine einphasige Festschreibung anstelle der zweiphasigen Festschreibung durchführen, wenn nur ein Ressourcenmanager beteiligt ist oder wenn ein Ressourcenmanager antwortet, dass seine Transaktionsverzweigung im Nur-Lese-Modus arbeitet.

Ressourcenmanager (RM)

Ein Ressourcenmanager (RM) stellt den Zugriff auf gemeinsame Ressourcen wie Datenbanken zur Verfügung.

DB2 als Ressourcenmanager einer Datenbank kann an einer *globalen Transaktion* beteiligt sein, die von einem dem XA-Standard entsprechenden Transaktionsmanager koordiniert wird. Wie für die XA-Schnittstelle erforderlich stellt der Datenbankmanager eine externe C-Variable *db2xa_switch* des Typs xa_switch_t bereit, um die XA-Schalterstruktur an den Transaktionsmanager zurückzugeben. Diese Datenstruktur enthält die Adressen der verschiedenen XA-Routinen, die vom Transaktionsmanager aufzurufen sind, und die Betriebsmerkmale des Ressourcenmanagers. Weitere Informationen zu XA-Funktionen, die vom Datenbankmanager unterstützt werden, finden Sie in „Unterstützte XA-Funktion" auf Seite 190.

Es gibt zwei Methoden, mit denen der Ressourcenmanager seine Teilnahme an der jeweiligen globalen Transaktion registrieren kann: *statische Registrierung* und *dynamische Registrierung*:

- Für die statische Registrierung ist es erforderlich, dass der Transaktionsmanager (für jede Transaktion) die Folge von Aufrufen **xa_start**, **xa_end** und **xa_prepare** an alle Ressourcenmanager absetzt, die für die Server-Anwendung definiert sind, unabhängig davon, ob der einzelne Ressourcenmanager von der Transaktion verwendet wird. Dies ist nicht effizient, wenn nicht jeder Ressourcenmanager an jeder Transaktion beteiligt ist. Der Grad der Ineffizienz ist proportional zur Anzahl der definierten Ressourcenmanager.
- Die (von DB2 genutzte) dynamische Registrierung ist flexibel und effizient. Ein Ressourcenmanager registriert sich dem Transaktionsmanager gegenüber nur dann mit Hilfe eines **ax_reg**-Aufrufs, wenn der Ressourcenmanager eine Anforderung für seine Ressource empfängt. Beachten Sie, dass es keine Leistungsnachteile bei dieser Methode gibt, wenn lediglich ein Ressourcenmanager definiert ist oder wenn jeder Ressourcenmanager von jeder Transaktion verwendet wird, da die Aufrufe **ax_reg** und **xa_start** ähnliche Pfade im Transaktionsmanager haben.

Eine XA-Schnittstelle stellt eine Zweiwegeübertragung zwischen einem Transaktionsmanager und einem Ressourcenmanager her. Bei der XA-Schnittstelle handelt es sich um eine Systemebenenschnittstelle zwischen den beiden DTP-Softwarekomponenten, und nicht um eine gewöhnliche Schnittstelle für Anwendungsprogramme, für die ein Entwickler Aufrufe codiert. Anwendungsentwickler sollten jedoch mit den durch DTP-Softwarekomponenten bedingten Programmiereinschränkungen vertraut sein. Informationen zu Gesichtspunkten, die bei Programmierung unter Verwendung der X/Open XA-Schnittstelle zu beachten sind, finden Sie im Handbuch *Application Development Guide*.

Obwohl die XA-Schnittstelle unveränderlich ist, kann jeder dem XA-Standard entsprechende Transaktionmanager einen Ressourcenmanager auf eine eigene, produktspezifische Weise integrieren. Informationen zur Integration Ihres DB2-Produkts als Ressourcenmanager mit einem bestimmten Transaktionsmanager finden Sie in der Dokumentation des entsprechenden Transaktionsmanagerprodukts. Informationen zur Integration in Bezug auf die gängigsten TP-Monitore finden Sie in „Konfigurieren des XA-Transaktionsmanagers zur Verwendung von DB2 UDB" auf Seite 194.

Einrichten einer Datenbank als Ressourcenmanager

Jede Datenbank wird als ein separater Ressourcenmanager (RM) beim Transaktionsmanager (TM) definiert, und die Datenbank muss durch eine XA-Zeichenfolge zum Öffnen identifiziert werden. Eine Beschreibung des DB2-Formats der XA-Zeichenfolge zum Öffnen finden Sie unter „Verwenden der XA-Zeichenfolgen zum Öffnen und Schließen".

Verwenden der XA-Zeichenfolgen zum Öffnen und Schließen

Die XA-Zeichenfolge (xa_open) des Datenbankmanagers zum Öffnen besitzt zwei akzeptierte Formate. Ein Format wird in DB2 Version 7 neu eingeführt. Das zweite Format wird von früheren Versionen von DB2 verwendet und bleibt aus Gründen der Kompatibilität mit früheren Versionen erhalten. Neue Implementierungen sollten mit dem neuen Format arbeiten, und ältere Implementierungen sollten nach Möglichkeit auf das neue Format umgestellt werden. Zukünftige Versionen von DB2 werden möglicherweise das ältere Format für die XA-Zeichenfolge zum Öffnen nicht unterstützen. Informationen zum ursprünglichen Format der XA-Zeichenfolge zum Öffnen finden Sie in „Format der XA-Zeichenfolge zum Öffnen für frühere Versionen von DB2" auf Seite 183.

Bei der Einrichtung einer Datenbank als Ressourcenmanager ist keine XA-Zeichenfolge zum Schließen (xa_close) erforderlich. Falls eine solche Zeichenfolge angegeben wird, wird sie vom Datenbankmanager ignoriert.

Neues Format der XA-Zeichenfolge zum Öffnen für DB2 Version 7

Das folgende XA-Zeichenfolgenformat wird mit DB2 Version 7 neu eingeführt:

```
parm_id1 = <parm-wert>,parm_id2 = <parm-wert>, ...
```

Es spielt keine Rolle, in welcher Reihenfolge diese Parameter angegeben werden. Die gültigen Werte für *parm_id* werden in der folgenden Tabelle beschrieben.

Tabelle 22. Gültige Werte für parm_id

Parametername	Wert	Verbindlich?	Groß-/Kleinschreibung beachtet?	Standardwert
DB	Aliasname der Datenbank	Ja	Nein	Kein
Der Aliasname der Datenbank, der von der Anwendung für den Zugriff auf die Datenbank verwendet wird.				
UID	Benutzer-ID	Nein	Ja	Kein
Eine Benutzer-ID mit Berechtigung zur Verbindung mit der Datenbank. Erforderlich, wenn ein Kennwort angegeben wird.				
PWD	Kennwort	Nein	Ja	Kein
Ein zur Benutzer-ID gehöriges Kennwort. Erforderlich, wenn eine Benutzer-ID angegeben wird.				
TPM	Name des TP-Monitors	Nein	Nein	Kein
Name des verwendeten TP-Monitors. Unterstützte Werte siehe „Werte für TPM und TP_MON_NAME" auf Seite 180. Dieser Parameter kann angegeben werden, um mehreren TP-Monitoren die Verwendung eines einzigen DB2-Exemplars zu ermöglichen. Der angegebene Wert überschreibt den im Konfigurationsparameter *tp_mon_name* des Datenbankmanagers definierten Wert.				
AXLIB	Bibliothek, die die Funktionen **ax_reg** und **ax_unreg** des TP-Monitors enthält	Nein	Ja	Kein
Dieser Wert wird von DB2 zum Abrufen der Adressen der erforderlichen Funktionen **ax_reg** und **ax_unreg** verwendet. Er kann genutzt werden, um aufgrund des Parameters TPM angenommene Werte zu überschreiben, oder er kann von TP-Monitoren genutzt werden, die nicht in der Liste für TPM aufgeführt sind.				
CHAIN_END	xa_end-Kettungsmarkierung. Gültige Werte sind `T`, `F` oder kein Wert.	Nein	Nein	F

Tabelle 22. Gültige Werte für parm_id (Forts.)

Parametername	Wert	Verbindlich?	Groß-/Kleinschreibung beachtet?	Standardwert
Die XA_END-Kettung ist eine Optimierungsmaßnahme, die von DB2 zur Verringerung von Netzwerkübertragungen genutzt werden kann. Wenn die TP-Monitorumgebung so eingerichtet ist, dass gewährleistet werden kann, dass **xa_prepare** innerhalb desselben Thread oder Prozesses unmittelbar nach einem Aufruf von **xa_end** aufgerufen wird und CHAIN_END aktiviert ist, wird die xa_end-Markierung mit dem Befehl **xa_prepare** verkettet, um so eine Netzwerkübertragung einzusparen. Der Wert `T` bedeutet, dass CHAIN_END aktiviert ist. Der Wert `F` bedeutet, dass CHAIN_END inaktiv ist. Kein Wert bedeutet, dass CHAIN_END aktiviert ist. Dieser Parameter kann dazu dienen, die aus einem angegebenen TPM-Wert abgeleitete Einstellung zu überschreiben.				
SUSPEND_CURSOR	Gibt an, ob Cursor beizubehalten sind, wenn ein Transaktions-Thread der Steuerung angehalten wird. Gültige Werte sind `T`, `F` oder kein Wert.	Nein	Nein	F
TP-Monitore, die eine Transaktionsverzweigung anhalten, können den angehaltenen Thread oder Prozess für andere Transaktionen wiederverwenden. In diesen Fällen müssen Cursor geschlossen werden, so dass die neue Transaktion diese nicht übernimmt. Wenn eine angehaltene Transaktion wieder aufgenommen wird, muss die Anwendung den Cursor erneut abrufen. Wenn SUSPEND_CURSOR aktiv ist, werden keine geöffneten Cursor geschlossen, jedoch kann der Thread oder Prozess nicht für andere Transaktionen wiederverwendet werden. Es ist nur die Wiederaufnahme der unterbrochenen Transaktion zulässig. Der Wert `T` bedeutet, dass SUSPEND_CURSOR aktiviert ist. Der Wert `F` bedeutet, dass SUSPEND_CURSOR inaktiv ist. Kein angegebener Wert bedeutet, dass SUSPEND_CURSOR aktiviert ist. Dieser Parameter kann dazu dienen, die aus einem angegebenen TPM-Wert abgeleitete Einstellung zu überschreiben.				
HOLD_CURSOR	Gibt an, ob Cursor über Festschreibungen von Transaktionen hinweg beibehalten werden. Gültige Werte sind `T`, `F` oder kein Wert.	Nein	Nein	F

Tabelle 22. Gültige Werte für parm_id (Forts.)

Parametername	Wert	Verbindlich?	Groß-/Klein-schreibung beachtet?	Standardwert
TP-Monitore verwenden Threads oder Prozesse in der Regel für mehrere Anwendungen wieder. Um sicherzustellen, dass neu geladene Anwendungen keine von einer früheren Anwendung geöffneten Cursor übernehmen, werden Cursor nach einer Festschreibung (COMMIT) geschlossen. Wenn HOLD_CURSOR aktiviert ist, werden Cursor über Festschreibungen von Transaktionen hinweg beibehalten. Der Wert `T` bedeutet, dass HOLD_CURSOR aktiviert ist. Der Wert `F` bedeutet, dass HOLD_CURSOR inaktiv ist. Kein angegebener Wert bedeutet, dass HOLD_CURSOR aktiviert ist. Dieser Parameter kann dazu dienen, die aus einem angegebenen TPM-Wert abgeleitete Einstellung zu überschreiben.				

Werte für TPM und TP_MON_NAME

Der Parameter TPM für die XA-Zeichenfolge zum Öffnen (xa_open) und der Konfigurationsparameter *tp_mon_name* des Datenbankmanagers dienen dazu, für DB2 zu definieren, welcher TP-Monitor verwendet wird. Der Wert für *tp_mon_name* gilt für das gesamte DB2-Exemplar. Der Parameter TPM gilt nur für den speziellen XA-Ressourcenmanager. Der Wert für TPM überschreibt den Parameter *tp_mon_name*. Die folgenden Werte für die Parameter TPM und *tp_mon_name* sind gültig:

Tabelle 23. Gültige Werte für TPM und tp_mon_name

TPM-Wert	TP-Monitor-produkt	Interne Einstellungen

CICS	IBM TxSeries CICS	`AXLIB=libEncServer (für Windows)`
		`=/usr/lpp/encina/lib/libEncServer`
		`(für auf UNIX basierende Systeme)`
		`HOLD_CURSOR=T`
		`CHAIN_END=T`
		`SUSPEND_CURSOR=F`

ENCINA	IBM TxSeries Encina Monitor	`AXLIB=libEncServer (für Windows)`
		`=/usr/lpp/encina/lib/libEncServer`
		`(für auf UNIX basierende Systeme)`
		`HOLD_CURSOR=F`
		`CHAIN_END=T`
		`SUSPEND_CURSOR=F`

MQ	IBM MQSeries	`AXLIB=mqmax (für Windows)`
		`=/usr/mqm/lib/libmqmax.a`
		`(für AIX)`
		`=/opt/mqm/lib/libmqmax.a`
		`(für Solaris)`
		`HOLD_CURSOR=F`
		`CHAIN_END=F`
		`SUSPEND_CURSOR=F`

Tabelle 23. Gültige Werte für TPM und tp_mon_name (Forts.)

TPM-Wert	TP-Monitor-produkt	Interne Einstellungen	

CB	IBM Component Broker	`AXLIB=somtrx1i (für Windows)`
		`=libsomtrx1`
		`(für auf UNIX basierende Systeme)`
		`HOLD_CURSOR=F`
		`CHAIN_END=T`
		`SUSPEND_CURSOR=F`	

SF	IBM San Francisco	`AXLIB=ibmsfDB2`
		`HOLD_CURSOR=F`
		`CHAIN_END=T`
		`SUSPEND_CURSOR=F`	

TUXEDO	BEA Tuxedo	`AXLIB=libtux`
		`HOLD_CURSOR=F`
		`CHAIN_END=F`
		`SUSPEND_CURSOR=F`	

MTS	Microsoft Transaction Server		Es ist nicht nötig, DB2 für MTS zu konfigurieren. MTS wird vom ODBC-Treiber von DB2 automatisch erkannt.
JTA	Java Transaction API		Es ist nicht nötig, DB2 für Enterprise Java Servers (EJS) wie IBM WebSphere zu konfigurieren. Der JDBC-Treiber von DB2 erkennt diese Umgebung automatisch.

Beispiele

1. Sie verwenden IBM TxSeries CICS unter Windows NT. Der Dokumentation zu TxSeries ist zu entnehmen, dass Sie den Parameter *tp_mon_name* mit dem Wert `libEncServer:C` konfigurieren müssen. Dies ist immer noch ein akzeptables Format. Allerdings haben Sie mit DB2 UDB oder DB2 Connect Version 7 folgende Auswahl:
   - Angeben des Wertes `CICS` für den Parameter *tp_mon_name* (empfohlen für dieses Szenario):

     ```
     db2 update dbm cfg using tp_mon_name CICS
     ```

Für jede Datenbank, die in der Initialisierungszeichenfolge Region—> Resources—> Product—> XAD—> Resource manager für CICS definiert ist, geben Sie Folgendes an:

```
db=dbalias,uid=benutzer-id,pwd=kennwort
```

- Für jede Datenbank, die in der Initialisierungszeichenfolge Region—> Resources—> Product—> XAD—> Resource manager für CICS definiert ist, geben Sie Folgendes an:

```
db=dbalias,uid=benutzer-id,pwd=kennwort,tpm=cics
```

2. Sie verwenden IBM MQSeries unter Windows NT. Der Dokumentation zu MQSeries ist zu entnehmen, dass Sie den Parameter *tp_mon_name* mit dem Wert `mqmax` konfigurieren müssen. Dies ist immer noch ein akzeptables Format. Allerdings haben Sie mit DB2 UDB oder DB2 Connect Version 7 folgende Auswahl:
   - Angeben des Wertes `MQ` für den Parameter *tp_mon_name* (empfohlen für dieses Szenario):

     ```
     db2 update dbm cfg using tp_mon_name MQ
     ```

     Für jede Datenbank, die in der Initialisierungszeichenfolge Region—> Resources—> Product—> XAD—> Resource manager für CICS definiert ist, geben Sie Folgendes an:

     ```
     uid=benutzer-id,db=dbalias,pwd=kennwort
     ```

   - Für jede Datenbank, die in der Initialisierungszeichenfolge Region—> Resources—> Product—> XAD—> Resource manager für CICS definiert ist, geben Sie Folgendes an:

     ```
     uid=benutzer-id,db=dbalias,pwd=kennwort,tpm=mq
     ```

3. Sie verwenden sowohl IBM TxSeries CICS als auch IBM MQSeries unter Windows NT. Es wird ein einziges DB2-Exemplar verwendet. In diesem Szenario würden Sie wie folgt konfigurieren:
   a. Für jede Datenbank, die in der Initialisierungszeichenfolge Region—> Resources—> Product—> XAD—> Resource manager für CICS definiert ist, geben Sie Folgendes an:

      ```
      pwd=kennwort,uid=benutzer-id,tpm=cics,db=dbalias
      ```

   b. Für jede Datenbank, die als Ressource in den Eigenschaften des Warteschlangenmanagers definiert ist, geben Sie eine XA-Zeichenfolge zum Öffnen wie folgt an:

      ```
      db=dbalias,uid=benutzer-id,pwd=kennwort,tpm=mq
      ```

4. Sie entwickeln einen eigenen XA-konformen Transaktionsmanager (XA TM) unter Windows NT, und Sie wollen DB2 mitteilen, dass die Bibliothek "meinaxlib" die erforderlichen Funktionen **ax_reg** und **ax_unreg** enthält. Die Bibliothek "meinaxlib" befindet sich in einem in der Anweisung PATH definierten Verzeichnis. Sie haben die folgende Auswahl:
   - Angeben des Wertes `meinaxlib` für den Parameter *tp_mon_name*:

```
db2 update dbm cfg using tp_mon_name meinaxlib
```

Und für jede Datenbank, die für den XA TM definiert ist, Angeben einer XA-Zeichenfolge zum Öffnen:

```
db=dbalias,uid=benutzer-id,pwd=kennwort
```

- Für jede Datenbank, die für den XA TM definiert ist, Angeben einer XA-Zeichenfolge zum Öffnen:

```
db=dbalias,uid=benutzer-id,pwd=kennwort,axlib=meinaxlib
```

5. Sie entwickeln einen eigenen XA-konformen Transaktionsmanager (XA TM) unter Windows NT, und Sie wollen DB2 mitteilen, dass die Bibliothek "meinaxlib" die erforderlichen Funktionen **ax_reg** und **ax_unreg** enthält. Die Bibliothek "meinaxlib" befindet sich in einem in der Anweisung PATH definierten Verzeichnis. Außerdem wollen Sie die XA-END-Kettung aktivieren. Sie haben die folgende Auswahl:
   - Für jede Datenbank, die für den XA TM definiert ist, Angeben einer XA-Zeichenfolge zum Öffnen:

```
db=dbalias,uid=benutzer-id,pwd=kennwort,axlib=meinaxlib,chain_end=T
```

   - Für jede Datenbank, die für den XA TM definiert ist, Angeben einer XA-Zeichenfolge zum Öffnen:

```
db=dbalias,uid=benutzer-id,pwd=kennwort,axlib=meinaxlib,chain_end
```

Format der XA-Zeichenfolge zum Öffnen für frühere Versionen von DB2

Frühere Versionen von DB2 arbeiten mit dem im Folgenden beschriebenen Format der XA-Zeichenfolge zum Öffnen (xa_open). Dieses Format wird aus Gründen der Kompatibilität weiterhin unterstützt. Anwendungen sollten nach Möglichkeit auf das neue Format (siehe „Neues Format der XA-Zeichenfolge zum Öffnen für DB2 Version 7" auf Seite 177) umgestellt werden.

Jede Datenbank wird als ein separater Ressourcenmanager (RM) beim Transaktionsmanager (TM) definiert, und die Datenbank muss durch eine XA-Zeichenfolge zum Öffnen mit der folgenden Syntax identifiziert werden:

```
"db_alias<,benutzer-id,kennwort>"
```

Der *db_alias* ist erforderlich, um den Aliasnamen der Datenbank anzugeben. Der Aliasname ist derselbe Name wie der Datenbankname, sofern Sie nicht explizit einen Aliasnamen nach der Erstellung der Datenbank katalogisiert haben. Der Benutzername und das Kennwort sind wahlfrei und werden je nach Methode zur Identifikationsüberprüfung verwendet, um Informationen zur Identifikationsüberprüfung an die Datenbank weiterzugeben.

Bei der Einrichtung einer Datenbank als Ressourcenmanager ist keine XA-Zeichenfolge zum Schließen (xa_close) erforderlich. Falls eine solche Zeichenfolge angegeben wird, wird sie vom Datenbankmanager ignoriert.

Aktualisieren von Host- oder AS/400-Datenbank-Servern

Host- und AS/400-Datenbank-Server können aktualisierbar sein, je nach Architektur des XA-Transaktionsmanagers. Zur Unterstützung von Festschreibungssequenzen aus verschiedenen Prozessen muss der DB2 Connect-Konzentrator aktiviert werden. Zur Aktivierung des DB2 Connect EE-Konzentrators setzen Sie den Konfigurationsparameter *max_logicagents* des Datenbankmanagers auf einen Wert, der größer ist als *maxagents*. Beachten Sie, dass für den DB2 Connect EE-Konzentrator ein Client mit DB2 Version 7.1 zur Unterstützung von XA-Festschreibungssequenzen aus verschiedenen Prozessen erforderlich ist. Informationen zu SQL-Anweisungen, die in dieser Umgebung zulässig sind, finden Sie im Handbuch *Application Development Guide*. Informationen zum Konzentrator finden Sie im *DB2 Connect Benutzerhandbuch*.

Wenn Sie die Host- oder AS/400-Datenbank-Server aktualisieren wollen, benötigen Sie DB2 Connect mit konfiguriertem DB2-Synchronisationspunktmanager (SPM). Anweisungen finden Sie in einem der Handbücher *Einstieg*.

Überlegungen zu Datenbankverbindungen

Dieser Abschnitt behandelt die folgenden Themen:

- „Anweisung RELEASE"
- „Transaktionen, die auf partitionierte Datenbanken zugreifen"

Anweisung RELEASE

Wenn die Anweisung RELEASE verwendet wurde, um die Verbindung zu einer Datenbank freizugeben, sollte die Anweisung CONNECT und nicht die Anweisung SET CONNECTION zur erneuten Herstellung der Verbindung zu dieser Datenbank verwendet werden.

Transaktionen, die auf partitionierte Datenbanken zugreifen

In einer Umgebung mit partitionierten Datenbanken können Benutzerdaten auf mehrere Datenbankpartitionen verteilt werden. Eine Anwendung, die auf eine solche Datenbank zugreift, stellt die Verbindung her und sendet Anforderungen an eine der Partitionen der Datenbank (den Koordinatorknoten). Verschiedene Anwendungen können die Verbindung zu verschiedenen Datenbankpartitionen herstellen, und dieselbe Anwendung kann verschiedene Datenbankpartitionen für verschiedene Verbindungen auswählen.

Für Transaktionen, die an einer Datenbank in einer Umgebung mit partitionierten Datenbanken ausgeführt werden, muss der gesamte Zugriff über *gleiche* Datenbankpartition erfolgen. Das bedeutet, dass die gleiche Datenbankpartition vom Beginn der Transaktion bis zu dem Zeitpunkt (einschließlich) verwendet werden muss, zu dem die Transaktion festgeschrieben wird.

Jede Transaktion für die partitionierte Datenbank muss festgeschrieben werden, bevor die Verbindung getrennt wird.

Treffen einer heuristischen Entscheidung

Ein dem XA-Standard entsprechender Transaktionsmanager arbeitet mit einem ähnlichen zweiphasigen Festschreibeprozess wie der DB2-Transaktionsmanager, der in „Prozess der zweiphasigen Festschreibung" auf Seite 164 beschrieben wird. Der Hauptunterschied zwischen diesen beiden Umgebungen besteht darin, dass hier der TP-Monitor die Funktionen zur Protokollierung und Steuerung zur Verfügung stellt, und nicht der DB2-Transaktionsmanager und die Transaktionsmanagerdatenbank.

Bei der Verwendung eines dem XA-Standard entsprechenden Transaktionsmanagers können Fehler auftreten, die den für den DB2-Transaktionsmanager beschriebenen Fehlern ähnlich sind (siehe „Beheben von Problemen bei der zweiphasigen Festschreibung" auf Seite 167). Ähnlich wie der DB2-Transaktionsmanager versucht auch ein XA-Transaktionsmanager unbestätigte Transaktionen zu resynchronisieren.

Wenn Sie aus einem bestimmten Grund nicht darauf warten können, dass der Transaktionsmanager unbestätigte Transaktionen automatisch auflöst, haben Sie die Möglichkeit, Maßnahmen zur manuellen Auflösung unbestätigter Transaktionen zu ergreifen. Dieser manuelle Prozess wird manchmal als das „Treffen einer heuristischen Entscheidung" bezeichnet.

Der Befehl LIST INDOUBT TRANSACTIONS (mit der Option WITH PROMPTING) bzw. eine entsprechende Gruppe von APIs ermöglicht es, unbestätigte Transaktionen abzufragen, festzuschreiben (COMMIT) oder rückgängig zu machen (ROLLBACK). Darüber hinaus können mit diesem Befehl Transaktionen ignoriert werden („FORGET"), die heuristisch festgeschrieben oder rückgängig gemacht wurden, indem die Protokollsätze entfernt und der Protokollspeicherbereich freigegeben werden. Zum Abfragen von Informationen über unbestätigte Transaktionen aus DB2 UDB auf UNIX-Systemen, auf dem Windows-Betriebssystem oder OS/2, stellen Sie eine Verbindung zu der Datenbank her und setzen den Befehl LIST INDOUBT TRANSACTIONS WITH PROMPTING ab bzw. verwenden die entsprechende API. Weitere Informationen zu diesem Befehl oder zu den entsprechenden administrativen APIs finden Sie im Handbuch *Command Reference* bzw. *Administrative API Reference*.

Zum Abrufen von Informationen zu unbestätigten Transaktionen in Bezug auf Host- oder AS/400-Datenbank-Server stehen zwei Möglichkeiten zur Verfügung:

- Sie können Informationen zu unbestätigten Transaktionen direkt vom Host- oder AS/400-Server abrufen.

  Zum Abrufen von Informationen zu unbestätigten Transaktionen direkt aus DB2 für OS/390 rufen Sie den Befehl DISPLAY THREAD TYPE(INDOUBT) auf. Mit Hilfe des Befehls RECOVER können Sie eine heuristische Entschei-

dung treffen. Zum direkten Abrufen von Informationen zu unbestätigten Transaktionen aus DB2 für OS/400 rufen Sie den Befehl **wrkcmtdfn** auf.

- Sie können Informationen zu unbestätigten Transaktionen direkt aus dem DB2 Connect-Server abrufen, der für den Zugriff auf den Host- oder AS/400-Datenbank-Server verwendet wird.

  Zum Abrufen von Informationen zu unbestätigten Transaktionen vom DB2 Connect-Server stellen Sie zunächst eine Verbindung zum DB2-Synchronisationspunktmanager her, indem Sie eine Verbindung zu dem DB2-Exemplar herstellen, das durch den Wert des Konfigurationsparameters *spm_name* des Datenbankmanagers definiert ist. Anschließend setzen Sie den Befehl LIST DRDA INDOUBT TRANSACTIONS WITH PROMPTING ab, um unbestätigte Transaktionen anzuzeigen und heuristische Entscheidungen zu treffen.

Verwenden Sie diese Befehle (bzw. die entsprechenden APIs) nur mit *äußerster Vorsicht* und nur als allerletzte Möglichkeit. Die beste Strategie ist, zu warten, bis der Transaktionsmanager den Prozess der Resynchronisation durchgeführt hat. Es kann zu Problemen mit der Datenintegrität kommen, wenn Sie eine Transaktion in einer der beteiligten Datenbanken manuell festschreiben oder rückgängig machen und die entgegengesetzte Maßnahme für eine andere beteiligte Datenbank durchgeführt wird. Die Behebung von Problemen der Datenintegrität setzt voraus, dass Sie die Logik der Anwendung und die geänderten oder rückgängig gemachten Daten kennen, und Sie dann eine Wiederherstellung bis zu einem bestimmten Zeitpunkt durchführen bzw. die Datenbankänderungen manuell rückgängig machen oder erneut anwenden.

Wenn Sie nicht darauf warten können, bis der Transaktionsmanager den Prozess zur Resynchronisation initialisiert, und die Ressourcen, die von einer unbestätigten Transaktion blockiert werden, freigeben müssen, sind heuristische Maßnahmen erforderlich. Diese Situation kann auftreten, wenn der Transaktionsmanager für einen längeren Zeitraum zur Durchführung der Resynchronisation nicht verfügbar ist und die unbestätigte Transaktion Ressourcen sperrt, die dringend benötigt werden. Eine unbestätigte Transaktion blockiert Ressourcen, die dieser Transaktion zugeordnet wurden, als der Transaktionsmanager oder die Ressourcenmanager noch verfügbar waren. Im Fall des Datenbankmanagers gehören zu diesen Ressourcen zum Beispiel die Sperren für Tabellen und Indizes, der Protokollspeicherbereich und der Speicher, der von der Transaktion belegt wird. Jede unbestätigte Transaktion verringert außerdem die maximale Anzahl gleichzeitiger Transaktionen (um den Wert 1), die von der Datenbank verarbeitet werden können.

Obwohl es keine absolut sichere Methode zur Durchführung heuristischer Maßnahmen gibt, werden im Folgenden einige allgemeine Richtlinien beschrieben:

1. Stellen Sie die Verbindung zu der Datenbank her, für die alle Transaktionen abgeschlossen werden sollen.
2. Verwenden Sie den Befehl LIST INDOUBT TRANSACTIONS, um die unbestätigten Transaktionen anzuzeigen. Die *xid* stellt die ID der globalen Transaktion dar und ist mit der *xid* identisch, die vom Transaktionsmanager und von anderen Ressourcenmanagern, die an der Transaktion beteiligt sind, verwendet wird.
3. Bestimmen Sie mit Hilfe Ihrer Kenntnisse über die Anwendung und die Betriebsumgebung für jede unbestätigte Transaktion die anderen beteiligten Ressourcenmanager.
4. Stellen Sie fest, ob der Transaktionsmanager verfügbar ist:
   - Wenn der Transaktionsmanager verfügbar ist und die unbestätigte Transaktion in einem Ressourcenmanager dadurch verursacht wurde, dass der Ressourcenmanager in der zweiten Phase der Festschreibung nicht verfügbar war, oder für einen früheren Resynchronisationsprozess, sollten Sie das Protokoll des Transaktionsmanagers überprüfen, um festzustellen, welche Aktion für die anderen Ressourcenmanager durchgeführt wurde. Anschließend sollten Sie dieselbe Aktion für die Datenbank durchführen, d. h. mit dem Befehl LIST INDOUBT TRANSACTIONS die Transaktion entweder heuristisch festschreiben (COMMIT) oder heuristisch rückgängig machen (ROLLBACK).
   - Wenn der Transaktionsmanager *nicht* verfügbar ist, müssen Sie den Status der Transaktion in den anderen beteiligten Ressourcenmanagern verwenden, um festzustellen, welche Aktion durchzuführen ist:
     – Wenn mindestens einer der anderen Ressourcenmanager die Transaktion festgeschrieben hat, dann sollten Sie die Transaktion in allen anderen Ressourcenmanagern heuristisch festschreiben (COMMIT).
     – Wenn mindestens einer der anderen Ressourcenmanager die Transaktion rückgängig gemacht hat, sollten Sie die Transaktion heuristisch rückgängig machen (ROLLBACK).
     – Wenn die Transaktion den Status "prepared" (unbestätigt) in allen beteiligten Ressourcenmanagern hat, sollten Sie die Transaktion heuristisch rückgängig machen.
     – Wenn einer oder mehrere der anderen Ressourcenmanager nicht verfügbar sind, sollten Sie die Transaktion heuristisch rückgängig machen.

Führen Sie die heuristische Funktion FORGET nicht aus, sofern nicht eine heuristisch festgeschriebene oder rückgängig gemachte Transaktion zu der Bedingung führt, dass das Protokoll voll ist, die in der Ausgabe des Befehls LIST INDOUBT TRANSACTIONS angezeigt wird. Die heuristische Funktion FORGET gibt den Protokollspeicherbereich frei, der durch eine unbestätigte Transaktion belegt wird. Dies könnte zur Folge haben, dass, wenn ein Transaktionsmanager schließlich einen Resynchronisationsprozess für diese

unbestätigte Transaktion durchführt, er eine falsche Entscheidung, andere Ressourcenmanager festzuschreiben oder rückgängig zu machen, treffen könnte, weil für die Transaktion in diesem Ressourcenmanager kein Protokollsatz mehr vorhanden wäre. Allgemein impliziert ein „fehlender" Protokollsatz, dass der Ressourcenmanager die Transaktion rückgängig gemacht hat.

Überlegungen zur Sicherheit

Der TP-Monitor ordnet eine Gruppe von Server-Prozessen im Voraus zu und führt die Transaktionen von verschiedenen Benutzern unter den IDs der Server-Prozesse aus. Für die Datenbank erscheint jeder Server-Prozess wie eine große Anwendung, die viele Arbeitseinheiten vereinigt, die alle unter derselben, dem Server-Prozess zugeordneten ID ausgeführt werden.

Wenn zum Beispiel in einer AIX-Umgebung mit CICS eine TXSeries CICS-Region gestartet wird, wird ihr der AIX-Benutzername zugeordnet, unter dem sie definiert ist. Alle CICS-Anwendungs-Server-Prozesse werden ebenfalls unter dieser TXSeries CICS-Haupt-ID ausgeführt, die in der Regel als ″cics″ definiert ist. CICS-Benutzer können CICS-Transaktionen unter ihrer DCE-Anmelde-ID aufrufen und, während sie in CICS arbeiten, ihre ID mit Hilfe der CESN-Anmeldetransaktion auch ändern. In beiden Fällen ist für den Ressourcenmanager die Endbenutzer-ID nicht verfügbar. Folglich kann ein CICS-Anwendungsprozess Transaktionen für viele Benutzer ausführen, aber sie erscheinen dem Ressourcenmanager wie ein einziges Programm mit vielen Arbeitseinheiten von derselben ID ″cics″. Wahlfrei können Sie eine Benutzer-ID und ein Kennwort in der XA-Zeichenfolge zum Öffnen angeben, so dass diese Benutzer-ID anstelle der ID ″cics″ für die Verbindung mit der Datenbank verwendet wird.

Die Auswirkungen sind für statische SQL-Anweisungen nicht groß, da zum Zugriff auf die Datenbank die Zugriffsrechte der Person, die das Paket gebunden hat, und nicht die Zugriffsrechte des Endbenutzers verwendet werden. Das bedeutet aber, dass das Zugriffsrecht EXECUTE der Datenbankpakete der Server-ID, und nicht der Endbenutzer-ID erteilt werden muss.

Für dynamische Anweisungen, für deren Zugriff die Authentifizierung zur Laufzeit erfolgt, müssen die Zugriffsrechte der Datenbankobjekte der Server-ID, und nicht dem tatsächlichen Benutzer dieser Objekte erteilt werden. Anstatt sich bei der Steuerung des Zugriffs bestimmter Benutzer auf die Datenbank zu stützen, müssen Sie über das TP-Monitorsystem festlegen, welche Benutzer welche Programme ausführen können. Der Server-ID müssen alle Zugriffsrechte erteilt werden, die für die SQL-Benutzer erforderlich sind.

Um festzustellen, wer auf eine Datenbanktabelle oder -sicht zugegriffen hat, können Sie folgende Schritte unternehmen:

1. Erstellen Sie sich aus der Katalogsicht SYSCAT.PACKAGEDEP eine Liste aller Pakete, die von der Tabelle oder Sicht abhängig sind.

2. Bestimmen Sie die Namen der Server-Programme (z. B. CICS-Programme), die diesen Paketen aufgrund der in Ihrer Installation verwendeten Namenskonvention entsprechen.
3. Bestimmen Sie die Client-Programme (z. B. CICS-Transaktions-IDs), die diese Programme aufrufen konnten, und verwenden Sie anschließend das Protokoll des TP-Monitors (z. B. das CICS-Protokoll), um festzustellen, wer diese Transaktionen oder Programme wann ausgeführt hat.

Überlegungen zur Konfiguration

Bei der Einrichtung der TP-Monitorumgebung sollten die folgenden Konfigurationsparameter berücksichtigt werden:

- *tp_mon_name*

  Mit diesem Konfigurationsparameter des Datenbankmanagers wird der Name des verwendeten TP-Monitorprodukts (z. B. ″CICS″ oder ″ENCINA″) angegeben.

- *tpname*

  Mit diesem Konfigurationsparameter des Datenbankmanagers wird der Name des fernen Transaktionsprogramms angegeben, das der Datenbank-Client verwenden muss, wenn er eine Zuordnungsanforderung an den Datenbank-Server über das APPC-Übertragungsprotokoll absetzt. Der Wert wird in der Konfigurationsdatei auf dem Server definiert und muss mit dem im SNA-Transaktionsprogramm konfigurierten TP-Namen für den Transaktionsprozessor übereinstimmen. Weitere Informationen finden Sie in den Handbüchern *Einstieg*.

- *tm_database*

  Da DB2 Transaktionen in der XA-Umgebung *nicht* koordiniert, wird dieser Konfigurationsparameter des Datenbankmanagers für XA-koordinierte Transaktionen nicht verwendet.

- *maxappls*

  Mit diesem Konfigurationsparameter der Datenbank wird die zulässige maximale Anzahl aktiver Anwendungen angegeben. Der Wert dieses Parameters muss gleich oder größer sein als die Summe der verbundenen Anwendungen plus die Anzahl dieser Anwendungen, die sich gleichzeitig im Prozess einer zweiphasigen Festschreibung bzw. einer ROLLBACK-Operation befinden können. Diese Summe sollte dann um die vorab geschätzte Anzahl unbestätigter Transaktionen erhöht werden, die zu einem beliebigen Zeitpunkt gleichzeitig vorhanden sein könnten. Weitere Informationen zu unbestätigten Transaktionen finden Sie in „Beheben von Problemen bei der zweiphasigen Festschreibung” auf Seite 167.

  Für eine TP-Monitorumgebung (z. B. TXSeries CICS) müssen Sie den Wert des Parameters *maxappls* erhöhen. Dies hilft bei der Sicherstellung, dass alle TP-Monitorprozesse verarbeitet werden können.

- *autorestart*

  Mit diesem Konfigurationsparameter der Datenbank wird festgelegt, ob die Routine RESTART DATABASE bei Bedarf aufgerufen wird. Der Standardwert ist YES (d. h. aktiviert).

  Für eine Datenbank, die unbestätigte Transaktionen enthält, ist eine RESTART DATABASE-Operation für den Neutstart erforderlich. Wenn *autorestart* nicht aktiviert ist und die letzte Verbindung zur Datenbank getrennt wurde, schlägt die nächste Verbindungsanforderung fehl, so dass ein explizites Aufrufen des Befehls RESTART DATABASE erforderlich wird. Diese Bedingung bleibt solange bestehen, bis die unbestätigten Transaktionen entweder durch die Resynchronisationsoperation des Transaktionsmanagers oder durch eine vom Administrator eingeleitete heuristische Operation beseitigt wird. Wenn der Befehl RESTART DATABASE abgesetzt wird, wird eine Nachricht zurückgegeben, wenn unbestätigte Transaktionen in der Datenbank vorhanden sind. Der Administrator kann dann mit Hilfe des Befehls LIST INDOUBT TRANSACTIONS und anderer Befehle des Befehlszeilenprozessors Informationen über diese unbestätigten Transaktionen erhalten.

Unterstützte XA-Funktion

DB2 Universal Database unterstützt die Spezifikation XA91, die in *X/Open CAE Specification Distributed Transaction Processing: The XA Specification* definiert ist, mit folgenden Ausnahmen:

- Asynchrone Services

  Die XA-Spezifikation ermöglicht der Schnittstelle die Verwendung asynchroner Services, so dass das Ergebnis einer Anforderung zu einem späteren Zeitpunkt überprüft werden kann. Für den Datenbankmanager müssen die Anforderungen im synchronen Modus aufgerufen werden.

- Statische Registrierung

  Die XA-Schnittstelle ermöglicht zwei Methoden zur Registrierung eines Ressourcenmanagers: statische und dynamische Registrierung. DB2 Universal Database unterstützt nur die dynamische Registrierung, die ausgereifter und effizienter ist. Weitere Informationen zu diesen beiden Methoden finden Sie in „Ressourcenmanager (RM)" auf Seite 176.

- Migration von Zuordnungen

  DB2 Universal Database unterstützt die Transaktionsmigration zwischen Threads der Steuerung nicht.

Informationen zur Verwendung der XA-Zeichenfolgen zum Öffnen und zum Schließen (xa_open und xa_close) finden Sie in „Verwenden der XA-Zeichenfolgen zum Öffnen und Schließen" auf Seite 177.

Syntax und Position der XA-Schalter

Wie für die XA-Schnittstelle erforderlich stellt der Datenbankmanager eine externe C-Variable *db2xa_switch* des Typs xa_switch_t bereit, um die XA-Schalterstruktur an den Transaktionsmanager zurückzugeben. Neben den Adressen verschiedener XA-Funktionen werden folgende Felder zurückgegeben:

Feld	Wert
name	Der Produktname des Datenbankmanagers. Zum Beispiel: DB2 für AIX.
flags	TMREGISTER | TMNOMIGRATE Gibt explizit an, dass DB2 Universal Database die dynamische Registrierung verwendet und dass der TM keine Migration von Zuordnungen verwenden soll. Gibt implizit an, dass ein asynchroner Betrieb nicht unterstützt wird.
version	Muss null sein.

Verwenden des XA-Schalters von DB2 Universal Database

Die XA-Architektur verlangt, dass ein Ressourcenmanager (RM) einen *Schalter* bereitstellt, der dem XA-Transaktionsmanager (TM) Zugriff auf die **xa**_-Routinen des Ressourcenmanagers gibt. Ein RM-Schalter verwendet eine Struktur, die als xa_switch_t bezeichnet wird. Der Schalter enthält den Namen des RMs, Nicht-NULL-Zeiger auf die XA-Eingangspunkte des RMs, eine Markierung (Flag) und eine Versionsnummer.

Auf UNIX basierende Systeme und OS/2: Der DB2 UDB-Schalter kann durch eine der folgenden Methoden abgerufen werden:

- Über eine weitere Zwischenstufe. In einem C-Programm kann dies durch Definieren des Makros:

  ```
  #define db2xa_switch (*db2xa_switch)
  ```

  vor Verwendung von *db2xa_switch* erreicht werden.
- Durch Aufrufen von **db2xacic**

  DB2 UDB stellt diese API zur Verfügung, die die Adresse der *db2xa_switch*-Struktur liefert. Der Prototyp dieser Funktion lautet:

  ```
  struct xa_switch_t * SQL_API_FN  db2xacic( )
  ```

Bei beiden Methoden müssen Sie die Anwendung mit `libdb2` (bei auf UNIX basierenden Systemen) oder `db2api.lib` (unter OS/2) verbinden (″linken″).

Windows NT: Der Zeiger auf die *xa_switch*-Struktur, *db2xa_switch*, wird in Form von DLL-Daten exportiert (d. h. bereitgestellt). Dies heißt für eine Anwendung unter Windows NT, die diese Struktur verwendet, dass sie auf die Struktur mit Hilfe einer von drei Methoden zugreifen muss:

- Über eine weitere Zwischenstufe. In einem C-Programm kann dies durch Definieren des Makros:

  ```
  #define db2xa_switch (*db2xa_switch)
  ```

  vor Verwendung von *db2xa_switch* erreicht werden.
- Bei Verwendung des Microsoft Visual C++-Compilers kann *db2xa_switch* folgendermaßen definiert werden:

  ```
  extern __declspec(dllimport) struct xa_switch_t db2xa_switch
  ```
- Durch Aufrufen von **db2xacic**

  DB2 UDB stellt diese API zur Verfügung, die die Adresse der *db2xa_switch*-Struktur liefert. Der Prototyp dieser Funktion lautet:

  ```
  struct xa_switch_t * SQL_API_FN  db2xacic( )
  ```

Bei jeder dieser Methoden müssen Sie die Anwendung mit `db2api.lib` verbinden (″linken″).

C-Beispielcode: Der folgende Code veranschaulicht die verschiedenen Methoden des Zugriffs auf *db2xa_switch* über ein C-Programm auf einer beliebigen DB2 UDB-Plattform. Stellen Sie sicher, dass die Anwendung mit der entsprechenden Bibliothek verbunden wird.

```
#include <stdio.h>
#include <xa.h>

struct xa_switch_t * SQL_API_FN  db2xacic( );

#ifdef DECLSPEC_DEFN
extern __declspec(dllimport) struct xa_switch_t db2xa_switch;
#else
#define db2xa_switch (*db2xa_switch)
extern struct xa_switch_t db2xa_switch;
#endif
main( )
   {
      struct xa_switch_t *foo;
      printf ( "%s \n", db2xa_switch.name );
      foo = db2xacic();
      printf ( "%s \n", foo->name );
      return ;
   }
```

Bestimmung von XA-Schnittstellenfehlern

Wenn ein Fehler während einer XA-Anforderung vom Transaktionsmanager festgestellt wird, ist das Anwendungsprogramm eventuell nicht in der Lage, den Fehlercode vom Transaktionsmanager zu ermitteln. Wenn Ihr Programm abnormal beendet wird oder einen unklaren Rückkehrcode vom TP-Monitor oder Transaktionsmanager empfängt, sollten Sie das Serviceprotokoll des DB2-Diagnoseprogramms überprüfen, in dem XA-Fehlerinformationen aufgezeichnet werden, wenn Diagnosestufe 3 oder höher in Kraft ist. Weitere Informationen zum Serviceprotokoll des DB2-Diagnoseprogramms finden Sie im Handbuch *Troubleshooting Guide*.

Sie sollten auch die Informationen der Konsolnachricht, der Fehlerdatei des Transaktionsmanagers oder andere produktspezifische Informationen über die verwendete externe Software zur Verarbeitung von Transaktionen berücksichtigen.

Der Datenbankmanager schreibt alle XA-spezifischen Fehler mit dem SLQ-CODE -998 (Fehler bei Transaktion oder heuristischer Maßnahme) und den entsprechenden Ursachencodes in das Serviceprotokoll des DB2-Diagnoseprogramms. Es folgen einige der häufigeren Fehlerursachen:

- Ungültige Syntax in der XA-Zeichenfolge zum Öffnen
- Fehler bei der Verbindung mit der in der XA-Zeichenfolge zum Öffnen angegebenen Datenbank aus einem der folgenden Gründe:
  - Die Datenbank ist nicht katalogisiert.
  - Die Datenbank wurde nicht gestartet.
  - Der Benutzername bzw. das Kennwort der Server-Anwendung ist zur Verbindung mit der Datenbank nicht berechtigt.
- Übertragungsfehler

Das folgende Beispiel zeigt ein Fehlerprotokoll für einen xa_open-Fehler (wegen einer fehlender XA-Zeichenfolge zum Öffnen), das unter AIX generiert wurde:

```
Tue Apr  4 15:59:08 1995
toop pid(83378) process (xatest) XA DTP Support      sqlxa_open Probe:101
DIA4701E Datenbank "" konnte nicht für verteilte
Transaktionsverarbeitung geöffnet werden.
String Title : XA Interface SQLCA  pid(83378)
SQLCODE = -998  REASON CODE: 4  SUBCODE: 1
Dump File : /u/toop/diagnostics/83378.dmp Data : SQLCA
```

Konfigurieren des XA-Transaktionsmanagers zur Verwendung von DB2 UDB

In den folgenden Abschnitten wird beschrieben, wie bestimmte Produkte zur Verwendung von DB2 als Ressourcenmanager konfiguriert werden. Möglich sind:

- „Konfigurieren von IBM TXSeries CICS"
- „Konfigurieren von IBM TXSeries Encina"
- „Konfigurieren von BEA Tuxedo" auf Seite 197
- „Konfigurieren von Microsoft Transaction Server" auf Seite 199.

Konfigurieren von IBM TXSeries CICS

Informationen zur Konfiguration von IBM TXSeries CICS zur Verwendung von DB2 als Ressourcenmanager finden Sie im Handbuch *IBM TXSeries CICS Administration Guide*. Die TXSeries-Dokumentation steht online unter http://www.transarc.com/Library/documentation/websphere/WAS-EE/en_US/html/ zur Verfügung.

Host- und AS/400-Datenbank-Server können an CICS-koordinierten Transaktionen teilnehmen.

Konfigurieren von IBM TXSeries Encina

Im folgenden werden die verschiedenen APIs und Konfigurationsparameter aufgeführt, die für die Integration von Encina Monitor und DB2 Universal Database-Servern oder DB2 für MVS, DB2 für OS/390, DB2 für AS/400 oder DB2 für VSE & VM erforderlich sind, wenn über DB2 Connect darauf zugegriffen wird. Die TXSeries-Dokumentation steht online unter http://www.transarc.com/Library/documentation/websphere/WAS-EE/en_US/html/ zur Verfügung.

Host- und AS/400-Datenbank-Server können an CICS-koordinierten Transaktionen teilnehmen.

Konfigurieren von DB2

Gehen Sie wie folgt vor, um DB2 zu konfigurieren:

1. Jeder Datenbankname muss im DB2-Datenbankverzeichnis definiert werden. Wenn es sich bei der Datenbank um eine ferne Datenbank handelt, muss auch ein Eintrag im Knotenverzeichnis definiert werden. Sie können die Konfiguration mit dem Programm Client-Konfiguration - Unterstützung oder mit dem DB2-Befehlszeilenprozessor ausführen. Beispiel:

   ```
   DB2 CATALOG DATABASE inventdb AS inventdb AT NODE host1 AUTH SERVER
   DB2 CATALOG TCPIP NODE host1 REMOTE hostname1 SERVER svcname1
   ```

2. Der DB2-Client kann seine interne Verarbeitung für Encina optimieren, wenn er weiß, dass er es mit Encina zu tun hat. Sie können dies angeben, indem Sie den Konfigurationsparameter *tp_mon_name* des Datenbankmanagers auf `ENCINA` setzen. In der Standardeinstellung erfolgt keine spezielle Optimierung. Wenn *tp_mon_name* definiert ist, muss die Anwendung sicherstellen, dass der Thread, der die Arbeitseinheit ausführt, die Arbeit nach Beendigung auch sofort festschreibt. Es darf keine andere Arbeitseinheit gestartet werden. Wenn dies *nicht* Ihre Umgebung ist, stellen Sie sicher, dass der Wert von *tp_mon_name* `NONE` ist (oder der über den Befehlszeilenprozessor auf `NULL` gesetzt wird). Der Parameter kann über die Steuerzentrale oder den Befehlszeilenprozessor aktualisiert werden. Der Befehl für den Befehlszeilenprozessor (CLP) lautet:

   ```
   db2 update dbm cfg using tp_mon_name ENCINA
   ```

Konfigurieren von Encina für jeden Ressourcenmanager

Bei der Konfiguration von Encina für jeden Ressourcenmanager (RM) muss ein Administrator die Zeichenfolge zum Öffnen, die Zeichenfolge zum Schließen und den Thread der Steuerungsvereinbarung für jede DB2-Datenbank als Ressourcenmanager definieren, damit der Ressourcenmanager für Transaktionen in einer Anwendung registriert werden kann. Die Konfiguration kann mit der Enconcole-Gesamtanzeigeschnittstelle oder mit der Encina-Befehlszeilenschnittstelle ausgeführt werden. Beispiel:

```
monadmin create rm inventdb -open "db=inventdb,uid=benutzer1,pwd=kennwort1"
```

Es gibt eine Ressourcenmanagerkonfiguration für jede DB2-Datenbank, und jede Konfiguration eines Ressourcenmanagers(RM) muss einen rm-Namen („logischen RM-Namen") haben. Dieser sollte zur Vereinfachung mit dem Datenbanknamen übereinstimmen.

Die XA-Zeichenfolge zum Öffnen (xa_open) enthält Informationen, die erforderlich sind, um eine Verbindung zur Datenbank herzustellen. Der Inhalt der Zeichenfolge ist RM-spezifisch. Die XA-Zeichenfolge zum Öffnen von DB2 UDB enthält den Aliasnamen der Datenbank, die geöffnet werden soll, und wahlweise eine Benutzer-ID und ein Kennwort, die der Verbindung zugeordnet werden sollen. Beachten Sie, dass der hier definierte Datenbankname auch in dem normalen Datenbankverzeichnis katalogisiert werden muss, das für alle Datenbankzugriffe erforderlich ist. Informationen über die XA-Zeichenfolge zum Öffnen von DB2 finden Sie in „Einrichten einer Datenbank als Ressourcenmanager" auf Seite 177.

Die XA-Zeichenfolge zum Schließen (xa_close) wird von DB2 nicht verwendet.

Der Thread der Steuerungsvereinbarung (Thread of Control Agreement) bestimmt, ob ein Anwendungsagent-Thread mehrere Transaktionen gleichzeitig verarbeiten kann. DB2 UDB unterstützt den Standardwert von TMXA_SE-

RIALIZE_ALL_OPERATIONS, wonach ein Thread nur dann wiederverwendet werden kann, wenn eine Transaktion vollständig beendet ist.

Wenn Sie auf DB2 für OS/390, DB2 für MVS, DB2 für AS/400 oder DB2 für VSE & VM zugreifen, müssen Sie den DB2-Synchronisationspunktmanager verwenden. Weitere Informationen zur Konfiguration finden Sie im Handbuch *DB2 Connect Enterprise Edition für OS/2 und Windows Einstieg*.

Verweisen auf eine DB2-Datenbank aus einer Encina-Anwendung

Gehen Sie wie folgt vor, um auf eine DB2-Datenbank aus einer Encina-Anwendung zu verweisen:

1. Verwenden Sie die Encina Scheduling Policy-API, um anzugeben, wie viele Anwendungsagenten aus einem einzelnen TP-Monitoranwendungsprozess ausgeführt werden können. Beispiel:

   ```
   rc = mon_SetSchedulingPolicy (MON_EXCLUSIVE)
   ```

   Für DB2 (DB2 Universal Database, Host- oder AS/400-Datenbank-Server) müssen Sie die Standardeinstellung MON_EXCLUSIVE verwenden. Damit wird sichergestellt, dass:

   - Der Anwendungsprozess während der Dauer der Transaktion gesperrt wird.
   - Die Anwendung als ein einziger Thread arbeitet.

   Anmerkung: Wenn Sie ODBC oder DB2 Call Level Interface verwenden, müssen Sie die Multithreading-Unterstützung inaktivieren. Sie können dazu den Konfigurationsparameter der Befehlszeilenschnittstelle folgendermaßen einstellen: `DISABLEMULTITHREAD = 1` (inaktiviert das Multithreading). Der Standardwert für DB2 Universal Database ist `DISABLEMULTITHREAD = 0` (aktiviert das Multithreading). Weitere Informationen finden Sie im Handbuch *CLI Guide and Reference*.

2. Verwenden Sie die Encina RM Registration-API, um den XA-Schalter und den logischen RM-Namen bereitzustellen, die von Encina zur Angabe des RMs in einem Anwendungsprozess verwendet werden sollen. Beispiel:

   ```
   rc = mon_RegisterRmi ( &db2xa_switch,   /* xa-Schalter */
                          "inventdb",      /* logischer RM-Name */
                          &rmiId );        /* interne RM-ID */
   ```

   Der XA-Schalter enthält die Adressen der XA-Routinen im RM, die der TM aufrufen kann. Er gibt auch die Funktionalität an, die vom RM bereitgestellt wird. Der XA-Schalter von DB2 Universal Database ist `db2xa_switch`, und er befindet sich in der DB2-Client-Bibliothek (`db2app.dll` auf Windows-Betriebssystemen und OS/2 sowie `libdb2` auf UNIX-basierten Systemen).

Von Encina wird der logische RM-Name und nicht der eigentliche Name der Datenbank verwendet, die von der SQL-Anwendung verwendet wird, die unter Encina ausgeführt wird. Der tatsächliche Name der Datenbank wird in der XA-Zeichenfolge zum Öffnen in der Encina RM Registration-API angegeben. Der logische RM-Name in diesem Beispiel stimmt mit dem Datenbanknamen überein.

Der dritte Parameter gibt eine interne Kennung zurück, die vom TM verwendet wird, um auf diese Verbindung zu verweisen.

Anmerkung: Wenn Encina für die Transaktionsverarbeitung mit DB2 über die TM-XA-Schnittstelle verwendet wird, ist zu beachten, dass verschachtelte Encina-Transaktionen von der DB2-XA-Schnittstelle zurzeit nicht unterstützt werden. Eine Verwendung dieser Transaktionen sollte nach Möglichkeit vermieden werden. Wenn dies nicht möglich ist, stellen Sie sicher, dass die SQL-Arbeit in nur einem Mitglied der Encina-Transaktionsfamilie durchgeführt wird.

Konfigurieren von BEA Tuxedo

Führen Sie die folgenden Schritte aus, um Tuxedo zur Verwendung von DB2 als Ressourcenmanager zu konfigurieren:

1. Installieren Sie Tuxedo, wie in der Dokumentation für das Produkt angegeben. Stellen Sie sicher, dass Sie die gesamte Basiskonfiguration von Tuxedo durchführen, einschließlich Protokolldateien und Umgebungsvariablen.

   Sie benötigen auch einen Compiler und das DB2 Application Development Client. Installieren Sie diese, falls erforderlich.
2. Definieren Sie auf der Tuxedo-Server-ID die Umgebungsvariable DB2INSTANCE, so dass sie auf das Exemplar verweist, das die Datenbanken enthält, die Tuxedo verwenden soll. Definieren Sie die Variable PATH, so dass sie die Programmverzeichnisse von DB2 enthält. Bestätigen Sie, dass die Tuxedo-Server-ID eine Verbindung zu den DB2-Datenbanken herstellen kann.
3. Aktualisieren Sie den Konfigurationsparameter *tp_mon_name* des Datenbankmanagers mit dem Wert `TUXEDO`.

4. Fügen Sie eine Definition für DB2 zur Tuxedo-Ressourcenmanagerdefinitionsdatei hinzu. In den folgenden Beispielen ist UDB_XA der lokal definierte Tuxedo-Ressourcenmanagername für DB2, und *db2xa_switch* ist der DB2-definierte Name für eine Struktur des Typs xa_switch_t:
   - Für AIX. Fügen Sie in der Datei ${TUXDIR}/udataobj/RM folgende Definition hinzu:

     ```
     # DB2 UDB
     UDB_XA:db2xa_switch:-L${DB2DIR} /lib -ldb2
     ```

     Dabei ist {TUXDIR} das Verzeichnis, in dem Sie Tuxedo installiert haben, und {DB2DIR} das Verzeichnis des DB2-Exemplars.
   - Für Windows NT. Fügen Sie in der Datei %TUXDIR%\udataobj\rm folgende Definition hinzu:

     ```
     # DB2 UDB
     UDB_XA;db2xa_switch;%DB2DIR%\lib\db2api.lib
     ```

     Dabei ist %TUXDIR% das Verzeichnis, in dem Sie Tuxedo installiert haben, und %DB2DIR% das Verzeichnis des DB2-Exemplars.
5. Erstellen Sie das Tuxedo-Transaktionsmonitor-Server-Programm für DB2:
   - Für AIX:

     ```
     ${TUXDIR}/bin/buildtms -r UDB_XA -o ${TUXDIR}/bin/TMS_UDB
     ```

     Dabei ist {TUXDIR} das Verzeichnis, in dem Sie Tuxedo installiert haben.
   - Für Windows NT:

     ```
     %TUXDIR%\bin\buildtms -r UDB_XA -o %TUXDIR%\bin\TMS_UDB
     ```
6. Erstellen Sie die Anwendungs-Server. In den folgenden Beispielen gibt die Option -r den Ressourcenmanagernamen, die Option -f (einmal oder mehrmals verwendet) die Dateien, die die Anwendungsservices enthalten, die Option -s die Anwendungsservicenamen für diesen Server und die Option -o den Ausgabe-Server-Dateinamen an:
   - Für AIX:

     ```
     ${TUXDIR}/bin/buildserver -r UDB_XA -f svcfile.o -s SVC1,SVC2
        -o UDBserver
     ```

     Dabei ist {TUXDIR} das Verzeichnis, in dem Sie Tuxedo installiert haben.
   - Für Windows NT:

     ```
     %TUXDIR%\bin\buildserver -r UDB_XA -f svcfile.o -s SVC1,SVC2
        -o UDBserver
     ```

     Dabei ist %TUXDIR% das Verzeichnis, in dem Sie Tuxedo installiert haben.

7. Richten Sie die Tuxedo-Konfigurationsdatei so ein, dass auf den DB2-Server verwiesen wird. Fügen Sie im Abschnitt *GROUPS der Datei UDBCONFIG einen Eintrag wie den folgenden hinzu:

```
UDB_GRP   LMID=simp GRPNO=3
  TMSNAME=TMS_UDB TMSCOUNT=2
  OPENINFO="UDB_XA:db=sample,uid=db2_benutzer,
            pwd=db2_benutzer_kwt"
```

   Dabei gibt der Parameter TMSNAME das Transaktionsmonitor-Server-Programm an, das Sie zuvor erstellt haben, und der Parameter OPENINFO gibt den Ressourcenmanagernamen an. Darauf folgen der Datenbankname und DB2-Benutzer und Kennwort, die für die Authentifizierung verwendet werden. Die Anwendungs-Server, die Sie zuvor erstellt haben, werden im Abschnitt *SERVERS der Tuxedo-Konfigurationsdatei angegeben.

8. Wenn die Anwendung auf Daten zugreift, die sich in DB2 für OS/390, DB2 für OS/400 oder DB2 für VM&VSE befinden, ist der DB2 Connect XA-Konzentrator erforderlich. Einzelangaben zur Konfiguration und zu Einschränkungen finden Sie im *DB2 Connect Benutzerhandbuch*.

9. Starten Sie Tuxedo:

```
tmboot -y
```

   Nach Beendigung des Befehls sollten Tuxedo-Nachrichten angeben, dass die Server gestartet wurden. Außerdem sollten Sie, wenn Sie den DB2-Befehl LIST APPLICATIONS ALL absetzen, zwei Verbindungen sehen, die (in diesem Fall) vom Parameter TMSCOUNT in der UDB-Gruppe in der Tuxedo-Konfigurationsdatei UDBCONFIG angegeben werden.

Konfigurieren von Microsoft Transaction Server

DB2 UDB Version 5.2 und spätere Versionen können vollständig mit Microsoft Transaction Server (MTS) Version 2.0 integriert werden. Anwendungen, die unter MTS auf 32-Bit-Windows-Betriebssystemen ausgeführt werden, können MTS verwenden, um eine zweiphasige Festschreibung mit mehreren DB2 UDB-, Host- und AS/400-Datenbank-Servern sowie mit anderen MTS-konformen Ressourcenmanagern zu koordinieren.

Anmerkung: Zusätzliche technische Informationen stehen eventuell auf der IBM Web-Site bereit, um Ihnen bei der Installation und Konfiguration der MTS-Unterstützung von DB2 zu helfen. Geben Sie die URL-Adresse http://www.ibm.com/software/data/db2/library/ an, und suchen Sie nach einem DB2 Universal Database-Artikel (“Technote”) mit dem Suchbegriff MTS.

Aktivieren der MTS-Unterstützung in DB2

Die MTS-Unterstützung wird automatisch aktiviert. Obgleich Sie den Konfigurationsparameter *tp_mon_name* des Datenbankmanagers auf `MTS` setzen können, ist dies nicht notwendig und wird ignoriert.

MTS-Softwareanforderungen

Die MTS-Unterstützung benötigt DB2 Client Application Enabler (CAE) Version 5.2 oder später, und MTS muss Version 2.0 mit Hotfix 0772 oder höheren Releases haben.

Bei der Installation des DB2-ODBC-Treibers auf 32-Bit-Windows-Betriebssysteme wird automatisch ein neues Schlüsselwort in die Registrierdatenbank eingefügt:

```
HKEY_LOCAL_MACHINE\software\ODBC\odbcinit.ini\IBM DB2 ODBC Driver:
Wert des Schlüsselworts: CPTimeout
Datentyp: REG_SZ
Wert: 60
```

Installation und Konfiguration

Der folgende Abschnitt bietet eine Zusammenfassung von Überlegungen zur Installation und Konfiguration von MTS. Zur Verwendung der MTS-Unterstützung von DB2 müssen folgende Voraussetzungen erfüllt werden:

1. Installieren Sie MTS und den DB2-Client auf der gleichen Maschine, auf der die MTS-Anwendung ausgeführt wird.
2. Gehen Sie wie folgt vor, wenn Host- oder IBM AS/400-Datenbank-Server an einer Aktualisierung auf mehreren Systemen beteiligt sein sollen:
   a. Installieren Sie DB2 Connect Enterprise Edition (EE) entweder auf Ihrer lokalen Maschine oder auf einer fernen Maschine. DB2 Connect EE ermöglicht Host- oder IBM AS/400-Datenbank-Servern an einer Aktualisierung auf mehreren Systemen teilzunehmen.
   b. Stellen Sie sicher, dass Ihr DB2 Connect EE-Server für die Aktualisierung auf mehreren Systemen aktiviert ist. Informationen zum Aktivieren von DB2 Connect für eine Aktualisierung auf mehreren Systemen finden Sie im Handbuch *DB2 Connect Enterprise Edition Einstieg* für Ihre Plattform.

Wenn Sie CLI/ODBC-Anwendungen ausführen, dürfen die Standardwerte der folgenden Konfigurationsschlüsselwörter (die in der Datei `db2cli.ini` festgelegt sind) nicht geändert werden:

- Schlüsselwort CONNECTYPE (Standardwert 1)
- Schlüsselwort MULTICONNECT (Standardwert 1)
- Schlüsselwort DISABLEMULTITHREAD (Standardwert 0)
- Schlüsselwort CONNECTIONPOOLING (Standardwert 0)
- Schlüsselwort KEEPCONNECTION (Standardwert 0)

CLI-Anwendungen, die für die Verwendung der MTS-Unterstützung geschrieben wurden, dürfen nicht die Attributwerte ändern, die den obigen Schlüsselwörtern entsprechen. Außerdem dürfen die Anwendungen nicht die Standardwerte der folgenden Attribute ändern:

- Attribut SQL_ATTR_CONNECT_TYPE (Standardwert SQL_CONCURRENT-_TRANS)
- Attribut SQL_ATTR_CONNECTON_POOLING (Standardwert SQL_C-P_OFF)

Prüfen der Installation

1. Konfigurieren Sie den DB2-Client und DB2 Connect EE für den Zugriff auf den DB2 UDB-, Host- oder IBM AS/400-Server.
2. Prüfen Sie die Verbindung von der DB2 CAE-Maschine zu den DB2 UDB-Datenbank-Servern.
3. Prüfen Sie die Verbindung von der DB2 Connect-Maschine zu Ihrem Host- oder AS/400-Datenbank-Server mit dem DB2-Befehlszeilenprozessor, und setzen Sie einige Abfragen ab.
4. Prüfen Sie die Verbindung von der DB2 CAE-Maschine über den DB2 Connect-Gateway zu Ihrem Host- oder AS/400-Datenbank-Server, und setzen Sie einige Abfragen ab.

Unterstützte DB2-Datenbank-Server

Die folgenden Server werden für die Aktualisierung auf mehreren Systemen mit Hilfe von MTS-koordinierten Transaktionen unterstützt:

- DB2 Universal Database Enterprise Edition Version 6 und spätere Versionen
- DB2 Universal Database Enterprise - Extended Edition Version 6 und spätere Versionen
- DB2 für OS/390
- DB2 für MVS
- DB2 für AS/400
- DB2 für VM & VSE

MTS-Transaktionszeitlimit und DB2-Verbindungsverhalten

Sie können den Wert für das Transaktionszeitlimit im MTS Explorer-Tool festlegen. Weitere Informationen finden Sie online im *MTS Administrator Guide*.

Wenn eine Transaktion länger dauert als das Transaktionszeitlimit (der Standardwert ist 60 Sekunden), setzt MTS asynchron eine Abbruchanforderung an alle betroffenen Ressourcenmanager ab, und die gesamte Transaktion wird abgebrochen.

Für die Verbindung zu einem DB2-Server wird die Abbruchanforderung in eine DB2-Anforderung ROLLBACK übersetzt. Wie alle anderen Datenbankanforderungen wird die ROLLBACK-Anforderung auf der Verbindung serialisiert, um die Integrität der Daten auf dem Datenbank-Server zu gewährleisten.

Ergebnis:

- Wenn sich die Verbindung im Leerlauf befindet, wird die ROLLBACK-Operation sofort ausgeführt.
- Wenn eine SQL-Anweisung mit langer Ausführungsdauer verarbeitet wird, wartet die ROLLBACK-Anforderung, bis die SQL-Anweisung beendet ist.

Verbindungspool

Der Verbindungspool erlaubt es einer Anwendung, eine Verbindung aus einem Pool von Verbindungen zu verwenden, so dass die Verbindung nicht für jede Verwendung erneut erstellt werden muss. Nachdem eine Verbindung erstellt und einem Pool hinzugefügt wurde, kann eine Anwendung diese Verbindung erneut verwenden, ohne einen vollständigen Verbindungsprozess ausführen zu müssen. Die Verbindung wird dem Pool hinzugefügt, wenn die Anwendung die Verbindung zur ODBC-Datenquelle trennt, und einer neuen Verbindung übergeben, deren Attribute identisch sind.

Der Verbindungspool war eine Funkion des ODBC-Treibermanagers 2.x. Beim neuesten ODBC-Treibermanager (Version 3.5), der mit MTS geliefert wurde, gibt es für den Verbindungspool einige Konfigurationsänderungen und eine neue Funktionsweise für ODBC-Verbindungen von MTS COM-Objekten für Transaktionen (siehe „Wiederverwenden von ODBC-Verbindungen zwischen COM-Objekten, die an der gleichen Transaktion teilnehmen" auf Seite 204).

Der ODBC-Treibermanager 3.5 verlangt, dass der ODBC-Treiber ein neues Schlüsselwort in der Registrierdatenbank registriert, damit der Verbindungspool aktiviert werden kann. Das Schlüsselwort ist:

```
Schlüsselwort: SOFTWARE\ODBC\ODBCINST.INI\IBM DB2 ODBC DRIVER
Name: CPTimeout
Typ: REG_SZ
Wert: 60
```

Der DB2-ODBC-Treiber ab Version 6 für das 32-Bit-Betriebssystem Windows unterstützt vollständig den Verbindungspool. Daher wird dieses Schlüsselwort registriert. Clients unter Version 5.2 müssen das FixPak 3 (WR09024) oder höher installieren.

Der Standardwert 60 bedeutet, dass eine Verbindung für 60 Sekunden im Pool behalten wird, bevor sie getrennt wird.

In einer Umgebung mit hoher Auslastung ist es besser, den Wert CPTimeout möglichst hoch zu setzen (Microsoft schlägt für bestimmte Umgebungen 10 Minuten vor), um zu häufiges Herstellen und Trennen physischer Verbindungen zu verhindern, weil diese große Mengen an Systemressourcen, einschließlich Systemspeicher- und Übertragungsstapelressourcen, beanspruchen.

Darüber hinaus müssen Sie auf einer Mehrprozessormaschine die Unterstützung für mehrere Pools pro Prozessor inaktivieren, um sicherzustellen, dass dieselbe Verbindung zwischen Objekten in der gleichen Transaktion verwendet wird. Dazu kopieren Sie die folgende Registrierdatenbankeinstellung in eine Datei namens `odbcpool.reg`, sichern diese Datei als Textdatei und setzen den Befehl **odbcpool.reg** ab. Das Windows-Betriebssystem importiert daraufhin diese Registrierdatenbankeinstellung.

```
REGEDIT4

[HKEY_LOCAL_MACHINE\SOFTWARE\ODBC\ODBCINST.INI\ODBC Connection Pooling]
"NumberOfPools"="1"
```

Wenn dieses Schlüsselwort nicht auf den Wert 1 gesetzt ist, kann MTS Verbindungen in verschiedenen Pools anlegen, so dass nicht dieselbe Verbindung wiederverwendet wird.

MTS-Verbindungspoolfunktion mit ADO 2.1 und späteren Versionen

Wenn die MTS-COM-Objekte mit Hilfe von ADO auf die Datenbank zugreifen, muss die Poolfunktion für OLEDB-Ressourcen inaktiviert werden, so dass der Microsoft-OLEDB-Anbieter für ODBC (MSDASQL) keinen störenden Einfluss auf die ODBC-Verbindungspoolfunktion hat. Diese Einrichtung wird mit dem Wert `OFF` in ADO 2.0, in ADO 2.1 jedoch mit dem Wert `ON` initialisiert. Zum Ausschalten der OLEDB-Ressourcenpoolfunktion kopieren Sie die folgenden Zeilen in eine Datei namens `oledb.reg`, sichern die Datei als Textdatei und setzen den Befehl **oledb.reg** ab. Das Windows-Betriebssystem importiert daraufhin diese Registrierdatenbankeinstellungen.

```
REGEDIT4

[HKEY_CLASSES_ROOT\CLSID\{c8b522cb-5cf3-11ce-ade5-00aa0044773d}]
@="MSDASQL"
"OLEDB_SERVICES"=dword:fffffffc
```

Wiederverwenden von ODBC-Verbindungen zwischen COM-Objekten, die an der gleichen Transaktion teilnehmen

Für ODBC-Verbindungen in MTS COM-Objekten ist der Verbindungspool automatisch eingeschaltet (unabhängig davon, ob es sich um ein COM-Objekt für Transaktionen handelt).

Wenn mehrere MTS COM-Objekte an der gleichen Transaktion teilnehmen, kann die Verbindung zwischen zwei oder mehr COM-Objekten auf folgende Weise wiederverwendet werden.

Angenommen, es gibt zwei COM-Objekte, COM1 und COM2, die eine Verbindung zur gleichen ODBC-Datenquelle herstellen und an der gleichen Transaktion teilnehmen.

Nachdem COM1 die Verbindung hergestellt und seine Arbeit ausgeführt hat, trennt es die Verbindung, und die Verbindung wird im Pool behalten. Diese Verbindung wird jedoch für die Verwendung durch andere COM-Objekte der gleichen Transaktion reserviert. Sie ist für andere Transaktionen erst dann verfügbar, wenn die aktuelle Transaktion beendet ist.

Wenn COM2 in der gleichen Transaktion aufgerufen wird, erhält es die gleiche Verbindung aus dem Pool. MTS stellt sicher, dass die Verbindung nur die COM-Objekte erhalten, die an der gleichen Transaktion teilnehmen.

Wenn andererseits COM1 die Verbindung nicht explizit trennt, belegt es die Verbindung, bis die Transaktion beendet ist. Wenn COM2 in der gleichen Transaktion aufgerufen wird, wird eine separate Verbindung hergestellt. Als Folge belegt diese Transaktion zwei Verbindungen statt einer.

Die Funktion zur Wiederverwendung von Verbindungen für COM-Objekte, die an der gleichen Transaktion teilnehmen, ist aus folgenden Gründen vorteilhaft:

- Sie verwendet weniger Ressourcen sowohl beim Client als auch beim Server. Es wird nur eine Verbindung benötigt.
- Durch sie wird die Möglichkeit ausgeschaltet, dass zwei Verbindungen, die an der gleichen Transaktion teilnehmen (auf den gleichen Datenbank-Server und auf die gleichen Daten zugreifen), sich gegenseitig sperren, weil DB2-Server verschiedene Verbindungen von MTS COM-Objekten als separate Transaktionen behandeln.

Optimieren von TCP/IP-Übertragungen

Wenn ein kleiner CPTimeout-Wert in einer Umgebung mit hoher Auslastung verwendet wird, in der zu viele physische Verbindungen zur gleichen Zeit hergestellt und getrennt werden, kann es beim TCP/IP-Stapel zu Ressourcenknappheit kommen.

Zur Milderung dieses Problems sollten Sie die TCP/IP-Registrierungseinträge verwenden. Diese werden im Handbuch *Windows NT Resource Guide*, Volume 1 beschrieben. Die Registrierungsschlüsselwerte befinden sich in HKEY_LOCAL_MACHINE—> SYSTEM—> CurrentControlSet—> Services—> TCPIP—> Parameters.

Die Standardwerte und die vorgeschlagenen Einstellungen sind:

Name	Standardwert	Vorgeschlagener Wert
KeepAlive time	7200000 (2 Stunden)	Derselbe
KeepAlive interval	1000 (1 Sekunde)	10000 (10 Sekunden)
TcpKeepCnt	120 (2 Minuten)	240 (4 Minuten)
TcpKeepTries	20 (20 Wiederholungen)	Derselbe
TcpMaxConnectAttempts	3	6
TcpMaxConnectRetransmission	3	6
TcpMaxDataRetransmission	5	8
TcpMaxRetransmissionAttempts	7	10
Wenn der Registrierungswert nicht definiert ist, erstellen Sie ihn.		

Testen von DB2 mit der MTS-Beispielanwendung ″BANK″

Sie können das mit MTS gelieferte Beispielprogramm ″BANK″ verwenden, um die Konfiguration der Client-Produkte und von MTS zu testen.

Führen Sie die folgenden Schritte aus:

1. Ändern Sie die Datei `\Program Files\Common Files\ODBC\Data Sources\MTSSamples.dsn`, so dass sie folgendermaßen aussieht:

```
[ODBC]
DRIVER=IBM DB2 ODBC DRIVER
UID=ihre_benutzer_id
PWD=ihr_kennwort
DSN=ihr_datenbank_aliasname
Description=MTS Samples
```

Dabei gilt folgendes:

- *ihre_benutzer_id* und *ihr_kennwort* sind die Benutzer-ID und das Kennwort für die Verbindung zum Host.
- *ihr_datenbank_aliasname* ist der Aliasname für die Datenbank, der für die Verbindung zum Datenbank-Server verwendet wird.

2. Rufen Sie ODBC-Administrator in der **Systemsteuerung** auf, wählen Sie die Registerkarte **System-DSN** aus und fügen Sie die Datenquelle hinzu:
   a. Wählen Sie IBM ODBC-Driver und anschließend **Fertigstellen** aus.
   b. Wenn die Liste der Aliasnamen für die Datenbank angezeigt wird, wählen Sie den Namen aus, der zuvor angegeben wurde.
   c. Wählen Sie **OK** aus.
3. Stellen Sie mit Hilfe des DB2-Befehlszeilenprozessors unter der obigen ID *ihre_benutzer_id* eine Verbindung zu einer DB2-Datenbank her.
   a. Binden Sie die Datei `db2cli.lst`:

      ```
      db2 bind @C:\sqllib\bnd\db2cli.lst blocking all grant public
      ```

   b. Binden Sie die Dienstprogramme.

      Wenn der Server ein DRDA-Host-Server ist, binden Sie die Datei `ddcsmvs.lst`, `ddcs400.lst` oder `ddcsvm.lst`, je nach Host, zu dem Sie eine Verbindung herstellen (OS/390, AS/400 oder VSE&VM). Beispiel:

      ```
      db2 bind @C:\sqllib\bnd\@ddcsmvs.lst blocking all grant public
      ```

      Binden Sie andernfalls die Datei `db2ubind.lst`:

      ```
      db2 bind @C:\sqllib\bnd\@db2ubind.lst blocking all grant public
      ```

   c. Erstellen Sie wie folgt die Beispieltabelle und -daten für die MTS-Beispielanwendung:

      ```
      db2 create table account (accountno int, balance int)
      db2 insert into account values(1, 1)
      ```

4. Stellen Sie auf dem DB2-Client sicher, dass der Konfigurationsparameter *tp_mon_name* des Datenbankmanagers auf `MTS` gesetzt ist.
5. Führen Sie die Anwendung "BANK" aus. Wählen Sie den Druckknopf **Account** und die Option **Visual C++** aus, und übergeben Sie dann die Anforderung. Andere Optionen verwenden möglicherweise SQL Server-spezifisches SQL und funktionieren vielleicht nicht.

Kapitel 11. Einführung in die Unterstützung der hohen Verfügbarkeit und der Funktionsübernahme

Erfolgreiche e-business-Unternehmen hängen von der ständigen Verfügbarkeit von Transaktionsverarbeitungssystemen ab, welche ihrerseits von Datenbankverwaltungssystemen gesteuert werden, z. B. von DB2. Diese Systeme müssen rund um die Uhr und an allen Wochentagen verfügbar sein („24 x 7").

Hohe Verfügbarkeit

Mit *hoher Verfügbarkeit* (HA - High Availability) wird ausgedrückt, dass Systeme aus der Sicht von Kunden praktisch immer aktiv und verfügbar sind. Dafür müssen die folgenden Voraussetzungen erfüllt sein:

- Transaktionen müssen effizient verarbeitet werden, ohne dass sich die Leistung in Zeiten von Spitzenbelastungen nennenswert vermindert (oder es gar zu einem völligen Leistungsausfall kommt). In einer Umgebung mit partitionierten Datenbanken kann DB2 sowohl die partitionsinterne Parallelität als auch die partitionsübergreifende Parallelität nutzen, um Transaktionen effizient zu verarbeiten. Die *partitionsinterne Parallelität* kann in einer SMP-Umgebung eingesetzt werden, um verschiedene Komponenten einer komplexen SQL-Anweisung gleichzeitig zu verarbeiten. Die *partitionsübergreifende Parallelität* in einer Umgebung mit partitionierten Datenbanken bedeutet dagegen, dass eine Abfrage auf allen beteiligten Knoten gleichzeitig verarbeitet wird, wobei jeder Knoten eine Untermenge von Tabellenzeilen verarbeitet. Weitere Informationen zur Parallelverarbeitung finden Sie im Handbuch „Arten der Parallelität" auf Seite 42.
- Die Systeme müssen nach einem Hardware- oder Softwarefehler oder einem Störfall schnell wiederherstellbar sein. DB2 verfügt über ein erweitertes fortlaufendes Prüfpunktsystem und eine parallele Wiederherstellungsfunktion, die eine extrem schnelle Wiederherstellung nach einem Systemabsturz ermöglichen.

  Die schnelle Wiederherstellung kann auch vom Vorhandensein einer getesteten Sicherungs- und Wiederherstellungsstrategie abhängen.
  Weitere Informationen zu Wiederherstellungsstrategien finden Sie in *Datenwiederherstellung und hohe Verfügbarkeit Handbuch und Referenz*.
- Software für die Unternehmensdatenbanken muss für die Transaktionsverarbeitung ständig aktiv und verfügbar sein. Dazu muss bei einem Ausfall des Datenbankmanagers ein anderer Datenbankmanager dessen Aufgaben übernehmen können. Dies wird als Funktionsübernahme bezeichnet.

Die *Funktionsübernahme* ermöglicht bei einem Hardwarefehler eine automatische Übertragung der Arbeitsbelastung von einem System auf ein anderes System.

Zur Funktionsübernahme muss auf einer anderen Maschine eine Datenbankkopie gepflegt werden, die ständig die Protokolldateien aktualisierend wiederherstellt. Die *Protokollübertragung* ist das Kopieren ganzer Protokolldateien auf eine Bereitschaftsmaschine, entweder von einer Archivierungseinheit aus oder mit Hilfe eines Benutzer-Exit-Programms, das für die primäre Datenbank ausgeführt wird. Mit dieser Methode wird die primäre Datenbank auf der Bereitschaftsmaschine wiederhergestellt, wobei das DB2-Wiederherstellungsdienstprogramm oder die Funktion zur Erstellung einer Spiegeldatenbank verwendet wird. Sie können die neue ausgesetzte E/A-Unterstützung verwenden, um die neue Datenbank schnell zu initialisieren (siehe „Hohe Verfügbarkeit durch Unterstützung der ausgesetzten E/A und der Onlineteilung einer Spiegeldatenbank" auf Seite 210). Die sekundäre Datenbank auf der Bereitschaftsmaschine stellt die Protokolldateien ständig aktualisierend wieder her. Wenn die primäre Datenbank ausfällt, werden alle übrigen Protokolldateien auf die Bereitschaftsmaschine kopiert. Nach einer aktualisierenden Wiederherstellung bis zum Ende der Protokolle und bis zur Stoppoperation werden alle Clients mit der sekundären Datenbank auf der Bereitschaftsmaschine verbunden.

Die Unterstützung der Funktionsübernahme erhalten Sie auch über plattformspezifische Software, die Sie auf dem System installieren können. Zum Beispiel:

- High Availability Cluster Multi-Processing, Enhanced Scalability für AIX

  Weitere Informationen zu HACMP/ES finden Sie in *Datenwiederherstellung und hohe Verfügbarkeit Handbuch und Referenz* oder im White Paper mit dem Titel „IBM DB2 Universal Database Enterprise Edition for AIX and HACMP/ES", das auf der Website „DB2 UDB and DB2 Connect Online Support" unter http://www.ibm.com/software/data/pubs/papers/ verfügbar ist.
- Microsoft Cluster Server für Windows NT oder Windows 2000

  Weitere Informationen zu MSCS finden Sie in *Datenwiederherstellung und hohe Verfügbarkeit Handbuch und Referenz*.
- Sun Cluster oder VERITAS Cluster Server für die Solaris-Betriebsumgebung

  Weitere Informationen zu Sun Cluster 2.x finden Sie in *Datenwiederherstellung und hohe Verfügbarkeit Handbuch und Referenz*; Informationen zu Sun Cluster 3.0 finden Sie im White Paper mit dem Titel „DB2 and High Availability on Sun Cluster 3.0", das auf der Website „DB2 UDB and DB2 Connect Online Support" unter http://www.ibm.com/software/data/pubs/papers/ verfügbar ist.

Weitere Informationen zu VERITAS Cluster Server finden Sie im White Paper mit dem Titel „DB2 and High Availability on VERITAS Cluster Server", das ebenfalls auf der Website „DB2 UDB and DB2 Connect Online Support" verfügbar ist.

- Multi-Computer/ServiceGuard für Hewlett-Packard

  Weitere Informationen zu HP MC/ServiceGuard finden Sie im White Paper mit dem Titel „IBM DB2 EE v.7.1 Implementation and Certification With Hewlett-Packard's MC/ServiceGuard High Availability Software", das auf der Website „DB2 UDB and DB2 Connect Online Support" unter http://www.ibm.com/software/data/pubs/papers/ verfügbar ist.

Strategien zur Funktionsübernahme basieren normalerweise auf Clustern aus mehreren Systemen. Ein *Cluster* ist eine Gruppe verbundener Systeme, die nach außen hin wie ein einzelnes System zusammenarbeitet. Die einzelnen Prozessoren werden innerhalb des Clusters als Knoten bezeichnet. Durch das Clustering können Server sich gegenseitig gegen Ausfälle schützen, indem sie die Arbeitsbelastung des ausgefallenen Servers übernehmen.

Die IP-Adressübernahme (IP-Übernahme) ist die Fähigkeit, bei einem Ausfall eines Servers eine Server-IP-Adresse von einer Maschine auf eine andere zu übertragen. Aus der Sicht der Client-Anwendung erscheinen die Maschinen zu unterschiedlichen Zeiten wie ein und derselbe Server.

In der Software für Funktionsübernahme können *Überwachungssignale* oder *Keepalive-Pakete* zwischen Systemen verwendet werden, um die Verfügbarkeit zu bestätigen. Überwachungssignale bedeuten, dass Systemservices zwischen allen Knoten eines Clusters ständig kommunizieren. Wenn kein Überwachungssignal erkannt wird, beginnt die Funktionsübernahme durch ein Ausweichsystem. Endbenutzer bemerken gewöhnlich nicht, dass ein System ausgefallen ist.

Die zwei häufigsten Strategien für Funktionsübernahme auf dem Markt sind der *Bereitschaftsmodus (Idle Standby)* und der *Modus der gegenseitigen Übernahme (Mutual Takeover)*, obwohl die Konfigurationen, die zu diesen Modi gehören, je nach Lieferant auch anders bezeichnet werden können:

Bereitschaftsmodus (Idle Standby)
: Bei dieser Konfiguration führt ein System ein DB2-Exemplar aus; das zweite System befindet sich „im Bereitschaftsmodus" (Idle Standby) und ist bereit, das Exemplar zu übernehmen, falls das Betriebssystem oder die Hardware für das erste System ausfällt. Auf die Gesamtleitung des Systems wirkt sich dies nicht aus, da das Ausweichsystem erst bei Bedarf eingesetzt wird.

Gegenseitige Übernahme (Mutual Takeover)
Bei dieser Konfiguration ist jedes System als Ausweichsystem für ein anderes zugeordnet. Dies kann sich auf die Gesamtleistung des Systems auswirken, da das Ausweichsystem nach einer Funktionsübernahme zusätzliche Arbeit leisten muss: Es muss die eigenen Aufgaben erfüllen und zusätzlich die Aufgaben des ausgefallenen Systems.

Mit Strategien zur Funktionsübernahme können die Funktionen eines Exemplars, einer Partition oder mehrerer logischer Knoten übernommen werden.

Hohe Verfügbarkeit durch Unterstützung der ausgesetzten E/A und der Onlineteilung einer Spiegeldatenbank

Ausgesetzte E/A unterstützt die ständige Systemverfügbarkeit, indem für die Bearbeitung der Onlineteilung von Spiegeldatenbank eine vollständige Implementierung, d. h. das Teilen einer Spiegeldatenbank ohne Herunterfahren des Systems, bereitgestellt wird. Eine *geteilte Spiegeldatenbank* ist eine „Momentaufnahme" der Datenbank, die erstellt wird, indem die Festplatten mit den Daten gespiegelt werden; wenn eine Kopie benötigt wird, wird die Spiegeldatenbank geteilt. *Plattenspiegelung* ist das Schreiben aller Daten auf zwei separate Festplatten, wobei eine Platte das Spiegelbild der anderen ist. *Teilen einer Spiegeldatenbank* bedeutet, eine Sicherungskopie der Spiegeldatenbank zu erstellen.

Wenn Sie vermeiden möchten, eine große Datenbank mit dem DB2-Sicherungsdienstprogramm zu sichern, können Sie mit der Funktion für ausgesetzte E/A und der Funktion zur Erstellung einer Spiegeldatenbank von einem gespiegelten Image Kopien erstellen. Außerdem erreichen Sie mit dieser Methode Folgendes:

- Sie vermeiden hohen Sicherungsaufwand auf der Produktionsmaschine.
- Sie verfügen über eine schnelle Methode, Systeme zu klonen.
- Sie verfügen über eine schnelle Implementierung der Funktionsübernahme aus dem Bereitschaftsmodus. Eine einleitende Wiederherstellung findet nicht statt, und sollte eine aktualisierende Wiederherstellung sich als zu langsam erweisen oder sollten Fehler auftreten, ist die erneute Initialisierung sehr schnell.

Mit dem Befehl **db2inidb** wird die geteilte Spiegeldatenbank initialisiert, so dass Sie sie wie folgt einsetzen können:

- Zur Erstellung eines Datenbankklons

  Einen Lesezugriffsklon der primären Datenbank können Sie z. B. verwenden, um Berichte zu erstellen.
- Als Bereitschaftsdatenbank
- Als Sicherungsimage

Diesen Befehl können Sie nur für die getrennte Spiegeldatenbank absetzen, und auf dieser Datenbank müssen Sie vor ihrem Einsatz zuerst das Programm **db2inidb** ausführen (siehe *Datenwiederherstellung und hohe Verfügbarkeit Handbuch und Referenz*).

In einer Umgebung mit partitionierten Datenbanken müssen Sie den Befehl **db2inidb** auf jeder Partition ausführen, bevor Sie das getrennte Image einer beliebigen Partition verwenden können. Das Tool können Sie auf allen Partitionen gleichzeitig ausführen.

Erstellen eines Datenbankklons

Ein Klon einer Datenbank kann eine „Offlinesicherung" der primären Datenbank (Live-Datenbank) sein. Sie können jedoch die geklonte Datenbank nicht sichern, noch ihr Image auf dem Originalsystem wiederherstellen oder es mit Hilfe der Protokolldateien, die auf dem Originalsystem generiert wurden, aktualisierend wiederherstellen.

Gehen Sie wie folgt vor, um eine Datenbank zu klonen:

1. Setzen Sie für die primäre Datenbank die E/A aus:
   ```
   db2 set write suspend for database
   ```
2. Verwenden Sie geeignete Befehle auf Betriebssystemebene, um die Spiegeldatenbank(en) aus der primären Datenbank zu erstellen.
3. Nehmen Sie den E/A-Betrieb auf der primären Datenbank wieder auf:
   ```
   db2 set write resume for database
   ```
4. Stellen Sie von einer anderen Maschine aus die Verbindung zur gespiegelten Datenbank her.
5. Starten Sie das Datenbankexemplar:
   ```
   db2start
   ```
6. Initialisieren Sie die gespiegelte Datenbank als Klon der primären Datenbank:
   ```
   db2inidb aliasname-der-datenbank as snapshot
   ```

   Anmerkung: Mit diesem Befehl werden Transaktionen rückgängig gemacht, die während des Teilens gerade verarbeitet werden.

Verwenden der geteilten Spiegeldatenbank als Bereitschaftsdatenbank

Während die gespiegelte Datenbank (Bereitschaftsdatenbank) ständig anhand der Protokolle aktualisierend wiederhergestellt wird, werden neue Protokolle, die gerade von der primären Datenbank erstellt werden, ständig aus dem primären System abgerufen. Gehen Sie wie folgt vor, um die geteilte Spiegeldatenbank als Bereitschaftsdatenbank zu verwenden:

1. Setzen Sie für die primäre Datenbank die E/A aus:

   ```
   db2 set write suspend for database
   ```

2. Verwenden Sie geeignete Befehle auf Betriebssystemebene, um die Spiegeldatenbank(en) aus der primären Datenbank zu erstellen.
3. Nehmen Sie den E/A-Betrieb auf der primären Datenbank wieder auf:

   ```
   db2 set write resume for database
   ```

4. Stellen Sie die Verbindung zwischen der gespiegelten Datenbank und einem anderen Exemplar her.
5. Versetzen Sie die gespiegelte Datenbank in den Status *Aktualisierende Wiederherstellung anstehend*:

   ```
   db2inidb aliasname-der-datenbank as standby
   ```

   Wenn Sie nur über DMS-Tabellenbereiche (vom Datenbankmanager verwaltete Tabellenbereiche) verfügen, können Sie eine vollständige Datenbanksicherung erstellen, damit der Systemaufwand für eine Sicherung auf der Produktionsdatenbank entfällt.
6. Definieren Sie ein Benutzer-Exit-Programm, um die Protokolldateien vom primären System abzurufen.
7. Stellen Sie die Datenbank bis zum Ende der Protokolle aktualisierend wieder her.
8. Rufen Sie Protokolldateien ab, und stellen Sie die Datenbank bis zum Ende der Protokolle aktualisierend wieder her, bis die primäre Datenbank ausfällt.

Verwenden der geteilten Spiegeldatenbank als Sicherungsimage

Gehen Sie wie folgt vor, um die geteilte Spiegeldatenbank als „Sicherungsimage" zu verwenden:

1. Setzen Sie für die primäre Datenbank die E/A aus:

   ```
   db2 set write suspend for database
   ```

2. Verwenden Sie geeignete Befehle auf Betriebssystemebene, um die Spiegeldatenbank(en) aus der primären Datenbank zu erstellen.
3. Nehmen Sie den E/A-Betrieb auf der primären Datenbank wieder auf:

   ```
   db2 set write resume for database
   ```

4. Ein Fehler tritt auf dem primären System auf und macht eine Wiederherstellung von einer Sicherung erforderlich.
5. Verwenden Sie Befehle auf Betriebssystemebene, um die abgeteilten Daten auf das primäre System zu kopieren. Kopieren Sie nicht die abgeteilten Protokolle, da die primären Protokolle für die aktualisierende Wiederherstellung benötigt werden.
6. Starten Sie das primäre Datenbankexemplar:

   ```
   db2start
   ```

7. Initialisieren Sie die primäre Datenbank:

   db2inidb *aliasname-der-datenbank* as mirror

8. Stellen Sie die primäre Datenbank bis zum Ende der Protokolle aktualisierend wieder her.

Teil 4. Anhänge und Schlussteil

Anhang A. Namenskonventionen

Lesen Sie den Abschnitt, der die Namenskonventionen beschreibt, zu denen Sie weitere Informationen wünschen:

- „Allgemeine Namenskonventionen"
- „Namenskonventionen für Objektnamen" auf Seite 218
- „Beibehaltung von Werten in Groß-/Kleinschreibung in einem System zusammengeschlossener Datenbanken" auf Seite 222

Allgemeine Namenskonventionen

Soweit nicht anders angegeben, können alle Namen die folgenden Zeichen enthalten:

- A bis Z. Bei den meisten Namen werden die Zeichen A bis Z von Klein- in Großschreibung umgesetzt.
- 0 - 9
- @, #, $ und _ (Unterstreichungszeichen)

Namen dürfen nicht mit einer Ziffer oder einem Unterstreichungszeichen beginnen.

Verwenden Sie zur Benennung von Tabellen, Sichten, Spalten, Indizes und Berechtigungs-IDs keine für SQL reservierten Wörter. Eine Liste der für SQL reservierten Wörter finden Sie im Handbuch *SQL Reference*.

Es gibt weitere Sonderzeichen, die abhängig vom Betriebssystem und dem Einsatzort von DB2 möglicherweise verwendet werden können. Es gibt jedoch keine Garantie für die Verwendung dieser Zeichen. Es wird nicht empfohlen, diese weiteren Sonderzeichen bei der Benennung von Objekten in der Datenbank zu verwenden.

Namenskonventionen für Objektnamen

Für alle Objekte gelten die allgemeinen Namenskonventionen. Zusätzlich gelten für einige Objekte weitere Einschränkungen, die unten erläutert werden.

Tabelle 24. Namenskonventionen für Datenbanknamen, Aliasnamen der Datenbank und Exemplarnamen

Objekte	Richtlinien
• Datenbanken • Aliasnamen der Datenbank • Exemplare	• Datenbanknamen müssen innerhalb der Position, in der sie katalogisiert werden, eindeutig sein. In auf UNIX basierenden Implementierungen von DB2 ist diese Position ein Verzeichnispfad, während in Windows-Implementierungen diese Position ein Laufwerkbuchstabe ist. • Aliasnamen der Datenbank müssen innerhalb des Systemdatenbankverzeichnisses eindeutig sein. Wenn eine neue Datenbank erstellt wird, ist der Standardwert des Aliasnamens der Datenbankname. Daher können Sie keine Datenbank mit einem Namen erstellen, der bereits als Datenbankaliasname verwendet wird, selbst wenn es keine andere Datenbank mit diesem Namen gibt. • Datenbanknamen, Aliasnamen der Datenbank und Exemplarnamen können bis zu 8 Byte umfassen. • Bei Sytemen mit Windows NT und Windows 2000 darf kein Exemplar einen Servicenamen tragen. **Anmerkung:** Zur Vermeidung von Problemen sollten Sie die Sonderzeichen @, # und $ nicht in einem Datenbanknamen benutzen, wenn Sie beabsichtigen, die Datenbank in einer Übertragungsumgebung zu verwenden. Darüber hinaus sollten Sie diese Zeichen nicht benutzen, wenn Sie die Datenbank in einer anderen Sprache verwenden möchten, weil diese Zeichen nicht auf allen Tastaturen in gleicher Weise verfügbar sind.

Tabelle 25. Namenskonventionen für Datenbankobjekte

Objekte	Richtlinien
• Aliasnamen • Pufferpools • Spalten • Ereignismonitore • Indizes • Methoden • Knotengruppen • Schemata • Gespeicherte Prozeduren • Tabellen • Tabellenbereiche • Auslöser • Benutzerdefinierte Funktionen (UDFs) • Benutzerdefinierte Datentypen (UDTs) • Sichten	Können bis zu 18 Byte umfassen. Es gibt folgende *Ausnahmen*: • Tabellennamen (einschließlich Namen von Sichten, Übersichtstabellen, Aliasnamen und Korrelationsnamen), die bis zu 128 Byte umfassen können. • Spaltennamen, die bis zu 30 Byte umfassen können. • Schemennamen, die bis zu 30 Byte umfassen können. • Objektnamen können ferner folgende Zeichen enthalten: – Gültige diakritische Zeichen (zum Beispiel ö) – Mehrbytezeichen außer Mehrbyteleerzeichen (für Mehrbyteumgebungen)

Zusätzliche Informationen zu Schemennamen

- Tabelle mit Schemennamen, die länger als 18 Byte sind, können nicht repliziert werden.
- Benutzerdefinierte Datentypen (UDTs) dürfen keine Schemennamen aufweisen, die länger als 8 Byte sind.
- Die folgenden Schemennamen sind reservierte Wörter und dürfen nicht verwendet werden: SYSCAT, SYSFUN, SYSIBM, SYSSTAT.
- Um zukünftigen Migrationsproblemen vorzubeugen, sollten Sie keine Schemanamen verwenden, die mit der Zeichenfolge SYS beginnen. Der Datenbankmanager lässt die Erstellung von Auslösern, benutzerdefinierten Datentypen bzw. benutzerdefinierten Funktionen mit Schemennamen, die mit SYS beginnen, nicht zu.
- Es wird empfohlen, das Wort SESSION nicht als Schemennamen zu verwenden. Deklarierte temporäre Tabellen müssen mit SESSION qualifiziert werden. Daher ist es möglich, dass eine Anwendung eine temporäre Datei mit einem Namen deklariert, der mit dem einer permanenten Tabelle identisch ist. In einem solchen Fall kann die Anwendungslogik übermäßig kompliziert werden. Außer im Zusammenhang mit deklarierten temporären Tabellen sollte die Verwendung des Schemennamens SESSION also vermieden werden.

Tabelle 26. Namenskonventionen für Benutzername, Benutzer-ID und Gruppenname

Objekte	Richtlinien
• Gruppennamen • Benutzernamen • Benutzer-IDs	• Gruppennamen können bis zu 8 Byte umfassen. • Benutzer-IDs in UNIX-basierten Systemen können bis zu 8 Zeichen enthalten. • Benutzernamen unter Windows können bis zu 30 Zeichen enthalten. Windows NT und Windows 2000 haben momentan praktisch eine Begrenzung auf 20 Zeichen. • Bei der Verwendung von DCE-Authentifizierung sind Benutzernamen auf 8 Zeichen begrenzt. • Wenn weder DCE- noch Client-Authentizierung verwendet wird, werden Benutzernamen mit mehr als 8 Zeichen auf nicht-Windows 32-Bit-Clients, die mit Windows NT oder Windows 2000 verbunden sind, unterstützt, wenn Benutzername und Kennwort explizit angegeben werden. • Namen und IDs dürfen nicht: – USERS, ADMINS, GUESTS, PUBLIC, LOCAL oder wie eines der für SQL reservierten Wörter lauten, die im Handbuch *SQL Reference* aufgeführt sind. – mit IBM, SQL oder SYS beginnen. – diakritische Zeichen enthalten. In auf UNIX basierenden Systemen können Gruppen und Benutzer denselben Namen haben. Bei der Anweisung GRANT müssen Sie angeben, ob eine Gruppe oder ein Benutzer gemeint ist. Bei der Anweisung REVOKE hängt die Angabe eines Benutzers oder einer Gruppe davon ab, ob für GRANTEE mehrere Zeilen in den Tabellen des Berechtigungskatalogs mit verschiedenen Werten für GRANTEETYPE enthalten sind. Unter Windows NT dürfen lokale und globale Gruppen und Benutzer nicht dieselben Namen haben. Unter OS/2 dürfen Gruppen und Benutzer nicht dieselben Namen haben. **Anmerkungen:** 1. Bei einigen Betriebssystemen sind Benutzer-IDs und Kennwörter mit Unterscheidung von Groß-/Kleinschreibung möglich. Prüfen Sie die Dokumentation zu Ihrem Betriebssystem, um festzustellen, ob dies auf Ihr Betriebssystem zutrifft. 2. Die Berechtigungs-ID, die nach der erfolgreichen Ausführung von CONNECT oder ATTACH zurückgegeben wird, wird nach 8 Zeichen abgeschnitten. Eine Auslassung (...) wird der Berechtigungs-ID angehängt und die Felder SQLWARN enthalten Warnungen, die auf das Abschneiden hinweisen. Weitere Informationen finden Sie im Abschnitt zur Anweisung CONNECT in *SQL Reference*.

Zusätzliche Informationen zu Kennwörtern

Sie müssen möglicherweise Aufgaben zur Kennwortverwaltung ausführen. Da solche Aufgaben auf dem Server auszuführen sind und viele Benutzer mit der Arbeit in der Server-Umgebung nicht vertraut sind oder sich dabei nicht wohl fühlen, kann diese Tätigkeit eine große Herausforderung bedeuten. DB2 UDB stellt eine Möglichkeit zur Aktualisierung und Überprüfung von Kennwörtern ohne Arbeiten auf dem Server zur Verfügung. Zum Beispiel unterstützt DB2 für OS/390 Version 5 diese Methode, um das Kennwort eines Benutzers zu ändern. Wenn eine Fehlernachricht SQL1404N „Kennwort abgelaufen" empfangen wird, ändern Sie mit der Anweisung CONNECT das Kennwort auf folgende Weise:

```
CONNECT TO <datenbank> USER <benutzer-id> USING <kennwort>
   NEW <neues_kennwort> CONFIRM <neues_kennwort>
```

Das Dialogfenster „Kennwort ändern" der DB2-Client-Konfiguration - Unterstützung kann ebenfalls zur Änderung des Kennworts verwendet werden. Weitere Informationen zu diesen Methoden zur Änderung des Kennworts finden Sie im Handbuch *SQL Reference* und in der Online-Hilfefunktion der DB2-Client-Konfiguration - Unterstützung.

Tabelle 27. Namenskonventionen für Objekte in zusammengeschlossenen Datenbanken

Objekte	Richtlinien
• Funktionszuordnungen • Indexspezifikationen • Kurznamen • Server • Typzuordnungen • Benutzerzuordnungen • Oberflächen	• Kurznamen, Namen für Zuordnungen, Indexspezifikationen, Server und Oberflächen dürfen nicht länger als 128 Byte sein. • Optionen für Server und Kurznamen und Optionseinstellungen sind auf 255 Byte begrenzt. • Namen für Objekte in zusammengeschlossenen Datenbanken können folgende Zeichen enthalten: – Gültige diakritische Zeichen (zum Beispiel ö) – Ein Mehrbytezeichen außer Mehrbyteleerzeichen (für Mehrbyteumgebungen)

Verwendung begrenzter Bezeichner in Objektnamen

Schlüsselwörter dürfen verwendet werden. Bei Verwendung eines Schlüsselworts in einem Kontext, in dem es auch als SQL-Schlüsselwort interpretiert werden könnte, muss es als begrenzter Bezeichner angegeben werden.

Mit Hilfe der begrenzten Bezeichner ist es möglich, ein Objekt zu erstellen, dessen Name gegen diese Namenskonventionen verstößt. Jedoch können bei nachfolgender Verwendung eines solchen Objekts Fehler auftreten. Wenn Sie zum Beispiel eine Spalte mit einem Namen erstellen, in dem ein Pluszeichen + oder ein Minuszeichen – vorkommt, und Sie in der Folge diese Spalte in einem Index verwenden, treten Probleme auf, wenn Sie versuchen, die Tabelle zu reorganisieren. Weitere Informationen zu begrenzten Bezeichnern finden Sie im Abschnitt ″SQL Identifiers″ im Handbuch *SQL Reference*.

Beibehaltung von Werten in Groß-/Kleinschreibung in einem System zusammengeschlossener Datenbanken

In verteilten Anforderungen müssen Sie manchmal Kennungen und Kennwörter angeben, deren Groß-/Kleinschreibung auf der Datenquelle unterschieden wird. Beachten Sie die folgenden Richtlinien, um sicherzustellen, dass die Schreibweise beim Übergeben an die Datenquelle korrekt ist:

- Geben Sie die IDs und Kennwörter in der erforderlichen Schreibweise (Groß-/Kleinschreibung) an, und setzen Sie sie in doppelte Anführungszeichen.
- Wenn Sie eine Benutzer-ID angeben, setzen Sie die Serveroption *fold_id* für die Datenquelle auf ″n″ („Nein, Groß-/Kleinschreibung nicht ändern″). Wenn Sie ein Kennwort angeben, setzen Sie die Serveroption *fold_pw* für die Datenquelle auf ″n″.

  Es gibt eine Alternative für Benutzer-IDs und Kennwörter. Wenn bei einer Datenquelle eine Benutzer-ID in Kleinbuchstaben angegeben werden muss, können Sie sie in beliebiger Schreibweise angeben und die Serveroption *fold_id* auf ″l″ („Diese ID in Kleinbuchstaben an die Datenquelle senden″) setzen. Wenn bei der Datenquelle die Benutzer-ID in Großbuchstaben angegeben werden muss, können Sie sie in beliebiger Schreibweise angeben und die Serveroption *fold_id* auf ″u″ („Diese ID in Großbuchstaben an die Datenquelle senden″) setzen. Auf dieselbe Art und Weise können Sie, wenn bei einer Datenquelle ein Kennwort in Klein- oder in Großbuchstaben angegeben werden muss, diese Anforderung erfüllen, indem Sie die Serveroption *fold_pw* auf ″l″ bzw. ″u″ setzen.

  Weitere Informationen zu Serveroptionen finden Sie in ″Vereinfachen der Datenquellendefinitionen und der Authentifizierungsverarbeitung durch Serveroptionen″ in *Systemverwaltung: Implementierung* .

- Wenn Sie eine Kennung oder ein Kennwort mit zu beachtender Groß-/Kleinschreibung in einer Eingabeaufforderung des Betriebssystems in doppelte Anführungszeichen setzen, müssen Sie sicherstellen, dass das System die doppelten Anführungszeichen syntaktisch korrekt analysiert. Gehen Sie dazu wie folgt vor:
  - Bei einem auf UNIX basierenden Betriebssystem schließen Sie die Anweisung in einfache Anführungszeichen ein.
  - Beim Betriebssystem Windows NT stellen Sie jedem Anführungszeichen einen umgekehrten Schrägstrich voran.

  Viele begrenzte Bezeichner in SB2-Datenquellen unterscheiden zum Beispiel Groß-/Kleinschreibung. Angenommen, Sie wollen einen Kurznamen `KURZ1` für eine Sicht von DB2 für CS, `"mein_schema"."wöch_geh"`, in einer Datenquelle namens `NORBASE` erstellen.

  In die Eingabeaufforderung eines auf UNIX basierenden Systems würden Sie Folgendes eingeben:

  ```
  db2 'create nickname kurz1 for norbase."mein_schema"."wöch_geh"'
  ```

  In eine Eingabeaufforderung von Windows NT würden Sie Folgendes eingeben:

  ```
  db2 create nickname kurz1 for norbase.\"mein_schema\".\"wöch_geh\"
  ```

  Wenn Sie die Anweisung an der DB2-Eingabeaufforderung (Dialogmodus) oder in einem Anwendungsprogramm eingeben, sind die einfachen Anführungszeichen oder die umgekehrten Schrägstriche nicht notwendig. In die DB2-Eingabeaufforderung eines auf UNIX basierenden Systems oder eines Windows NT-Systems würden Sie Folgendes eingeben:

  ```
  create nickname kurz1 for norbase."mein_schema"."wöch_geh"
  ```

Anhang B. Planen der Datenbankmigration

Dieser Abschnitt enthält einen Überblick über den Migrationsprozess. Beachten Sie, dass Datenbanken von DB2 UDB Version 6 nicht nach Version 7 migriert werden müssen. Detaillierte Informationen zur Migration von Datenbanken aus DB2 UDB Version 5.*x* finden Sie im Handbuch *Einstieg* zu Ihrem Betriebssystem.

Wenn Sie eine Datenbank migrieren:

- Folgende Bestandteile der Datenbank werden migriert:
  - Konfigurationsdatei der Datenbank
  - Systemkatalogtabellen der Datenbank
  - Datenbankverzeichnisse
  - Kopfsatz der Datenbankprotokolldatei
- Die Systemkatalogtabellen werden folgendermaßen geändert:
  - Neue Spalten werden hinzugefügt.
  - Neue Tabellen werden erstellt.
  - Eine Gruppe von Katalogsichten wird migriert, und eine Gruppe neuer Katalogsichten wird im Schema SYSCAT erstellt.
  - Eine Gruppe aktualisierbarer Katalogsichten wird im Schema SYSSTAT erstellt.
  - Eine Gruppe von Skalarfunktionen für allgemeine Zwecke wird behalten, und eine Gruppe neuer Skalarfunktionen für allgemeine Zwecke wird im Schema SYSFUN erstellt. Nur die Skalarfunktion SYSFUN.DIFFERENCE wird während der Datenbankmigration entfernt und neu erstellt.
- Eine Datenbankprotokolldatei wird mit zugehöriger Spiegelkopie im Datenbankverzeichnis erstellt. Diese Datei enthält eine Zusammenfassung der Sicherungsinformationen, die verwendet werden können, wenn eine Datenbank wiederhergestellt werden muss. Sie wird immer dann aktualisiert, wenn spezifische Operationen an der Datenbank ausgeführt werden. Eine Zusammenfassung von Sicherungsinformationen wird auch für Sicherungs- und Wiederherstellungsoperationen von Tabellenbereichen gespeichert.

Überlegungen zur Migration

Zur erfolgreichen Migration einer Datenbank, die mit einer vorherigen Version des Datenbankmanagers erstellt wurde, sind folgende Faktoren zu berücksichtigen:

- „Migrationseinschränkungen"
- „Sicherheit und Berechtigungen" auf Seite 227
- „Speicherbedarf" auf Seite 227
- „Durch Release-Wechsel bedingte Inkompatibilitäten" auf Seite 227

Migrationseinschränkungen

Für die Migration einer Datenbank auf Version 7 gelten bestimmte Bedingungen bzw. Einschränkungen, die zu beachten sind:

- Eine Migration wird nur von Version 5.x oder Version 6 unterstützt. Die Migration von DB2 Version 1.2 Parallel Edition wird nicht unterstützt. Frühere Version von DB2 (Datenbankmanager) müssen zunächst auf Version 5.x oder V6 umgestellt werden, bevor sie auf Version 7 migriert werden können.
- Das Absetzen des Migrationsbefehls von einem Client der Version 7 zur Migration einer Datenbank auf einen Server der Version 7 wird unterstützt. Jedoch wird das Absetzen eines Migrationsbefehls von älteren DB2-Clients zur Migration einer Datenbank auf einen Server der Version 7 nicht unterstützt.
- Migration zwischen Plattformen wird nicht unterstützt.
- Benutzerobjekte innerhalb Ihrer Datenbank können keine für Version 7 reservierte Schemennamen als Objektqualifikationsmerkmale haben. Zu diesen reservierten Schemennamen gehören: SYSCAT, SYSSTAT und SYSFUN.
- Benutzerdefinierte einzigartige Datentypen mit den Namen BIGINT, REAL, DATALINK bzw. REFERENCE müssen vor der Migration der Datenbank umbenannt werden.
- Es ist nicht möglich, eine Datenbank zu migrieren, die sich in einem der folgenden Status befindet:
  - Sicherung anstehend
  - Aktualisierende Wiederherstellung anstehend
  - Einer oder mehrere Tabellenbereiche nicht im Normalstatus
  - Transaktion inkonsistent
- Die Wiederherstellung von Datenbanksicherungen früherer Versionen (Version 5.x oder Version 6) wird unterstützt. Die aktualisierende Wiederherstellung (ROLLFORWARD) von Protokollen früherer Versionen wird jedoch nicht unterstützt.

Sicherheit und Berechtigungen

Sie benötigen die Berechtigung SYSADM, um Ihre Datenbank migrieren zu können.

Speicherbedarf

Während der Migration wird sowohl für die alten als auch für die neuen Kataloge Speicherplatz benötigt. Die Menge des erforderlichen Plattenspeicherplatzes hängt von der Größe und Komplexität der Datenbank sowie von der Anzahl und der Größe der Datenbankobjekte ab. Zu diesen Objekten gehören sämtliche Tabellen und Sichten. Sie sollten mindestens das Doppelte des Plattenspeicherplatzes verfügbar machen, den der Datenbankkatalog momentan einnimmt. Reicht der Plattenspeicherplatz nicht aus, schlägt die Migration fehl.

Wenn Ihr SYSCAT-Tabellenbereich ein SMS-Tabellenbereich ist, sollten Sie außerdem in Betracht ziehen, die Konfigurationsparameter der Datenbank zu aktualisieren, die sich auf die Protokolldateien auswirken. Sie sollten die Werte der Parameter *logfilsiz*, *logprimary* und *logsecond* erhöhen, um zu verhindern, dass der Speicherbereich für die Dateien aufgebraucht wird (SQL1704N mit Ursachencode 3). In diesem Fall erhöhen Sie die Werte der Parameter für den Protokollspeicherbereich und setzen den Befehl MIGRATE DATABASE erneut ab.

Durch Release-Wechsel bedingte Inkompatibilitäten

Berücksichtigen Sie bei der Planung der Migration einer Datenbank die Auswirkungen der Inkompatibilitäten zwischen den beiden Versionen des Produkts.

Um die Erweiterungen von Version 7 nutzen zu können, sollten Sie Ihre Datenbank- und Datenbankmanagerkonfiguration nach der Migration Ihrer Datenbanken optimieren. Zur Vereinfachung dieser Optimierung können Sie die Werte der Konfigurationsparameter vor und nach der Migration aufzeichnen und vergleichen. (Eine Beschreibung der Befehle GET DATABASE CONFIGURATION und GET DATABASE MANAGER CONFIGURATION finden Sie im Handbuch *Command Reference*.)

Umstellen einer Datenbank

Die folgenden Schritte müssen zur Migration einer Datenbank ausgeführt werden. Bevor Sie mit der Migration beginnen, müssen Sie den Datenbankmanager starten.

PRÄMIGRATION:

Anmerkung: Die Schritte der Prämigration müssen an einem vorigen Release (d. h. an dem aktuellen Release vor dem Umstellen auf das neue Release bzw. vor der Installation des neuen Release) ausgeführt werden.

1. Stellen Sie sicher, dass es keine geklärten Punkte im Sinne der „Migrationseinschränkungen" auf Seite 226 gibt.
2. Trennen Sie alle Anwendungen und Endbenutzer von jeder zu migrierenden Datenbank (verwenden Sie nach Bedarf den Befehl LIST APPLICATIONS oder FORCE APPLICATIONS).
3. Verwenden Sie das Prämigrationsdienstprogramm DB2CKMIG, um festzustellen, ob die Datenbank migriert werden kann (detaillierte Informationen zu diesem Dienstprogramm finden Sie im Handbuch *Einstieg* für Ihre Plattform). Beachten Sie, dass Sie unter Windows NT oder OS/2 aufgefordert werden, dieses Tool während der Installation auszuführen, während es auf UNIX-basierten Systemen automatisch bei der Exemplarmigration aufgerufen wird.
4. Erstellen Sie eine Sicherungskopie der Datenbank.
   Die Migration ist ein Prozess, der sich nicht rückgängig machen lässt. Wenn Sie Ihre Datenbank sichern, bevor die unter Version 6 reservierten Schemennamen geändert werden, sind Sie nicht in der Lage, die Datenbank mit Hilfe von DB2 UDB Version 7 wiederherzustellen. Zur Wiederherstellung der Datenbank müssen Sie Ihre frühere Version des Datenbankmanagers verwenden.

   Achtung! Wenn Sie keine Sicherungskopie Ihrer Datenbank angelegt haben und der Migrationsversuch fehlschlägt, gibt es keine Möglichkeit, Ihre Datenbank mit Hilfe von DB2 UDB Version 7 oder der vorherigen Version Ihres Datenbankmanagers wiederherzustellen.

   Beachten Sie außerdem, dass alle Datenbanktransaktionen, die im Zeitraum zwischen der Erstellung der Sicherung und der Migration auf Version 7 erfolgen, nicht wiederhergestellt werden können. Das bedeutet, dass, wenn zu einem Zeitpunkt nach der Beendigung der Installation und Migration auf Version 7 die Datenbank wiederhergestellt werden muss (auf eine Stufe der Version 7), die Protokolldateien, die vor der Installation von Version 7 geschrieben wurden, bei der aktualisierenden Wiederherstellung nicht verwendet werden können.

MIGRATION:

5. Migrieren Sie die Datenbank mit einer der folgenden Methoden:
   - Befehl MIGRATE DATABASE
   - Befehl RESTORE DATABASE bei der Wiederherstellung einer Gesamtsicherung der Datenbank
   - API sqlemgdb - API zur Datenbankmigration

Unter OS/2: Das Migrationshilfsprogramm DB2CIDMG, das in einer Umgebung mit CID-Architektur (CID - Configuration/Installation/Distribution) ausführbar ist, steht nur unter DB2 für OS/2 zur Verfügung. Es ermöglicht eine ferne nichtüberwachte Installation und Konfiguration auf Workstations in einem LAN. Ihr LAN muss über NetView DM/2 verfügen, damit die CID-Migration verwendet werden kann.

Auf UNIX-basierten Systemen: Das Handbuch *Einstieg* für Ihre Plattform beschreibt, was zu tun ist, wenn Sie nicht alle Datenbanken in einem bestimmten Exemplar migrieren wollen.

POSTMIGRATION:

6. Verwenden Sie wahlfrei das Dienstprogramm DB2UIDDL, das die Verwaltung einer gestaffelten Migration eindeutiger Indizes nach eigenem Zeitplan erleichtert. (DB2-Datenbanken, die in Version 5 erstellt wurden, benötigen dieses Tool nicht, um eine verzögerte Überprüfung auf Eindeutigkeit nutzen zu können, da alle eindeutigen Indizes, die in Version 5 erstellt wurden, über diese Semantik bereits verfügen. Jedoch ist für Datenbanken, die zuvor auf Version 5 migriert wurden, diese Semantik nicht automatisch, sofern Sie nicht das Dienstprogramm DB2UIDDL zum Ändern der eindeutigen Indizes verwenden.) Dieses Dienstprogramm generiert Anweisungen CREATE UNIQUE INDEX für eindeutige Indizes auf Benutzertabellen und schreibt diese in eine Datei. Das Ausführen dieser Datei als DB2-CLP-Befehlsdatei bewirkt, dass der eindeutige Index in die Semantik von Version 7 umgewandelt wird. Detaillierte Informationen zur Verwendung dieses Dienstprogramms finden Sie in einem der Handbücher *Einstieg*.
7. Führen Sie wahlfrei den Befehl RUNSTATS für Tabellen aus, die besonders entscheidend für die Leistung von SQL-Abfragen sind. Die alten Statistikdaten werden in der migrierten Datenbank beibehalten und erst aktualisiert, wenn Sie den Befehl RUNSTATS ausführen.
8. Verwenden Sie wahlfrei das Dienstprogramm DB2RBIND, um alle Pakete wieder gültig zu machen, oder lassen Sie die Gültigkeit von Paketen implizit bei der ersten Verwendung eines Pakets wiederherstellen.
9. Migrieren Sie wahlfrei EXPLAIN-Tabellen, wenn Sie beabsichtigen, sie in Version 7 zu verwenden. Weitere Informationen finden Sie in "SQL-EXPLAIN-Einrichtung" im Handbuch *Systemverwaltung: Optimierung*.
10. Optimieren Sie die Konfigurationsparameter Ihrer Datenbank und Ihres Datenbankmanagers, um die Erweiterungen von Version 7 zu nutzen.

Anhang C. Inkompatibilitäten zwischen Releases

Dieser Abschnitt beschreibt Inkompatibilitäten, die zwischen DB2 Universal Database und früheren Releases von DB2 bestehen.

Eine *Inkompatibilität* ist ein Bestandteil von DB2 Universal Database, dessen Funktionsweise im Vergleich zu einem früheren Release von DB2 geändert wurde. Bei Verwendung in einer vorhandenen Anwendung führt sie zu einem unerwarteten Ergebnis, macht eine Änderung in der Anwendung erforderlich oder beeinträchtigt die Leistung. In diesem Kontext bezieht sich die Bezeichnung *Anwendung* auf folgendes:

- Anwendungsprogrammcode
- Dienstprogramme anderer Hersteller
- Interaktive SQL-Abfragen
- Aufrufe von Befehlen oder APIs

Es werden Inkompatibilitäten beschrieben, die mit DB2 Universal Database Version 6 und Version 7 eingeführt wurden. Sie werden nach folgenden Kategorien untergliedert:

- Systemkatalogsichten
- Anwendungsprogrammierung
- SQL
- Datenbanksicherheit und Optimierung
- Dienstprogramme und Tools
- Konnektivität und Koexistenz
- Konfigurationsparameter

Jeder Inkompatibilitätsabschnitt einhält eine Beschreibung der Inkompatibilität, das Symptom oder den Effekt der Inkompatibilität und mögliche Lösungsmaßnahmen. Zu Anfang jeder Inkompatibilitätsbeschreibung gibt ein Indikator an, auf welches Betriebssystem sich die Inkompatibilität bezieht:

WIN Von DB2 unterstützte Microsoft Windows-Plattformen

UNIX Von DB2 unterstützte UNIX-basierte Plattformen

OS/2 OS/2

Anmerkung: Mit dem Stand von DB2 Universal Database Version 6 werden Clients der Version 1.x und Version 2.x, einschließlich der Clients, die mit DB2 Parallel Edition Version 1.2-Servern geliefert werden, nicht länger unterstützt.

Geplante Inkompatibilitäten von DB2 Universal Database

Dieser Abschnitt beschreibt zukünftige Inkompatibilitäten, die Benutzer von DB2 Universal Database bei der Codierung neuer Anwendungen bzw. der Änderung vorhandener Anwendungen berücksichtigen sollten. Dadurch wird die Migration auf künftige Versionen von DB2 UDB vereinfacht.

Schreibgeschützte Sichten in einer künftigen Version von DB2 Universal Database

WIN	UNIX	OS/2

Änderung
Die Systemkatalogsichten werden schreibgeschützte Sichten. Die SYSSTAT-Sichten bleiben weiterhin aktualisierbar.

Symptom
UPDATE-Anweisungen, die bisher für Spalten in den SYSCAT-Sichten funktionierten, schlagen nun fehl.

Erläuterung
Tools oder Anwendungen sind so codiert, dass sie Werte im Katalog durch Aktualisieren der Spalte gemäß der Definition in der SYSCAT-Sicht ändern.

Maßnahme
Ändern Sie das Tool oder die Anwendung so, dass der Katalog durch Aktualisieren der Spalte gemäß der Definition in der SYSSTAT-Sicht geändert wird.

PK_COLNAMES und FK_COLNAMES in einer künftigen Version von DB2 Universal Database

WIN	UNIX	OS/2

Änderung
Die SYSCAT.REFERENCES-Spalten PK_COLNAMES und FK_COLNAMES sind nicht mehr verfügbar.

Symptom
Die Spalte ist nicht mehr vorhanden, und es wird ein Fehler zurückgegeben.

Erläuterung
Tools oder Anwendungen sind so codiert, dass sie die veralteten Spalten PK_COLNAMES und FK_COLNAMES verwenden.

Maßnahme
Ändern Sie das Tool oder die Anwendung so, dass stattdessen die Sicht SYSCAT.KEYCOLUSE verwendet wird.

COLNAMES in einer künftigen Version von DB2 Universal Database nicht mehr verfügbar

WIN	UNIX	OS/2

Änderung
Die SYSCAT.INDEXES-Spalte COLNAMES ist nicht mehr verfügbar.

Symptom
Die Spalte ist nicht mehr vorhanden, und es wird ein Fehler zurückgegeben.

Erläuterung
Tools oder Anwendungen sind so codiert, dass sie die veraltete Spalte COLNAMES verwenden.

Maßnahme
Ändern Sie das Tool oder die Anwendung so, dass stattdessen die Sicht SYSCAT.INDEXCOLUSE verwendet wird.

Inkompatibilitäten in DB2 Universal Database Version 7

Dieser Abschnitt beschreibt die Inkompatibilitäten, die mit DB2 Universal Database Version 7 eingeführt werden.

Anwendungsprogrammierung

Query Patroller Universal Client

WIN	UNIX	OS/2

Änderung: Diese neue Version des Client Application Enabler (CAE) funktioniert nur mit Query Patroller Server Version 7, weil neue gespeicherte Prozeduren vorhanden sind. CAE ist die Anwendungsschnittstelle für DB2, die letzten Endes alle Anwendungen passieren müssen, um auf die Datenbank zuzugreifen.

Symptom: Wenn diese CAE-Version mit einem Server einer früheren Version ausgeführt wird, wird SQL29001 zurückgegeben.

Objektumsetzungsfunktionen und strukturierte Typen

WIN	UNIX	OS/2

Änderung: Es gibt eine kleinere und entfernt mögliche Inkompatibilität zwischen einem Client einer Version vor Version 7 und einem Server der Version 7, die im Zusammenhang mit Änderungen am SQL-Deskriptorbereich

(SQLDA) stehen. Wie im Handbuch *Application Development Guide* erläutert, kann Byte 8 des zweiten SQLVAR-Feldes nun den Wert X'12' (neben den Werten X'00' und X'01') annehmen. Anwendungen, die den neuen Wert nicht abfangen, können von dieser Erweiterung beeinträchtigt werden.

Maßnahme: Da es in zukünftigen Releases noch weitere Erweiterungen an diesem Feld geben kann, sind Entwickler gut beraten, dieses Feld nur auf explizit definierte Werte zu testen.

Von JVM verwendete Versionen von Class- und Jar-Dateien

WIN	UNIX	OS/2

Änderung: Wenn früher eine gespeicherte Java-Prozedur oder eine benutzerdefinierte Funktion (UDF) gestartet wurde, sperrte Java Virtual Machine (JVM) alle Dateien, die in CLASSPATH angegeben waren (einschließlich der Dateien in `sqllib/function`). JVM verwendete diese Dateien, bis der Datenbankmanager gestoppt wurde. Abhängig von der Umgebung, in der Sie eine gespeicherte Prozedur oder UDF ausführen (d. h. abhängig vom Wert des Konfigurationsparameters *keepdari* des Datenbankmanagers und davon, ob die gespeicherte Prozedur abgeschirmt (fenced) wird), ermöglicht Ihnen das Aktualisieren (Refresh) von Klassen ein Ersetzen von CLASS- und JAR-Dateien, ohne den Datenbankmanager zu stoppen. Dies unterscheidet sich von der früheren Funktionsweise.

Geänderte Funktionalität der Befehle zum Installieren, Ersetzen und Entfernen eines JAR

WIN	UNIX	OS/2

Änderung: In der Vergangenheit bewirkte die Installation eines JAR ein Abbrechen aller DARI-Prozesse (Database Application Remote Interface). Auf diese Weise wurde sichergestellt, dass eine neue gespeicherte Prozedurklasse garantiert mit dem nächsten Aufruf ausgewählt wurde. Zur Zeit brechen keine JAR-Befehle DARI-Prozesse ab. Um sicherzustellen, dass Klassen aus neu installierten oder ersetzten JARs ausgewählt werden, müssen Sie den Befehl SQLEJ.REFRESH_CLASSES explizit absetzen.

Eine weitere Inkompatibilität, die durch das Nichtabrechen von DARI-Prozessen eingeführt wird, ist die Tatsache, dass für abgeschirmte (fenced) gespeicherte Prozeduren, wenn der Wert des Konfigurationsparameters *keepdari* des Datenbankmanagers auf "YES" gesetzt ist, Clients verschiedene Versionen der JAR-Dateien erhalten können. Betrachten Sie folgendes Szenario:

1. Benutzer A ersetzt ein JAR und aktualisiert (Refresh) die Klassen nicht.
2. Benutzer A ruft anschließend eine gespeicherte Prozedur aus dem JAR auf. Unter der Annahme, dass dieser Aufruf denselben DARI-Prozess verwendet, erhält Benutzer A eine alte Version der JAR-Datei.
3. Benutzer B ruft dieselbe gespeicherte Prozedur auf. Dieser Aufruf arbeitet mit einer neuen DARI, was bedeutet, dass das neu erstellte Klassenladeprogramm (Class loader) die neue Version der JAR-Datei auswählt.

Mit anderen Worten, wenn Klassen nach JAR-Operationen nicht aktualisiert (Refresh) werden, kann eine gespeicherte Prozedur aus verschiedenen Versionen von JARs aufgerufen werden, je nachdem, welche DARI-Prozesse verwendet werden. Dies unterscheidet sich von der früheren Funktionsweise, bei der (durch Abbrechen (Flush) der DARI-Prozesse) sichergestellt war, dass neue Klassen immer verwendet wurden.

Inkompatibilität bei 32-Bit-Anwendungen

	UNIX	

Änderung: Ausführbare 32-Bit-Codedateien (DB2-Anwendungen) funktionieren mit der neuen 64-Bit-Datenbanksteuerkomponente nicht.

Symptom: Die Anwendung kann nicht verbunden ("gelinkt") werden. Wenn Sie versuchen, 32-Bit-Objekte mit der 64-Bit-Anwendungsbibliothek von DB2 zu verbinden, wird eine Betriebssystemfehlernachricht des Verbindungseditors (Linker) angezeigt.

Maßnahme: Die Anwendung muss als ausführbare 64-Bit-Codedatei neu kompiliert und wieder mit den neuen 64-Bit-Bibliotheken von DB2 verbunden (gelinkt) werden.

Ändern des Längenfelds des Arbeitspuffers

WIN	UNIX	OS/2

Änderung: Jede benutzerdefinierte Funktion (UDF), die das Längenfeld des Arbeitspuffers ändert, der an die UDF übergeben wurde, empfängt nun einen SQLCODE-Wert -450.

Symptom: Eine UDF, die das Längenfeld des Arbeitspuffers ändert, schlägt fehl. Die aufrufende Anweisung empfängt den SQLCODE-Wert -450 mit Angabe des Schemas und des bestimmten Namens der Funktion.

Maßnahme: Schreiben Sie den Hauptteil der UDF um, so dass das Längenfeld des Arbeitspuffers nicht geändert wird.

SQL

Anwendungen, die mit dem Schema SESSION qualifizierte reguläre Tabellen verwenden

WIN	UNIX	OS/2

Änderung: Das Schema SESSION ist das einzig zulässige Schema für temporäre Tabellen und wird nun von DB2 verwendet, um anzuzeigen, dass ein mit SESSION qualifizierter Tabellenname auf eine temporäre Tabelle verweisen kann. Jedoch ist SESSION kein reserviertes Schlüsselwort für temporäre Tabellen und kann als Schema für reguläre Basistabellen verwendet werden. Aus diesem Grund kann eine Anwendung vielleicht sowohl eine richtige Tabelle SESSION.T1 als auch eine deklarierte temporäre Tabelle SESSION.T1 gleichzeitig vorfinden. Wenn beim Binden eines Pakets eine statische Anweisung angetroffen wird, die einen (explizit oder implizit) durch SESSION qualifizierten Tabellenverweis enthält, wird weder ein Abschnitt (Section) noch eine Abhängigkeit für diese Anweisung in den Katalogen gespeichert. Statt dessen muss dieser Abschnitt zur Laufzeit inkrementell gebunden werden. Dadurch wird eine Kopie des Abschnitts in den Cache für dynamisches SQL gestellt, wo die zwischengespeicherte Kopie nur für das eindeutige Exemplar der Anwendung privat ist. Wenn zur Laufzeit eine deklarierte temporäre Tabelle mit übereinstimmendem Tabellennamen vorhanden ist, wird die deklarierte temporäre Tabelle verwendet, selbst wenn eine permanente Basistabelle desselben Namens existiert.

Symptom: In Version 6 (und früheren) verweisen alle Pakete mit statischen Anweisungen, die Tabellen mit durch SESSION qualifizierten Namen betreffen, immer auf eine permanente Basistabelle. Beim Binden eines Pakets werden ein Abschnitt (Section) sowie relevante Abhängigkeitseinträge für diese Anweisung in den Katalogen gespeichert. In Version 7 werden diese Anweisungen nicht zur Bindezeit gebunden und können zur Laufzeit möglicherweise in eine deklarierte temporäre Tabelle des gleichen Namens aufgelöst werden. Daraus können folgende Situationen entstehen:

- Bei Migration von Version 5. Wenn ein solches Paket in Version 5 vorhanden war, wird es in Version 6 erneut gebunden, und die statischen Anweisungen werden nun inkrementell gebunden. Dies kann sich auf die Leistung auswirken, da diese inkrementell gebundenen Abschnitte (Sections) sich wie im Cache zwischengespeichertes dynamisches SQL verhalten, mit der Ausnahme, dass der im Cache zwischengespeicherte Abschnitt nicht mit anderen Anwendungen gemeinsam benutzt werden kann (nicht einmal von verschiedenen Exemplaren derselben ausführbaren Anwendungsdatei).
- Bei Migration von Version 6 zu Version 7. Wenn ein solches Paket in Version 6 vorhanden war, wird es in Version 7 nicht unbedingt erneut gebunden. Statt dessen werden die Anweisungen weiterhin als reguläres stati-

sches SQL unter Verwendung des Abschnitts (Section) ausgeführt, der im Katalog zu ursprünglichen Bindezeit gespeichert wurde. Wenn dieses Paket jedoch erneut gebunden wird (implizit oder explizit), werden die Anweisungen im Paket mit durch SESSION qualifizierten Tabellenverweisen nicht mehr gespeichert und erfordern inkrementelles Binden. Dadurch kann die Leistung beeinträchtigt werden.

Zusammengefasst lässt sich sagen, dass jedes Paket, dass in Version 7 mit statischen Anweisungen gebunden wird, die auf durch SESSION qualifizierte Tabellen verweisen, sich nicht länger wie statisches SQL verhalten, weil sie ein inkrementelles Binden erfordern. Wenn der Anwendungsprozess eine Anweisung DECLARE GLOBAL TEMPORARY TABLE für eine Tabelle absetzt, die den gleichen Namen wie eine vorhandene, mit SESSION qualifizierte Tabelle, Sicht bzw. ein solcher Aliasname besitzt, werden Verweise auf diese Objekte immer auf die deklarierte temporäre Tabelle bezogen.

Maßnahme: Ändern Sie nach Möglichkeit die Schemennamen permanenter Tabellen, so dass nicht SESSION verwendet wird. Ansonsten gibt es keine Abhilfe, außer sich die Auswirkungen auf die Leistung und den möglichen Konflikt mit deklarierten temporären Tabellen bewusst zu machen.

Die folgende Abfrage kann zur Erkennung von Tabellen, Sichten und Aliasnamen verwendet werden, die betroffen sein könnten, wenn eine Anwendung mit temporären Tabellen arbeitet:

```
select tabschema, tabname from SYSCAT.TABLES where tabschema = 'SESSION'
```

Die folgende Abfrage kann zur Ermittlung von gebundenen Paketen der Version 7 dienen, für die statische Abschnitte (Sections) in den Katalogen gespeichert sind und deren Funktionsweise sich ändern könnte, wenn das Paket erneut gebunden wird (nur bei Migration von Version 6 zu Version 7 relevant):

```
select pkgschema, pkgname, bschema, bname from syscat.packagedep
   where bschema = 'SESSION' and btype in ('T', 'V', 'I')
```

Dienstprogramme und Tools

Data Links File Manager und File System Filter unter Solaris

	UNIX	

Änderung: Data Links File Manager und File System Filter werden unter Solaris OS 2.5.1 nicht unterstützt.

db2set unter AIX und Solaris

	UNIX	

Änderung: Der Befehl ″db2set -ul (user level)″ und die zugehörigen Funktionen wurden nicht auf AIX oder Solaris portiert.

Data Links File System und Norton Utilities**

WIN		

Änderung: Windows NT Data Links File System ist mit Norton Utilities nicht kompatibel.

Symptom: Wenn eine Datei von einem Laufwerk gelöscht wird, das von DLFS gesteuert wird, tritt eine Kernelausnahmebedingung auf: error 0x1E (Kernel Mode Exception Not Handled). Die Ausnahmebedinung ist 0xC00000005 (Zugriffsverletzung).

Erläuterung: Dieser ungültige Zugriff ist darauf zurückzuführen, dass der Treiber für Norton Utilities nach dem Laden des DLFS-Filtertreibers geladen wird.

Maßnahme: Dieser Fehler lässt sich vermeiden, indem der DLFSD-Treiber nach dem Treiber für Norton Utilities geladen wird. Ändern Sie die Startart des DLFSD-Treibers in manuellen Start. Klicken Sie dazu ″Start″ an, und wählen Sie Einstellungen—> Systemsteuerung—> Geräte—> DLFSD aus, und setzen Sie den Treiber auf ″Manuell″.

Sie können eine Stapeldatei erstellen, die den DLFSD-Treiber und den DLFM-Dienst beim Systemstart lädt. Die Stapeldatei hat folgenden Inhalt:

```
net start dlfsd
net start "dlfm service"
```

Geben Sie dieser Stapeldatei den Namen start_dlfs.bat, und kopieren Sie sie in das Verzeichnis
WINNT\Profiles\Administrator\Startmenü\Programme\Autostart. Nur ein Administrator besitzt die Berechtigung, den DLFS-Filtertreiber und den DLFM-Dienst zu laden.

Konnektivität und Koexistenz

32-Bit-Client-Inkompatibilität

WIN	UNIX	OS/2

Änderung: 32-Bit-Clients können keine Verbindung zu Exemplaren (Attach) oder Datenbanken (Connect) auf 64-Bit-Servern herstellen.

Symptom: Wenn sowohl der Client als auch der Server mit Code der Version 7 arbeiten, wird SQL1434N zurückgegeben. Ansonsten schlägt der Verbindungsversuch (Attach oder Connect) mit SQLCODE-Wert -30081 fehl.

Maßnahme: Verwenden Sie 64-Bit-Clients.

Inkompatibilitäten in DB2 Universal Database Version 6

Dieser Abschnitt beschreibt die Inkompatibilitäten, die mit DB2 Universal Database Version 6 eingeführt werden.

Systemkatalogsichten

Systemkatalogsichten in DB2 Universal Database Version 6

WIN	UNIX	OS/2

Änderung: In den Systemkatalogsichten wurden neue Codes eingeführt: „U" für typisierte Tabellen und „W" für typisierte Sichten.

Symptom: Abfragen, die in den Systemkatalogen mit dem Typencode „T" nach Tabellen bzw. „V" nach Sichten suchen, finden keine typisierten Tabellen und Sichten mehr.

Erläuterung: Verschiedene Systemkataloge, einschließlich der Systemkatalogsichten TABLES, PACKAGEDEP, TRIGDEP und VIEWDEP, haben eine Spalte namens TYPE oder BTYPE, die einen aus einem Buchstaben bestehenden Typencode enthält. In Version 5.2 wurde der Typencode „T" für alle Tabellen und der Typencode „V" für alle Sichten verwendet. In Version 6 haben nicht typisierte Tabellen weiterhin den Typencode „T". Typisierte Tabelle haben dagegen den neuen Typencode „U". Analog dazu haben nicht typisierte Sichten weiterhin den Typencode „V". Typisierte Sichten haben dagegen den neuen Typencode „W". Darüber hinaus wird eine neue Art von Tabelle, eine so genannte Hierarchietabelle, die nicht direkt von Benutzern erstellt, sondern vom System zur Implementierung von Tabellenhierarchien verwendet wird, in den Systemkatalogtabellen mit dem Typencode „H" angezeigt.

Maßnahme: Ändern Sie das Tool oder die Anwendung so, dass die Codes für typisierte Tabellen und Sichten erkannt werden. Wenn das Tool oder die Anwendung eine logische Sicht von Tabellen benötigt, sollten die Typencodes „T", „U", „V" und „W" verwendet werden. Benötigt das Tool oder die Anwendung eine physische Sicht von Tabellen, einschließlich Hierarchietabellen, sollten die Typencodes „T" und „H" verwendet werden.

Primär- und Fremdschlüsselspaltennamen in DB2 Universal Database Version 6

WIN	UNIX	OS/2

Änderung: In zwei SYSCAT.REFERENCES-Spalten, PK_COLNAMES und FK_COLNAMES, wird der Datentyp von VARCHAR(320) in VARCHAR(640) geändert.

Symptom: Primärschlüssel- oder Fremdschlüsselspaltennamen werden abgeschnitten, sind nicht korrekt oder fehlen.

Erläuterung: Wenn in einem Primärschlüssel oder Fremdschlüssel Spaltennamen verwendet werden, die länger als 18 Byte sind, kann das Format, mit dem die Liste von Spaltennamen in diesen beiden Spalten gespeichert ist, nicht gleich bleiben. Die mit Leerzeichen begrenzten 20 Byte langen Spaltennamen, die auf die Spalte folgen, deren Länge (*n*) größer als 18 Byte ist, werden um *n*-18 Byte nach rechts verschoben. Wenn die Liste von Spaltennamen 640 Byte übersteigt, enthält die Spalte auch die leere Zeichenfolge.

Maßnahme: Die Sicht SYSCAT.KEYCOLUSE enthält die Liste der Spalten, aus denen ein Primär-, Fremd- oder eindeutiger Schlüssel besteht, und sollte anstelle der Spalten in SYSCAT.REFERENCES verwendet werden. Alternativ dazu können Benutzer die Länge der Spaltennamen auf 18 Byte oder die Gesamtlänge der Spaltenliste auf 640 Byte begrenzen.

SYSCAT.VIEWS-Spalte TEXT in DB2 Universal Database Version 6

WIN	UNIX	OS/2

Änderung: Sichttext in der SYSCAT.VIEWS-Spalte TEXT wird nicht mehr auf mehrere Zeilen verteilt. Der Datentyp wird von VARCHAR(3600) in CLOB(64K) geändert.

Symptom: Das Tool oder die Anwendung zeigt nicht den vollständigen Sichttext an.

Erläuterung: Tools oder Anwendungen, die so codiert sind, dass sie erwarten, dass höchstens 3600 (oder möglicherweise 3900) Byte auf einmal von der Spalte TEXT zurückgegeben werden, können die erhöhte Größe dieses Felds nicht handhaben. Der Mechanismus zum Abrufen mehrerer Zeilen und zum Wiederherstellen des Sichttexts über das Feld SEQNO ist nicht mehr notwendig. Der Wert für SEQNO ist immer 1.

Maßnahme: Ändern Sie das Tool oder die Anwendung so, dass Werte aus der Spalte TEXT gehandhabt werden können, die größer als 3600 Byte sind. Alternativ dazu könnte der Sichttext so umgeschrieben werden, dass er 3600 Byte nicht übersteigt.

SYSCAT.STATEMENTS-Spalte TEXT in DB2 Universal Database Version 6

WIN	UNIX	OS/2

Änderung: Anweisungstext in der SYSCAT.STATEMENTS-Spalte TEXT wird nicht mehr auf mehrere Zeilen verteilt. Der Datentyp wird von VARCHAR(3600) in CLOB(64K) geändert.

Symptom: Das Tool oder die Anwendung zeigt nicht den vollständigen Anweisungstext an.

Erläuterung: Tools oder Anwendungen, die so codiert sind, dass sie erwarten, dass höchstens 3600 (oder möglicherweise 3900) Byte auf einmal von der Spalte TEXT zurückgegeben werden, können die erhöhte Größe dieses Felds nicht handhaben. Der Mechanismus zum Abrufen mehrerer Zeilen und zum Wiederherstellen des Anweisungstexts über das Feld SEQNO ist nicht mehr notwendig. Der Wert für SEQNO ist immer 1.

Maßnahme: Ändern Sie das Tool oder die Anwendung so, dass Werte aus der Spalte TEXT gehandhabt werden können, die größer als 3600 Byte sind. Alternativ dazu könnte der Anweisungstext so umgeschrieben werden, dass er 3600 Byte nicht übersteigt.

SYSCAT.INDEXES-Spalte COLNAMES in DB2 Universal Database Version 6

WIN	UNIX	OS/2

Änderung: Der Datentyp der SYSCAT.INDEXES-Spalte COLNAMES wird von VARCHAR(320) in VARCHAR(640) geändert.

Symptom: Es fehlen Spaltennamen aus einem Index.

Erläuterung: Tools oder Anwendungen, die so codiert sind, dass sie Daten aus einer Spalte mit dem Datentyp VARCHAR(320) abrufen, können die erhöhte Größe dieses Felds nicht handhaben.

Maßnahme: Die Sicht SYSCAT.INDEXCOLUSE enthält die Liste von Spalten, aus denen ein Index besteht. Sie sollte anstelle der Spalte COLNAMES verwendet werden. Alternativ dazu können Sie eine Spalte aus dem Index entfer-

nen oder die Größe des Spaltennamens verringern, so dass die Liste von Spaltennamen (mit führendem + oder −) 320 Byte nicht übersteigt.

SYSCAT.CHECKS-Spalte TEXT in DB2 Universal Database Version 6

WIN	UNIX	OS/2

Änderung: Der Datentyp der SYSCAT.CHECKS-Spalte TEXT wird von CLOB(32K) in CLOB(64K) geändert.

Symptom: Die Klausel für die Prüfung auf Integritätsbedingung ist unvollständig.

Erläuterung: Tools oder Anwendungen, die so codiert sind, dass sie Daten aus einer Spalte mit dem Datentyp CLOB(32K) abrufen, können die erhöhte Größe dieses Felds nicht handhaben.

Maßnahme: Ändern Sie das Tool oder die Anwendung so, dass Werte aus der Spalte TEXT gehandhabt werden können, die länger als 32 KB sind. Alternativ dazu können Sie die Klausel für die Prüfung auf Integritätsbedingung so umschreiben, dass sie weniger Zeichen verwendet und so 32 KB nicht übersteigt.

Spaltendatentyp in DB2 Universal Database Version 6 in BIGINT geändert

WIN	UNIX	OS/2

Änderung: Bei mehreren Spalten von Systemkatalogsichten wurde der Datentyp von INTEGER in BIGINT geändert.

Symptom: Werte sind viel kleiner (oder größer) als erwartet, insbesondere bei statistischen Daten.

Erläuterung: Tools oder Anwendungen, die so codiert sind, dass sie Daten aus einer Spalte mit dem Datentyp INTEGER abrufen, können die erhöhte Größe dieses Felds nicht handhaben.

Maßnahme: Ändern Sie das Tool oder die Anwendung so, dass Werte gehandhabt werden können, die größer als der Höchstwert oder kleiner als Mindestwert sind, der in einem INTEGER-Feld gespeichert werden kann. Alternativ dazu können Sie die zugrundeliegende Struktur oder den zugrundeliegenden SQL-Code ändern, durch die/den der Wert aus dem Bereich herausfällt, der durch ein INTEGER-Feld dargestellt werden kann.

Spaltenabweichung in DB2 Universal Database Version 6

WIN	UNIX	OS/2

Änderung: Neue Spalten werden in der SYSCAT-Sichtdefinition nicht am Ende von Sichten eingefügt.

Symptom: Die erneute Vorverarbeitung schlägt mit mehreren Abweichungen der Spalte oder des Spaltendatentyps fehl.

Erläuterung: Neue Spalten werden in die Systemkatalogsichten eingeführt und für eine Sofortabfrageumgebung sinnvoll angeordnet, d. h. kürzere Spalten werden vor sehr langen Spalten angeordnet, und die Spalte REMARKS ist immer die letzte Spalte.

Maßnahme: Nennen Sie die Spalten in der SELECT-Liste explizit, statt „SELECT *" zu verwenden.

SYSCAT.COLUMNS und SYSCAT.ATTRIBUTES in DB2 Universal Database Version 6

WIN	UNIX	OS/2

Änderung: SYSCAT.COLUMNS und SYSCAT.ATTRIBUTES enthalten jetzt Einträge für übernommene Spalten und Attribute.

Symptom: Abfragen auf SYSCAT.COLUMNS zum Abrufen der Spalten einer typisierten Tabelle oder Sicht sowie Abfragen auf SYSCAT.ATTRIBUTES zum Abrufen der Attribute eines strukturierten Typs können in Version 6 mehr Zeilen zurückgeben als in Version 5.2, wenn das Ziel der Abfrage eine untergeordnete Tabelle, eine untergeordnete Sicht oder ein Subtyp ist.

Erläuterung: In Version 5.2 enthielten die Kataloge COLUMNS und ATTRIBUTES für eine bestimmte Tabelle, eine bestimmte Sicht oder einen bestimmten strukturierten Typ nur Einträge für Spalten und Attribute, die von dieser Tabelle, dieser Sicht oder diesem Typ eingeführt wurden. Spalten und Attribute, die von übergeordneten Tabellen oder Typen übernommen wurden, wurden nicht in den Katalogen dargestellt. In Version 6 enthalten jetzt die Kataloge COLUMNS und ATTRIBUTES jedoch Einträge für übernommene Spalten und Attribute.

Maßnahme: Ändern Sie das Tool oder die Anwendung so, dass die neuen Einträge in den Katalogen COLUMNS und ATTRIBUTES erkannt werden.

OBJCAT-Sichten in DB2 Universal Database Version 6 nicht mehr unterstützt

WIN	UNIX	OS/2

Änderung: Die rekursiven Katalogsichten im Schema OBJCAT von Version 5.2 werden nicht mehr mit DB2 Universal Database mitgeliefert.

Symptom: Abfragen, die für die OBJCAT-Katalogsichten geschrieben sind, werden nicht mehr erfolgreich ausgeführt.

Maßnahme: Die meisten der Informationen, die bisher in den OBJCAT-Sichten enthalten waren, sind jetzt in normalen SYSCAT-Katalogsichten enthalten. In den meisten Fällen können Sie die Informationen aus den Systemkatalogsichten abrufen. Wenn Sie von Version 5.2 migrieren und die OBJCAT-Katalogsichten vorhanden sind, sollten diese gelöscht werden. Dies kann mit der Befehlszeilenprozessorprozedur `objcatdp.db2` im Unterverzeichnis `misc` des Verzeichnisses `sqllib` erfolgen.

Sie können auch Ihre eigenen OBJCAT-Sichten erstellen, die zu den in Version 5.2 unterstützten Katalogsichten äquivalent sind.

In Version 5.2 wurden die Benutzer in Anhang E des Handbuchs *SQL Reference* darauf hingewiesen, dass die OBJCAT-Katalogsichten temporär sind und in zukünftigen Versionen nicht mehr unterstützt werden.

Abhängigkeitscodes in DB2 Universal Database Version 6 geändert

WIN	UNIX	OS/2

Änderung: In den Systemkatalogsichten werden die hierarchischen Abhängigkeiten, die bisher mit dem Code „H" bezeichnet wurden, jetzt mit dem Code „O" bezeichnet.

Symptom: Abfragen, die in den Katalogsichten mit dem Code „H" nach hierarchischen Abhängigkeiten suchen, funktionieren nicht mehr korrekt.

Erläuterung: Verschiedene Systemkataloge, einschließlich der Systemkatalogsichten PACKAGEDEP, TRIGDEP und VIEWDEP, haben eine Spalte namens BTYPE. In Version 5.2 bezeichneten die OBJCAT-Sichten hierarchische Abhängigkeiten mit dem Code „H". In Version 6 werden diese Abhängigkeiten mit dem Code „O" bezeichnet.

Maßnahme: Überarbeiten Sie diese Abfragen, so dass sie nach Code „O" suchen.

SYSIBM-Basiskatalogtabellen in DB2 Universal Database Version 6

WIN	UNIX	OS/2

Änderung: Die folgenden Änderungen wurden an den SYSIBM-Basiskatalogtabellen vorgenommen, die Sie vielleicht immer noch statt der SYSCAT-Sichten verwenden:

- Gelöschte Felder (jedoch in den SYSCAT-Sichten weiter vorhanden):
  - SYSSTMT.SEQNO
  - SYSVIEWS.SEQNO
- Umbenannte Katalogtabelle: SYSTRIGDEP wurde in SYSDEPENDENCIES geändert. Die Spalten BCREATOR und DCREATOR wurden in BSCHEMA bzw. DSCHEMA umbenannt. Die Sicht SYSCAT.TRIGDEP wurde nicht geändert.
- Gelöschte Felder (waren nie in den SYSCAT-Sichten enthalten):
  - SYSATTRIBUTES.DEFAULT_VALUE
  - SYSATTRIBUTES.NULLS
  - SYSCOLUMNS.SERVERTYPE
  - SYSDATATYPES.REFREP_TYPENAME
  - SYSDATATYPES.REFREP_TYPESCHEMA
  - SYSDATATYPES.REFREP_LENGTH
  - SYSDATATYPES.REFREP_SCALE
  - SYSDATATYPES.REFREP_CODEPAGE
  - SYSINDEXES.TEXT

    (War in der Sicht enthalten, aber nur zur künftigen Nutzung reserviert.)
  - SYSPLANDEP.PUBLICPRIV
  - SYSSECTION.SEQNO
  - SYSTABAUTH.UPDATE_BY_COLS
  - SYSTABAUTH.REF_BY_COLS
  - SYSTABLES.MINPDLENGTH
  - SYSTABLESPACES.READONLY
  - SYSTABLESPACES.REMOVABLEMEDIA
- Datentypänderungen:
  - SYSSECTION.SECTION: von VARCHAR(3600) in CLOB(10M)
  - SYSPLANDEP.COLUSAGE: von VARCHAR(3000) FOR BIT DATA in BLOB(5K)

Anwendungsprogrammierung

Datentyp VARCHAR in DB2 Universal Database Version 6

WIN	UNIX	OS/2

Änderung: Die mögliche Maximalgröße des Datentyps VARCHAR (VARGRAPHIC) wurde in Version 6 von 4000 Zeichen (2000 Doppelbytezeichen) auf 32672 Zeichen (16336 Doppelbytezeichen) erhöht.

Symptom: Eine Anwendung, die Puffer mit einer festen Länge von 4000 Byte für den Datentyp VARCHAR (VARGRAPHIC) verwendet, überschreibt möglicherweise Puffer oder schneidet Daten ab, wenn sie ein VARCHAR-Feld mit mehr als 4000 Byte in einen zu kleinen Puffer abruft. Die CLI-Funktion `SQLGetTypeInfo()` gibt jetzt die Größe von VARCHAR als 32672 zurück. CLI-Anwendungen, die diesen Wert in Tabellen-DDLs verwenden, erhalten möglicherweise Fehler, weil keine Tabellenbereiche mit ausreichender Seitengröße verfügbar sind. Weitere Informationen zur Größe von Tabellenbereichen finden Sie in „Benutzertabellendaten" auf Seite 108.

Maßnahme: Bei der Codierung der Anwendung ist zu empfehlen, zuerst die Spalten der Ergebnismenge (mit der Anweisung DESCRIBE) zu beschreiben und dann Puffer zu verwenden, deren Größe sich nach der von der Anweisung DESCRIBE zurückgelieferten Länge richtet.

Positionierte UPDATE- und DELETE-Anweisungen bei Java-Programmierung in DB2 Universal Database Version 6

WIN	UNIX	OS/2

Änderung: Beim Programmieren von Java in Version 6 verwenden positionierte UPDATE- und DELETE-Anweisungen die Standardberechtigungskennung der Person, die das Cursorpaket gebunden hat. Dies unterscheidet sich von Version 5.2, bei der die Berechtigungskennung der Person verwendet wird, die das Paket ausführt.

Symptom: Das Paket mit den positionierten UPDATE- und DELETE-Anweisungen wird möglicherweise nicht ausgeführt, da die Berechtigungskennung der Person, die das Paket gebunden hat, keine ausreichende Berechtigung hat.

Maßnahme: Der Berechtigungskennung der Person, die das Paket bindet, muss eine ausreichende Berechtigung zum Ausführen der positionierten UPDATE- und DELETE-Anweisungen im Paket erteilt werden. Erteilen Sie die korrekten Zugriffsrechte, und binden Sie das Paket dann erneut.

Syntaxänderung in der Klausel FOR UPDATE in DB2 Universal Database Version 6

WIN	UNIX	OS/2

Änderung: In Version 5.2 kann in einem SQLJ-Programm die Klausel FOR UPDATE in einer SELECT-Anweisungen verwendet werden, um die Spalten anzugeben, die in nachfolgenden positionierten UPDATE-Anweisungen aktualisiert werden können. Die Syntax wurde für Version 6 geändert.

Symptom: Sie erhalten die Fehlernachricht SQJ0204E, wenn eine SELECT-Anweisung eine Klausel FOR UPDATE enthält.

Maßnahme: Entfernen Sie die Klausel FOR UPDATE aus der SELECT-Anweisung. Geben Sie einen aktualisierbaren Iterator über die Iteratordeklarationsklausel an. Beispiel:

```
#sql public iterator DelByName implements sqlj.runtime.ForUpdate
   (String EmpNo) with updateColumns = (salary);
```

Wenn Sie explizit angeben wollen, welche Spalten aktualisierbar sind, geben Sie diese über das Schlüsselwort `updateColumns` in Verbindung mit der Klausel WITH an.

Weitere Informationen zur Deklaration positionierter Iteratoren finden Sie im Handbuch *Application Development Guide*.

Zeichennamenlängen in DB2 Universal Database Version 6

WIN	UNIX	OS/2

Änderung: DB2 Universal Database Version 6 unterstützt 128 Byte lange Tabellen-, Sichten- und Aliasnamen sowie 30 Byte lange Spaltennamen. Bisher wurden 18 Byte lange Namen für alle diese Kategorien unterstützt.

Die Sonderregister USER und CURRENT SCHEMA wurden von CHAR(8) in VARCHAR(128) geändert. Das Sonderregister CURRENT EXPLAIN MODE wurde von CHAR(8) in VARCHAR(254) geändert. Die Ausgabe für die integrierten Funktionen TYPE_SCHEMA und TABLE_SCHEMA wurde von CHAR(8) in VARCHAR(128) geändert.

Symptom: Wenn Anwendungen, die vor Version 6 entwickelt wurden, für eine Datenbank der Version 6 ausgeführt werden, die die längeren Begrenzungen nicht verwendet, sollte sich das Anwendungsverhalten nicht ändern. Wenn Sie jedoch diese Anwendungen für eine Datenbank der Version 6 ausführen, die längere Namen verwendet, könnte es je nach Codierung dieser Anwendungen zu gewissen Nebeneffekten können.

Es folgen einige Beispiele:

- Eine vorhandene Anwendung ruft mit einer FETCH-Anweisung einen Tabellen- oder Spaltennamen (typischerweise aus einer Katalogsicht) in eine Host-Variable ab, die mit einer Länge von 18 Byte definiert wurde. Da 18 Byte bis Version 6 die maximale Größe für den Tabellen- oder Spaltennamen waren, prüft diese Anwendung möglicherweise nicht das `sqlwarn1`-Bit des SQL-Kommunikationsbereichs. Sie geht (fälschlicherweise) davon aus, dass ein Abschneiden der Daten niemals auftritt.
- Eine Anwendung ruft mit einer FETCH-Anweisung einen Tabellen- oder Spaltennamen (typischerweise aus einer Katalogsicht) in einen SQL-Deskriptorbereich (SQLDA) ab, bei dem die Größe des Felds `sqldata` auf der Basis des Feldes `sqllen` aus einer DESCRIBE-Operation der SELECT-Anweisung zugeordnet wurde. Dies führt dazu, dass das korrekte (nicht abgeschnittene) Ergebnis an die Anwendung zurückgegeben wird, obwohl die Größe der Tabellen- oder Spaltennamen sich möglicherweise erhöht hat. Wenn eine Anwendungslogik von der Annahme ausgeht, dass Spaltennamen auf 18 Byte begrenzt sind, können die zurückgegebenen längeren Namen auf unerwartete Weise behandelt werden. Zum Beispiel kann die Anzeige längerer Spaltennamen bei 18 Byte abgeschnitten werden.
- Da das Token-Feld des SQL-Kommunikationsbereichs (`sqlerrmc`) auf 70 Byte begrenzt ist, können vorhandene Anwendungen, die versuchen, eine Zeile in eine Tabelle einzufügen, betroffen sein. Als Reaktion auf die Fehlernachricht SQL0204N ermitteln solche Anwendungen den Namen der Tabelle aus dem Feld `sqlerrmc` des SQL-Kommunikationsbereichs und führen dann einige Operationen mit Hilfe dieses Objektnamens durch. Bei früheren Versionen von DB2 wurde durch die Begrenzung der Tabellen- oder Schemenkennungen sichergestellt, dass der gesamte Tabellenname in den SQL-Kommunikationsbereich aufgenommen wurde. Dies ist in Version 6 nicht der Fall.
- Eine Anwendung, die eine frühere Version einer API verwendet, erhält nur die ersten 18 Byte eines Tabellennamens.
- Auf vorhandene CLI- und ODBC-Anwendungen, die die Schemenfunktionen (wie SQLTables(), oder SQLColumns() u. a.) verwenden, ergibt sich eine Auswirkung, wenn eine Verbindung zu einem Server hergestellt wird, der längere Namen als 18 Byte unterstützt. Obwohl Warnungen ausgegeben werden, dass abgeschnitten wird, prüft die Anwendung möglicherweise diese Warnung nicht, und fährt mit einem abgeschnittenen Namen fort.

Maßnahme: Die beste Möglichkeit, Probleme dieser Art zu beheben, besteht darin, die Anwendung so neu zu codieren, dass sie längere Tabellen- und Spaltenname handhaben kann. Stellen Sie andernfalls sicher, dass diese Anwendungen nicht für Datenbanken der Version 6 ausgeführt werden, die Namen mit mehr als 18 Byte verwenden.

PC/IXF-Formatänderungen in DB2 Universal Database Version 6

WIN	UNIX	OS/2

Änderung: DB2 Universal Database Version 6 unterstützt 128 Byte lange Tabellen-, Sichten- und Aliasnamen sowie 30 Byte lange Spaltennamen. Bisher wurden 18 Byte lange Namen für alle diese Kategorien unterstützt.

Symptom: Ein Client unter DB2 Universal Database Version 5 kann keine PC/IXF-Datei importieren, die von einem Client unter DB2 Universal Database Version 6 exportiert wurde (Fehlernachricht SQL3059N). Eine PC/IXF-Datei (exportiert aus einem Client unter DB2 Universal Database Version 6) kann nicht in eine Datenbank von DB2 Universal Database Version 5 geladen werden (Fehlernachricht SQL3059N).

Maßnahme: Verwenden Sie beim Importieren oder Laden von PC/IXF-Daten kompatible Versionen von DB2 Universal Database.

SQLNAME in nicht verdoppelten SQLVAR-Einträgen in DB2 Universal Database Version 6

WIN	UNIX	OS/2

Änderung: DB2 Universal Database Version 6 unterstützt 30 Byte lange Spaltennamen. Bisher wurden 18 Byte lange Namen unterstützt. In Version 5 war die dokumentierte Funktionsweise, dass „0xFF" in das 30. Byte eines Felds SQLNAME für nicht verdoppelte SQLVAR-Einträge platziert wurde. Für systemgenerierte Namen und benutzerdefinierte Spaltennamen, die in einer Klausel AS angegeben wurden, wurde auch „0x00" in das 30. Byte eingefügt.

In Version 6 wird „0xFF" nur dann im 30. Byte zurückgegeben, wenn der Name systemgeneriert ist.

Symptom: Alle Anwendungen, die mit Hilfe des 30. Byte des Felds SQLNAME festzustellen, ob es sich um einen benutzerdefinierten Spaltennamen oder um einen vom System generierten Namen handelt, erhalten unerwartete Ergebnisse bei Logikprüfungen, wenn der benutzerdefinierte Spaltenname 30 Zeichen lang ist. Dies sollte selten vorkommen.

Maßnahme: Diese Anwendungen sollten so geändert werden, dass sie nur auf „0xFF" im 30. Byte des Felds SQLNAME prüfen, wenn die Länge dieses Feldes kleiner als 30 ist. In diesem Fall ist der Name benutzerdefiniert.

Veraltete CLI-/ODBC-Konfigurationsschlüsselwörter in DB2 Universal Database Version 6

WIN		

Änderung: Bei der Migration zu einer neuen Version von DB2 UDB können Sie die Funktionsweise des DB2-CLI-/ODBC-Treibers durch Angeben einer Gruppe wahlfreier Schlüsselwörter in der Datei `db2cli.ini` ändern.

In Version 6 sind die Schlüsselwörter TRANSLATEDLL und TRANSLATEOPTION veraltet.

Symptom: Diese Schlüsselwörter werden ignoriert, wenn sie noch angegeben sind. Das Wegfallen dieser Einstellungen kann zu erkennbaren Veränderungen in der Funktionsweise führen.

Maßnahme: Entscheiden Sie anhand der neuen Liste der gültigen Parameter, welche Schlüsselwörter und Einstellungen Ihrer Umgebung entsprechen. Informationen zu diesen Schlüsselwörtern finden Sie im Handbuch *CLI Guide and Reference*.

Ausgabedatenstromformat des Ereignismonitors in DB2 Universal Database Version 6

WIN	UNIX	OS/2

Änderung: Ausgabedatenströme des Ereignismonitors haben keine Versionssteuerung. Wenn daher Unterstützung für Tabellennamen hinzugefügt wird, die länger als 18-byte Byte sind, muss zu einem Ausgabedatenstromformat gewechselt werden.

Symptom: Anwendungen, die die Ausgabedatenströme des Ereignismonitors syntaktisch analysieren, funktionieren nicht mehr ordnungsgemäß.

Maßnahme: Es gibt zwei Möglichkeiten:

- Aktualisieren Sie die Anwendung, so dass sie den neuen Datenstrom verwendet.
- Setzen Sie die Registrierungsvariable

  ```
  DB2OLDEVMON=evmonname1,evmonname2,...
  ```

  Dabei ist *evmonname* der Name des Ereignismonitors (Event Monitor), der im alten Datenformat geschrieben werden soll. Beachten Sie, dass auf neue Felder im Ereignismonitor unter dem alten Datenformat nicht zugegriffen werden kann.

SQL

DATALINK-Spalten in DB2 Universal Database Version 6

	UNIX	

Änderung: Datenübertragungsverbindungswerte (DATALINK-Werte), die under DB2 Universal Database Version 6 eingefügt werden, benötigen 4 Byte mehr Platz im Spaltenwertdeskriptor.

Symptom: Wenn in Version 5.2 erstellte DATALINK-Spalten aktualisiert werden, sind auf der Datenseite weitere 4 Byte erforderlich, um den neuen Spaltenwert zu speichern. Es ist daher möglicherweise nicht genügend Platz in der Datenseite vorhanden, um die Aktualisierung durchzuführen, und sie muss vielleicht auf eine neue Seite versetzt werden. Dadurch könnte der Aktualisierung der Platz ausgehen.

Maßnahme: Sie müssen auf Ihrem System mehr Platz für Aktualisierungen hinzufügen.

SYSFUN-Zeichenfolgefunktionskennungen in DB2 Universal Database Version 6

WIN	UNIX	OS/2

Änderung: Eine Reihe von Zeichenfolgefunktionen im Schema SYSFUN haben jetzt verbesserte Versionen, die im Schema SYSIBM definiert sind (integrierte Funktionen). Die Funktionsnamen sind LCASE, LTRIM, RTRIM und UCASE.

Symptom: Beim Vorbereiten von Anweisungen oder beim Erstellen von Sichten können sich die von diesen Funktionen zurückgegebenen Datentypen in Version 6 unterscheiden. Dies kommt daher, dass die integrierten Funktionen (unter dem Schema SYSIBM) normalerweise vor Funktionen im Schema SYSFUN aufgelöst werden.

Maßnahme: Es ist keine Maßnahme erforderlich. Normalerweise wird die integrierte Funktion vor der Funktion im Schema SYSFUN ausgewählt. Die frühere Funktionsweise kann durch Tauschen des SQL-Pfads (so dass SYSFUN vor SYSIBM steht) wiederhergestellt werden, jedoch wird die Leistung dadurch beeinträchtigt. Die Funktion der früheren Version kann auch dadurch aufgerufen werden, dass der Funktionsname mit dem Schemenname SYSFUN qualifiziert wird.

Migrierte Pakete, Sichten, Übersichtstabellen, Auslöser und Integritätsbedingungen verwenden weiter die Version aus dem Schema SYSFUN, sofern

keine explizite Aktion wie das Binden des Pakets oder das Neuerstellen der Sicht, der Übersichtstabelle, des Auslösers oder der Integritätsbedingung vorgenommen wird.

SET INTEGRITY ersetzt SET CONSTRAINTS

WIN	UNIX	OS/2

Änderung: Die Anweisung SET CONSTRAINTS wurde durch die Anweisung SET INTEGRITY ersetzt. Aus Gründen der Abwärtskompatibilität werden beide Anweisungen in DB2 Version 6 und Version 7 akzeptiert.

SYSTABLE-Spaltenänderung mit neuem Integritätsstatus in DB2 Universal Database Version 6

WIN	UNIX	OS/2

Änderung: Der Status „U" in der Spalte CONST_CHECKED von SYSCAT-.TABLES ändert sich anders, wenn eine Anweisung SET INTEGRITY ... OFF ausgeführt wird.

Symptom: Vor Version 6 änderte sich der Status „U" in CONST_CHECKED in „N", wenn eine Anweisung SET INTEGRITY ... OFF ausgeführt wurde. Jetzt ändert sich der Status „U" in den Status „W".

Maßnahme: Es ist keine Maßnahme erforderlich. Der neue Status „W" in der Spalte CONST_CHECKED wird verwendet, um anzugeben, dass der Typ der Integritätsbedingung bisher durch den Benutzer geprüft wurde und dass einige Daten in der Tabelle eventuell auf Integrität geprüft werden müssen.

Der Status „N" klärt nicht darüber auf, ob es alte Daten gibt, die noch nicht durch den Datenbankmanager überprüft wurden. In einer nachfolgenden Anweisung SET INTEGRITY ... IMMEDIATE CHECKED INCREMENTAL muss der Datenbankmanager einen Fehler zurückgeben, weil die Datenintegrität nicht gewährleistet werden kann, wenn nur neue Änderungen geprüft wurden. Andererseits kann der Status „W" zurück in den Status „U" geändert werden (wenn die Option INCREMENTAL angegeben ist), um anzugeben, dass der Benutzer weiterhin für die Integrität der Daten in der Tabelle verantwortlich ist. Wenn die Option INCREMENTAL nicht angegeben wird, wählt der Datenbankmanager die vollständige Verarbeitung, ändert den Status „W" in den Status „Y" und übernimmt die Verantwortung für die Aufrechterhaltung der Datenintegrität.

Datenbanksicherheit und Optimierung

Erstellen von Datenbanken mit Clients in DB2 Universal Database Version 6

WIN	UNIX	OS/2

Änderung: Die von den Clients verwendete Methode zum Erstellen einer Datenbank hat sich geändert.

Symptom: Die Verwendung eines Clients einer früheren Version zur Erstellung einer Datenbank führt zu Fehlern.

Maßnahme: Stellen Sie bei der Verwendung eines Clients zur Erstellung einer Datenbank sicher, dass der Client und der Server mit der gleichen Stufe des DB2-Codes arbeiten.

SELECT-Zugriffsrecht in DB2 Universal Database Version 6 in Hierarchie erforderlich

WIN 32-Bit	UNIX	OS/2

Änderung: Die Angabe des Schlüsselworts ONLY (für eine Tabelle) setzt jetzt voraus, dass der Benutzer das SELECT-Zugriffsrecht für alle untergeordneten Tabellen der angegebenen typisierten Tabelle hat. Analog dazu setzt die Angabe des Schlüsselworts ONLY (für eine Sicht) jetzt voraus, dass der Benutzer das SELECT-Zugriffsrecht für alle untergeordneten Sichten der angegebenen typisierten Tabelle hat. In früheren Versionen von DB2 war das SELECT-Zugriffsrecht nur für die angegebene Tabelle oder Sicht erforderlich.

Symptom: Es gibt zwei mögliche Symptome:

- Ein Berechtigungsfehler (SQLCODE -551, SQLSTATE 42501) tritt beim erneuten Binden eines Pakets mit einer SQL-Anweisung auf, in der das Schlüsselwort ONLY in einer Klausel FROM angegeben ist, wenn die Berechtigungs-ID, unter der das Paket gebunden wurde, nicht das SELECT-Zugriffsrecht für die untergeordneten Tabellen der angegebenen typisierten Tabelle (oder Sicht) hat.
- Wenn die Definition einer Sicht oder eines Auslösers das Schlüsselwort ONLY in einer Klausel FROM enthält, arbeitet die Sicht oder der Auslöser weiter normal. Die Definition der Sicht oder des Auslösers kann jedoch nicht mehr verwendet werden, um eine eine neue Sicht oder einen neuen Auslöser zu erstellen, wenn der Ersteller nicht das SELECT-Zugriffsrecht für alle untergeordneten Tabellen der angegebenen Tabelle (oder Sicht) hat.

Maßnahme: Der Berechtigungs-ID, die ein Paket erneut binden oder eine neue Sicht oder einen neuen Auslöser erstellen soll, sollte das SELECT-Zugriffsrecht für alle untergeordneten Tabellen (und untergeordnete Sichten) der Tabelle (oder Sicht) erteilt werden, die nach dem Schlüsselwort ONLY angegeben ist.

Veraltete Profilregistrierdatenbank und Umgebungsvariablen in DB2 Universal Database Version 6

WIN	UNIX	OS/2

Änderung: Die folgenden Profilregistrierdatenbank- oder Umgebungsvariablen sind veraltet:

- DB2_VECTOR

Maßnahme: Diese Variablen werden nicht mehr benötigt.

Dienstprogramme und Tools

Aktueller EXPLAIN-Modus in DB2 Universal Database Version 6

WIN	UNIX	OS/2

Änderung: Der Typ des Sonderregisters CURRENT EXPLAIN MODE wurde von CHAR(8) in VARCHAR(254) geändert.

Symptom: Wenn die Anwendung davon ausgeht, dass der Typ immer noch CHAR(8) ist, kann der Wert von 254 auf 8 Byte abgeschnitten werden.

Maßnahme: Ändern Sie den Typ aller Host-Variablen, die das Sonderregister lesen, von CHAR(8) in VARCHAR(254).

Diese Änderung ist erforderlich, um zwei neue Werte im Sonderregister CURRENT EXPLAIN MODE unterzubringen. Diese neuen Werte sind EVALUATE INDEXES und RECOMMEND INDEXES.

Parameter USING und SORT BUFFER in DB2 Universal Database Version 6

WIN	UNIX	OS/2

Änderung: Ab Version 6 werden die Parameter USING und SORT BUFFER des Befehls LOAD nicht mehr unterstützt. Diese Parameter werden ignoriert.

Symptom: Es wird eine Warnung zurückgegeben, die besagt, dass die Parameter USING und SORT BUFFER nicht länger unterstützt und vom Dienstprogramm LOAD ignoriert werden.

Maßnahme: Ignorieren Sie die Warnung. Weitere Informationen finden Sie im Handbuch *Versetzen von Daten Dienstprogramme und Referenz*.

Konnektivität und Koexistenz

Ersetzen von RUMBA durch PCOMM in DB2 Universal Database Version 6

WIN		

Änderung: In Version 6 wird RUMBA durch PCOMM unter Windows NT, Windows 98 und Windows 95 (jedoch nicht unter Windows 3.1) ersetzt.

Symptom: Keine

Maßnahme: Keine

Konfigurationsparameter

Veraltete Datenbankkonfigurationsparameter

WIN	UNIX	OS/2

Änderung: Die folgenden Datenbankkonfigurationsparameter sind veraltet:

- DL_NUM_BACKUP (ersetzt durch den Datenbankkonfigurationsparameter NUM_DB_BACKUP)

Maßnahme: Entfernen Sie alle Verweise auf diese Parameter aus Ihren Anwendungen.

Anhang D. Unterstützung von Landessprachen

Dieser Abschnitt enthält Informationen zur Unterstützung von Landessprachen (National Language Support, NLS), die von DB2 bereitgestellt wird, einschließlich Informationen zu den unterstützten Ländern/Regionen, Sprachen und Codepages (codierten Zeichensätzen) sowie Angaben zu Konfiguration und Verwendung von DB2 NLS-Funktionen in Datenbanken und Anwendungen.

NLS-Versionen

DB2 Version 7 ist in folgenden Sprachen verfügbar: Englisch, Französisch, Deutsch, Italienisch, Spanisch, brasilianisches Portugiesisch, Japanisch, Koreanisch, vereinfachtes Chinesisch, traditionelles Chinesisch, Dänisch, Finnisch, Norwegisch, Schwedisch, Tschechisch, Niederländisch, Ungarisch, Polnisch, Türkisch, Russisch, Bulgarisch und Slowenisch.

Unterstützung von Landes-/Regionalcodes und Codepages

Tabelle 28 auf Seite 258 zeigt die von den Datenbankservern unterstützten Sprachen und codierten Zeichensätze sowie die Zuordnung dieser Werte zu den vom Datenbankmanager verwendeten Werten für den Landes-/Regionalcode und die Codepage.

Es folgt eine Erläuterung zu jeder Spalte der Tabelle:

- Unter **Codepage** wird die von IBM definierte Codepage gezeigt, wie sie vom codierten Zeichensatz des Betriebssystems zugeordnet wird.
- Unter **Gruppe** wird angegeben, ob eine Codepage Einzelbytezeichen („S") oder Doppelbytezeichen („D") enthält. Die Angabe „-n" ist eine Zahl, die zur Erstellung einer Buchstabe-Zahl-Kombination verwendet wird. Übereinstimmende Kombinationen geben an, wo die Verbindung und die Umsetzung von DB2 erlaubt ist. Beispielsweise können alle „S-1"-Gruppen zusammenarbeiten.
- Unter **Codierter Zeichens.** wird der codierte Zeichensatz für die unterstützte Sprache aufgeführt. Der codierte Zeichensatz wird der DB2-Codepage zugeordnet.
- Unter **Geb.** ist die Gebietskennung angegeben.
- Unter **Landes-/Regionalcode** finden Sie den Code, den der Datenbankmanager intern zum Bereitstellen der regionenspezifischen Unterstützung verwendet.

- Unter **Länderspez. Angaben** werden die Werte der vom Datenbankmanager unterstützten länderspezifischen Angaben angezeigt.
- Unter **BS** wird das Betriebssystem angezeigt, das die Sprachen und codierten Zeichensätze unterstützt.
- Unter **Land/Region** wird der Name der Region, des Landes oder der Länder angegeben.

Tabelle 28. Unterstützte Sprachen und codierte Zeichensätze

Code-page	Gruppe	Codierter Zeichens.	Geb.	Landes-/ Regional-code	Ländersp. Angaben	BS	Land/ Region
437	S-1	IBM-437	AL	355	-	OS2	Albanien
850	S-1	IBM-850	AL	355	-	OS2	Albanien
819	S-1	ISO8859-1	AL	355	sq_AL	AIX	Albanien
850	S-1	IBM-850	AL	355	Sq_AL	AIX	Albanien
819	S-1	iso88591	AL	355	-	HP	Albanien
1051	S-1	roman8	AL	355	-	HP	Albanien
819	S-1	ISO8859-1	AL	355	-	Sun	Albanien
1252	S-1	1252	AL	355	-	WIN	Albanien
1275	S-1	1275	AL	355	-	Mac	Albanien
37	S-1	IBM-37	AL	355	-	HOST	Albanien
1140	S-1	IBM-1140	AL	355	-	HOST	Albanien
864	S-6	IBM-864	AA	785	-	OS2	Arabische Länder
1046	S-6	IBM-1046	AA	785	Ar_AA	AIX	Arabische Länder
1089	S-6	ISO8859-6	AA	785	ar_AA	AIX	Arabische Länder
1089	S-6	iso88596	AA	785	ar_SA.iso88596	HP	Arabische Länder
1256	S-6	1256	AA	785	-	WIN	Arabische Länder
420	S-6	IBM-420	AA	785	-	HOST	Arabische Länder
437	S-1	IBM-437	AU	61	-	OS2	Australien
850	S-1	IBM-850	AU	61	-	OS2	Australien
819	S-1	ISO8859-1	AU	61	en_AU	AIX	Australien
850	S-1	IBM-850	AU	61	En_AU	AIX	Australien
819	S-1	iso88591	AU	61	-	HP	Australien
1051	S-1	roman8	AU	61	-	HP	Australien
819	S-1	ISO8859-1	AU	61	en_AU	Sun	Australien
819	S-1	ISO8859-1	AU	61	en_AU	SCO	Australien
1252	S-1	1252	AU	61	-	WIN	Australien
1275	S-1	1275	AU	61	-	Mac	Australien
37	S-1	IBM-37	AU	61	-	HOST	Australien
1140	S-1	IBM-1140	AU	61	-	HOST	Australien

Tabelle 28. Unterstützte Sprachen und codierte Zeichensätze (Forts.)

Code-page	Gruppe	Codierter Zeichens.	Geb.	Landes-/ Regional-code	Ländersp. Angaben	BS	Land/ Region
437	S-1	IBM-437	AT	43	-	OS2	Österreich
850	S-1	IBM-850	AT	43	-	OS2	Österreich
819	S-1	ISO8859-1	AT	43	ge_AT	AIX	Österreich
850	S-1	IBM-850	AT	43	Ge_AT	AIX	Österreich
819	S-1	iso88591	AT	43	-	HP	Österreich
1051	S-1	roman8	AT	43	-	HP	Österreich
819	S-1	ISO8859-1	AT	43	de_AT	SCO	Österreich
819	S-1	ISO-8859-1	AT	43	de_AT	Linux	Österreich
819	S-1	ISO8859-1	AT	43	de_AT	Sun	Österreich
1252	S-1	1252	AT	43	-	WIN	Österreich
1275	S-1	1275	AT	43	-	Mac	Österreich
37	S-1	IBM-37	AT	43	-	HOST	Österreich
1140	S-1	IBM-1140	AT	43	-	HOST	Österreich
915	S-5	ISO8859-5	BY	375	-	OS2	Weißrussland
915	S-5	ISO8859-5	BY	375	be_BY	AIX	Weißrussland
1131	S-5	IBM-1131	BY	375	-	OS2	Weißrussland
1251	S-5	1251	BY	375	-	WIN	Weißrussland
1283	S-5	1283	BY	375	-	Mac	Weißrussland
1025	S-5	IBM-1025	BY	375	-	HOST	Weißrussland
274	S-1	IBM-274	BE	32	-	HOST	Belgien
437	S-1	IBM-437	BE	32	-	OS2	Belgien
850	S-1	IBM-850	BE	32	-	OS2	Belgien
819	S-1	ISO8859-1	BE	32	nl_BE	AIX	Belgien
850	S-1	IBM-850	BE	32	Nl_BE	AIX	Belgien
819	S-1	iso88591	BE	32	-	HP	Belgien
819	S-1	ISO8859-1	BE	32	fr_BE	SCO	Belgien
819	S-1	ISO8859-1	BE	32	nl_BE	SCO	Belgien
819	S-1	ISO-8859-1	BE	32	nl_BE	Linux	Belgien
819	S-1	ISO8859-1	BE	32	nl_BE	Sun	Belgien
1252	S-1	1252	BE	32	-	WIN	Belgien
1275	S-1	1275	BE	32	-	Mac	Belgien
500	S-1	IBM-500	BE	32	-	HOST	Belgien
1148	S-1	IBM-1148	BE	32	-	HOST	Belgien
855	S-5	IBM-855	BG	359	-	OS2	Bulgarien
915	S-5	ISO8859-5	BG	359	-	OS2	Bulgarien
915	S-5	ISO8859-5	BG	359	bg_BG	AIX	Bulgarien
915	S-5	iso88595	BG	359	bg_BG.iso88595	HP	Bulgarien
1251	S-5	1251	BG	359	-	WIN	Bulgarien
1283	S-5	1283	BG	359	-	Mac	Bulgarien
1025	S-5	IBM-1025	BG	359	-	HOST	Bulgarien

Tabelle 28. Unterstützte Sprachen und codierte Zeichensätze (Forts.)

Code-page	Gruppe	Codierter Zeichens.	Geb.	Landes-/ Regional-code	Ländersp. Angaben	BS	Land/ Region
850	S-1	IBM-850	BR	55	-	OS2	Brasilien
850	S-1	IBM-850	BR	55	-	AIX	Brasilien
819	S-1	ISO8859-1	BR	55	pt_BR	AIX	Brasilien
819	S-1	ISO8859-1	BR	55	-	HP	Brasilien
819	S-1	ISO8859-1	BR	55	pt_BR	SCO	Brasilien
819	S-1	ISO8859-1	BR	55	pt_BR	Sun	Brasilien
819	S-1	ISO-8859-1	BR	55	pt_BR	Linux	Brasilien
1252	S-1	1252	BR	55	-	WIN	Brasilien
37	S-1	IBM-37	BR	55	-	HOST	Brasilien
1140	S-1	IBM-1140	BR	55	-	HOST	Brasilien
850	S-1	IBM-850	CA	1	-	OS2	Kanada
850	S-1	IBM-850	CA	1	En_CA	AIX	Kanada
819	S-1	ISO8859-1	CA	1	en_CA	AIX	Kanada
819	S-1	iso88591	CA	1	fr_CA.iso88591	HP	Kanada
1051	S-1	roman8	CA	1	fr_CA.roman8	HP	Kanada
819	S-1	ISO8859-1	CA	1	en_CA	SCO	Kanada
819	S-1	ISO8859-1	CA	1	fr_CA	SCO	Kanada
819	S-1	ISO8859-1	CA	1	en_CA	Sun	Kanada
819	S-1	ISO8859-1	CA	1	en_CA	Sun	Kanada
819	S-1	ISO-8859-1	CA	1	en_CA	Linux	Kanada
1252	S-1	1252	CA	1	-	WIN	Kanada
1275	S-1	1275	CA	1	-	Mac	Kanada
37	S-1	IBM-37	CA	1	-	HOST	Kanada
1140	S-1	IBM-1140	CA	1	-	HOST	Kanada
863	S-1	IBM-863	CA	2	-	OS2	Kanada (Französisch)
1381	D-4	IBM-1381	CN	86	-	OS2	China (VRC)
1386	D-4	GBK	CN	86	-	OS2	China (VRC)
1383	D-4	IBM-eucCN	CN	86	zh_CN	AIX	China (VRC)
1386	D-4	GBK	CN	86	Zh_CN.GBK	AIX	China (VRC)
1383	D-4	hp15CN	CN	86	zh_CN.hp15CN	HP	China (VRC)
1383	D-4	eucCN	CN	86	zh_CN	SCO	China (VRC)
1383	D-4	eucCN	CN	86	zh_CN.eucCN	SCO	China (VRC)
1383	D-4	gb2312	CN	86	zh	Sun	China (VRC)
1381	D-4	IBM-1381	CN	86	-	WIN	China (VRC)
1386	D-4	GBK	CN	86	-	WIN	China (VRC)
935	D-4	IBM-935	CN	86	-	HOST	China (VRC)
1388	D-4	IBM-1388	CN	86	-	HOST	China (VRC)
5488**	D-4		CN	86	-		China (PRC)

** Codepage 5488 kann nur mit den Dienstprogrammen LOAD oder IMPORT zum Versetzen von Daten aus Codepage 5488 in eine DB2-Unicode-Datenbank bzw. mit dem Dienstprogramm EXPORT zum Exportieren aus einer DB2-Unicode-Datenbank in Codepage 5488 verwendet werden. Weitere Informationen enthält der Abschnitt über das Versetzen von Daten in den Release-Informationen zu Version 7.2 FixPak 4.

Tabelle 28. Unterstützte Sprachen und codierte Zeichensätze (Forts.)

Code-page	Gruppe	Codierter Zeichens.	Geb.	Landes-/ Regional-code	Ländersp. Angaben	BS	Land/ Region
852	S-2	IBM-852	HR	385	-	OS2	Kroatien
912	S-2	ISO8859-2	HR	385	hr_HR	AIX	Kroatien
912	S-2	iso88592	HR	385	hr_HR.iso88592	HP	Kroatien
912	S-2	ISO8859-2	HR	385	hr_HR.ISO8859-2	SCO	Kroatien
912	S-2	ISO-8859-2	HR	385	hr_HR	Linux	Kroatien
1250	S-2	1250	HR	385	-	WIN	Kroatien
1282	S-2	1282	HR	385	-	Mac	Kroatien
870	S-2	IBM-870	HR	385	-	HOST	Kroatien
852	S-2	IBM-852	CZ	421	-	OS2	Tschechische Republik
912	S-2	ISO8859-2	CZ	421	cs_CZ	AIX	Tschechische Republik
912	S-2	iso88592	CZ	421	cs_CZ.iso88592	HP	Tschechische Republik
912	S-2	ISO8859-2	CZ	421	cs_CZ.ISO8859-2	SCO	Tschechische Republik
912	S-2	ISO-8859-2	CZ	421	cs_CZ	Linux	Tschechische Republik
1250	S-2	1250	CZ	421	-	WIN	Tschechische Republik
1282	S-2	1282	CZ	421	-	Mac	Tschechische Republik
870	S-2	IBM-870	CZ	421	-	HOST	Tschechische Republik
850	S-1	IBM-850	DK	45	-	OS2	Dänemark
819	S-1	ISO8859-1	DK	45	da_DK	AIX	Dänemark
850	S-1	IBM-850	DK	45	Da_DK	AIX	Dänemark
819	S-1	iso88591	DK	45	da_DK.iso88591	HP	Dänemark
1051	S-1	roman8	DK	45	da_DK.roman8	HP	Dänemark
819	S-1	ISO8859-1	DK	45	da	SCO	Dänemark
819	S-1	ISO8859-1	DK	45	da_DA	SCO	Dänemark
819	S-1	ISO8859-1	DK	45	da_DK	SCO	Dänemark
819	S-1	ISO8859-1	DK	45	da	Sun	Dänemark
819	S-1	ISO8859-1	DK	45	da	Sun	Dänemark
819	S-1	ISO-8859-1	DK	45	da_DK	Linux	Dänemark
1252	S-1	1252	DK	45	-	WIN	Dänemark
1275	S-1	1275	DK	45	-	Mac	Dänemark
277	S-1	IBM-277	DK	45	-	HOST	Dänemark
1142	S-1	IBM-1142	DK	45	-	HOST	Dänemark
922	S-10	IBM-922	EE	372	-	OS2	Estland
922	S-10	IBM-922	EE	372	Et_EE	AIX	Estland
1257	S-10	1257	EE	372	-	WIN	Estland
1122	S-10	IBM-1122	EE	372	-	HOST	Estland

Tabelle 28. Unterstützte Sprachen und codierte Zeichensätze (Forts.)

Code-page	Gruppe	Codierter Zeichens.	Geb.	Landes-/ Regional-code	Ländersp. Angaben	BS	Land/ Region
437	S-1	IBM-437	FI	358	-	OS2	Finnland
850	S-1	IBM-850	FI	358	-	OS2	Finnland
819	S-1	ISO8859-1	FI	358	fi_FI	AIX	Finnland
850	S-1	IBM-850	FI	358	Fi_FI	AIX	Finnland
819	S-1	iso88591	FI	358	fi_FI.iso88591	HP	Finnland
819	S-1	ISO8859-1	FI	358	fi	SCO	Finnland
819	S-1	ISO8859-1	FI	358	fi_FI	SCO	Finnland
819	S-1	ISO8859-1	FI	358	sv_FI	SCO	Finnland
819	S-1	ISO8859-1	FI	358	-	Sun	Finnland
819	S-1	ISO-8859-1	FI	358	fi_FI	Linux	Finnland
1051	S-1	roman8	FI	358	-	HP	Finnland
1252	S-1	1252	FI	358	-	WIN	Finnland
1275	S-1	1275	FI	358	-	Mac	Finnland
278	S-1	IBM-278	FI	358	-	HOST	Finnland
1143	S-1	IBM-1143	FI	358	-	HOST	Finnland
855	S-5	IBM-855	MK	389	-	OS2	Mazedonien
915	S-5	ISO8859-5	MK	389	-	OS2	Mazedonien
915	S-5	ISO8859-5	MK	389	mk_MK	AIX	Mazedonien
915	S-5	iso88595	MK	389	-	HP	Mazedonien
1251	S-5	1251	MK	389	-	WIN	Mazedonien
1283	S-5	1283	MK	389	-	Mac	Mazedonien
1025	S-5	IBM-1025	MK	389	-	HOST	Mazedonien
437	S-1	IBM-437	FR	33	-	OS2	Frankreich
850	S-1	IBM-850	FR	33	-	OS2	Frankreich
819	S-1	ISO8859-1	FR	33	fr_FR	AIX	Frankreich
850	S-1	IBM-850	FR	33	Fr_FR	AIX	Frankreich
819	S-1	iso88591	FR	33	fr_FR.iso88591	HP	Frankreich
1051	S-1	roman8	FR	33	fr_FR.roman8	HP	Frankreich
819	S-1	ISO8859-1	FR	33	fr	Sun	Frankreich
819	S-1	ISO8859-1	FR	33	fr	SCO	Frankreich
819	S-1	ISO8859-1	FR	33	fr_FR	SCO	Frankreich
819	S-1	ISO-8859-1	FR	33	fr_FR	Linux	Frankreich
1252	S-1	1252	FR	33	-	WIN	Frankreich
1275	S-1	1275	FR	33	-	Mac	Frankreich
297	S-1	IBM-297	FR	33	-	HOST	Frankreich
1147	S-1	IBM-1147	FR	33	-	HOST	Frankreich

Tabelle 28. Unterstützte Sprachen und codierte Zeichensätze (Forts.)

Code-page	Gruppe	Codierter Zeichens.	Geb.	Landes-/ Regional-code	Ländersp. Angaben	BS	Land/ Region
437	S-1	IBM-437	DE	49	-	OS2	Deutschland
850	S-1	IBM-850	DE	49	-	OS2	Deutschland
819	S-1	ISO8859-1	DE	49	de_DE	AIX	Deutschland
850	S-1	IBM-850	DE	49	De_DE	AIX	Deutschland
819	S-1	iso88591	DE	49	de_DE.iso88591	HP	Deutschland
1051	S-1	roman8	DE	49	de_DE.roman8	HP	Deutschland
819	S-1	ISO8859-1	DE	49	de	SCO	Deutschland
819	S-1	ISO8859-1	DE	49	de_DE	SCO	Deutschland
819	S-1	ISO8859-1	DE	49	de	Sun	Deutschland
819	S-1	ISO-8859-1	DE	49	de_DE	Linux	Deutschland
1252	S-1	1252	DE	49	-	WIN	Deutschland
1275	S-1	1275	DE	49	-	Mac	Deutschland
273	S-1	IBM-273	DE	49	-	HOST	Deutschland
1141	S-1	IBM-1141	DE	49	-	HOST	Deutschland
819	S-1	ISO8859-1	DE	49	De_DE.88591	SINIX	Deutschland
819	S-1	ISO8859-1	DE	49	De_DE.6937	SINIX	Deutschland
813	S-7	ISO8859-7	GR	30	-	OS2	Griechenland
869	S-7	IBM-869	GR	30	-	OS2	Griechenland
813	S-7	ISO8859-7	GR	30	el_GR	AIX	Griechenland
813	S-7	iso88597	GR	30	el_GR.iso88597	HP	Griechenland
813	S-7	ISO8859-7	GR	30	el_GR.ISO8859-7	SCO	Griechenland
813	S-7	ISO-8859-7	GR	30	gr_GR	Linux	Griechenland
737	S-7	737	GR	30	-	WIN	Griechenland
1253	S-7	1253	GR	30	-	WIN	Griechenland
1280	S-7	1280	GR	30	-	Mac	Griechenland
423	S-7	IBM-423	GR	30	-	HOST	Griechenland
875	S-7	IBM-875	GR	30	-	HOST	Griechenland
852	S-2	IBM-852	HU	36	-	OS2	Ungarn
912	S-2	ISO8859-2	HU	36	hu_HU	AIX	Ungarn
912	S-2	iso88592	HU	36	hu_HU.iso88592	HP	Ungarn
912	S-2	ISO8859-2	HU	36	hu_HU.ISO8859-2	SCO	Ungarn
912	S-2	ISO-8859-2	HU	36	hu_HU	Linux	Ungarn
1250	S-2	1250	HU	36	-	WIN	Ungarn
1282	S-2	1282	HU	36	-	Mac	Ungarn
870	S-2	IBM-870	HU	36	-	HOST	Ungarn

Tabelle 28. Unterstützte Sprachen und codierte Zeichensätze (Forts.)

Code-page	Gruppe	Codierter Zeichens.	Geb.	Landes-/ Regional-code	Ländersp. Angaben	BS	Land/ Region
850	S-1	IBM-850	IS	354	-	OS2	Island
819	S-1	ISO8859-1	IS	354	is_IS	AIX	Island
850	S-1	IBM-850	IS	354	Is_IS	AIX	Island
819	S-1	iso88591	IS	354	is_IS.iso88591	HP	Island
1051	S-1	roman8	IS	354	is_IS.roman8	HP	Island
819	S-1	ISO8859-1	IS	354	is	SCO	Island
819	S-1	ISO8859-1	IS	354	is_IS	SCO	Island
819	S-1	ISO8859-1	IS	354	-	Sun	Island
819	S-1	ISO-8859-1	IS	354	is_IS	Linux	Island
1252	S-1	1252	IS	354	-	WIN	Island
1275	S-1	1275	IS	354	-	Mac	Island
871	S-1	IBM-871	IS	354	-	HOST	Island
1149	S-1	IBM-1149	IS	354	-	HOST	Island
437	S-1	IBM-437	IE	353	-	OS2	Irland
850	S-1	IBM-850	IE	353	-	OS2	Irland
819	S-1	ISO8859-1	IE	353	en_IE	AIX	Irland
850	S-1	IBM-850	IE	353	En_IE	AIX	Irland
819	S-1	iso88591	IE	353	-	HP	Irland
1051	S-1	roman8	IE	353	-	HP	Irland
819	S-1	ISO8859-1	IE	353	en_IE	Sun	Irland
819	S-1	ISO8859-1	IE	353	en_IE.ISO8859-1	SCO	Irland
819	S-1	ISO-8859-1	IE	353	en_IE	Linux	Irland
1252	S-1	1252	IE	353	-	WIN	Irland
1275	S-1	1275	IE	353	-	Mac	Irland
285	S-1	IBM-285	IE	353	-	HOST	Irland
1146	S-1	IBM-1146	IE	353	-	HOST	Irland
806	S-12	IBM-806	IN	91	hi_IN	-	Indien
1137	S-12	IBM-1137	IN	91	-	HOST	Indien
862	S-8	IBM-862	IL	972	-	OS2	Israel
916	S-8	ISO8859-8	IL	972	iw_IL	AIX	Israel
856	S-8	IBM-856	IL	972	Iw_IL	AIX	Israel
916	S-8	ISO-8859-8	IL	972	iw_IL	Linux	Israel
1255	S-8	1255	IL	972	-	WIN	Israel
424	S-8	IBM-424	IL	972	-	HOST	Israel

Tabelle 28. Unterstützte Sprachen und codierte Zeichensätze (Forts.)

Code-page	Gruppe	Codierter Zeichens.	Geb.	Landes-/ Regional-code	Ländersp. Angaben	BS	Land/ Region
437	S-1	IBM-437	IT	39	-	OS2	Italien
850	S-1	IBM-850	IT	39	-	OS2	Italien
819	S-1	ISO8859-1	IT	39	it_IT	AIX	Italien
850	S-1	IBM-850	IT	39	It_IT	AIX	Italien
819	S-1	iso88591	IT	39	it_IT.iso88591	HP	Italien
1051	S-1	roman8	IT	39	it_IT.roman8	HP	Italien
819	S-1	ISO8859-1	IT	39	it	SCO	Italien
819	S-1	ISO8859-1	IT	39	it_IT	SCO	Italien
819	S-1	ISO8859-1	IT	39	it	Sun	Italien
819	S-1	ISO-8859-1	IT	39	it_IT	Linux	Italien
1252	S-1	1252	IT	39	-	WIN	Italien
1275	S-1	1275	IT	39	-	Mac	Italien
280	S-1	IBM-280	IT	39	-	HOST	Italien
1144	S-1	IBM-1144	IT	39	-	HOST	Italien
932	D-1	IBM-932	JP	81	-	OS2	Japan
942	D-1	IBM-942	JP	81	-	OS2	Japan
943	D-1	IBM-943	JP	81	-	OS2	Japan
954	D-1	IBM-eucJP	JP	81	ja_JP	AIX	Japan
943*	D-1	IBM-943	JP	81	Ja_JP	AIX	Japan
954	D-1	eucJP	JP	81	ja_JP.eucJP	HP	Japan
5039	D-1	SJIS	JP	81	ja_JP.SJIS	HP	Japan
954	D-1	eucJP	JP	81	ja	SCO	Japan
954	D-1	eucJP	JP	81	ja_JP	SCO	Japan
954	D-1	eucJP	JP	81	ja_JP.EUC	SCO	Japan
954	D-1	eucJP	JP	81	ja_JP.eucJP	SCO	Japan
954	D-1	eucJP	JP	81	ja	Sun	Japan
943	D-1	IBM-943	JP	81	ja_JP.PCK	Sun	Japan
954	D-1	EUC-JP	JP	81	ja_JP	Linux	Japan
943	D-1	IBM-943	JP	81	-	WIN	Japan
930	D-1	IBM-930	JP	81	-	HOST	Japan
939	D-1	IBM-939	JP	81	-	HOST	Japan
5026	D-1	IBM-5026	JP	81	-	HOST	Japan
5035	D-1	IBM-5035	JP	81	-	HOST	Japan
1390	D-1		JP	81	-	HOST	Japan
1399	D-1		JP	81	-	HOST	Japan
1394**	D-1		JP	81	-		Japan

* Unter AIX 4.3 oder einer späteren Version ist die Codepage 943. Bei Verwendung von AIX 4.2 oder einer früheren Version ist die Codepage 932.
** Codepage 1394 kann nur mit den Dienstprogrammen LOAD oder IMPORT zum Versetzen von Daten aus Codepage 1394 in eine DB2-Unicode-Datenbank bzw. mit dem Dienstprogramm EXPORT zum Exportieren aus einer DB2-Unicode-Datenbank in Codepage 1394 verwendet werden. Weitere Informationen enthält der Abschnitt über das Versetzen von Daten in den Release-Informationen zu Version 7.2 FixPak 4.

Tabelle 28. Unterstützte Sprachen und codierte Zeichensätze (Forts.)

Code-page	Gruppe	Codierter Zeichens.	Geb.	Landes-/ Regional-code	Ländersp. Angaben	BS	Land/ Region
949	D-3	IBM-949	KR	82	-	OS2	Korea, Süd-
970	D-3	IBM-eucKR	KR	82	ko_KR	AIX	Korea, Süd-
970	D-3	eucKR	KR	82	ko_KR.eucKR	HP	Korea, Süd-
970	D-3	eucKR	KR	82	ko_KR.eucKR	SGI	Korea, Süd-
970	D-3	5601	KR	82	ko	Sun	Korea, Süd-
1363	D-3	1363	KR	82	-	WIN	Korea, Süd-
933	D-3	IBM-933	KR	82	-	HOST	Korea, Süd-
1364	D-3	IBM-1364	KR	82	-	HOST	Korea, Süd-
437	S-1	IBM-437	Lat	3	-	OS2	Lateinamerika
850	S-1	IBM-850	Lat	3	-	OS2	Lateinamerika
819	S-1	ISO8859-1	Lat	3	-	AIX	Lateinamerika
850	S-1	IBM-850	Lat	3	-	AIX	Lateinamerika
819	S-1	iso88591	Lat	3	-	HP	Lateinamerika
819	S-1	ISO8859-1	Lat	3	-	Sun	Lateinamerika
819	S-1	ISO-8859-1	Lat	3	-	Linux	Lateinamerika
1051	S-1	roman8	Lat	3	-	HP	Lateinamerika
1252	S-1	1252	Lat	3	-	WIN	Lateinamerika
1275	S-1	1275	Lat	3	-	Mac	Lateinamerika
284	S-1	IBM-284	Lat	3	-	HOST	Lateinamerika
1145	S-1	IBM-1145	Lat	3	-	HOST	Lateinamerika
921	S-10	IBM-921	LV	371	-	OS2	Lettland
921	S-10	IBM-921	LV	371	Lv_LV	AIX	Lettland
1257	S-10	1257	LV	371	-	WIN	Lettland
1112	S-10	IBM-1112	LV	371	-	HOST	Lettland
921	S-10	IBM-921	LT	370	-	OS2	Litauen
921	S-10	IBM-921	LT	370	Lt_LT	AIX	Litauen
1257	S-10	1257	LT	370	-	WIN	Litauen
1112	S-10	IBM-1112	LT	370	-	HOST	Litauen
437	S-1	IBM-437	NL	31	-	OS2	Niederlande
850	S-1	IBM-850	NL	31	-	OS2	Niederlande
819	S-1	ISO8859-1	NL	31	nl_NL	AIX	Niederlande
850	S-1	IBM-850	NL	31	Nl_NL	AIX	Niederlande
819	S-1	iso88591	NL	31	nl_NL.iso88591	HP	Niederlande
1051	S-1	roman8	NL	31	nl_NL.roman8	HP	Niederlande
819	S-1	ISO8859-1	NL	31	nl	SCO	Niederlande
819	S-1	ISO8859-1	NL	31	nl_NL	SCO	Niederlande
819	S-1	ISO8859-1	NL	31	nl	Sun	Niederlande
819	S-1	ISO-8859-1	NL	31	nl_NL	Linux	Niederlande
1252	S-1	1252	NL	31	-	WIN	Niederlande
1275	S-1	1275	NL	31	-	Mac	Niederlande
37	S-1	IBM-37	NL	31	-	HOST	Niederlande
1140	S-1	IBM-1140	NL	31	-	HOST	Niederlande

Tabelle 28. Unterstützte Sprachen und codierte Zeichensätze (Forts.)

Code-page	Gruppe	Codierter Zeichens.	Geb.	Landes-/ Regional-code	Ländersp. Angaben	BS	Land/ Region
850	S-1	IBM-850	NZ	64	-	OS2	Neuseeland
850	S-1	IBM-850	NZ	64	En_NZ	AIX	Neuseeland
819	S-1	ISO8859-1	NZ	64	en_NZ	AIX	Neuseeland
819	S-1	ISO8859-1	NZ	64	-	HP	Neuseeland
819	S-1	ISO8859-1	NZ	64	en_NZ	SCO	Neuseeland
819	S-1	ISO8859-1	NZ	64	en_NZ	Sun	Neuseeland
1252	S-1	1252	NZ	64	-	WIN	Neuseeland
37	S-1	IBM-37	NZ	64	-	HOST	Neuseeland
1140	S-1	IBM-1140	NZ	64	-	HOST	Neuseeland
850	S-1	IBM-850	NO	47	-	OS2	Norwegen
819	S-1	ISO8859-1	NO	47	no_NO	AIX	Norwegen
850	S-1	IBM-850	NO	47	No_NO	AIX	Norwegen
819	S-1	iso88591	NO	47	no_NO.iso88591	HP	Norwegen
1051	S-1	roman8	NO	47	no_NO.roman8	HP	Norwegen
819	S-1	ISO8859-1	NO	47	no	SCO	Norwegen
819	S-1	ISO8859-1	NO	47	no_NO	SCO	Norwegen
819	S-1	ISO8859-1	NO	47	no	Sun	Norwegen
819	S-1	ISO-8859-1	NO	47	no_NO	Linux	Norwegen
1252	S-1	1252	NO	47	-	WIN	Norwegen
1275	S-1	1275	NO	47	-	Mac	Norwegen
277	S-1	IBM-277	NO	47	-	HOST	Norwegen
1142	S-1	IBM-1142	NO	47	-	HOST	Norwegen
852	S-2	IBM-852	PL	48	-	OS2	Polen
912	S-2	ISO8859-2	PL	48	pl_PL	AIX	Polen
912	S-2	iso88592	PL	48	pl_PL.iso88592	HP	Polen
912	S-2	ISO8859-2	PL	48	pl_PL.ISO8859-2	SCO	Polen
912	S-2	ISO-8859-2	PL	48	pl_PL	Linux	Polen
1250	S-2	1250	PL	48	-	WIN	Polen
1282	S-2	1282	PL	48	-	Mac	Polen
870	S-2	IBM-870	PL	48	-	HOST	Polen
860	S-1	IBM-860	PT	351	-	OS2	Portugal
850	S-1	IBM-850	PT	351	-	OS2	Portugal
819	S-1	ISO8859-1	PT	351	pt_PT	AIX	Portugal
850	S-1	IBM-850	PT	351	Pt_PT	AIX	Portugal
819	S-1	iso88591	PT	351	pt_PT.iso88591	HP	Portugal
1051	S-1	roman8	PT	351	pt_PT.roman8	HP	Portugal
819	S-1	ISO8859-1	PT	351	pt	SCO	Portugal
819	S-1	ISO8859-1	PT	351	pt_PT	SCO	Portugal
819	S-1	ISO8859-1	PT	351	pt	Sun	Portugal
819	S-1	ISO-8859-1	PT	351	pt_PT	Linux	Portugal
1252	S-1	1252	PT	351	-	WIN	Portugal
1275	S-1	1275	PT	351	-	Mac	Portugal
37	S-1	IBM-37	PT	351	-	HOST	Portugal
1140	S-1	IBM-1140	PT	351	-	HOST	Portugal

Tabelle 28. Unterstützte Sprachen und codierte Zeichensätze (Forts.)

Code-page	Gruppe	Codierter Zeichens.	Geb.	Landes-/ Regional-code	Ländersp. Angaben	BS	Land/ Region
852	S-2	IBM-852	RO	40	-	OS2	Rumänien
912	S-2	ISO8859-2	RO	40	ro_RO	AIX	Rumänien
912	S-2	iso88592	RO	40	ro_RO.iso88592	HP	Rumänien
912	S-2	ISO8859-2	RO	40	ro_RO.ISO8859-2	SCO	Rumänien
912	S-2	ISO-8859-2	RO	40	ro_RO	Linux	Rumänien
1250	S-2	1250	RO	40	-	WIN	Rumänien
1282	S-2	1282	RO	40	-	Mac	Rumänien
870	S-2	IBM-870	RO	40	-	HOST	Rumänien
866	S-5	IBM-866	RU	7	-	OS2	Russland
915	S-5	ISO8859-5	RU	7	-	OS2	Russland
915	S-5	ISO8859-5	RU	7	ru_RU	AIX	Russland
915	S-5	iso88595	RU	7	ru_RU.iso88595	HP	Russland
915	S-5	ISO8859-5	RU	7	ru_RU.ISO8859-5	SCO	Russland
915	S-5	ISO-8859-5	RU	7	ru_RU	Linux	Russland
1251	S-5	1251	RU	7	-	WIN	Russland
1283	S-5	1283	RU	7	-	Mac	Russland
1025	S-5	IBM-1025	RU	7	-	HOST	Russland
855	S-5	IBM-855	SP	381	-	OS2	Serbien/Montenegro
915	S-5	ISO8859-5	SP	381	-	OS2	Serbien/Montenegro
915	S-5	ISO8859-5	SP	381	sr_SP	AIX	Serbien/Montenegro
915	S-5	iso88595	SP	381	-	HP	Serbien/Montenegro
1251	S-5	1251	SP	381	-	WIN	Serbien/Montenegro
1283	S-5	1283	SP	381	-	Mac	Serbien/Montenegro
1025	S-5	IBM-1025	SP	381	-	HOST	Serbien/Montenegro
852	S-2	IBM-852	SK	422	-	OS2	Slowakei
912	S-2	ISO8859-2	SK	422	sk_SK	AIX	Slowakei
912	S-2	iso88592	SK	422	sk_SK.iso88592	HP	Slowakei
912	S-2	ISO8859-2	SK	422	sk_SK.ISO8859-2	SCO	Slowakei
1250	S-2	1250	SK	422	-	WIN	Slowakei
1282	S-2	1282	SK	422	-	Mac	Slowakei
870	S-2	IBM-870	SK	422	-	HOST	Slowakei
852	S-2	IBM-852	SI	386	-	OS2	Slowenien
912	S-2	ISO8859-2	SI	386	sl_SI	AIX	Slowenien
912	S-2	iso88592	SI	386	sl_SI.iso88592	HP	Slowenien
912	S-2	ISO8859-2	SI	386	sl_SI.ISO8859-2	SCO	Slowenien
912	S-2	ISO-8859-2	SI	386	sl_SI	Linux	Slowenien
1250	S-2	1250	SI	386	-	WIN	Slowenien
1282	S-2	1282	SI	386	-	Mac	Slowenien
870	S-2	IBM-870	SI	386	-	HOST	Slowenien

Tabelle 28. Unterstützte Sprachen und codierte Zeichensätze (Forts.)

Code-page	Gruppe	Codierter Zeichens.	Geb.	Landes-/ Regional-code	Ländersp. Angaben	BS	Land/ Region
437	S-1	IBM-437	ZA	27	-	OS2	Südafrika
850	S-1	IBM-850	ZA	27	-	OS2	Südafrika
819	S-1	ISO8859-1	ZA	27	en_ZA	AIX	Südafrika
850	S-1	IBM-850	ZA	27	En_ZA	AIX	Südafrika
819	S-1	iso88591	ZA	27	-	HP	Südafrika
1051	S-1	roman8	ZA	27	-	HP	Südafrika
819	S-1	ISO8859-1	ZA	27	-	Sun	Südafrika
819	S-1	ISO8859-1	ZA	27	en_ZA.ISO8859-1	SCO	Südafrika
1252	S-1	1252	ZA	27	-	WIN	Südafrika
1275	S-1	1275	ZA	27	-	Mac	Südafrika
285	S-1	IBM-285	ZA	27	-	HOST	Südafrika
1146	S-1	IBM-1146	ZA	27	-	HOST	Südafrika
437	S-1	IBM-437	ES	34	-	OS2	Spanien
850	S-1	IBM-850	ES	34	-	OS2	Spanien
819	S-1	ISO8859-1	ES	34	es_ES	AIX	Spanien
850	S-1	IBM-850	ES	34	Es_ES	AIX	Spanien
819	S-1	iso88591	ES	34	es_ES.iso88591	HP	Spanien
1051	S-1	roman8	ES	34	es_ES.roman8	HP	Spanien
819	S-1	ISO8859-1	ES	34	es	Sun	Spanien
819	S-1	ISO8859-1	ES	34	es	SCO	Spanien
819	S-1	ISO8859-1	ES	34	es_ES	SCO	Spanien
819	S-1	ISO-8859-1	ES	34	es_ES	Linux	Spanien
1252	S-1	1252	ES	34	-	WIN	Spanien
1275	S-1	1275	ES	34	-	Mac	Spanien
284	S-1	IBM-284	ES	34	-	HOST	Spanien
1145	S-1	IBM-1145	ES	34	-	HOST	Spanien
437	S-1	IBM-437	SE	46	-	OS2	Schweden
850	S-1	IBM-850	SE	46	-	OS2	Schweden
819	S-1	ISO8859-1	SE	46	sv_SE	AIX	Schweden
850	S-1	IBM-850	SE	46	Sv_SE	AIX	Schweden
819	S-1	iso88591	SE	46	sv_SE.iso88591	HP	Schweden
1051	S-1	roman8	SE	46	sv_SE.roman8	HP	Schweden
819	S-1	ISO8859-1	SE	46	sv	SCO	Schweden
819	S-1	ISO8859-1	SE	46	sv_SE	SCO	Schweden
819	S-1	ISO8859-1	SE	46	sv	Sun	Schweden
819	S-1	ISO-8859-1	SE	46	sv_SE	Linux	Schweden
1252	S-1	1252	SE	46	-	WIN	Schweden
1275	S-1	1275	SE	46	-	Mac	Schweden
278	S-1	IBM-278	SE	46	-	HOST	Schweden
1143	S-1	IBM-1143	SE	46	-	HOST	Schweden

Tabelle 28. Unterstützte Sprachen und codierte Zeichensätze (Forts.)

Code-page	Gruppe	Codierter Zeichens.	Geb.	Landes-/ Regional-code	Ländersp. Angaben	BS	Land/ Region
437	S-1	IBM-437	CH	41	-	OS2	Schweiz
850	S-1	IBM-850	CH	41	-	OS2	Schweiz
819	S-1	ISO8859-1	CH	41	de_CH	AIX	Schweiz
850	S-1	IBM-850	CH	41	De_CH	AIX	Schweiz
819	S-1	iso88591	CH	41	-	HP	Schweiz
1051	S-1	roman8	CH	41	-	HP	Schweiz
819	S-1	ISO8859-1	CH	41	de_CH	SCO	Schweiz
819	S-1	ISO8859-1	CH	41	fr_CH	SCO	Schweiz
819	S-1	ISO8859-1	CH	41	it_CH	SCO	Schweiz
819	S-1	ISO8859-1	CH	41	de_CH	Sun	Schweiz
819	S-1	ISO-8859-1	CH	41	de_CH	Linux	Schweiz
1252	S-1	1252	CH	41	-	WIN	Schweiz
1275	S-1	1275	CH	41	-	Mac	Schweiz
500	S-1	IBM-500	CH	41	-	HOST	Schweiz
1148	S-1	IBM-1148	CH	41	-	HOST	Schweiz
938	D-2	IBM-938	TW	88	-	OS2	Taiwan
948	D-2	IBM-948	TW	88	-	OS2	Taiwan
950	D-2	big5	TW	88	-	OS2	Taiwan
950	D-2	big5	TW	88	Zh_TW	AIX	Taiwan
964	D-2	IBM-eucTW	TW	88	zh_TW	AIX	Taiwan
950	D-2	big5	TW	88	zh_TW.big5	HP	Taiwan
964	D-2	eucTW	TW	88	zh_TW.eucTW	HP	Taiwan
950	D-2	big5	TW	88	big5	Sun	Taiwan
964	D-2	cns11643	TW	88	zh_TW	Sun	Taiwan
950	D-2	big5	TW	88	-	WIN	Taiwan
937	D-2	IBM-937	TW	88	-	HOST	Taiwan
874	S-20	TIS620-1	TH	66	-	OS2	Thailand
874	S-20	TIS620-1	TH	66	Th_TH	AIX	Thailand
874	S-20	tis620	TH	66	th_TH.tis620	HP	Thailand
874	S-20	TIS620-1	TH	66	-	WIN	Thailand
838	S-20	IBM-838	TH	66	-	HOST	Thailand
857	S-9	IBM-857	TR	90	-	OS2	Türkei
920	S-9	ISO8859-9	TR	90	tr_TR	AIX	Türkei
920	S-9	iso88599	TR	90	tr_TR.iso88599	HP	Türkei
920	S-9	ISO8859-9	TR	90	tr_TR.ISO8859-9	SCO	Türkei
920	S-9	ISO-8859-9	TR	90	tr_TR	Linux	Türkei
1254	S-9	1254	TR	90	-	WIN	Türkei
1281	S-9	1281	TR	90	-	Mac	Türkei
1026	S-9	IBM-1026	TR	90	-	HOST	Türkei

Tabelle 28. Unterstützte Sprachen und codierte Zeichensätze (Forts.)

Code-page	Gruppe	Codierter Zeichens.	Geb.	Landes-/ Regional-code	Ländersp. Angaben	BS	Land/ Region
437	S-1	IBM-437	GB	44	-	OS2	GB
850	S-1	IBM-850	GB	44	-	OS2	GB
819	S-1	ISO8859-1	GB	44	en_GB	AIX	GB
850	S-1	IBM-850	GB	44	En_GB	AIX	GB
819	S-1	iso88591	GB	44	en_GB.iso88591	HP	GB
1051	S-1	roman8	GB	44	en_GB.roman8	HP	GB
819	S-1	ISO8859-1	GB	44	en_UK	Sun	GB
819	S-1	ISO8859-1	GB	44	en_GB	SCO	GB
819	S-1	ISO8859-1	GB	44	en	SCO	GB
819	S-1	ISO-8859-1	GB	44	en_GB	Linux	GB
1252	S-1	1252	GB	44	-	WIN	GB
1275	S-1	1275	GB	44	-	Mac	GB
285	S-1	IBM-285	GB	44	-	HOST	GB
1146	S-1	IBM-1146	GB	44	-	HOST	GB
819	S-1	88591	GB	44	En_GB.88591	SINIX	GB
819	S-1	ISO8859-1	GB	44	En_GB.6937	SINIX	GB
1125	S-5	IBM-1125	UA	380	-	OS2	Ukraine
1124	S-5	IBM-1124	UA	380	uk_UA	AIX	Ukraine
1251	S-5	1251	UA	380	-	WIN	Ukraine
1123	S-5	IBM-1123	UA	380	-	HOST	Ukraine
437	S-1	IBM-437	US	1	-	OS2	USA
850	S-1	IBM-850	US	1	-	OS2	USA
819	S-1	ISO8859-1	US	1	en_US	AIX	USA
850	S-1	IBM-850	US	1	En_US	AIX	USA
819	S-1	iso88591	US	1	en_US.iso88591	HP	USA
1051	S-1	roman8	US	1	en_US.roman8	HP	USA
819	S-1	ISO8859-1	US	1	en_US	Sun	USA
819	S-1	ISO8859-1	US	1	en_US	SGI	USA
819	S-1	ISO8859-1	US	1	en_US	SCO	USA
819	S-1	ISO-8859-1	US	1	en_US	Linux	USA
1252	S-1	1252	US	1	-	WIN	USA
1275	S-1	1275	US	1	-	Mac	USA
37	S-1	IBM-37	US	1	-	HOST	USA
1140	S-1	IBM-1140	US	1	-	HOST	USA
1163	S-11	IBM-1163	VN	84	-	OS2	Vietnam
1163	S-11	IBM-1163	VN	84	vi_VN	AIX	Vietnam
1258	S-11	1258	VN	84	-	WIN	Vietnam
1164	S-11	IBM-1164	VN	84	-	HOST	Vietnam

Tabelle 28. Unterstützte Sprachen und codierte Zeichensätze (Forts.)

Code-page	Gruppe	Codierter Zeichens.	Landes-/ Regional- Geb. code	Ländersp. Angaben	BS	Land/ Region

---

```
Übersicht der arabischen Länder (AA):
-----------------------------------------
    /* Arabisch (Saudi-Arabien) */
    /* Arabisch (Irak) */
    /* Arabisch (Ägypten) */
    /* Arabisch (Libyen) */
    /* Arabisch (Algerien) */
    /* Arabisch (Marokko) */
    /* Arabisch (Tunesien) */
    /* Arabisch (Oman) */
    /* Arabisch (Jemen) */
    /* Arabisch (Syrien) */
    /* Arabisch (Jordanien) */
    /* Arabisch (Libanon) */
    /* Arabisch (Kuwait) */
    /* Arabisch (Vereinigte Arabische Emirate) */
    /* Arabisch (Bahrain) */
    /* Arabisch (Katar) */
```

```
Übersicht über Englisch (US):
--------------------------------
    /* Englisch (Jamaika) */
    /* Englisch (Karibik) */
```

```
Übersicht über Lateinamerika (Lat):
---------------------------------------
   /* Spanisch (Mexiko) */
   /* Spanisch (Guatemala) */
   /* Spanisch (Costa Rica) */
   /* Spanisch (Panama) */
   /* Spanisch (Dominikanische Republik) */
   /* Spanisch (Venezuela) */
   /* Spanisch (Kolumbien) */
   /* Spanisch (Peru) */
   /* Spanisch (Argentinien) */
   /* Spanisch (Ecuador) */
   /* Spanisch (Chile) */
   /* Spanisch (Uruguay) */
   /* Spanisch (Paraguay) */
   /* Spanisch (Bolivien) */
```

Anmerkungen:

1. Von der Codepage 950 für Solaris werden die folgenden Zeichen in IBM 950 nicht unterstützt:

Codebereich	Beschreibung	Sun Big-5	IBM Big-5
C6A1-C8FE	Symbole	Reservierter Bereich	Symbole
F9D6-F9FE	Erweiterung ETen	Reservierter Bereich	Erweiterung ETen
F286-F9A0	Von IBM ausgewählte Zeichen	Reservierter Bereich	Von IBM ausgewählt

2. Die Unterstützung für das Euro-Symbol ist in dieser Version von DB2 UDB integriert. Die ANSI-Codepages von Microsoft Windows werden gemäß der neuesten Definition von Microsoft geändert. Diese enthalten nun Euro-Symbol an Position 0x80. Diese Position war bisher nicht definiert. Zusätzlich wurde die Definition von Codepage 850 geändert. Diese enthält anstelle des Zeichens ″i ohne Punkt″ (an Position 0xD5) das Euro-Symbol. DB2 UDB verwendet die neuen Definitionen dieser Codepages als Standardeinstellung für die Unterstützung des Euro-Symbols. Diese Implementierung ist der geeignete Standardwert für die derzeitigen DB2 UDB Kunden, die die Unterstützung des Euro-Symbols benötigen. Für Kunden, die das Euro-Symbol nicht benötigen, sollten sich keine Änderungen ergeben. Wenn Sie jedoch die bisherigen Definitionen dieser Codepages weiter verwenden wollen, können Sie die folgenden Dateien:
   - 12520850.cnv
   - 08501252.cnv
   - IBM00850.ucs
   - IBM01252.ucs

   aus dem Verzeichnis

   ```
   sqllib/conv/alt/
   ```

   in das Verzeichnis

   ```
   sqllib/conv/
   ```

   nach Ende der Installation kopieren. Es empfiehlt sich, die vorhandenen Dateien `IBM01252.usc` und `IBM00850.ucs` zu sichern, bevor Sie die Versionen ohne Unterstützung des Euro-Symbols darüber kopieren. Nach dem Kopieren der Dateien unterstützt DB2 UDB das Euro-Symbol nicht mehr.

Ländereinstellungen für den DB2-Verwaltungsserver

Stellen Sie sicher, dass die Ländereinstellung (Locale) des DB2-Verwaltungsserverexemplars mit der Ländereinstellung des DB2-Exemplars kompatibel ist. Ansonsten kann das DB2-Exemplar nicht mit dem DB2-Verwaltungsserver kommunizieren.

Wenn die Umgebungsvariable LANG im Benutzerprofil des DB2-Verwaltungsservers nicht definiert ist, wird der DB2-Verwaltungsserver mit der Standardländereinstellung des Systems gestartet. Falls die Standardländereinstellung des Systems nicht definiert ist, wird der DB2-Verwaltungsserver mit Codepage 819 gestartet. Wenn das DB2-Exemplar mit einer der DBCS-Ländereinstel- .br lungen arbeitet und der DB2-Verwaltungsserver mit Codepage 819 gestartet wird, ist das Exemplar nicht in der Lage, mit dem DB2-Verwaltungsserver zu kommunizieren. Die Ländereinstellung des DB2-Verwaltungsservers und die Ländereinstellung des DB2-Exemplars müssen kompatibel sein.

Zum Beispiel sollte für ein Linux-System mit vereinfachtem Chinesisch im Benutzerprofil des DB2-Verwaltungsservers die Variable `LANG=zh_CN` definiert sein.

Unterstützte übersetzte Locales für UNIX-basierte Plattformen

Auf UNIX-basierten Plattformen können die Produktnachrichten und die Bibliothek von DB2 in mehreren verschiedenen Sprachen installiert werden. Das DB2-Installationsdienstprogramm legt die Dateigruppen der Nachrichtenkataloge im gängigsten Locale-Verzeichnis für eine bestimmte Plattform ab, wie den folgenden Tabellen zu entnehmen ist. Tabelle 29 auf Seite 275 enthält Informationen zu AIX, HP-UX und Solaris. Tabelle 30 auf Seite 276 enthält Informationen zu Linux, Linux/390, SGI und Dynix.

Wenn Ihr System die gleichen Codepages, jedoch andere Locale-Namen als die in Tabelle 29 auf Seite 275 und Tabelle 30 auf Seite 276 gezeigten verwendet, können Sie die übersetzten Nachrichten immer noch anzeigen, indem Sie eine Verbindung (Link) zum entsprechenden Nachrichtenverzeichnis herstellen.

Wenn das Standard-Locale Ihrer AIX-Maschine `ja_JP.IBM-eucJP` und die Codepage `ja_JP.IBM-eucJP` 954 ist, können Sie eine Verbindung von `/usr/lpp/db2_07_01/msg/ja_JP.IBM-eucJP` zu `/usr/lpp/db2_07_01/msg/ja_JP` erstellen, indem Sie den folgenden Befehl ausführen:

ln -s /usr/lpp/db2_07_01/msg/ja_JP /usr/lpp/db2_07_01/msg/ja_JP.IBM-eucJP

Nach der Ausführung dieses Befehls werden alle DB2-Nachrichten in Japanisch angezeigt.

Tabelle 29. Verzeichnis-Locales für AIX, HP-UX, Solaris

	AIX		**HP-UX**		**Solaris**	
Sprache	**Locale**	**Code-page**	**Locale**	**Code-page**	**Locale**	**Code-page**
Französisch	fr_FR Fr_FR	819 850	fr_FR.iso88591 fr_FR.roman8	819 1051	fr	819
Deutsch	de_DE De_DE	819 850	de_DE.iso88591 de_DE.roman8	819 1051	de	819
Italienisch	it_IT It_IT	819 850	it_IT.iso88591 it_IT.roman8	819 1051	it	819
Spanisch	es_ES Es_ES	819 850	es_ES.iso88591 es_ES.roman8	819 1051	es	819
Brasilianisches Portugiesisch	pt_BR	819			pt_BR	819
Japanisch	ja_JP Ja_JP	954 932	ja_JP.eucJP	954	ja	954
Koreanisch	ko_KR	970	ko_KR.eucKR	970	ko	970
Vereinfachtes Chinesisch	zh_CN Zh_ CN.GBK	1383 1386	zh_CN.hp15CN	1383	zh	1383
Traditionelles Chinesisch	zh_TW Zh_TW	964 950	zh_TW.eucTW zh_TW.big5	964 950	zh_TW zh_TW.BIG5	964 950
Dänisch	da_DK Da_DK	819 850	da_DK.iso88591 da_DK.roman8	819 1051	da	819
Finnisch	fi_FI Fi_FI	819 850	fi_FI.iso88591 fi_FI.roman8	819 1051	fi	819
Norwegisch	no_NO No_NO	819 850	no_NO.iso88591 no_NO.roman8	819 1051	no	819
Schwedisch	sv_SE Sv_SE	819 850	sv_SE.iso88591 sv_SE.roman8	819 1051	sv	819
Tschechisch	cs_CZ	912				

Tabelle 29. Verzeichnis-Locales für AIX, HP-UX, Solaris (Forts.)

	AIX		HP-UX		Solaris	
Sprache	**Locale**	**Code-page**	**Locale**	**Code-page**	**Locale**	**Code-page**
Ungarisch	hu_HU	912				
Polnisch	pl_PL	912				
Niederländisch	nl_NL Nl_NL	819 850				
Türkisch	tr_TR	920				
Russisch	ru_RU	915				
Bulgarisch	bg_BG	915	bg_BG.iso88595	915		
Slowenisch	sl_SI	912	sl_SI.iso88592	912	sl_SI	912

Tabelle 30. Verzeichnis-Locales für Linux, Linux/390, SGI, Dynix

	Linux		Linux/390		SGI		Dynix	
Sprache	**Locale**	**Code-page**	**Locale**	**Code-page**	**Locale**	**Code-page**	**Locale**	**Code-page**
Französisch	fr	819	fr	819			fr	819
Deutsch	de	819	de	819			de	819
Italienisch							es	819
Spanisch								
Brasilianisches Portugiesisch								
Japanisch	ja_JP.ujis	954	ja_JP.ujis	954			ja_JP.EUC	954
Koreanisch	ko	970	ko	970	ko_KO.euc	970		
Vereinfachtes Chinesisch	zh zh_CN.GBK	1386	zh zh_CN.GBK	1386				
Traditionelles Chinesisch	zh_TW.Big5	950	zh_TW.Big5	950				
Dänisch								
Finnisch								
Norwegisch								
Schwedisch								
Tschechisch								

Tabelle 30. Verzeichnis-Locales für Linux, Linux/390, SGI, Dynix (Forts.)

	Linux		Linux/390		SGI		Dynix	
Sprache	**Locale**	**Code-page**	**Locale**	**Code-page**	**Locale**	**Code-page**	**Locale**	**Code-page**
Ungarisch								
Polnisch								
Niederländisch							nl	819
Türkisch								
Russisch								
Bulgarisch								
Slowenisch								

Dateigruppen für die Steuerzentrale und die Dokumentation

Die Dateigruppen für die Steuerzentrale, für die Hilfe der Steuerzentrale und für die Dokumentation werden in folgenden Verzeichnissen auf der Ziel-Workstation gespeichert:

- DB2 für AIX:
  - /usr/lpp/db2_07_01/cc/%L
  - /usr/lpp/db2_07_01/java/%L
  - /usr/lpp/db2_07_01/doc/%L
  - /usr/lpp/db2_07_01/qp/%L
  - /usr/lpp/db2_07_01/spb/%L
- DB2 für HP-UX:
  - /opt/IBMdb2/V7.1/cc/%L
  - /opt/IBMdb2/V7.1/java/%L
  - /opt/IBMdb2/V7.1/doc/%L
- DB2 für Linux:
  - /usr/IBMdb2/V7.1/cc/%L
  - /usr/IBMdb2/V7.1/java/%L
  - /usr/IBMdb2/V7.1/doc/%L
- DB2 für Solaris:
  - /opt/IBMdb2/V7.1/cc/%L
  - /usr/IBMdb2/V7.1/java/%L
  - /opt/IBMdb2/V7.1/doc/%L

Die Dateigruppen für die Steuerzentrale liegen in der Unicode-Codepage vor. Die Dateigruppen für die Dokumentation und die Hilfe der Steuerzentrale liegen in einem für Browser verständlichen codierten Zeichensatz vor. Wenn Ihr System einen anderen Locale-Namen als den angegebenen verwendet, können Sie die übersetzte Version der Steuerzentrale immer noch ausführen und die übersetzte Version der Hilfetexte anzeigen, indem Sie Verbindungen zu den entsprechenden Sprachverzeichnissen erstellen.

Wenn das Standard-Locale Ihrer AIX-Maschine zum Beispiel ja_JP.IBM-eucJP ist, können Sie Verbindungen von /usr/lpp/db2_07_01/cc/ja_JP.IBM-eucJP zu /usr/lpp/db2_07_01/cc/ja_JP und von /usr/lpp/db2_07_01/doc/ja_JP.IBM-eucJP zu /usr/lpp/db2_07_01/doc/ja_JP erstellen, indem Sie die folgenden Befehle ausführen:

- **ln -s /usr/lpp/db2_07_01/cc/ja_JP /usr/lpp/db2_07_01/cc/ja_JP.IBM-eucJP**
- **ln -s /usr/lpp/db2_07_01/doc/ja_JP /usr/lpp/db2_07_01/doc/ja_JP.IBM-eucJP**

Nach der Ausführung dieser Befehle werden die Steuerzentrale und die Hilfe in Japanisch angezeigt.

Anmerkung: Web Control Center wird auf Linux/390 oder NUMA-Q nicht vom System aus unterstützt. Diese Komponente kann von einer Client-Workstation aus zur Verwaltung von Datenbanken auf diesen Plattformen verwendet werden.

Ableiten der Werte von Codepages

Die *Codepage der Anwendung* wird aus der aktiven Umgebung abgeleitet, wenn die Datenbankverbindung hergestellt wird. Wenn die Registrierungsvariable DB2CODEPAGE gesetzt ist, wird ihr Wert als Codepage der Anwendung verwendet. Normalerweise braucht die Registrierungsvariable DB2CODEPAGE nicht definiert zu werden, weil DB2 die Codepage-Informationen automatisch aus dem Betriebssystem ableitet. Die Einstellung der Registrierungsvariablen DB2CODEPAGE auf falsche Werte kann zu unvorhersehbaren Ergebnissen führen.

Wenn Anwendungen so codiert sind, dass sie mit der CLI-Schnittstelle arbeiten, verwendet die CLI-Codeschicht die Locale-Einstellungen in einigen Fällen auch dann, wenn der Benutzer die Registrierungsvariable DB2CODEPAGE definiert hat.

Die *Codepage der Datenbank* wird aus dem Wert abgeleitet, der zur Zeit der Datenbankerstellung (explizit oder standardmäßig) angegeben wurde. Zum Beispiel wird im Folgenden definiert, wie die *aktive Umgebung* in den unterschiedlichen Betriebsumgebungen bestimmt wird:

UNIX
Unter auf UNIX basierenden Betriebssystemen wird die aktive Umgebung durch die Einstellung für die länderspezifischen Angaben bestimmt, die Informationen zur Sprache, zu Gebiet (Territory) und zum codierten Zeichensatz umfassen.

OS/2
Unter OS/2 wird die primäre und sekundäre Codepage in der Datei `CONFIG.SYS` angegeben. Mit dem Befehl **chcp** können Sie die Codepage anzeigen und innerhalb einer Sitzung dynamisch ändern.

Macintosh	Für das Macintosh Betriebssystem wird die Macintosh Codepage aus dem regionalen Versionscode des installierten Skripts abgeleitet, wenn die Registrierdatenbankvariable DB2CODEPAGE nicht gesetzt ist.
Windows	Für alle 32-Bit-Windows-Betriebssysteme wird die Codepage aus der Einstellung für die ANSI-Codepage in der Registrierdatenbank abgeleitet, wenn die Registrierungsvariable DB2CODEPAGE nicht gesetzt ist.

Eine vollständige Liste der Umgebungszuordnungen für Codepage-Werte finden Sie in Tabelle 28 auf Seite 258.

Zeichensätze

Im Allgemeinen wird der für eine Anwendung verfügbare Zeichensatz vom Datenbankmanager nicht eingeschränkt. Eine detaillierte Erläuterung zu Mehrbytezeichensätzen (MBCS), die von DB2 unterstützt werden, finden Sie im Handbuch *Application Development Guide*.

Zeichensatz für Bezeichner

Der Standardzeichensatz, der für Datenbanknamen verwendet werden kann, besteht aus den großen und kleinen Einzelbytebuchstaben des lateinischen Alphabets (A...Z, a...z), den arabischen Ziffern (0...9) und dem Unterstreichungszeichen (_). Diese Liste wird noch um drei Sonderzeichen (#, @ und $) erweitert, um Kompatibilität mit den Host-Datenbankprodukten zu gewährleisten. Die Sonderzeichen #, @ und $ sind in einer NLS-Umgebung mit Vorsicht zu verwenden, da sie nicht im unveränderlichen Zeichensatz für NLS-Hosts (EBCDIC) enthalten sind. Je nach verwendeter Codepage können auch Zeichen aus dem erweiterten Zeichensatz verwendet werden. Wenn Sie die Datenbank in einer Umgebung mit mehreren Codepages verwenden, müssen Sie darauf achten, dass alle Codepages die Elemente aus dem erweiterten Zeichensatz unterstützen, die Sie verwenden möchten.

Beim Benennen von Datenbankobjekten (wie Tabellen und Sichten), Programmkennsätzen, Host-Variablen und Cursorn können auch Elemente aus dem erweiterten Zeichensatz (z. B. Buchstaben mit diakritischen Zeichen) verwendet werden. Welche Zeichen im Einzelnen verfügbar sind, hängt von der verwendeten Codepage ab. Wenn Sie die Datenbank in einer Umgebung mit mehreren Codepages verwenden, müssen Sie darauf achten, dass alle Codepages die Elemente aus dem erweiterten Zeichensatz unterstützen, die Sie verwenden möchten. Informationen zu begrenzten Bezeichnern, die Zeichen außerhalb des erweiterten Zeichensatzes enthalten, jedoch in SQL-Anweisungen verwendet werden können, finden Sie im Handbuch *SQL Reference*.

Definition des erweiterten Zeichensatzes für DBCS-Bezeichner

In DBCS-Umgebungen umfasst der erweiterte Zeichensatz alle Zeichen des Standardzeichensatzes plus die folgenden Elemente:

- Alle Doppelbytezeichen in jeder DBCS-Codepage, mit Ausnahme des Doppelbyteleerzeichens, sind gültige Buchstaben.
- Das Doppelbyteleerzeichen ist ein Sonderzeichen.
- Die in jeder Misch-Codepage verfügbaren Einzelbytezeichen werden verschiedenen Kategorien zugeordnet:

Kategorie	Gültige Codepunkte in jeder Misch-Codepage
Ziffern	x30-39
Buchstaben	x23-24, x40-5A, x61-7A, xA6-DF (A6-DF nur für Codepages 932 und 942)
Sonderzeichen	Alle anderen gültigen Codepunkte für Einzelbytezeichen

Codieren von SQL-Anweisungen

Die Codierung von SQL-Anweisungen ist nicht sprachenabhängig. SQL-Schlüsselwörter können in Großbuchstaben, Kleinbuchstaben oder in gemischter Schreibweise eingegeben werden. Die Namen von Datenbankobjekten und Host-Variablen sowie Programmbezeichnungen in einer SQL-Anweisung dürfen keine Zeichen enthalten, die nicht in dem in „Definition des erweiterten Zeichensatzes für DBCS-Bezeichner" beschriebenen erweiterten Zeichensatz enthalten sind.

CCSID-Unterstützung für bidirektionale Zeichen

Die folgenden bidirektionalen Attribute sind erforderlich für die korrekte Behandlung bidirektionaler Daten auf unterschiedlichen Plattformen:

```
- Textart (LOGICAL oder VISUAL)
- Gestaltung (SHAPED oder UNSHAPED)
- Ausrichtung (RIGHT-TO-LEFT oder LEFT-TO-RIGHT)
- Zeichengestaltung (ARABIC oder HINDI)
- Symmetrische Auslagerung (YES oder NO)
```

Da die Standardeinstellungen auf den unterschiedlichen Plattformen nicht gleich sind, können Probleme auftreten, wenn DB2-Daten von einer Plattform auf eine andere übertragen werden. Das Windows-Betriebssystem verwendet beispielsweise Daten im Format LOGICAL UNSHAPED, während OS/390 im Allgemeinen mit Daten im Format SHAPED VISUAL arbeitet. Daten, die ohne Unterstützung für bidirektionale Attribute von DB2 Universal Database für OS/390 auf DB2 UDB unter 32-Bit-Windows-Betriebssysteme übertragen werden, können daher falsch angezeigt werden.

Spezifische CCSIDs für bidirektionale Zeichen

DB2 unterstützt bidirektionale Datenattribute über spezielle bidirektionale IDs für codierte Zeichensätze (CCSIDs). Die folgenden CCSIDs für bidirektionale Zeichen wurden definiert und mit DB2 UDB implementiert:

CCSID (dez)	CCSID (hex)	Code-page	Zeichenfolge-art
00420	x'01A4'	420	4
00424	x'01A8'	424	4
08612	x'21A4'	420	5
08616	x'21A8'	424	10
00856	x'0358'	856	5
00862	x'035E'	862	4
00864	x'0360'	864	5
00916	x'0394'	916	5
01046	x'0416'	1046	5
01089	x'0441'	1089	5
01255	x'04E7'	1255	5
01256	x'04E8'	1256	5
62208	x'F300'	856	4
62209	x'F301'	862	10
62210	x'F302'	916	4
62211	x'F303'	424	5
62213	x'F305'	862	5
62215	x'F307'	1255	4
62218	x'F30A'	864	4
62220	x'F30C'	856	6
62221	x'F30D'	862	6
62222	x'F30E'	916	6
62223	x'F30F'	1255	6
62224	x'F310'	420	6
62225	x'F311'	864	6
62226	x'F312'	1046	6
62227	x'F313'	1089	6
62228	x'F314'	1256	6
62229	x'F315'	424	8
62230	x'F316'	856	8
62231	x'F317'	862	8
62232	x'F318'	916	8
62233	x'F319'	420	8
62234	x'F31A'	420	9
62235	x'F31B'	424	6
62236	x'F31C'	856	10
62237	x'F31D'	1255	8
62238	x'F31E'	916	10
62239	x'F31F'	1255	10
62240	x'F320'	424	11
62241	x'F321'	856	11
62242	x'F322'	862	11
62243	x'F323'	916	11
62244	x'F324'	1255	11

62245	x'F325'	424	10
62246	x'F326'	1046	8
62247	x'F327'	1046	9
62248	x'F328'	1046	4
62249	x'F329'	1046	12
62250	x'F32A'	420	12

Dabei sind CDRA-Zeichenfolgearten wie folgt definiert:

Zeichenfolgeart	Textart	Zeichengestaltung	Ausrichtung	Gestaltung	Symmetrische Auslagerungsfunktion
4	VISUAL	PASSTHRU	LTR	SHAPED	OFF
5	IMPLICIT	ARABIC	LTR	UNSHAPED	ON
6	IMPLICIT	ARABIC	RTL	UNSHAPED	ON
7(*)	VISUAL	ARABIC	CONTEXTUAL(*)	UNSHAPED-Lig	OFF
8	VISUAL	ARABIC	RTL	SHAPED	OFF
9	VISUAL	PASSTHRU	RTL	SHAPED	ON
10	IMPLICIT	PASSTHRU	CONTEXTUAL-L	UNSHAPED	ON
11	IMPLICIT	PASSTHRU	CONTEXTUAL-R	UNSHAPED	ON
12	IMPLICIT	ARABIC	RTL	SHAPED	ON

Anmerkung: (*) Die Feldausrichtung ist von links nach rechts (LTR), wenn das erste alphabetische Zeichen ein lateinisches Zeichen ist, und von rechts nach links (RTL), wenn es ein bidirektionales Zeichen ist. Zeichen sind UNSHAPED, LamAlef-Ligaturen werden jedoch beibehalten und nicht in Bestandteile getrennt.

Implementierung von Unterstützung bidirektionaler Zeichen in DB2 Universal Database

Bidirektionale Layoutumsetzungen werden in DB2 Universal Database mit den neuen CCSID-Definitionen implementiert. Bei den neuen spezifischen CCSIDs für bidirektionale Zeichen werden Layoutumsetzungen anstatt oder zusätzlich zur Umsetzung von Codepages durchgeführt. Damit diese Unterstützung verwendet werden kann, muss die Registrierungsvariable DB2BIDI auf YES gesetzt werden. Standardmäßig ist diese Variable nicht gesetzt. Sie wird vom Server für alle Umsetzungen verwendet und kann nur gesetzt werden, wenn der Server gestartet ist. Das Einstellen von DB2BIDI auf YES kann sich wegen der zusätzlichen Prüfung und der Layoutumsetzungen auf die Leistung auswirken. Wenn Sie eine bestimmte CCSID für bidirektionale Zeichen in einer Nicht-DRDA-Umgebung angeben wollen, wählen Sie die CCSID (aus obiger Tabelle) aus, die den Kenndaten Ihres Clients entspricht, und stellen DB2CODEPAGE auf diesen Wert ein. Wenn Sie bereits eine Verbindung zur Datenbank haben, müssen Sie den Befehl TERMINATE absetzen, und die Verbindung anschließend erneut herstellen, damit die Einstellung von DB2CODEPAGE wirksam wird. Wenn Sie eine CCSID auswählen, die für die Codepage oder die Zeichenfolgeart Ihrer Client-Plattform nicht geeignet ist, erhalten Sie möglicherweise unerwartete Ergebnisse. Wenn Sie eine inkompatible CCSID (z.B. CCSID für Hebräisch zur Verbindung zu einer arabischen

Datenbank) verwenden oder wenn DB2BIDI nicht für den Server eingestellt wurde, erhalten Sie eine Fehlernachricht, wenn Sie versuchen, die Verbindung herzustellen.

Wenn bei DRDA-Umgebungen die HOST-EBCDIC-Plattform diese CCSIDs für bidirektionale Zeichen ebenfalls unterstützt, müssen Sie nur den Wert für DB2CODEPAGE einstellen. Wenn die HOST-Plattform diese CCSIDs jedoch nicht unterstützt, müssen Sie eine CCSID-Überschreibung für den HOST-Datenbankserver angeben, zu dem Sie eine Verbindung herstellen. Dies ist notwendig, da in einer DRDA-Umgebung Umsetzungen von Codepages und Layoutumsetzungen vom Empfänger der Daten durchgeführt werden. Wenn der HOST-Server diese CCSIDs jedoch nicht unterstützt, wird keine Layoutumsetzung für die Daten durchgeführt, die er von DB2 UDB empfängt. Wenn Sie eine CCSID-Überschreibung verwenden, führt der Client unter DB2 UDB die Layoutumsetzung auch für die abgehenden Daten durch. Weitere Informationen zur Einstellung einer CCSID-Überschreibung finden Sie im *DB2 Connect Benutzerhandbuch*.

Die CCSID-Überschreibung wird nicht unterstützt, wenn die HOST-EBCDIC-Plattform der Client und DB2 UDB der Server ist.

Implementierung von Unterstützung bidirektionaler Zeichen in DB2 Connect

Wenn Daten zwischen DB2 Connect und einer Datenbank auf dem Server ausgetauscht werden, ist es normalerweise die Empfängermaschine, die die Umsetzung der ankommenden Daten ausführt. Dasselbe würde normalerweise auch auf bidirektionale Layoutumsetzungen zutreffen, die zusätzlich zur üblichen Codepage-Umsetzung erfolgen. DB2 Connect kann wahlfrei bidirektionale Layoutumsetzungen an den Daten ausführen, die gerade an die Server-Datenbank gesendet werden, zusätzlich zu den Daten, die von der Server-Datenbank empfangen wurden.

Damit DB2 Connect bidirektionale Layoutumsetzungen an abgehenden Daten für eine Server-Datenbank ausführen kann, muss die bidirektionale CCSID der Server-Datenbank überschrieben werden. Dies wird durch die Verwendung des Parameters BIDI im Feld PARMS des DCS-Datenbankverzeichniseintrags für die Server-Datenbank erreicht.

Anmerkung: Wenn Sie möchten, dass DB2 Connect eine Layoutumsetzung an den Daten ausführt, die gerade an die DB2-Host-Datenbank gesendet werden, müssen Sie ihre CCSID nicht überschreiben, aber Sie müssen den Parameter BIDI dem Feld PARMS des DCS-Datenbankverzeichnisses hinzufügen. In diesem Fall sollten Sie als CCSID die Standard-CCSID der DB2-Host-Datenbank angeben.

Der Parameter BIDI muss als neunter Parameter im Feld PARMS zusammen mit der bidirektionalen CCSID, mit der Sie die bidirektionale Standard-CCSID der Server-Datenbank überschreiben wollen, angegeben werden:

```
",,,,,,,,BIDI=xyz"
```

Dabei gilt folgendes: *xyz* ist die CCSID-Überschreibung.

Anmerkung: Die Registrierungsvariable DB2BIDI muss auf den Wert YES gesetzt werden, damit der Parameter BIDI wirksam werden kann.

Eine Liste der unterstützten bidirektionalen CCSIDs finden Sie zusammen mit ihren Zeichenfolgearten in „Spezifische CCSIDs für bidirektionale Zeichen” auf Seite 282.

Die Verwendung dieser Einrichtung wird am besten an einem Beispiel erläutert.

Angenommen, Sie haben einen hebräischen DB2-Client, der CCSID 62213 (bidirektionale Zeichenfolgeart 5) ausführt, und Sie wollen auf eine DB2-Host-Datenbank zugreifen, die CCSID 00424 (bidirektionale Zeichenfolgeart 4) ausführt. Sie wissen aber, dass die in der DB2-Host-Datenbank enthaltenen Daten auf der CCSID 08616 (bidirektionale Zeichenfolgeart 6) basieren.

Hier gibt es zwei Probleme: Das erste Problem besteht darin, dass die DB2-Host-Datenbank den Unterschied zwischen den bidirektionalen Zeichenfolgearten mit den CCSIDs 00424 und 08616 nicht kennt. Das zweite Problem ist, dass die DB2-Host-Datenbank die DB2-Client-CCSID (62213) nicht erkennt. Sie unterstützt nur CCSID 00862, die auf derselben Codepage wie CCSID 62213 basiert.

Sie müssen zunächst sicherstellen, dass die an die DB2-Host-Datenbank gesendeten Daten das Format der bidirektionalen Zeichenfolgeart 6 haben. Ferner müssen Sie DB2 Connect bekannt geben, dass eine bidirektionale Umsetzung der Daten ausgeführt werden soll, die von der DB2-Host-Datenbank empfangen werden. Sie müssen den folgenden Katalogisierungsbefehl für die DB2-Host-Datenbank verwenden:

```
db2 catalog dcs database nydb1 as telaviv parms ",,,,,,,,BIDI=08616"
```

Mit diesem Befehl wird DB2 Connect mitgeteilt, dass die CCSID 00424 der DB2-Host-Datenbank mit der CCSID 08616 überschrieben werden soll. Diese Überschreibung wird wie folgt verarbeitet:

1. DB2 Connect stellt eine Verbindung zur DB2-Host-Datenbank mit CCSID 00862 her.
2. DB2 Connect führt eine bidirektionale Layoutumsetzung der Daten aus, die gerade an die DB2-Host-Datenbank *gesendet* werden sollen. CCSID 62213 (bidirektionale Zeichenfolgeart 5) wird in die CCSID 62221 (bidirektionale Zeichenfolgeart 6) umgesetzt.
3. DB2 Connect führt eine bidirektionale Layoutumsetzung der Daten aus, die von der DB2-Host-Datenbank *empfangen* werden. Diese Umsetzung erfolgt von CCSID 08616 (bidirektionale Zeichenfolgeart 6) in CCSID 62213 (bidirektionale Zeichenfolgeart 5).

Anmerkung: In einigen Fällen kann die Verwendung einer bidirektionalen CCSID die SQL-Abfrage selbst so ändern, dass sie nicht mehr vom DB2-Server erkannt wird. Sie sollten vor allem die Verwendung von CCSIDs mit IMPLICIT CONTEXTUAL und IMPLICIT RIGHT-TO-LEFT vermeiden, wenn eine andere Zeichenfolgeart verwendet werden kann. CONTEXTUAL-CCSIDs können zu unvorhersehbaren Ergebnissen führen, wenn die SQL-Abfrage Zeichenfolgen in Anführungszeichen enthält. Vermeiden Sie daher in SQL-Anweisungen Zeichenfolgen in Anführungszeichen. Verwenden Sie nach Möglichkeit überall Host-Variablen.

Wenn eine bestimmte bidirektionale CCSID Probleme hervorruft, die nicht durch das Befolgen dieser Empfehlungen korrigiert werden können, setzen Sie DB2BIDI auf den Wert NO.

Sortierfolge

Der Datenbankmanager vergleicht Zeichendaten mit Hilfe einer *Sortierfolge*. Dies ist eine Reihenfolge für eine Zeichengruppe, mit der festgelegt wird, ob ein bestimmtes Zeichen vor, nach oder gleichwertig mit einem anderen angeordnet wird. Mit einer Sortierfolge kann beispielsweise angegeben werden, dass Großbuchstaben und Kleinbuchstaben gleich bewertet werden sollen.

Die Sortierfolge wird bei der Datenbankerstellung angegeben und kann später nicht mehr geändert werden.

Der Datenbankmanager ermöglicht das Erstellen von Datenbanken mit benutzerdefinierten Sortierfolgen über die API (Anwendungsprogrammierschnittstelle). Informationen zur Implementierung einer benutzerdefinierten Sortierfolgentabelle finden Sie im Handbuch *Application Development Guide*.

Anmerkung: Daten aus Zeichenfolgen, die mit dem Attribut FOR BIT DATA definiert wurden, und BLOB-Daten werden mit der Binärsortierfolge sortiert.

Allgemeine Überlegungen

Wenn eine Sortierfolge definiert ist, werden alle zukünftigen Zeichenvergleiche für diese Datenbank mit dieser Sortierfolge ausgeführt. Mit Ausnahme von Zeichendaten, die mit dem Attribut FOR BIT DATA oder BLOB definiert wurden, wird die Sortierfolge für alle SQL-Vergleiche und Klauseln ORDER BY verwendet, sowie für das Erstellen von Indizes und Statistiken. Weitere Informationen zur Verwendung der Datenbanksortierfolge finden Sie im Abschnitt über Vergleiche von Zeichenfolgen im Handbuch *SQL Reference*.

In folgenden Fällen können Probleme auftreten:

- Eine Anwendung mischt sortierte Daten aus einer Datenbank mit Anwendungsdaten, die mit einer anderen Sortierfolge sortiert wurden.
- Eine Anwendung mischt sortierte Daten aus einer Datenbank mit sortierten Daten aus einer anderen Datenbank, wobei beide unterschiedliche Sortierfolgen haben.
- Eine Anwendung geht von Voraussetzungen für die sortierten Daten aus, die für die betreffende Sortierfolge nicht stimmen. Für eine bestimmte Sortierfolge kann beispielsweise gelten, dass Zahlen nach Buchstaben angeordnet werden, oder nicht.

Ein letzter Punkt, der zu beachten ist, besteht darin, dass die Ergebnisse eines Sortiervorgangs, der auf einem direkten Vergleich von Zeichencodepunkten beruht, nur mit den Ergebnissen einer Abfrage übereinstimmen, die mit einer Identitätssortierfolge geordnet wurden.

Überlegungen zu zusammengeschlossenen Datenbanken

Die von Ihnen ausgewählte Datenbanksortierfolge kann sich auf die Leistung des Systems zusammengeschlossener Datenbanken auswirken. Wenn eine Datenquelle dieselbe Sortierfolge wie die zusammengeschlossene DB2-Datenbank verwendet, kann DB2 die von der Sortierfolge abhängige Verarbeitung von Zeichendaten in die Datenquelle auslagern. Verwendet eine Datenquelle eine von DB2 abweichende Sortierfolge, werden die Daten abgerufen, und die gesamte von der Sortierfolge abhängige Verarbeitung von Zeichendaten erfolgt lokal (dies kann die Leistung beeinträchtigen).

Beachten Sie die folgenden Punkte, wenn sie ermitteln wollen, ob DB2 und eine Datenquelle dieselbe Sortierfolge verwenden:

- Unterstützung von Landessprachen

  Die Sortierfolge bezieht sich auf die auf einem Server unterstützte Sprache. Vergleichen Sie die NLS-Informationen von DB2 und der Datenquelle.
- Kenndaten der Datenquelle

  Manche Datenquellen werden mit Sortierfolgen erstellt, die nicht zwischen Groß-/Kleinschreibung unterscheiden. Dadurch können von der Sortierfolge abhängige Operationen zu anderen Ergebnissen führen als in DB2.

- Anpassung

  Einige Datenquellen bieten mehrere Optionen für Sortierfolgen bzw. ermöglichen deren Anpassung.

Wählen Sie die Sortierfolge für eine zusammengeschlossene DB2-Datenbank entsprechend der Mischung der Datenquellen aus, auf welche diese Datenbank zugreift. Beispiel:

- Wenn eine DB2-Datenbank überwiegend auf Oracle-Datenbanken mit derselben Codepage (NLS) wie DB2 zugreift, geben Sie bei der Erstellung der Datenbank die Identitätssortierfolge an (Oracle-Datenbanken verwenden eine gleichwertige Sortierfolge).
- Wenn eine DB2-Datenbank nur auf DB2-UDB-Datenbanken zugreift, stellen Sie sicher, dass die Sortierfolgenwerte übereinstimmen.

Informationen zur Einrichtung einer MVS-Sortierfolge finden Sie in den Beispielen unter der Beschreibung der API **sqlecrea** (Create Database) zur Erstellung von Datenbanken im Handbuch *Administrative API Reference*. Diese Beispiele enthalten Sortierfolgetabellen für die EBCIDIC-Codepages 500, 37 und 5026/5035.

Denken Sie nach dem Festlegen der Sortierfolge für die DB2-Datenbank daran, die Server-Option *collating_sequence* für jeden Datenquellen-Server zu definieren. Diese Option gibt an, ob die Sortierfolge eines gegebenen Datenquellen-Servers mit der Sortierfolge der DB2-Datenbank übereinstimmt.

Setzen Sie die Option *collating_sequence* auf den Wert "Y", wenn die Sortierfolgen übereinstimmen. Diese Einstellung ermöglicht dem Optimierungsprogramm von DB2, von der Sortierfolge abhängige Verarbeitung in der Datenquelle vorzunehmen, was zu einer höheren Leistung führen kann. Unterscheidet sich jedoch die Sortierfolge der Datenquelle von der in DB2, erhalten Sie möglicherweise falsche Ergebnisse erhalten. Wenn Ihr Plan z. B. Mischverknüpfungen verwendet, lagert das DB2-Optimierungsprogramm Sortieroperationen soweit möglich in die Datenquellen aus. Wenn die Datenquelle eine andere Sortierfolge hat, können die Verknüpfungsergebnisse falsch sein.

Setzen Sie die Option *collating_sequence* auf den Wert "N", wenn die Sortierfolgen nicht übereinstimmen. Verwenden Sie diesen Wert, wenn sich die Sortierfolgen der Datenquelle von DB2 unterscheiden oder wenn die Sortieroperationen der Datenquelle möglicherweise die Groß-/Kleinschreibung nicht beachten. In einer Datenquelle mit einer deutschen Codepage ohne Beachtung von Groß-/Kleinschreibung z. B. werden die Begriffe KARUSSELL, KaRUsSeL und karussell als identisch betrachtet. Setzen Sie die Option *collating_sequence* auf "N", wenn Sie nicht sicher sind, dass die Sortierfolgen der Datenquelle und DB2 übereinstimmen.

Sortieren thailändischer Zeichen

Wenn Sie spezielle Vokale (″führende Vokale″), Tonmarkierungen und andere thailändische Sonderzeichen korrekt sortieren wollen, müssen Sie die Datenbank durch einen Befehl CREATE DATABASE mit der Klausel COLLATE USING NLSCHAR erstellen. Informationen zur Syntax finden Sie im Handbuch *Command Reference*.

Werte für Datum und Uhrzeit

Die Datentypen für Datum und Uhrzeit werden im Folgenden beschrieben. Werte für Datum und Uhrzeit können zwar in bestimmten arithmetischen Operationen und Zeichenfolgeoperationen verwendet werden und sind auch mit bestimmten Zeichenfolgen kompatibel, allerdings handelt es sich bei diesen Werten weder um Zeichenfolgen noch um Ziffern.

Datum

Datum ist ein Wert, der aus drei Teilen besteht (Jahr, Monat und Tag). Der Wertebereich für das Jahr liegt zwischen 0001 und 9999. Der Wertebereich für den Monat liegt zwischen 1 und 12. Der Wertebereich für den Tag liegt zwischen 1 und *x*, wobei *x* vom Monat abhängig ist.

Intern wird ein Datum durch eine Zeichenfolge aus 4 Byte dargestellt. Jedes dieser Byte besteht aus 2 gepackten Dezimalziffern. Die ersten zwei Byte stehen für das Jahr, das dritte Byte für den Monat und das letzte Byte für den Tag.

Die Länge einer Datumsspalte (DATE), wie im SQL-Deskriptorbereich beschrieben, beträgt 10 Byte. Dies ist die geeignete Länge für die Darstellung des Werts als Zeichenfolge.

Uhrzeit

Uhrzeit ist ein Wert, der aus drei Teilen besteht (Stunde, Minute und Sekunde) und eine Uhrzeit in der 24-Stunden-Zeiteinteilung bezeichnet. Der Wertebereich für die Stunde liegt zwischen 0 und 24. Der Wertebereich für die anderen Teile liegt zwischen 0 und 59. Wenn die Stunde 24 ist, sind die Minuten- und Sekundenangaben gleich null.

Intern wird Zeit durch eine Zeichenfolge aus 3 Byte dargestellt. Jedes dieser Byte besteht aus 2 gepackten Dezimalziffern. Das erste Byte steht für die Stunde, das zweite Byte für die Minute und das letzte Byte für die Sekunde.

Die Länge einer Zeitspalte (TIME), wie im SQL-Deskriptorbereich beschrieben, beträgt 8 Byte. Dies ist die geeignete Länge für die Darstellung des Werts als Zeichenfolge.

Zeitmarke

Eine *Zeitmarke* ist ein aus sieben Teilen bestehender Wert (Jahr, Monat, Tag, Stunde, Minute, Sekunde, Mikrosekunde), der einen Wert für ein Datum und

eine Uhrzeit (wie oben definiert) bezeichnet, nur dass hier für die Zeit zusätzlich Mikrosekunden angegeben werden.

Intern wird die Zeitmarke durch eine Zeichenfolge aus 10 Byte dargestellt. Jedes dieser Byte besteht aus 2 gepackten Dezimalziffern. Die ersten vier Byte stehen für das Datum, die nächsten drei Byte für die Uhrzeit und die letzten drei Byte für die Mikrosekunden.

Die Länge einer Zeitmarkenspalte (TIMESTAMP), wie im SQL-Deskriptorbereich beschrieben, beträgt 26 Byte. Dies ist die geeignete Länge für die Darstellung des Werts als Zeichenfolge.

Darstellung von Werten für Datum und Uhrzeit als Zeichenfolge

Werte mit dem Datentyp DATE, TIME oder TIMESTAMP werden intern in einer Form dargestellt, die für den SQL-Benutzer transparent ist. Datum, Uhrzeiten und Zeitmarken können allerdings auch als Zeichenfolgen dargestellt werden. Diese Darstellungen betreffen den SQL-Benutzer direkt, da es keine Konstanten oder Variablen mit den Datentypen DATE, TIME oder TIMESTAMP gibt. Daher muss ein Wert für Datum und Uhrzeit, der abgerufen werden soll, einer Zeichenfolgenvariablen zugeordnet werden. Die Darstellung als Zeichenfolge entspricht normalerweise dem Standardformat der Werte für Datum und Uhrzeit, die dem Landes-/Regionalcode des Clients zugeordnet sind, sofern diese nicht durch Angabe der Formatoption "F" bei der Vorkompilierung des Programms oder beim Binden an die Datenbank überschrieben werden. Eine Liste der Zeichenfolgenformate für die verschiedenen Landes-/Regionalcodes können Sie Tabelle 33 auf Seite 293 entnehmen.

Wenn eine gültige Zeichenfolge zur Darstellung des Werts für Datum und Uhrzeit in einer Operation mit einem internen Wert für Datum und Uhrzeit verwendet wird, wird die Zeichenfolge vor Ausführung der Operation in das interne Format für Datum, Zeit oder Zeitmarke umgesetzt. Gültige Zeichenfolgendarstellungen von Werten für Datum und Uhrzeit werden in den folgenden Abschnitten definiert.

Zeichenfolgen für das Datum

Zur Darstellung des Datums können Zeichenfolgen verwendet werden, die mit einer Ziffer beginnen und aus mindestens 8 Zeichen bestehen. Abschließende Leerzeichen sind zulässig. Führende Nullen bei den Monats- und Tagesangaben des Datums können wegfallen.

Gültige Zeichenfolgenformate für Datumsangaben sind in Tabelle 31 auf Seite 291 aufgelistet. Für jedes Format wird der Name, eine zugehörige Abkürzung und ein Verwendungsbeispiel angegeben.

Tabelle 31. Formate zur Darstellung des Datums als Zeichenfolge

Formatname	Abkürzung	Datumsformat	Beispiel
International Standards Organization	ISO	jjjj-mm-tt	1994-10-27
IBM USA Standard	USA	mm/tt/jjjj	10/27/1994
IBM Europäischer Standard	EUR	tt.mm.jjjj	27.10.1994
Japanischer Industriestandard	JIS	jjjj-mm-tt	1994-10-27
Standortabhängig (lokal)	LOC	Abhängig vom Landes-/Regionalcode der Datenbank	—

Zeichenfolgen für die Uhrzeit

Zur Darstellung der Uhrzeit können Zeichenfolgen verwendet werden, die mit einer Ziffer beginnen und aus mindestens 4 Zeichen bestehen. Abschließende Leerzeichen sind zulässig. Eine führende Null kann bei den Stundenangaben wegfallen, und die Sekunden können ganz wegfallen. Wenn Sie die Sekunden nicht angeben, wird eine implizite Angabe von 0 Sekunden angenommen. Dementsprechend ist `13.30` äquivalent zu `13.30.00`.

Gültige Zeichenfolgenformate für Uhrzeitangaben sind in Tabelle 32 aufgelistet. Für jedes Format wird der Name, eine zugehörige Abkürzung und ein Verwendungsbeispiel angegeben.

Tabelle 32. Formate zur Darstellung der Uhrzeit als Zeichenfolge

Formatname	Abkürzung	Zeitformat	Beispiel
International Standards Organization	ISO	hh.mm.ss	13.30.05
IBM USA Standard	USA	hh:mm AM oder PM	1:30 PM
IBM Europäischer Standard	EUR	hh.mm.ss	13.30.05
Japanischer Industriestandard	JIS	hh:mm:ss	13:30:05
Standortabhängig (lokal)	LOC	Abhängig vom Landes-/Regionalcode der Anwendung	—

Anmerkungen:

1. In den Uhrzeitformaten ISO, EUR oder JIS ist `.ss` (bzw. `:ss`) wahlfrei.

2. Im Uhrzeitformat USA kann die Angabe der Minuten weggelassen werden, was implizit eine Angabe von 00 Minuten bedeutet. Demnach ist 1 PM äquivalent zu 1:00 PM.
3. Im Uhrzeitformat USA kann die Stundenangabe nicht größer als 12 sein und darf, abgesehen vom Spezialfall 00:00 AM, nicht 0 sein. Bei Verwendung des ISO-Formats mit der 24-Stunden-Zeiteinteilung gelten die folgenden Entsprechungen zwischen dem Format USA und der 24-Stunden-Zeiteinteilung:
   - 12:01 AM bis 12:59 AM entspricht 00.01.00 bis 00.59.00.
   - 01:00 AM bis 11:59 AM entspricht 01.00.00 bis 11.59.00.
   - 12:00 PM (Mittag) bis 11:59 PM entspricht 12.00.00 bis 23.59.00.
   - 12:00 AM (Mitternacht) entspricht 24.00.00 und 00:00 AM (Mitternacht) entspricht 00.00.00.

Zeichenfolgen für Zeitmarken

Zur Darstellung einer Zeitmarke können Zeichenfolgen verwendet werden, die mit einer Ziffer beginnen und aus mindestens 16 Zeichen bestehen. Die vollständige Darstellung einer Zeitmarke hat das Format *jjjj-mm-tt-hh.mm.ss.nnnnnn*. Abschließende Leerzeichen sind zulässig. Führende Nullen können aus der Monats-, Tages- oder Stundenangabe der Zeitmarke weggelassen werden. Mikrosekunden können vollständig abgeschnitten oder weggelassen werden. Wenn Sie Ziffern in der Mikrosekundenangabe weglassen, wird eine implizite Angabe von 0 angenommen. Dementsprechend ist `1991-3-2-8.30.00` äquivalent zu `1991-03-02-08.30.00.000000`.

Überlegungen zu Zeichensätzen

Zeichenfolgen für Datum und Zeitmarke dürfen nur aus Einzelbytezeichen und -ziffern bestehen.

Datums- und Uhrzeitformate

Die Zeichenfolgen für Datums- und Zeitformate entsprechen dem Standardformat der Werte für Datum und Uhrzeit, die dem Landes-/Regionalcode der Anwendung zugeordnet sind. Dieses Standardformat kann durch Angeben der Formatoption ″F″ beim Vorkompilieren oder Binden an die Datenbank überschrieben werden.

Es folgt eine Beschreibung der Ein- und Ausgabeformate für Datum und Uhrzeit:

- Eingabezeitformat
  - Es gibt kein Standardformat für Uhrzeiteingaben.
  - Für alle Landes-/Regionalcodes können alle Zeitformate eingegeben werden.

- Ausgabezeitformat
  - Der Standardwert für die Ausgabe des Zeitangaben entspricht dem lokalen Zeitformat.
- Eingabedatumsformat
  - Es gibt kein Standardformat für Datumseingaben.
  - Wenn das lokale Format für das Datum und ein ISO-, JIS-, EUR- oder USA-Datumsformat nicht übereinstimmen, wird das lokale Format für die Datumseingabe angenommen. Siehe z. B. den Eintrag für Großbritannien in Tabelle 33.
- Ausgabedatumsformat
  - Das Standardformat für das Ausgabedatum wird in Tabelle 33 gezeigt.

Anmerkung: Tabelle 33 zeigt außerdem eine Liste der Zeichenfolgeformate für die verschiedenen Landes-/Regionalcodes.

Tabelle 33. Datums- und Zeitformate nach Landes-/Regionalcode

Landes-/ Regionalcode	Lokales Datumsformat	Lokales Zeitformat	Standardausgabedatumsformat	Eingabedatumsformate
355 Albanien	jjjj-mm-tt	JIS	LOC	LOC, USA, EUR, ISO
785 Arabisch	tt/mm/jjjj	JIS	LOC	LOC, EUR, ISO
001 Australien (1)	mm-tt-jjjj	JIS	LOC	LOC, USA, EUR, ISO
061 Australien	tt-mm-jjjj	JIS	LOC	LOC, USA, EUR, ISO
032 Belgien	tt/mm/jjjj	JIS	LOC	LOC, EUR, ISO
055 Brasilien	tt.mm.jjjj	JIS	LOC	LOC, EUR, ISO
359 Bulgarien	tt.mm.jjjj	JIS	EUR	LOC, USA, EUR, ISO
001 Kanada	mm-tt-jjjj	JIS	USA	LOC, USA, EUR, ISO
002 Kanada (Französisch)	tt-mm-jjjj	ISO	ISO	LOC, USA, EUR, ISO
385 Kroatien	jjjj-mm-tt	JIS	ISO	LOC, USA, EUR, ISO
042 Tschechische Republik	jjjj-mm-tt	JIS	ISO	LOC, USA, EUR, ISO

Tabelle 33. Datums- und Zeitformate nach Landes-/Regionalcode (Forts.)

Landes-/ Regionalcode	Lokales Datumsformat	Lokales Zeitformat	Standardausgabedatumsformat	Eingabedatumsformate
045 Dänemark	tt-mm-jjjj	ISO	ISO	LOC, USA, EUR, ISO
358 Finnland	tt/mm/jjjj	ISO	EUR	LOC, EUR, ISO
389 FJR Republik Mazedonien	tt.mm.jjjj	JIS	EUR	LOC, USA, EUR, ISO
033 Frankreich	tt/mm/jjjj	JIS	EUR	LOC, EUR, ISO
049 Deutschland	tt/mm/jjjj	ISO	ISO	LOC, EUR, ISO
030 Griechenland	tt/mm/jjjj	JIS	LOC	LOC, EUR, ISO
036 Ungarn	jjjj-mm-tt	JIS	ISO	LOC, USA, EUR, ISO
354 Island	tt-mm-jjjj	JIS	LOC	LOC, USA, EUR, ISO
091 Indien	tt/mm/jjjj	JIS	LOC	LOC, EUR, ISO
972 Israel	tt/mm/jjjj	JIS	LOC	LOC, EUR, ISO
039 Italien	tt/mm/jjjj	JIS	LOC	LOC, EUR, ISO
081 Japan	mm/tt/jjjj	JIS	ISO	LOC, USA, EUR, ISO
082 Korea	mm/tt/jjjj	JIS	ISO	LOC, USA, EUR, ISO
001 Lateinamerika (1)	mm-tt-jjjj	JIS	LOC	LOC, USA, EUR, ISO
003 Lateinamerika	tt-mm-jjjj	JIS	LOC	LOC, EUR, ISO
031 Niederlande	tt-mm-jjjj	JIS	LOC	LOC, USA, EUR, ISO
047 Norwegen	tt/mm/jjjj	ISO	EUR	LOC, EUR, ISO
048 Polen	jjjj-mm-tt	JIS	ISO	LOC, USA, EUR, ISO
351 Portugal	tt/mm/jjjj	JIS	LOC	LOC, EUR, ISO
086 VR China	mm/tt/jjjj	JIS	ISO	LOC, USA, EUR, ISO
040 Rumänien	jjjj-mm-tt	JIS	ISO	LOC, USA, EUR, ISO

Tabelle 33. Datums- und Zeitformate nach Landes-/Regionalcode (Forts.)

Landes-/ Regionalcode	Lokales Datumsformat	Lokales Zeitformat	Standardausgabedatumsformat	Eingabedatumsformate
007 Russland	tt/mm/jjjj	ISO	LOC	LOC, EUR, ISO
381 Serbien/ Montenegro	jjjj-mm-tt	JIS	ISO	LOC, USA, EUR, ISO
042 Slowakei	jjjj-mm-tt	JIS	ISO	LOC, USA, EUR, ISO
386 Slowenien	jjjj-mm-tt	JIS	ISO	LOC, USA, EUR, ISO
034 Spanien	tt/mm/jjjj	JIS	LOC	LOC, EUR, ISO
046 Schweden	tt/mm/jjjj	ISO	ISO	LOC, EUR, ISO
041 Schweiz	tt/mm/jjjj	ISO	EUR	LOC, EUR, ISO
088 Taiwan	mm-tt-jjjj	JIS	ISO	LOC, USA, EUR, ISO
066 Thailand (2)	tt/mm/jjjj	JIS	LOC	LOC, EUR, ISO
090 Türkei	tt/mm/jjjj	JIS	LOC	LOC, EUR, ISO
044 Großbritannien	tt/mm/jjjj	JIS	LOC	LOC, EUR, ISO
001 USA	mm-tt-jjjj	JIS	USA	LOC, USA, EUR, ISO
084 Vietnam	tt/mm/jjjj	JIS	LOC	LOC, EUR, ISO

Anmerkungen:

1. Ländern/Regionen, die die standardmäßige Ländereinstellung C (Locale C) verwenden, wird der Landes-/Regionalcode 001 zugeordnet.
2. jjjj in der buddhistischen Zeitrechnung entspricht der gregorianischen Zeitrechnung + 543 Jahre (nur Thailand).

Unicode-Unterstützung in DB2 UDB

Die Stufe der Unicode-Unterstützung in DB2 UDB ist hier dokumentiert.

Einführung

Der Unicode-Zeichencodierungsstandard ist ein Zeichencodierungsschema mit fester Länge, das Zeichen fast aller lebenden Sprachen der Welt umfasst.

Unicode arbeitet mit zwei Codierungsformen: 8-Bitcodierung und 16-Bitcodierung. Die Standardcodierungsform ist die 16-Bitcodierung, d. h. jedes Zeichen ist 16 Bit (zwei Byte) lang und wird gewöhnlich in der Form U+hhhh angegeben, wobei hhhh der Hexadezimalcodepunkt des Zeichens ist. Obwohl die sich daraus ergebenden über 65000 Codeelemente zur Codierung der meisten Zeichen der wichtigsten Sprachen der Welt ausreichen, bietet der Unicode-Standard außerdem einen Erweiterungsmechanismus, mit dem bis zu einer Million weiterer Zeichen codiert werden können. Der Erweiterungsmechanismus arbeitet mit einem Paar aus einem hohen und einem niedrigen Ersatzzeichen zur Codierung eines erweiterten Zeichens. Das erste (oder hohe) Ersatzzeichen besitzt einen Codewert zwischen U+D800 und U+DBFF und das zweite (oder niedrige) Ersatzzeichen einen Codewert zwischen U+DC00 and U+DFFF.

Der Standard 10646 (ISO/IEC 10646) der International Standards Organization (ISO) und der International Electrotechnical Commission (IEC) definiert den Universalzeichensatz Universal Multiple-Octet Coded Character Set (UCS), von dem es eine 16-Bit- (2-Byte-) und eine 32-Bitversion (4-Byteversion) gibt. UCS-2 ist mit der 16-Bit-Unicode-Form ohne Ersatzzeichen identisch. ISO/IEC 10646 definiert außerdem eine Erweiterungstechnik zum Codieren einiger UCS-4-Zeichen mit Hilfe von zwei UCS-2-Zeichen. Diese Erweiterung wird als UTF-16 bezeichnet und entspricht der 16-Bit-Unicode-Form mit Ersatzzeichen. Insgesamt besteht das UTF-16-Zeichenrepertoire aus allen UCS-2-Zeichen und der zusätzlichen eine Million von Zeichen, auf die über die Ersatzzeichenpaare zugegriffen werden kann.

Bei der Serialisierung von 16-Bit-Unicode-Zeichen in Byte platzieren einige Prozessoren das signifikanteste Byte in Anfangsposition (Big-Endian-Reihenfolge), während andere Prozessoren das am wenigsten signifikante Byte zuerst platzieren (Little-Endian-Reihenfolge). Die Standard-Bytereihenfolge für Unicode ist die Big-Endian-Reihenfolge.

Weitere Informationen zu Unicode stehen in der aktuellsten Ausgabe des Buchs "The Unicode Standard" sowie auf der Website von The Unicode Consortium (www.unicode.org) zur Verfügung.

UTF-8

16-Bit-Unicode-Zeichen stellen für byteorientierte und ASCII-basierte Anwendungen und Dateisystem ein großes Problem dar. Zum Beispiel könnten Anwendungen, die nicht auf Unicode ausgerichtet sind, die führenden acht Nullbit des Großbuchstabenzeichens 'A' (U+0041) als Einzelbytezeichen NULL fehlinterpretieren.

UTF-8 (UCS Transformation Format 8) ist eine algorithmische Umsetzung, mit der Unicode-Zeichen mit fester Länge in Bytefolgen variabler Länge umgesetzt werden. In UTF-8 werden ASCII-Zeichen und Steuerzeichen durch ihren gewöhnlichen Einzelbytecode dargestellt. Andere Zeichen werden jedoch zwei oder mehr Byte lang. Die Anzahl von Byte für jedes UTF-16-Zeichen im UTF-8-Format ist Tabelle 34 zu entnehmen.

Tabelle 34. UTF-8-Bitverteilung

Codewert (binär)	**UTF-16 (binär)**	**1. Byte (binär)**	**2. Byte (binär)**	**3. Byte (binär)**	**4. Byte (binär)**
00000000 0xxxxxxx	00000000 0xxxxxxx	0xxxxxxx			
00000yyy yyxxxxxx	00000yyy yyxxxxxx	110yyyyy	10xxxxxx		
zzzzyyyy yyxxxxxx	zzzzyyyy yyxxxxxx	1110zzzz	10yyyyyy	10xxxxxx	
uuuuu zzzzyyyy yyxxxxxx	110110ww wwzzzzyy 110111yy yyxxxxxx	11110uuu (mit uuuuu = wwww+1)	10uuzzzz	10yyyyyy	10xxxxxx

In den oben gezeigten Angaben entspricht eine Reihe von u, w, x, y und z jeweils der Bitdarstellung des Zeichens. Zum Beispiel wird U+0080 binär in 11000010 10000000 umgesetzt, und das Ersatzzeichenpaar U+D800 U+DC00 wird binär in 11110000 10010000 10000000 10000000 umgesetzt.

Unicode-Implementierung in DB2 UDB

DB2 UDB unterstützt UCS-2, d. h. Unicode ohne Ersatzzeichen.

In den Versionen von DB2 UDB vor Version 7.2 FixPak 4 behandelt DB2 UDB die beiden Zeichen in einem Ersatzzeichenpaar als zwei unabhängige Unicode-Zeichen. Daher führt die Umsetzung des Zeichenpaares von UTF-16/UCS-2 in UTF-8 zu zwei 3-Bytefolgen (siehe die vorletzte Zeile in Tabelle 34 auf Seite 297). Seit DB2 UDB Version 7.2 FixPak 4 erkennt DB2 UDB Ersatzzeichenpaare bei der Umsetzung zwischen UTF-16/UCS-2 und UTF-8, sodass ein Paar aus UTF-16-Ersatzzeichen in eine 4-Byte-UTF-8-Zeichenfolge umgesetzt wird (siehe letzte Zeile in Tabelle 34 auf Seite 297).

DB2 UDB behandelt jedes Unicode-Zeichen, einschließlich dieser Zeichen (außer Leerzeichen), wie das mit dem Akzentzeichen Akut kombinierende Zeichen (U+0301, KOMBINIERENDER AKZENT AKUT), als einzelnes Zeichen. Aus diesem Grund würde DB2 UDB nicht erkennen, dass das Zeichen LATIN KLEINER BUCHSTABE A MIT AKUT (U+00E1) kanonisch äquivalent zum Zeichen LATIN KLEINER BUCHSTABE A (U+0061), gefolgt vom Zeichen KOMBINIERENDER AKUT (U+0301), ist.

Nummern von Codepages/IDs für codierten Zeichensatz

Bei IBM wurde die UCS-2-Codepage als Codepage 1200 mit wachsendem Zeichensatz registriert. Das heißt, dass sich durch das Hinzufügen neuer Zeichen zu einer Codepage die Nummer der Codepage nicht ändert. Codepage 1200 verweist immer auf die aktuelle Version von Unicode.

Eine spezifische Version des UCS-Standards, wie durch Unicode 2.0 und ISO/IEC 10646-1 definiert, wurde auch bei IBM als ID für codierten Zeichensatz (CCSID) 13488 registriert. Diese ID für codierten Zeichensatz wird intern von DB2 UDB zum Speichern von Grafikzeichenfolgen in euc-Japan- und euc-Taiwan-Datenbanken verwendet. Die ID für codierten Zeichensatz 13488 und die Codepage 1200 verweisen beide auf UCS-2 und werden auf dieselbe Weise behandelt mit Ausnahme des Werts ihres „Doppelbyte"-Leerzeichens (DBCS):

CP/CCSID	Einzelbyte-Leerzeichen	Doppelbyte-Leerzeichen
1200	N/V	U+0020
13488	N/V	U+3000

HINWEIS: In einer UCS-2-Datenbank hat U+3000 keine spezielle Bedeutung.

Bei Konvertierungstabellen werden die gleichen Tabellen für beide verwendet, da Codepage 1200 eine Obermenge von CCSID 13488 ist.

Bei IBM wurde UTF-8 als ID für codierten Zeichensatz 1208 mit wachsendem Zeichensatz registriert (manchmal auch als Codepage 1208 bezeichnet). Wenn neue Zeichen zum Standard hinzugefügt werden, ändert sich diese Nummer (1208) nicht.

Die MBCS-Codepage-Nummer ist 1208. Dies ist die Codepage-Nummer der Datenbank und die Codepage von Zeichenfolgedaten in der Datenbank. Die Doppelbyte-Codepage-Nummer für UCS-2 ist 1200. Dies ist die Codepage von Grafikzeichenfolgedaten in der Datenbank. Wenn eine Unicode-Datenbank erstellt wird, werden CHAR-, VARCHAR-, LONG VARCHAR- und CLOB-Daten in UTF-8 und GRAPHIC-, VARGRAPHIC-, LONG VARGRAPHIC- und DBCLOB-Daten in UCS-2 gespeichert.

Erstellen einer Unicode-Datenbank

Standardmäßig werden Datenbanken in der Codepage der Anwendung erstellt, die sie erstellt. Wenn Sie daher Ihre Datenbank von einem Unicode-Client (UTF-8) aus erstellen (z. B. länderspezifische Angaben UNIVERSAL von AIX) oder wenn die Registrierungsvariable DB2CODEPAGE auf dem Client auf 1208 gesetzt ist, wird Ihre Datenbank als Unicode-Datenbank erstellt. Alternativ dazu können Sie explizit "UTF-8" als Namen für einen codierten Zeichensatz (CODESET) angeben und jeden gültigen Gebietscode (TERRITORY) verwenden, der von DB2 UDB unterstützt wird.

Setzen Sie z. B. den folgenden Befehl ab, um eine Unicode-Datenbank mit dem Gebietscode für die Vereinigten Staaten von Amerika zu erstellen:

```
DB2 CREATE DATABASE dbname USING CODESET UTF-8 TERRITORY US
```

Zum Erstellen einer Unicode-Datenbank mit der API **sqlecrea** sollten Sie die Werte in *sqledbcountryinfo* entsprechend setzen. Setzen Sie z. B. SQLDBCODESET auf den Wert `UTF-8` und SQLDBLOCALE auf einen gültigen Gebietscode (z. B. `US`).

Die Standardsortierfolge für eine UCS-2-Unicode-Datenbank ist IDENTITY, die die Zeichen nach ihren Codepunkten anordnet. Daher werden standardmäßig alle Unicode-Zeichen nach ihren Codepunkten geordnet und verglichen. Für die meisten Unicode-Zeichen sind die binären Sortierfolgen gleich, wenn sie in UTF-8 und UCS-2 codiert sind. Wenn Sie jedoch ein erweitertes Zeichen haben, zu dessen Codierung ein Ersatzzeichenpaar erforderlich ist, wird das Zeichen in der UTF-8-Codierung an das Ende sortiert, während das gleiche Zeichen in UCS-2-Codierung irgendwo in der Mitte eingereiht wird und seine beiden Ersatzzeichen getrennt werden können. Der Grund hierfür besteht darin, dass das erweiterte Zeichen bei der Codierung in UTF-8 einen 4-Bytewert der Form 11110xxx 10xxxxxx 10xxxxxx 10xxxxxx besitzt, der größer ist als die UTF-8-Codierung von U+FFFF. In UCS-2 ist das gleiche erweiterte Zeichen jedoch in Form von zwei UCS-2-Ersatzzeichen codiert, und DB2 UDB behandelt und sortiert jedes Ersatzzeichen einzeln.

Alle länderspezifischen Parameter wie Datums- oder Uhrzeitformat, Dezimaltrennzeichen u. a. basieren auf dem aktuellen Gebiet des Clients.

Eine Unicode-Datenbank erlaubt Verbindungen von jeder Codepage, die von DB2 UDB unterstützt wird. Codepage-Zeichenumsetzungen zwischen der Codepage des Clients und UTF-8 werden automatisch vom Datenbankmanager durchgeführt. Daten in Grafikzeichenfolgetypen sind immer in UCS-2 und werden keinen Codepage-Umsetzungen unterzogen. Die Umgebung des Befehlszeilenprozessors (CLP) ist eine Ausnahme. Wenn Sie mit einer SELECT-Anweisung Grafikzeichenfolgedaten (UCS-2) über den Befehlszeilenprozessor auswählen, werden die zurückgegebenen Grafikzeichenfolgedaten (vom Befehlszeilenprozessor) von UCS-2 in die Codepage Ihrer Client-Umgebung umgesetzt.

Jeder Client ist durch die Menge von Zeichen, die Eingabemethode und die von seiner Umgebung unterstützten Schriftarten beschränkt. Die UCS-2-Datenbank selbst jedoch akzeptiert und speichert alle UCS-2-Zeichen. Jeder Client arbeitet also mit einer Untergruppe von UCS-2-Zeichen, der Datenbankmanager lässt jedoch die Gesamtheit von UCS-2-Zeichen zu.

Wenn Zeichen von einer lokalen Codepage in UTF-8 umgesetzt werden, wird möglicherweise die Byteanzahl erweitert. Es gibt keine Erweiterung für ASCII-Zeichen, aber andere UCS-2-Zeichen werden um einen Faktor von zwei oder drei erweitert. Die Byteanzahl für jedes UCS-2-Zeichen im UTF-8-Format kann anhand der Tabelle in „UTF-8" auf Seite 297 bestimmt werden.

Datentypen

Alle von DB2 UDB unterstützten Datentypen werden auch in einer UCS-2-Datenbank unterstützt. Insbesondere Grafikzeichenfolgedaten werden für eine UCS-2-Datenbank unterstützt und in UCS-2/Unicode gespeichert. Jeder Client, einschließlich Clients mit SBCS-Zeichensatz, kann mit Grafikzeichenfolgedatentypen in UCS-2/Unicode arbeiten, wenn er mit einer UCS-2-Datenbank verbunden ist.

Eine UCS-2-Datenbank ist wie jede beliebige MBCS-Datenbank, bei der die Zeichenfolgedaten in der Anzahl von Byte gemessen werden. Beim Arbeiten mit Zeichenfolgedaten in UTF-8 sollte nicht davon ausgegangen werden, dass jedes Zeichen ein Byte lang ist. Bei UTF-8-Mehrbytecodierung ist jedes ASCII-Zeichen ein Byte lang, andere Zeichen nehmen jedoch jeweils zwei bis vier Byte ein. Dies sollte beim Definieren von CHAR-Feldern berücksichtigt werden. Je nach Verhältnis von ASCII-Zeichen zu anderen Zeichen kann ein CHAR-Feld der Größe von n Byte zwischen $n/4$ bis n Zeichen enthalten.

Die Verwendung der UTF-8-Zeichenfolgecodierung statt des UCS-2-Grafikzeichenfolgedatentyps wirkt sich ebenfalls auf die Gesamtspeicheranforderungen aus. Wenn die Mehrheit der Zeichen ASCII-Zeichen sind und

einige andere Zeichen dazwischenstehen, kann das Speichern von UTF-8-Daten die günstigere Alternative sein, da der Speicherbedarf eher einem Byte pro Zeichen entspricht. Wenn dagegen die Mehrheit der Zeichen keine ASCII-Zeichen sind, die auf drei oder vier Byte lange UTF-8-Folgen erweitert werden (z. B. Ideogramme), ist das UCS-2-Grafikzeichenfolgeformat möglicherweise eine bessere Alternative, da jede drei Byte lange UCS-2-Zeichenfolge zu einem 16-Bit-UCS-2-Zeichen wird, während jede vier Byte lange UFT-8-Zeichenfolge zu zwei 16-Bit-UCS-2-Zeichen wird.

In MBCS-Umgebungen operieren SQL-Skalarfunktionen, die mit Zeichenfolgen arbeiten, z. B. LENGTH, SUBSTR, POSSTR, MAX, MIN u. ä., mit der Anzahl von „Byte" statt mit der Anzahl von „Zeichen". Diese Funktionsweise ist in einer UCS-2-Datenbank unverändert. Sie sollten jedoch besonders vorsichtig sein, wenn Sie relative Positionen und Längen für eine UCS-2-Datenbank angeben, da diese Werte immer im Kontext der Datenbank-Codepage definiert werden. Im Fall einer UCS-2-Datenbank sollten diese relativen Positionen daher in UTF-8 definiert werden. Da einige Einzelbytezeichen in UTF-8 mehr als ein Byte benötigen, sind SUBSTR-Indizes, die für eine Einzelbyte-Datenbank gültig sind, möglicherweise für eine UCS-2-Datenbank nicht gültig. Wenn Sie falsche Indizes angeben, wird SQLCODE-Wert -191 (SQLSTATE 22504) zurückgegeben. Eine Beschreibung der Arbeitsweise dieser Funktionen finden Sie im Handbuch *SQL Reference*.

SQL-Datentypen CHAR werden in Benutzerprogrammen vom Datentyp `char` (in der Programmiersprache C) unterstützt. SQL-Datentypen GRAPHIC werden von `sqldbchar` in Benutzerprogrammen unterstützt. Bei einer UCS-2-Datenbank sind `sqldbchar`-Daten immer im Big-Endian-Format (hohes Byte zuerst). Wenn ein Anwendungsprogramm mit einer UCS-2-Datenbank verbunden ist, werden Zeichenfolgedaten zwischen der Codepage der Anwendung und UTF-8 durch DB2 UDB umgesetzt, Grafikzeichenfolgedaten bleiben jedoch immer in UCS-2.

Bezeichner

In einer UCS-2-Datenbank sind alle Bezeichner im UTF-8-Mehrbyteformat. Es ist daher möglich, jedes UCS-2-Zeichen in Bezeichnern zu verwenden, bei denen die Verwendung eines Zeichens im erweiterten Zeichensatz (z. B. ein Zeichen mit Akzent oder ein Mehrbytezeichen) von DB2 UDB zugelassen wird. Informationen darüber, welche Bezeichner die Verwendung erweiterter Zeichen zulassen, finden Sie in „Anhang A. Namenskonventionen" auf Seite 217.

Clients können alle Zeichen eingeben, die von ihrer Umgebung unterstützt werden. Alle Zeichen in den Bezeichnern werden vom Datenbankmanager in UTF-8 umgesetzt. Zwei Punkte müssen beim Angeben von Zeichen der Landessprache in Bezeichnern für eine UCS-2-Datenbank beachtet werden:

- Jedes Nicht-ASCII-Zeichen erfordert zwei bis vier Byte. Ein Bezeichner mit *n* Byte kann daher je nach Verhältnis von ASCII- zu anderen Zeichen nur zwischen *n*/4 und *n* Zeichen enthalten. Wenn Sie nur ein oder zwei Nicht-ASCII-Zeichen (z. B. Zeichen mit Akzent) haben, liegt die Grenze näher an *n* Zeichen, während für einen Bezeichner, der nur aus Nicht-ASCII-Zeichen besteht (z. B. in Japanisch), maximal *n*/4 bis *n*/3 Zeichen verwendet werden können.
- Wenn Bezeichner von unterschiedlichen Client-Umgebungen eingegeben werden sollen, sollten sie mit der gemeinsamen Untermenge von Zeichen definiert werden, die auf diesen Clients verfügbar sind. Wenn z. B. von Lateinisch-1-, arabischen oder japanischen Umgebungen auf eine UCS-2-Datenbank zugegriffen werden soll, sollten alle Bezeichner auf ASCII beschränkt sein.

Unicode-Literale

Unicode-Literale können auf zwei Arten angegeben werden:

- Als Grafikzeichenfolgekonstante mit dem Format G'...' oder N'....' gemäß der Beschreibung im Handbuch *SQL Reference*, Kapitel "Language Elements", Abschnitt "Graphic String Constants". Jedes auf diese Art angegebene Literal wird vom Datenbankmanager von der Codepage der Anwendung in 16-Bit-Unicode umgesetzt.
- Als hexadezimale Unicode-Zeichenfolge mit dem Format UX'....' oder GX'....': Die in Anführungszeichen nach UX oder GX angegebene Konstante muss ein Mehrfaches von vier hexadezimalen Ziffern in Big-Endian-Reihenfolge sein. Jede Gruppe aus vier Ziffern stellt jeweils einen 16-Bit-Unicode-Codepunkt dar. Beachten Sie, dass Ersatzzeichen immer in Paaren erscheinen, sodass Sie zur Darstellung des hohen und niedrigen Ersatzzeichens zwei vierziffrige Gruppen benötigen.

Bei Verwendung des Befehlszeilenprozessors (CLP) ist die erste Methode einfacher, wenn das UCS-2-Zeichen in der Codepage der lokalen Anwendung vorhanden ist (z. B. zur Eingabe von Zeichen der Codepage 850 von einem Terminal, das Codepage 850 verwendet). Die zweite Methode sollte für Zeichen verwendet werden, die außerhalb des Umfangs der Codepage der Anwendung liegen (z. B. zum Angeben von japanischen Zeichen von einem Terminal, das Codepage 850 verwendet).

Mustererkennung in einer UCS-2-Datenbank

Mustererkennung ist ein Bereich, in dem sich vorhandene MBCS-Datenbanken leicht von einer UCS-2-Datenbank unterscheiden.

MBCS-Datenbanken in DB2 UDB verhalten sich wie folgt: Wenn der Übereinstimmungsausdruck MBCS-Daten enthält, kann das Muster sowohl SBCS- als auch Nicht-SBCS-Zeichen enthalten. Die Sonderzeichen im Muster werden folgendermaßen interpretiert:

- Ein SBCS-Unterstreichungszeichen verweist auf ein SBCS-Zeichen.
- Ein DBCS-Unterstreichungszeichen verweist auf ein MBCS-Zeichen.
- Ein Prozentzeichen (SBCS oder DBCS) verweist auf eine Zeichenfolge von null oder mehr SBCS- oder Nicht-SBCS-Zeichen.

In einer Unicode-Datenbank gibt es keine echte Unterscheidung zwischen „Einzelbyte"- und „Doppelbyte"-Zeichen. Jedes 16-Bitzeichen nimmt zwei Byte ein. Obwohl das UTF-8-Format eine „Mischbyte"-Codierung von Unicode-Zeichen ist, gibt es in UTF-8 keine echte Unterscheidung zwischen SBCS- und MBCS-Zeichen. Jedes Zeichen ist ein Unicode-Zeichen, unabhängig von der Anzahl seiner Byte im UTF-8-Format. Bei der Angabe eines Zeichenfolge- oder Grafikzeichenfolgeausdrucks verweist ein Unterstreichungszeichen auf ein Unicode-Zeichen und ein Prozentzeichen auf eine Zeichenfolge von null oder mehr Unicode-Zeichen. Beachten Sie, dass Sie zwei Unterstreichungszeichen benötigen, um ein Ersatzpaarmuster anzugeben.

Auf der Client-Seite verwenden die Zeichenfolgeausdrücke die Codepage des Clients und werden vom Datenbankmanager in UTF-8 umgesetzt. SBCS-Codepages von Clients haben keine DBCS-Prozentzeichen oder -Unterstreichungszeichen. Jede unterstützte Codepage enthält jedoch ein Einzelbyte-Prozentzeichen (entspricht U+0025) und ein Einzelbyte-Unterstreichungszeichen (entspricht U+005F). Sonderzeichen werden in einer UCS-2-Datenbank folgendermaßen interpretiert:

- Ein SBCS-Unterstreichungszeichen (entspricht U+0025) verweist auf ein UCS-2-Zeichen in einem Grafikzeichenfolgeausdruck oder einem UTF-8-Zeichen in einem Zeichenfolgeausdruck.
- Ein SBCS-Prozentzeichen (entspricht U+005F) verweist auf eine Zeichenfolge von null oder mehr UCS-2-Zeichen in einem Grafikzeichenfolgeausdruck oder eine Zeichenfolge von null oder mehr UTF-8-Zeichen in einem Zeichenfolgeausdruck.

DBCS-Codepages unterstützen außerdem ein DBCS-Prozentzeichen (entspricht U+FF05) und ein DBCS-Unterstreichungszeichen (entspricht U+FF3F). Diese Zeichen haben für eine UCS-2-Datenbank keine besondere Bedeutung.

Für den wahlfreien „Escape-Ausdruck", der ein Zeichen angibt, das verwendet wird, um die spezielle Bedeutung des Unterstreichungszeichens und des Prozentzeichens aufzuheben, werden nur ASCII-Zeichen oder Zeichen unterstützt, die in eine UTF-8-Doppelbytefolge erweitert werden. Wenn Sie ein Escape-Zeichen angeben, das in einen 3 Byte langen UTF-8-Wert erweitert wird, wird eine Fehlermeldung (SQL0130N, SQLSTATE 22019) zurückgegeben.

Überlegungen zu IMPORT/EXPORT/LOAD
Die Dateiformate DEL, ASC und PC/IXF werden für eine UCS-2-Datenbank gemäß der Beschreibung in diesem Abschnitt unterstützt. Das Format WSF wird nicht unterstützt.

Beim Export von einer UCS-2-Datenbank in eine ASCII-DEL-Datei werden alle Zeichendaten in die Codepage der Anwendung umgesetzt. Sowohl Zeichenfolge- als auch Grafikzeichenfolgedaten werden in dieselbe SBCS- oder MBCS-Codepage des Clients umgesetzt. Dies ist die für den Export erwartete Funktionsweise von Datenbanken und kann nicht geändert werden, weil die gesamte ASCII-DEL-Datei nur eine Codepage haben kann. Wenn Sie also in eine ASCII-DEL-Datei exportieren, werden nur die UCS-2-Zeichen gespeichert, die in der Codepage Ihrer Anwendung vorhanden sind. Andere Zeichen werden durch die Standardsubstitutionszeichen für die Codepage der Anwendung ersetzt. Bei UTF-8-Clients (Codepage 1208) entsteht kein Datenverlust, weil alle UCS-2-Zeichen von UTF-8-Clients unterstützt werden.

Beim Importieren aus einer ASCII-Datei (DEL oder ASC) in eine UCS-2-Datenbank werden Zeichenfolgedaten aus der Codepage der Anwendung in UTF-8 und Grafikzeichenfolgedaten aus der Codepage der Anwendung in UCS-2 umgesetzt. Es entsteht kein Datenverlust. Wenn Sie ASCII-Daten importieren wollen, die unter einer anderen Codepage gespeichert wurden, sollten Sie die Codepage der Datendatei ändern, bevor Sie den Befehl IMPORT absetzen. Eine Möglichkeit hierzu besteht darin, DB2CODEPAGE auf die Codepage der ASCII-Datendatei zu setzen.

Der Bereich gültiger ASCII-Begrenzer für SBCS- und MBCS-Clients ist identisch mit dem derzeit von DB2 UDB für diese Clients unterstützten Bereich. Der Bereich gültiger Begrenzer für UTF-8-Clients ist X'01' bis X'7F' mit den üblichen Einschränkungen. Eine vollständige Liste dieser Einschränkungen finden Sie im Anhang zu Dateiformaten für die Dienstprogramme IMPORT/EXPORT/LOAD im Handbuch *Versetzen von Daten Dienstprogramme und Referenz*.

Beim Export aus einer UCS-2-Datenbank in eine PC/IXF-Datei werden alle Zeichenfolgedaten in die SBCS-/MBCS-Codepage des Clients umgesetzt. Grafikzeichenfolgedaten werden nicht umgesetzt und in UCS-2 (Codepage 1200) gespeichert. Es entsteht kein Datenverlust.

Beim Importieren aus einer PC/IXF-Datei in eine UCS-2-Datenbank wird bei Zeichenfolgedaten davon ausgegangen, dass sie in der in den PC/IXF-Kopfdaten gespeicherten SBCS-/MBCS-Codepage vorliegen, und bei Grafikzeichenfolgedaten, dass sie in der in den PC/IXF-Kopfdaten gespeicherten DBCS-Codepage vorliegen. Zeichenfolgedaten werden vom Importdienstprogramm von der in den PC/IXF-Kopfdaten angegebenen Codepage in die Codepage des Clients und dann (von der Anweisung INSERT) von der Codepage des Clients in UTF-8 umgesetzt. Grafikzeichenfolgedaten werden vom Importdienstprogramm von der in den PC/IXF-Kopfanweisung angegebenen DBCS-Codepage direkt in UCS-2 (Codepage 1200) umgesetzt.

Das Dienstprogramm LOAD setzt die Daten direkt in die Datenbank und geht standardmäßig davon aus, dass Daten in ASC- oder DEL-Dateien in der Codepage der Datenbank vorliegen. Es findet daher standardmäßig keine Codepage-Umsetzung für ASCII-Dateien statt. Wenn die Codepage für die Datendatei explizit (mit dem `codepage`-Wert) angegeben wurde, verwendet das Dienstprogramm LOAD diese Information zum Umsetzen der Daten aus der angegebenen Codepage in die Codepage der Datenbank, bevor es die Daten lädt. Bei PC/IXF-Dateien konvertiert das Dienstprogramm LOAD immer von den in den IXF-Kopfdaten angegebenen Codepages in die Codepage der Datenbank (1208 für CHAR und 1200 für GRAPHIC).

Die Codepage für DBCLOB-Dateien ist immer 1200 für UCS-2. Die Codepage von CLOB-Dateien entspricht der Codepage der importierten, geladenen oder exportierten Datendateien. Beim Laden oder Importieren von Daten mit dem Format PC/IXF wird beispielsweise angenommen, dass die CLOB-Datei in der in den PC/IXF-Kopfdaten angegebenen Codepage vorliegt. Wenn die DBCLOB-Datei das Format ASC oder DEL hat, nimmt das Dienstprogramm LOAD an, dass CLOB-Daten in der Codepage der Datenbank vorliegen (sofern dies nicht explizit mit dem `codepage`-Wert anders angegeben wird), während das Importdienstprogramm annimmt, dass die Daten in der Codepage der Client-Anwendung vorliegen.

Der nochecklengths-Wert wird für eine UCS-2-Datenbank aus folgenden Gründen immer angegeben:

- Jeder SBCS-Zeichensatz kann mit einer Datenbank verbunden sein, für die keine DBCS-Codepage vorhanden ist.
- Zeichenfolgen im UTF-8-Format haben in der Regel andere Längen als die Zeichenfolgen in Client-Codepages.

Weitere Informationen über die Dienstprogramme LOAD, IMPORT und EXPORT finden Sie im Handbuch *Versetzen von Daten Dienstprogramme und Referenz*.

Inkompatibilitäten

Für Anwendungen, die mit einer UCS-2-Datenbank verbunden sind, sind die Grafikzeichenfolgedaten immer in UCS-2 (Codepage 1200). Für Anwendungen, die mit Nicht-UCS-2-Datenbanken verbunden sind, haben die Grafikzeichenfolgedaten die DBCS-Codepage der Anwendung oder sind unzulässig, wenn die Codepage der Anwendung SBCS ist. Wenn z. B. ein 932-Client mit einer japanischen Nicht-UCS-2-Datenbank verbunden ist, haben die Grafikzeichenfolgedaten die Codepage 301. Für die 932-Client-Anwendungen, die mit einer UCS-2-Datenbank verbunden sind, sind die Grafikzeichenfolgedaten im UCS-2-Format.

Anhang E. Verwenden der DB2-Bibliothek

Die Bibliothek für DB2 Universal Database besteht aus Online-Hilfe, Handbüchern (PDF und HTML) und Beispielprogrammen in HTML-Format. Im Folgenden wird beschrieben, welche Informationen bereitgestellt werden und wie Sie darauf zugreifen können.

Über **Information - Unterstützung** können Sie online auf die Produktinformationen zugreifen. Weitere Informationen finden Sie in „Zugreifen auf Informationen mit ″Information - Unterstützung″“ auf Seite 326. Sie können sich im Web Informationen zu Tasks und zur Fehlerbehebung sowie DB2-Bücher, Beispielprogramme und DB2-Informationen anzeigen lassen.

PDF-Dateien und gedruckte Bücher für DB2

Informationen zu DB2

In der folgenden Tabelle sind die DB2-Handbücher in vier Kategorien unterteilt:

DB2-Benutzerhandbücher und -Referenzinformationen
Diese Bücher enthalten die allgemeinen DB2-Informationen für alle Plattformen.

DB2-Installations- und -Konfigurationsinformationen
Diese Bücher gelten für DB2 auf einer bestimmten Plattform. So steht beispielsweise jeweils ein separates Handbuch *Einstieg* (Quick Beginnings) für DB2 auf OS/2-, Windows- und UNIX-Plattformen zur Verfügung.

Plattformübergreifende Beispielprogramme in HTML
Bei diesen Beispielen handelt es sich um die HTML-Versionen der mit Application Development Client installierten Beispielprogramme. Sie dienen zur Information und können die Programme selbst nicht ersetzen.

Release-Informationen
Diese Dateien enthalten die neuesten Informationen, die in die DB2-Handbücher nicht mehr aufgenommen werden konnten.

Die Installationshandbücher, Release-Informationen und Lernprogramme können im HTML-Format direkt von der Produkt-CD-ROM angezeigt werden. Die meisten Handbücher stehen auf der Produkt-CD-ROM im HTML-Format zur Verfügung und können angezeigt werden. Auf der CD-ROM mit DB2-Veröffentlichungen stehen die Handbücher im PDF-Format zur Verfügung und

können mit Adobe Acrobat angezeigt und gedruckt werden. Darüber hinaus können Sie gedruckte Veröffentlichungen bei IBM bestellen. Siehe hierzu „Bestellen der gedruckten Handbücher" auf Seite 321. Die folgende Tabelle enthält eine Liste der Bücher, die bestellt werden können.

Auf OS/2- und Windows-Plattformen können Sie die HTML-Dateien im Verzeichnis `sqllib\doc\html` installieren. Die DB2-Informationen werden in verschiedene Sprachen übersetzt, jedoch nicht alle Informationen in alle Sprachen. Sind bestimmte Informationen in einer Sprache nicht verfügbar, wird stattdessen die englische Version dieser Informationen zur Verfügung gestellt.

Auf UNIX-Plattformen können Sie die HTML-Dateien in mehreren Sprachen installieren, und zwar in den Unterverzeichnissen `doc/%L/html`, wobei *%L* für den Code der jeweiligen Landessprache steht. Weitere Informationen finden Sie im entsprechenden Handbuch *Einstieg*.

Es gibt verschiedene Möglichkeiten, auf DB2-Bücher und -Informationen zuzugreifen:

- „Anzeigen von Online-Informationen" auf Seite 325
- „Suchen nach Online-Informationen" auf Seite 330
- „Bestellen der gedruckten Handbücher" auf Seite 321
- „Drucken der PDF-Handbücher" auf Seite 320

Tabelle 35. Informationen zu DB2

Name	Beschreibung	IBM Form PDF-Dateiname	HTML-Verzeichnis
DB2-Benutzerhandbücher und -Referenzinformationen			
Systemverwaltung	*Systemverwaltung: Konzept*. Dieses Handbuch enthält eine Übersicht über Datenbankkonzepte, Informationen zu Aspekten des Datenbankentwurfs (wie z. B. zum logischen und physischen Datenbankentwurf) sowie eine Erläuterung zur hohen Verfügbarkeit. *Systemverwaltung: Implementierung*. Dieses Handbuch enthält Informationen zu Implementierungsaspekten, wie beispielsweise zur Implementierung des Datenbankentwurfs, zum Zugriff auf Datenbanken sowie zu Prüfungs-, Sicherungs- und Wiederherstellungsverfahren. *Systemverwaltung: Optimierung*. Dieses Handbuch enthält Informationen zur Datenbankumgebung sowie zur Auswertung und Optimierung der Anwendungsleistung. Sie können die drei Bände des Handbuchs *Systemverwaltung* in englischer Sprache in den USA und Kanada über die Formnummer SBOF-8934 bestellen.	SC12-2879 db2d1g70 SC12-2878 db2d2g70 SC12-2877 db2d3g70	db2d0
Administrative API Reference	Dieses Handbuch enthält eine Beschreibung zu den DB2-Anwendungsprogrammierschnittstellen (APIs) und -Datenstrukturen, die Sie zum Verwalten Ihrer Datenbank verwenden können. Darüber hinaus wird in diesem Handbuch erläutert, wie Sie APIs von Ihren Anwendungen aus aufrufen können.	SC09-2947 db2b0e70	db2b0
Application Building Guide	Dieses Handbuch umfasst Informationen zur Umgebungskonfiguration sowie Anweisungsschritte zum Kompilieren, Verbinden und Ausführen von DB2-Anwendungen auf Windows-, OS/2- und UNIX-Plattformen.	SC09-2948 db2axe70	db2ax

Tabelle 35. Informationen zu DB2 (Forts.)

Name	Beschreibung	IBM Form PDF-Dateiname	HTML-Verzeichnis
APPC, CPI-C, and SNA Sense Codes	Dieses Handbuch enthält Basisinformationen zu APPC-, CPI-DFV- und SNA-Prüfcodes, die bei der Arbeit mit DB2 Universal Database-Produkten ausgegeben werden können. Nur im HTML-Format verfügbar.	Keine Formnummer db2ape70	db2ap
Application Development Guide	Dieses Handbuch enthält eine Erläuterung zur Entwicklung von Anwendungen, die mit Hilfe von eingebettetem SQL bzw. JAVA (JDBC und SQLJ) auf DB2-Datenbanken zugreifen. Unter anderem wird das Schreiben von gespeicherten Prozeduren, das Schreiben von benutzerdefinierten Funktionen, das Erstellen von benutzerdefinierten Typen, das Verwenden von Auslösern und das Entwickeln von Anwendungen in partitionierten Umgebungen oder mit Systemen zusammengeschlossener Datenbanken beschrieben.	SC09-2949 db2a0e70	db2a0
CLI Guide and Reference	Dieses Handbuch erklärt die Entwicklung von Anwendungen, die für den Zugriff auf DB2-Datenbanken DB2 Call Level Interface verwenden, eine aufrufbare SQL-Schnittstelle, die mit der Microsoft-ODBC-Spezifikation kompatibel ist.	SC09-2950 db2l0e70	db2l0
Command Reference	Dieses Handbuch enthält eine Erläuterung zur Verwendung des Befehlszeilenprozessors und eine Beschreibung der DB2-Befehle für die Datenbankverwaltung.	SC09-2951 db2n0e70	db2n0

Tabelle 35. Informationen zu DB2 (Forts.)

Name	Beschreibung	IBM Form PDF-Dateiname	HTML-Verzeichnis
Konnektivität Ergänzung	Dieses Handbuch enthält Konfigurations- und Referenzinformationen zur Verwendung von DB2 für AS/400, DB2 für OS/390, DB2 für MVS oder DB2 für VM als DRDA-Anwendungs-Requester mit DB2 Universal Database-Servern. Darüber hinaus enthält dieses Handbuch Informationen zur Verwendung von DRDA-Anwendungs-Servern mit DB2 Connect-Anwendungs-Requestern. Dieses Buch ist lediglich im HTML- und PDF-Format verfügbar.	Keine Formnummer db2h1g70	db2h1
Versetzen von Daten Dienstprogramme und Referenz	Dieses Handbuch enthält eine Erläuterung zur Verwendung der DB2-Dienstprogramme, wie beispielsweise IMPORT, EXPORT, LOAD, AUTOLOADER und DPROP, die das Verschieben von Daten vereinfachen.	SC12-2881 db2dmg70	db2dm
Data Warehouse-Zentrale Verwaltung	Dieses Handbuch enthält Informationen zur Erstellung und Verwaltung eines Data Warehouse mit Hilfe der Data Warehouse-Zentrale.	SC12-2885 db2ddg70	db2dd
Data Warehouse Center Application Integration Guide	Dieses Handbuch enthält Informationen, die Programmierer bei der Integration von Anwendungen in die Data Warehouse-Zentrale sowie in den Information Catalog Manager unterstützen.	SC26-9994 db2ade70	db2ad
DB2 Connect Benutzerhandbuch	Dieses Handbuch enthält eine Beschreibung der Konzepte der DB2 Connect-Produkte, allgemeine Informationen zur Verwendung sowie Informationen zur Programmierung dieser Produkte.	SC12-2880 db2c0g70	db2c0
DB2 Query Patroller Administration Guide	Dieses Handbuch enthält eine Übersicht über den Betrieb des DB2 Query Patroller-Systems, spezifische Informationen zum Systembetrieb und zur Verwaltung sowie Task-Informationen zu den GUI-Verwaltungsdienstprogrammen.	SC09-2958 db2dwe70	db2dw
DB2 Query Patroller User's Guide	In diesem Handbuch wird die Verwendung der Tools und Funktionen von DB2 Query Patroller beschrieben.	SC09-2960 db2wwe70	db2ww

Tabelle 35. Informationen zu DB2 (Forts.)

Name	Beschreibung	IBM Form PDF-Dateiname	HTML-Verzeichnis
Glossar	Dieses Handbuch enthält Definitionen zu den in DB2 und den zugehörigen Komponenten verwendeten Begriffen. Es ist im Handbuch *SQL Reference* enthalten und steht außerdem separat im HTML-Format zur Verfügung.	Keine Formnummer db2t0g70	db2t0
DB2 UDB Image, Audio und Video Extender Verwaltung und Programmierung	Dieses Handbuch enthält Basisinformationen zu DB2 Extender, Informationen zur Verwaltung und Konfiguration von IAV Extender sowie Informationen zur Programmierung mit Hilfe von IAV Extender. Es enthält Referenzinformationen, Diagnoseinformationen (mit Nachrichten) und Beispiele.	SC12-2892 dmbu7g70	dmbu7
Information Catalog Manager Systemverwaltung	Dieses Handbuch enthält eine Anleitung zur Verwaltung von Informationskatalogen.	SC12-2886 db2dig70	db2di
Information Catalog Manager Programming Guide and Reference	Dieses Handbuch enthält Definitionen für die Architekturschnittstellen für Information Catalog Manager.	SC26-9997 db2bie70	db2bi
Information Catalog Manager Benutzerhandbuch	Dieses Handbuch enthält Informationen zur Verwendung der Information Catalog Manager-Benutzerschnittstelle.	SC12-2887 db2aig70	db2ai
Installation und Konfiguration Ergänzung	Dieses Handbuch enthält Anweisungen zur Planung, Installation und Konfiguration von plattformspezifischen DB2-Clients. Darüber hinaus enthält es Informationen zu Bindevorgängen, zum Einrichten der Client/Server-Kommunikation, zu DB2-GUI-Tools, zu DRDR-AS, zur verteilten Installation, zur Konfiguration von verteilten Anforderungen sowie zum Zugriff auf heterogene Datenquellen.	GC12-2864 db2iyg70	db2iy

Tabelle 35. Informationen zu DB2 (Forts.)

Name	Beschreibung	IBM Form PDF-Dateiname	HTML-Verzeichnis
Fehlernachrichten	Dieses Handbuch enthält eine Liste der Nachrichten und Codes, die von DB2, vom Information Catalog Manager und von der Data Warehouse-Zentrale ausgegeben werden, sowie eine Beschreibung der jeweils erforderlichen Benutzeraktionen. Sie können beide Bände des Handbuchs *Fehlernachrichten* in englischer Sprache in den USA und Kanada unter der Formnummer SBOF-8932 bestellen.	Band 1 GC12-2875 db2m1g70 Band 2 GC12-2888 db2m2g70	db2m0
OLAP Integration Server Administration Guide	Dieses Handbuch enthält eine Erläuterung zur Verwendung der Komponente Administration Manager von OLAP Integration Server.	SC27-0782 db2dpe70	n/v
OLAP Integration Server Metaoutline User's Guide	Dieses Handbuch enthält eine Erläuterung zum Erstellen und Ausfüllen von OLAP-Metastrukturen mit Hilfe der OLAP Metaoutline-Standardschnittstelle (nicht mit Hilfe des OLAP Metaoutline Assistant).	SC27-0784 db2upe70	n/v
OLAP Integration Server Model User's Guide	Diese Handbuch enthält eine Erläuterung zum Erstellen von OLAP-Modellen mit Hilfe der OLAP Model-Standardschnittstelle (nicht mit Hilfe des OLAP Model Assistant).	SC27-0783 db2lpe70	n/v
OLAP Konfiguration und Benutzerhandbuch	Dieses Handbuch enthält Informationen zur Konfiguration und Einrichtung von OLAP Starter Kit.	SC12-2889 db2ipg70	db2ip
OLAP Tabellenkalkulations-Add-In Benutzerhandbuch für Excel	Dieses Handbuch enthält eine Beschreibung zur Verwendung des Tabellenkalkulationsprogramms Excel zum Analysieren von OLAP-Daten.	SC12-2890 db2epg70	db2ep
OLAP Tabellenkalkulations-Add-In Benutzerhandbuch für Lotus 1-2-3	Dieses Handbuch enthält eine Beschreibung zur Verwendung des Tabellenkalkulationsprogramms Lotus 1-2-3 zum Analysieren von OLAP-Daten.	SC12-2891 db2tpg70	db2tp
Replikation Benutzer- und Referenzhandbuch	Dieses Handbuch enthält Informationen zur Planung, Konfiguration, Verwaltung und Verwendung der mit DB2 gelieferten Replikations-Tools.	SC12-2884 db2e0g70	db2e0

Tabelle 35. Informationen zu DB2 (Forts.)

Name	Beschreibung	IBM Form PDF-Dateiname	HTML-Verzeichnis
Spatial Extender Benutzer- und Referenzhandbuch	Diese Handbuch enthält Informationen zur Installation, Konfiguration, Verwaltung, Programmierung und Fehlerbehebung für den Spatial Extender. Darüber hinaus enthält es zentrale Beschreibungen räumlicher Datenkonzepte sowie spezifische Referenzinformationen (Nachrichten und SQL) für den Spatial Extender.	SC12-2894 db2sbg70	db2sb
SQL Erste Schritte	Dieses Handbuch enthält eine Einführung in die SQL-Konzepte sowie Beispiele für eine Reihe von Konstrukten und Tasks.	SC12-2882 db2y0g70	db2y0
SQL Reference, Band 1 und Band 2	Dieses Handbuch beschreibt die Syntax, die Semantik und die Regeln von SQL. Darüber hinaus enthält das Handbuch Informationen zu Inkompatibilitäten zwischen Release-Ständen, Produkteinschränkungen und Katalogsichten. Sie können beide Bände des Handbuchs *SQL Reference* in englischer Sprache in den USA und Kanada unter der Formnummer SBOF-8933 bestellen.	Band 1 ^ SC09-2974 db2s1e70 Band 2 SC09-2975 db2s2e70	db2s0
System Monitor Guide and Reference	Dieses Handbuch enthält eine Beschreibung zum Sammeln unterschiedlicher Informationen zu Datenbanken und dem Datenbankmanager. In diesem Buch wird erläutert, wie Sie mit Hilfe dieser Informationen einen Einblick in Datenbankaktivitäten erhalten, die Leistung verbessern und Fehlerursachen feststellen können.	SC09-2956 db2f0e70	db2f0
Text Extender Verwaltung und Programmierung	Dieses Handbuch enthält Basisinformationen zu DB2 Extender, Informationen zur Verwaltung und Konfiguration von Text Extender sowie zur Programmierung mit Hilfe von Text Extender. Es bietet Referenzinformationen, Diagnoseinformationen (mit Nachrichten) und Beispiele.	SC12-2893 desu9g70	desu9

Tabelle 35. Informationen zu DB2 (Forts.)

Name	Beschreibung	IBM Form PDF-Dateiname	HTML-Verzeichnis
Troubleshooting Guide	Dieses Handbuch hilft Ihnen bei der Bestimmung von Fehlerquellen, bei der Fehlerbehebung sowie bei der Verwendung von Diagnose-Tools, wenn Sie den DB2-Kundendienst in Anspruch nehmen.	GC09-2850 db2p0e70	db2p0
Neue Funktionen	Dieses Handbuch enthält eine Beschreibung der neuen Einrichtungen, Funktionen und Erweiterungen in DB2 Universal Database Version 7.	SC12-2883 db2q0g70	db2q0
DB2-Installations- und -Konfigurationsinformationen			
DB2 Connect Enterprise Edition für OS/2 und Windows Einstieg	Dieses Handbuch enthält Informationen zur Planung, Migration, Installation und Konfiguration für DB2 Connect Enterprise Edition unter OS/2 und 32-Bit-Windows-Betriebssystemen. Darüber hinaus enthält dieses Handbuch Installations- und Konfigurationsinformationen für eine Reihe von unterstützten Clients.	GC12-2863 db2c6g70	db2c6
DB2 Connect Enterprise Edition für UNIX Einstieg	Dieses Handbuch enthält Informationen zur Planung, Migration, Installation, Konfiguration und Ausführung von Tasks für DB2 Connect Enterprise Edition auf UNIX-Plattformen. Darüber hinaus enthält dieses Handbuch Installations- und Konfigurationsinformationen für eine Reihe von unterstützten Clients.	GC12-2862 db2cyg70	db2cy
DB2 Connect Personal Edition Einstieg	Dieses Handbuch enthält Informationen zur Planung, Migration, Installation, Konfiguration und Ausführung von Tasks für DB2 Connect Personal Edition unter OS/2 und 32-Bit-Windows-Betriebssystemen. Darüber hinaus enthält dieses Handbuch Installations- und Konfigurationsinformationen für alle unterstützten Clients.	GC12-2869 db2c1g70	db2c1
DB2 Connect Personal Edition für Linux Einstieg	Dieses Handbuch enthält Informationen zur Planung, Installation, Migration und Konfiguration für DB2 Connect Personal Edition für alle unterstützten Linux-Varianten.	GC12-2865 db2c4g70	db2c4

Tabelle 35. Informationen zu DB2 (Forts.)

Name	Beschreibung	IBM Form PDF-Dateiname	HTML-Verzeichnis
DB2 Data Links Manager Einstieg	Dieses Handbuch enthält Informationen zur Planung, Installation, Konfiguration und Ausführung von Tasks für DB2 Data Links Manager unter AIX und 32-Bit-Windows-Betriebssystemen.	GC12-2868 db2z6g70	db2z6
DB2 Enterprise - Extended Edition für UNIX Einstieg	Dieses Handbuch enthält Informationen zur Planung, Installation und Konfiguration für DB2 Enterprise - Extended Edition auf UNIX-Plattformen. Darüber hinaus enthält dieses Handbuch Installations- und Konfigurationsinformationen für eine Reihe von unterstützten Clients.	GC12-2867 db2v3g70	db2v3
DB2 Enterprise - Extended Edition für Windows Einstieg	Dieses Handbuch enthält Informationen zur Planung, Installation und Konfiguration für DB2 Enterprise - Extended Edition unter 32-Bit-Windows-Betriebssystemen. Darüber hinaus enthält dieses Handbuch Installations- und Konfigurationsinformationen für eine Reihe von unterstützten Clients.	GC12-2866 db2v6g70	db2v6
DB2 für OS/2 Einstieg	Dieses Handbuch enthält Informationen zur Planung, Migration, Installation und Konfiguration von DB2 Universal Database für das Betriebssystem OS/2. Darüber hinaus enthält dieses Handbuch Installations- und Konfigurationsinformationen für eine Reihe von unterstützten Clients.	GC12-2870 db2i2g70	db2i2
DB2 für UNIX Einstieg	Dieses Handbuch enthält Informationen zur Planung, Migration, Installation und Konfiguration von DB2 Universal Database auf UNIX-Plattformen. Darüber hinaus enthält dieses Handbuch Installations- und Konfigurationsinformationen für eine Reihe von unterstützten Clients.	GC12-2872 db2ixg70	db2ix

Tabelle 35. Informationen zu DB2 (Forts.)

Name	Beschreibung	IBM Form PDF-Dateiname	HTML-Verzeichnis
DB2 für Windows Einstieg	Dieses Handbuch enthält Informationen zur Planung, Installation, Migration und Konfiguration für DB2 Universal Database unter 32-Bit-Windows-Betriebssystemen. Darüber hinaus enthält dieses Handbuch Installations- und Konfigurationsinformationen für eine Reihe von unterstützten Clients.	GC12-2873 db2i6g70	db2i6
DB2 Personal Edition Einstieg	Dieses Handbuch enthält Informationen zur Planung, Installation, Migration und Konfiguration für DB2 Universal Database Personal Edition unter OS/2 und 32-Bit-Windows-Betriebssystemen.	GC12-2871 db2i1g70	db2i1
DB2 Personal Edition für Linux Einstieg	Dieses Handbuch enthält Informationen zur Planung, Installation, Migration und Konfiguration für DB2 Universal Database Personal Edition für alle unterstützten Linux-Varianten.	GC12-2874 db2i4g70	db2i4
DB2 Query Patroller Installation Guide	Dieses Handbuch enthält Installationsinformationen zu DB2 Query Patroller.	GC09-2959 db2iwe70	db2iw
DB2 Warehouse Manager Installation	Dieses Handbuch enthält Installationsinformationen für Warehouse-Agenten, Warehouse-Umsetzungsprogramme und den Information Catalog Manager.	GC12-2876 db2ide70	db2id
Plattformübergreifende Beispielprogramme in HTML			
Beispielprogramme in HTML	Dieses Handbuch enthält die Beispielprogramme für die Programmiersprachen auf allen von DB2 unterstützten Plattformen im HTML-Format. Die Beispielprogramme werden lediglich zu Informationszwecken zur Verfügung gestellt. Nicht alle Beispiele sind für alle Programmiersprachen verfügbar. Die HTML-Beispiele stehen nur dann zur Verfügung, wenn der DB2 Application Development Client installiert ist. Weitere Informationen zu den Programmen finden Sie im Handbuch *Application Building Guide*.	Keine Formnummer	db2hs

Tabelle 35. Informationen zu DB2 (Forts.)

Name	Beschreibung	IBM Form PDF-Dateiname	HTML-Verzeichnis
	Release-Informationen		
DB2 Connect Release-Informationen	Dieses Dokument enthält die neuesten Informationen, die in die DB2 Connect-Handbücher nicht mehr aufgenommen werden konnten.	Siehe Anmerkung 2.	db2cr
DB2 Installationsinformationen	Dieses Dokument enthält die neuesten Informationen zur Installation, die in die DB2-Handbücher nicht mehr aufgenommen werden konnten.	Nur auf der Produkt-CD-ROM verfügbar.	
DB2-Release-Informationen	Dieses Dokument enthält die neuesten Informationen zu allen DB2-Produkten und -Funktionen, die in die DB2-Handbücher nicht mehr aufgenommen werden konnten.	Siehe Anmerkung 2.	db2ir

Anmerkungen:

1. Das Zeichen an der sechsten Stelle des Dateinamens gibt die Landessprache eines Buchs an. So kennzeichnet der Dateiname `db2d0e70` die englische Version des Handbuchs *Systemverwaltung*, der Dateiname `db2d0f70` kennzeichnet die französische Version des Buchs. Folgende Buchstaben werden an der sechsten Stelle des Dateinamens verwendet, um die Landessprache für ein Handbuch anzugeben:

Sprache	Kennung
Brasilianisches Portugiesisch	b
Bulgarisch	u
Dänisch	d
Deutsch	g
Englisch	e
Finnisch	y
Französisch	f
Griechisch	a
Italienisch	i
Japanisch	j
Koreanisch	k
Niederländisch	q
Norwegisch	n
Polnisch	p
Portugiesisch	v
Russisch	r
Schwedisch	s
Slowenisch	l
Spanisch	z
Trad. Chinesisch	t
Tschechisch	x
Türkisch	m
Ungarisch	h
Vereinf. Chinesisch	c

2. Kurzfristig verfügbare Informationen, die in die DB2-Handbücher nicht mehr aufgenommen werden können, sind in den Release-Informationen enthalten, die im HTML-Format und als ASCII-Datei verfügbar sind. Die HTML-Version steht über 'Information - Unterstützung' und auf den Produkt-CD-ROMs zur Verfügung. Gehen Sie wie folgt vor, um die ASCII-Dateien anzuzeigen:
   - Rufen Sie auf UNIX-Plattformen die Datei `Release.Notes` auf. Diese Datei befindet sich im Verzeichnis `DB2DIR/Readme/%L`. Dabei ist `%L` die länderspezifische Angabe und `DB2DIR` eine der folgenden Angaben:
     - `/usr/lpp/db2_07_01` (unter AIX)
     - `/opt/IBMdb2/V7.1` (unter HP-UX, PTX, Solaris und Silicon Graphics IRIX)
     - `/usr/IBMdb2/V7.1` (unter Linux)
   - Rufen Sie auf anderen Plattformen die Datei `RELEASE.TXT` auf. Diese Datei befindet sich in dem Verzeichnis, in dem das Produkt installiert ist. Auf OS/2-Plattformen können Sie auch den Ordner **IBM DB2** und anschließend das Symbol **Release-Informationen** doppelt anklicken.

Drucken der PDF-Handbücher

Wenn Sie eine gedruckte Version der Handbücher bevorzugen, können Sie die PDF-Dateien auf der CD-ROM mit DB2-Veröffentlichungen ausdrucken. Mit Adobe Acrobat Reader können Sie entweder das gesamte Handbuch oder bestimmte Teile des Handbuchs ausdrucken. Die Namen der einzelnen Handbücher in der Bibliothek finden Sie in Tabelle 35 auf Seite 309.

Die neueste Version von Adobe Acrobat Reader finden Sie auf der Adobe-Web-Site unter http://www.adobe.com.

Die PDF-Dateien befinden sich auf der CD-ROM mit DB2-Veröffentlichungen und haben die Dateierweiterung PDF. Führen Sie folgende Schritte aus, um auf die PDF-Dateien zuzugreifen:

1. Legen Sie die CD-ROM mit DB2-Veröffentlichungen in das CD-ROM-Laufwerk ein. Auf UNIX-Plattformen: Hängen Sie die CD-ROM mit den DB2-Veröffentlichungen an. Das Handbuch *Einstieg* enthält Anweisungen zu den Mount-Prozeduren.
2. Starten Sie Acrobat Reader.
3. Öffnen Sie die gewünschte PDF-Datei von einer der folgenden Positionen aus:
   - Auf OS/2- und Windows-Plattformen:

     Verzeichnis *x*`:\doc\`*sprache* . Dabei gibt *x* das CD-ROM-Laufwerk an, *sprache* den zweistelligen Landescode für die verwendete Sprache (z. B. EN für Englisch).

- Auf UNIX-Plattformen:

  Verzeichnis */cdrom*/doc/%*L* auf der CD-ROM. Dabei gibt */cdrom* den Mount-Punkt der CD-ROM an und %*L* den Namen der gewünschten länderspezifischen Angaben.

Sie können die PDF-Dateien auch von der CD-ROM in ein lokales Laufwerk oder ein Netzlaufwerk kopieren und sie von dort aus lesen.

Bestellen der gedruckten Handbücher

Sie können die gedruckten DB2-Handbücher einzeln bestellen. In den USA und Kanada ist es außerdem möglich, mehrere Bücher als Paket unter einer SBOF-Nummer zu bestellen. Setzen Sie sich mit Ihrem IBM Vertragshändler oder Vertriebsbeauftragten in Verbindung, oder bestellen Sie die Handbücher telefonisch bei IBM Direkt unter der Nummer 0180/55 090. Darüber hinaus können Sie die Handbücher über die Web-Seite mit Veröffentlichungen unter http://www.elink.ibmlink.ibm.com/pbl/pbl bestellen.

Es sind zwei Gruppen von Handbüchern verfügbar. Die Gruppe mit der Formnummer SBOF-8935 umfasst Referenzinformationen und Informationen zur Verwendung für DB2 Warehouse Manager. Die Gruppe mit der Formnummer SBOF-8931 umfasst Referenzinformationen und Informationen zur Verwendung für alle anderen DB2 Universal Database-Produkte und -Funktionen. Der Inhalt der SBOF-Gruppen ist in der folgenden Tabelle aufgeführt.

Tabelle 36. Bestellen der gedruckten Handbücher

SBOF-Nummer	In dieser Gruppe enthaltene Handbücher	
SBOF-8931	• Administration Guide: Planning • Administration Guide: Implementation • Administration Guide: Performance • Administrative API Reference • Application Building Guide • Application Development Guide • CLI Guide and Reference • Command Reference • Data Movement Utilities Guide and Reference • Data Warehouse Center Administration Guide • Data Warehouse Center Application Integration Guide • DB2 Connect User's Guide • Installation and Configuration Supplement • Image, Audio, and Video Extenders Administration and Programming • Message Reference, Volumes 1 and 2	• OLAP Integration Server Administration Guide • OLAP Integration Server Metaoutline User's Guide • OLAP Integration Server Model User's Guide • OLAP Integration Server User's Guide • OLAP Setup and User's Guide • OLAP Spreadsheet Add-in User's Guide for Excel • OLAP Spreadsheet Add-in User's Guide for Lotus 1-2-3 • Replication Guide and Reference • Spatial Extender Administration and Programming Guide • SQL Getting Started • SQL Reference, Volumes 1 and 2 • System Monitor Guide and Reference • Text Extender Administration and Programming • Troubleshooting Guide • What's New
SBOF-8935	• Information Catalog Manager Administration Guide • Information Catalog Manager User's Guide • Information Catalog Manager Programming Guide and Reference	• Query Patroller Administration Guide • Query Patroller User's Guide

DB2-Online-Dokumentation

Zugreifen auf die Online-Hilfefunktion

Die Online-Hilfefunktion ist für alle DB2-Komponenten verfügbar. In der folgenden Tabelle werden die verschiedenen Hilfearten beschrieben.

Hilfearten	Inhalt	Zugriff
Hilfe für Befehl	Erklärt die Syntax von Befehlen im Befehlszeilenprozessor.	Geben Sie im interaktiven Modus des Befehlszeilenprozessors Folgendes ein: `? befehl` Dabei stellt *befehl* ein Schlüsselwort bzw. den vollständigen Befehl dar. So kann beispielsweise durch die Eingabe von `? catalog` Hilfe für alle CATALOG-Befehle angezeigt werden, während mit `? catalog database` lediglich Hilfe für den Befehl CATALOG DATABASE angezeigt wird.
Hilfe für **Client-Konfiguration - Unterstützung** *Hilfe für die Befehlszentrale* *Hilfe für die Steuerzentrale* *Hilfe für die Data Warehouse-Zentrale* *Hilfe für* **Event Analyzer** *Hilfe für Information Catalog Manager* *Hilfe für die Satellitenverwaltungszentrale* *Hilfe für die Prozedurenzentrale*	Erläutert die Tasks, die Sie in einem Fenster oder Notizbuch ausführen können. Die Hilfe umfasst Übersichtsinformationen und unbedingt erforderliche Informationen sowie eine Beschreibung zur Verwendung der Steuerelemente im Fenster oder Notizbuch.	Klicken Sie in einem Fenster oder in einem Notizbuch den Druckknopf **Hilfe** an oder drücken Sie die Taste **F1**.

Hilfearten	Inhalt	Zugriff

Nachrichtenhilfe	Beschreibt die Ursache von Nachrichten sowie die auszuführenden Benutzeraktionen.	Geben Sie im interaktiven Modus des Befehlszeilenprozessors Folgendes ein:
		`? XXXnnnnn`
		Dabei ist *XXXnnnnn* eine gültige Nachrichtenkennung.
		Bei Eingabe von `? SQL30081` wird z. B. die Hilfe zur Nachricht SQL30081 angezeigt.
		Wenn Sie die Nachrichtenhilfe seitenweise anzeigen möchten, geben Sie den folgenden Befehl ein:
		`? XXXnnnnn | more`
		Geben Sie folgenden Befehl ein, um die Nachrichtenhilfe in einer Datei zu speichern:
		`? XXXnnnnn > datei.erw`
		Dabei ist *datei.erw* die Datei, in der Sie die Nachrichtenhilfe speichern möchten.

Hilfe für SQL	Erklärt die Syntax von SQL-Anweisungen.	Geben Sie im interaktiven Modus des Befehlszeilenprozessors Folgendes ein: `help anweisung` Dabei gibt *anweisung* eine SQL-Anweisung an. So kann beispielsweise durch die Eingabe von `help SELECT` die Hilfe zur Anweisung SELECT angezeigt werden. **Anmerkung:** Die Hilfe für SQL ist auf UNIX-Plattformen nicht verfügbar.
SQLSTATE-Hilfe	Erklärt SQLSTATE-Werte und SQL-Klassencodes.	Geben Sie im interaktiven Modus des Befehlszeilenprozessors Folgendes ein: `? sqlstate` oder `? klassencode` Dabei ist *sqlstate* ein gültiger, fünfstelliger SQL-Status, und *klassencode* stellt die ersten zwei Ziffern des SQL-Statuswerts dar. So kann beispielsweise durch die Eingabe von `? 08003` Hilfe für den SQL-Statuswert 08003 angezeigt werden, während mit `? 08` Hilfe für den Klassencode 08 angezeigt wird.

Anzeigen von Online-Informationen

Die zum Lieferumfang dieses Produkts gehörenden Handbücher werden als Softcopy im HTML-Format (HTML - Hypertext Markup Language) bereitgestellt. In einer Softcopy können Sie die Informationen auf einfache Art suchen und anzeigen und über Hypertextverbindungen auf zugehörige Informationen zugreifen. Außerdem wird die gemeinsame Nutzung der Bibliothek in Ihrem gesamten Unternehmen erleichtert.

Sie können die Online-Bücher und Beispielprogramme mit jedem Browser anzeigen, der den Spezifikationen von HTML Version 3.2 entspricht.

Führen Sie die nachfolgend beschriebenen Schritte aus, um Online-Bücher oder Beispielprogramme anzuzeigen:

- Wenn Sie DB2-Verwaltungs-Tools ausführen, verwenden Sie **Information - Unterstützung**.
- Klicken Sie in einem Browser **Datei—>Seite öffnen** an. Die geöffnete Seite enthält eine Übersicht über die DB2-Informationen und Verbindungen (Links) zu diesen Informationen:
  - Öffnen Sie auf UNIX-Plattformen die folgende Seite:

    ```
    INSTHOME/sqllib/doc/%L/html/index.htm
    ```

    Dabei ist *%L* die länderspezifische Angabe.
  - Öffnen Sie auf anderen Plattformen die folgende Seite:

    ```
    sqllib\doc\html\index.htm
    ```

    Der Pfad befindet sich auf dem Laufwerk, auf dem DB2 installiert ist.

    Wenn Sie **Information - Unterstützung** nicht installiert haben, können Sie die Seite öffnen, indem Sie das Symbol **DB2-Informationen** doppelt anklicken. Je nach verwendetem Betriebssystem befindet sich das Symbol im Hauptproduktordner bzw. unter Windows im Menü **Start**.

Installieren des Netscape-Browsers

Wenn Sie nicht bereits einen Web-Browser installiert haben, können Sie Netscape von der im Lieferumfang des Produkts enthaltenen Netscape-CD-ROM aus installieren. Führen Sie folgende Schritte aus, um ausführliche Informationen zur Installation zu erhalten:

1. Legen Sie die Netscape-CD-ROM ein.
2. Nur auf UNIX-Plattformen: Hängen Sie die CD-ROM an. Das Handbuch *Einstieg* enthält Anweisungen zu den Mount-Prozeduren.
3. Installationsanweisungen finden Sie in der Datei `CDNAV`*nn*`.txt`. Dabei ist *nn* die zweistellige Landeskennung. Die Datei befindet sich im Stammverzeichnis der CD-ROM.

Zugreifen auf Informationen mit "Information - Unterstützung"

Information - Unterstützung ermöglicht Ihnen den schnellen Zugriff auf DB2-Produktinformationen. **Information - Unterstützung** ist auf allen Plattformen mit DB2-Verwaltungs-Tools verfügbar.

Sie können 'Information - Unterstützung' öffnen, indem Sie das entsprechende Symbol doppelt anklicken. Abhängig vom verwendeten System befindet sich das Symbol im Hauptproduktordner im Ordner 'Information' bzw. unter Windows im Menü **Start**.

Sie können auf 'Information - Unterstützung' auch zugreifen, indem Sie die Funktionsleiste und das Menü **Hilfe** auf der DB2-Windows-Plattform verwenden.

Unter 'Information - Unterstützung' finden Sie sechs verschiedene Arten von Informationen. Klicken Sie die entsprechende Indexzunge an, um die für diese Informationsart verfügbaren Themen aufzurufen.

Funktionen Die Hauptfunktionen, die Sie mit DB2 ausführen können.

Referenz DB2-Referenzinformationen, wie beispielsweise Schlüsselwörter, Befehle und APIs.

Handbücher DB2-Handbücher.

Fehlerbehebung
Kategorien von Fehlernachrichten sowie die entsprechenden Benutzeraktionen.

Beispielprogramme
Beispielprogramme, die in DB2 Application Development Client enthalten sind. Wenn Sie DB2 Application Development Client nicht installiert haben, wird diese Indexzunge nicht angezeigt.

Web DB2-Informationen im World Wide Web. Sie müssen über Ihr System eine Verbindung zum Web herstellen können, um auf diese Informationen zugreifen zu können.

Wenn Sie einen Eintrag aus einer der Listen auswählen, startet **Information - Unterstützung** eine Funktion zum Anzeigen der Informationen. Bei der Anzeigefunktion kann es sich abhängig von der ausgewählten Informationsart um die Hilfeanzeige des Systems, einen Editor oder einen Web-Browser handeln.

In 'Information - Unterstützung' steht eine Suchfunktion zur Verfügung, mit der Sie nach einem bestimmten Thema suchen können, ohne in den Listen blättern zu müssen.

Rufen Sie über die Hypertextverbindung in 'Information - Unterstützung' das Suchformular **In DB2-Online-Informationen suchen** auf.

Der HTML-Such-Server wird normalerweise automatisch gestartet. Wenn eine Suche in HTML-Informationen fehlschlägt, müssen Sie möglicherweise mit einer der nachfolgend aufgeführten Methoden den Such-Server starten:

Unter Windows
Klicken Sie **Start** an und wählen Sie **Programme —> IBM DB2 —> Informationen —> HTML-Such-Server starten** aus.

Unter OS/2
Klicken Sie den Ordner **DB2 für OS/2** und anschließend das Symbol für **HTML-Such-Server starten** doppelt an.

Falls andere Probleme bei der Suche in HTML-Informationen auftreten, finden Sie möglicherweise entsprechende Hinweise in den Release-Informationen.

Anmerkung: Die Suchfunktion steht in Linux-, PTX- und Silicon Graphics IRIX-Umgebungen nicht zur Verfügung.

Verwenden der DB2-Assistenten

Assistenten unterstützen Sie bei der Ausführung bestimmter Verwaltungsaufgaben, indem sie Sie Schritt für Schritt durch jede Aufgabe führen. Assistenten stehen über die Steuerzentrale und 'Client-Konfiguration - Unterstützung' zur Verfügung. In der folgenden Tabelle sind die einzelnen Assistenten und deren Verwendungszweck aufgeführt.

Anmerkung: In Umgebungen mit partitionierten Datenbanken sind die Assistenten **Datenbank erstellen**, **Index erstellen**, **Aktualisierung auf mehreren Systemen konfigurieren** und **Leistungskonfiguration** verfügbar.

Assistent	Verwendung	Zugriff
Datenbank hinzufügen	Katalogisieren einer Datenbank auf einer Client-Workstation.	Klicken Sie in **Client-Konfiguration - Unterstützung** die Option **Hinzufügen** an.
Datenbank sichern	Festlegen, Erstellen und Terminieren eines Sicherungsplans.	Klicken Sie in der Steuerzentrale die zu sichernde Datenbank mit der rechten Maustaste an und wählen Sie **Sichern —> Datenbank mit Assistent** aus.
Aktualisierung auf mehreren Systemen konfigurieren	Konfigurieren einer Aktualisierung auf mehreren Systemen, einer verteilten Transaktion oder einer zweiphasigen Festschreibung.	Klicken Sie in der Steuerzentrale den Ordner **Datenbanken** mit der rechten Maustaste an und wählen Sie **Aktualisierung auf mehreren Systemen** aus.

Assistent	Verwendung	Zugriff
Datenbank erstellen	Erstellen einer Datenbank und Ausführen einiger grundlegender Konfigurationsfunktionen.	Klicken Sie in der Steuerzentrale den Ordner **Datenbanken** mit der rechten Maustaste an und wählen Sie **Erstellen —> Datenbank mit Assistent** aus.
Tabelle erstellen	Auswählen eines Basisdatentyps und Erstellen eines Primärschlüssels für die Tabelle.	Klicken Sie in der Steuerzentrale das Symbol **Tabellen** mit der rechten Maustaste an und wählen Sie **Erstellen —> Tabelle mit Assistent** aus.
Tabellenbereich erstellen	Erstellen eines neuen Tabellenbereichs.	Klicken Sie in der Steuerzentrale das Symbol **Tabellenbereiche** mit der rechten Maustaste an und wählen Sie **Erstellen —> Tabellenbereich mit Assistent** aus.
Index erstellen	Hinweise zum Erstellen und Löschen von Indizes für Ihre Abfragen.	Klicken Sie in der Steuerzentrale das Symbol **Index** mit der rechten Maustaste an und wählen Sie **Erstellen —> Index mit Assistent** aus.
Leistungskonfiguration	Optimieren der Leistung einer Datenbank durch Aktualisieren der Konfigurationsparameter, so dass sie den Anforderungen Ihres Unternehmens entsprechen.	Klicken Sie in der Steuerzentrale die Datenbank, die optimiert werden soll, mit der rechten Maustaste an und wählen Sie **Leistung mit Assistent konfigurieren** aus. Klicken Sie in einer Umgebung mit partitionierten Datenbanken in der Sicht für Datenbankpartitionen die erste Datenbankpartition, die optimiert werden soll, mit der rechten Maustaste an und wählen Sie **Leistung mit Assistent konfigurieren** aus.
Datenbank wiederherstellen	Wiederherstellen einer Datenbank nach einem Fehler. Dieser Assistent hilft Ihnen, zu entscheiden, welche Sicherungskopie Sie verwenden und welche Protokolle Sie erneut abarbeiten.	Klicken Sie in der Steuerzentrale die Datenbank, die wiederhergestellt werden soll, mit der rechten Maustaste an und wählen Sie **Wiederherstellen —> Datenbank mit Assistent** aus.

Einrichten eines Dokument-Servers

Die DB2-Informationen werden standardmäßig auf Ihrem lokalen System installiert. Das bedeutet, dass alle Benutzer, die Zugriff auf DB2-Informationen benötigen, dieselben Dateien installieren müssen. Führen Sie folgende Schritte aus, um die DB2-Informationen an einer einzigen Position zu speichern:

1. Kopieren Sie alle Dateien und Unterverzeichnisse aus dem Verzeichnis `\sqllib\doc\html` Ihres lokalen Systems auf einen Web-Server. Jedem Handbuch ist ein Unterverzeichnis zugeordnet, das alle erforderlichen HTML- und GIF-Dateien enthält, aus denen das Handbuch besteht. Stellen Sie sicher, dass die Verzeichnisstruktur erhalten bleibt.
2. Konfigurieren Sie den Web-Server so, dass er die Dateien an der neuen Speicherposition sucht. Informationen hierzu finden Sie im Anhang zu NetQuestion im Handbuch *Installation und Konfiguration Ergänzung*.
3. Wenn Sie die Java-Version von **Information - Unterstützung** verwenden, können Sie eine Basis-URL-Adresse für alle HTML-Dateien angeben. Sie sollten die URL-Adresse für das Bücherverzeichnis verwenden.
4. Wenn Sie die Buchdateien anzeigen können, ist es möglich, bei häufig aufgerufenen Themen Lesezeichen zu setzen. Es empfiehlt sich, folgende Seiten mit einem Lesezeichen zu versehen:
   - Bücherverzeichnis
   - Inhaltsverzeichnis häufig verwendeter Handbücher
   - Themen, auf die häufig verwiesen wird, wie beispielsweise zum Ändern von Tabellen
   - Suchformular

Informationen dazu, wie Sie die DB2 Universal Database-Online-Dokumentationsdateien auf einer zentralen Maschine zur Verfügung stellen können, finden Sie im Anhang zu NetQuestion im Handbuch *Installation und Konfiguration Ergänzung*.

Suchen nach Online-Informationen

Verwenden Sie eine der folgenden Methoden, um nach Informationen in den HTML-Dateien zu suchen:

- Klicken Sie im obersten Rahmen auf **Suchen**. Verwenden Sie das Suchformular, um nach einem bestimmten Thema zu suchen. Diese Funktion steht in Linux-, PIX- oder Silicon Graphics IRIX-Umgebungen nicht zur Verfügung.
- Klicken Sie im obersten Rahmen auf **Index**. Mit Hilfe des Indexes können Sie nach einem bestimmten Thema im Buch suchen.
- Rufen Sie das Inhaltsverzeichnis oder den Index der Hilfe oder des HTML-Buchs auf und verwenden Sie die Suchfunktion des Web-Browsers, um nach einem bestimmten Thema im Buch zu suchen.
- Mit Hilfe der Lesezeichenfunktion des Web-Browsers können Sie schnell zu einem bestimmten Thema zurückkehren.
- Mit Hilfe der Suchfunktion von **Information - Unterstützung** können Sie bestimmte Themen suchen. Weitere Informationen finden Sie in „Zugreifen auf Informationen mit ″Information - Unterstützung″“ auf Seite 326.

Anhang F. Bemerkungen

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 IBM Ansprechpartner erhältlich. Hinweise auf IBM Lizenzprogramme oder andere IBM Produkte bedeuten nicht, dass nur Programme, Produkte oder Dienstleistungen von IBM verwendet werden können. An Stelle der IBM Produkte, Programme oder Dienstleistungen können auch andere ihnen äquivalente Produkte, Programme oder Dienstleistungen verwendet werden, solange diese keine gewerblichen oder anderen Schutzrechte der IBM verletzen. Die Verantwortung für den Betrieb der Produkte, Programme oder Dienstleistungen in Verbindung mit Fremdprodukten und Fremddienstleistungen liegt beim Kunden, soweit nicht ausdrücklich solche Verbindungen erwähnt sind.

Für in diesem Handbuch beschriebene Erzeugnisse und Verfahren kann es IBM Patente oder Patentanmeldungen geben. Mit der Auslieferung dieses Handbuchs ist keine Lizenzierung dieser Patente verbunden. Lizenzanfragen sind schriftlich an IBM Europe, Director of Licensing, 92066 Paris La Defense Cedex, France, zu richten. Anfragen an obige Adresse müssen auf englisch formuliert werden.

Trotz sorgfältiger Bearbeitung können technische Ungenauigkeiten oder Druckfehler in dieser Veröffentlichung nicht ausgeschlossen werden. Die Angaben in diesem Handbuch werden in regelmäßigen Zeitabständen aktualisiert. Die Änderungen werden in Überarbeitungen bekanntgegeben. IBM kann jederzeit Verbesserungen und/oder Änderungen an den in dieser Veröffentlichung beschriebenen Produkten und/oder Programmen vornehmen.

Verweise in diesen Informationen auf Websites anderer Anbieter dienen lediglich als Benutzerinformationen und stellen keinerlei Billigung des Inhalts dieser Websites dar. Das über diese Websites verfügbare Material ist nicht Bestandteil des Materials für dieses IBM Produkt. Die Verwendung dieser Websites geschieht auf eigene Verantwortung.

Werden an IBM Informationen eingesandt, können diese beliebig verwendet werden, ohne dass eine Verpflichtung gegenüber dem Einsender entsteht.

Lizenznehmer des Programms, die Informationen zu diesem Produkt wünschen mit der Zielsetzung: (i) den Austausch von Informationen zwischen unabhängigen, erstellten Programmen und anderen Programmen (einschließlich des vorliegenden Programms) sowie (ii) die gemeinsame Nutzung der ausgetauschten Informationen zu ermöglichen, wenden sich an folgende Adresse:

IBM Canada Limited
Office of the Lab Director
1150 Eglinton Ave. East
North York, Ontario
M3C 1H7
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 Handbuch aufgeführten Lizenzprogramms sowie des zugehörigen Lizenzmaterials erfolgt im Rahmen der Allgemeinen Geschäftsbedingungen der IBM, der Internationalen Nutzungsbedingungen der IBM für Programmpakete oder einer äquivalenten Vereinbarung.

Alle in diesem Dokument enthaltenen Leistungsdaten stammen aus einer gesteuerten Umgebung. Die Ergebnisse, die in anderen Betriebsumgebungen erzielt werden, können daher erheblich von den hier erzielten Ergebnissen abweichen. Einige Daten stammen möglicherweise von Systemen, deren Entwicklung noch nicht abgeschlossen ist. Eine Garantie, dass diese Daten auch in allgemein verfügbaren Systemen erzielt werden, kann nicht gegeben werden. Darüber hinaus wurden einige Daten unter Umständen durch Extrapolation berechnet. Die tatsächlichen Ergebnisse können abweichen. Benutzer dieses Dokuments sollten die entsprechenden Daten in ihrer spezifischen Umgebung prüfen.

Informationen über Produkte anderer Hersteller als IBM wurden von den Herstellern dieser Produkte zur Verfügung gestellt, bzw. aus von ihnen veröffentlichten Ankündigungen oder anderen öffentlich zugänglichen Quellen entnommen. IBM hat diese Produkte nicht getestet und übernimmt im Hinblick auf Produkte anderer Hersteller keine Verantwortung für einwandfreie Funktion, Kompatibilität oder andere Ansprüche. Fragen zu den Leistungsmerkmalen von Produkten anderer Anbieter sind an den jeweiligen Anbieter zu richten.

Aussagen über Pläne und Absichten der IBM unterliegen Änderungen oder können zurückgenommen werden und repräsentieren nur die Ziele der IBM.

Diese Veröffentlichung enthält Beispiele für Daten und Berichte des alltäglichen Geschäftsablaufes. Sie sollen nur die Funktionen des Lizenzprogrammes illustrieren; sie können Namen von Personen, Firmen, Marken oder Produkten enthalten. Alle diese Namen sind frei erfunden, Ähnlichkeiten mit tatsächlichen Namen und Adressen sind rein zufällig.

COPYRIGHT-LIZENZ:

Diese Veröffentlichung enthält Beispielanwendungsprogramme, die in Quellensprache geschrieben sind. Sie dürfen diese Beispielprogramme kostenlos kopieren, ändern und verteilen, wenn dies zu dem Zweck geschieht, Anwendungsprogramme zu entwickeln, verwenden, vermarkten oder zu verteilen, die mit der Anwendungsprogrammierschnittstelle konform sind, für die diese Beispielprogramme geschrieben werden. Die in diesem Handbuch aufgeführten Beispiele sollen lediglich der Veranschaulichung und zu keinem anderen Zweck dienen. Diese Beispiele wurden nicht unter allen denkbaren Bedingungen getestet.

Kopien oder Teile der Beispielprogramme bzw. daraus abgeleiteter Code müssen folgenden Copyrightvermerk beinhalten:

© (Name Ihrer Firma) (Jahr). Teile des vorliegenden Codes wurden aus Beispielprogrammen der IBM Corp. abgeleitet. © Copyright IBM Corp. _Jahr/Jahre angeben_. Alle Rechte vorbehalten.

Marken

Folgende Namen sind in gewissen Ländern Marken der International Business Machines Corporation.

ACF/VTAM
AISPO
AIX
AIX/6000
AIXwindows
AnyNet
APPN
IBM System AS/400
BookManager
CICS
C Set++
C/370
DATABASE 2
DataHub
DataJoiner
DataPropagator
DataRefresher
DB2
DB2 Connect
DB2 Extender
DB2 OLAP Server
DB2 Universal Database
Distributed Relational
Database Architecture
DRDA
eNetwork
Extended Services
FFST
First Failure Support Technology
IBM
IMS
IMS/ESA
LAN DistanceMVS
MVS/ESA
MVS/XA
Net.Data
OS/2
OS/390
OS/400
PowerPC
QBIC
QMF
RACF
RS/6000
IBM System /370
SP
SQL/DS
SQL/400
System/370
IBM System /390
SystemView
VisualAge
VM/ESA
VSE/ESA
VTAM
WebExplorer
WIN-OS/2

Folgende Namen sind in gewissen Ländern Marken oder eingetragene Marken anderer Unternehmen:

Microsoft, Windows und Windows NT sind Marken oder eingetragene Marken von Microsoft Corporation.

Java und alle auf Java basierenden Marken und Logos sowie Solaris sind in gewissen Ländern Marken von Sun Microsystems, Inc.

Tivoli und NetView sind in gewissen Ländern Marken von Tivoli Systems Inc.

UNIX ist eine eingetragene Marke und wird ausschließlich von der X/Open Company Limited lizenziert.

Andere Namen von Unternehmen, Produkten oder Dienstleistungen können Marken anderer Unternehmen sein.

Index

E

F

G

H

I

J

K

L

M

N

O

P

R

S

T

U

V

W

X

Z

Kontaktaufnahme mit IBM

Bei technischen Problemen lesen Sie bitte die entsprechenden Korrekturmaßnahmen im Handbuch *Troubleshooting Guide* und führen Sie diese aus, bevor Sie sich mit der IBM Kundenunterstützung in Verbindung setzen. Mit Hilfe dieses Handbuchs können Sie Informationen sammeln, die die DB2-Kundenunterstützung zur Fehlerbehebung verwenden kann.

Wenn Sie weitere Informationen benötigen oder eines der DB2 Universal Database-Produkte bestellen möchten, setzen Sie sich mit einem IBM Ansprechpartner in einer lokalen Geschäftsstelle oder einem IBM Software-Vertriebspartner in Verbindung.

Telefonische Unterstützung erhalten Sie unter der folgenden Nummer:

- Unter 0180 3/313 233 erreichen Sie Hallo IBM, wo Sie Antworten zu allgemeinen Fragen erhalten.

Produktinformationen

Telefonische Unterstützung erhalten Sie unter den folgenden Nummern:

- Unter 0180 3/313 233 erreichen Sie Hallo IBM, wo Sie Antworten zu allgemeinen Fragen erhalten.
- Unter 0180/55 090 können Sie Handbücher telefonisch bestellen.

http://www.ibm.com/software/data/
Auf den DB2-World Wide Web-Seiten erhalten Sie aktuelle DB2-Informationen wie Neuigkeiten, Produktbeschreibungen, Schulungspläne und vieles mehr.

http://www.ibm.com/software/data/db2/library/
Mit **DB2 Product and Service Technical Library** können Sie auf häufig gestellte Fragen, Berichtigungen, Handbücher und aktuelle technische DB2-Informationen zugreifen.

Anmerkung: Diese Informationen stehen möglicherweise nur auf Englisch zur Verfügung.

http://www.elink.ibmlink.ibm.com/pbl/pbl/
Auf der Web-Site für die Bestellung internationaler Veröffentlichungen (International Publications) finden Sie Informationen zum Bestellverfahren.

http://www.ibm.com/education/certify/
Das 'Professional Certification Program' auf der IBM Web-Site stellt Zertifizierungstestinformationen für eine Reihe von IBM Produkten, u. a. auch DB2, zur Verfügung.

ftp.software.ibm.com
Melden Sie sich als *anonymous* an. Im Verzeichnis `/ps/products/db2` finden Sie Demo-Versionen, Berichtigungen, Informationen und Tools zu DB2 und vielen zugehörigen Produkten.

comp.databases.ibm-db2, bit.listserv.db2-l
Über diese Internet-Newsgroups können DB2-Benutzer Ihre Erfahrungen mit den DB2-Produkten austauschen.

Für Compuserve: GO IBMDB2
Geben Sie diesen Befehl ein, um auf IBM DB2 Family Forums zuzugreifen. Alle DB2-Produkte werden über diese Foren unterstützt.

In Anhang A des Handbuchs *IBM Software Support Handbook* finden Sie Informationen dazu, wie Sie sich mit IBM in Verbindung setzen können. Rufen Sie die folgende Web-Seite auf, um auf dieses Dokument zuzugreifen: http://www.ibm.com/support/. Wählen Sie anschließend die Verbindung zum IBM Software Support Handbook am unteren Rand der Seite aus.

Anmerkung: In einigen Ländern sollten sich die IBM Vertragshändler an die innerhalb ihrer Händlerstruktur vorgesehene Unterstützung wenden, nicht an die IBM Unterstützungsfunktion.

SC12-2879-01