IBM DB2 10.1
for Linux, UNIX and Windows

Datenbanksicherheit

SC12-4693-00

IBM DB2 10.1
for Linux, UNIX and Windows

Datenbanksicherheit

SC12-4693-00

Hinweis

Vor Verwendung dieser Informationen und des darin beschriebenen Produkts sollten die allgemeinen Informationen in Anhang B, „Bemerkungen", auf Seite 413 gelesen werden.

Diese Veröffentlichung ist eine Übersetzung des Handbuchs
IBM DB2 10.1 for Linux, UNIX, and Windows, Database Security Guide,
IBM Form SC27-3872-00,
herausgegeben von International Business Machines Corporation, USA

© Copyright International Business Machines Corporation 2012

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:
TSC Germany
Kst. 2877
Mai 2012

Inhaltsverzeichnis

Zu diesem Handbuch vii

Kapitel 1. DB2-Sicherheitsmodell 1
Authentifizierung 2
Berechtigung 3
Sicherheitsaspekte bei Installation und Verwendung des DB2-Datenbankmanagers 5
Vorausgesetzte Dateiberechtigungen für die Instanz- und Datenbankverzeichnisse 7
Details zur Authentifizierung 8
Authentifizierungsmethoden für den Server . . . 8
Authentifizierungsaspekte für ferne Clients . . . 14
Authentifizierungsaspekte bei partitionierten Datenbanken 15
Kerberos-Authentifizierung 15
Kennwortverwaltung auf Servern 22
Berechtigungen, Zugriffsrechte und Objekteigentumsrecht 23
Berechtigungen - Übersicht 30
Berechtigungen der Instanzebene 33
Datenbankberechtigungen 37
Zugriffsrechte 47
Berechtigungs-IDs in verschiedenen Kontexten . 53
Beim Erstellen einer Datenbank erteilte Standardzugriffsrechte 55
Erteilen und Entziehen von Zugriffsmöglichkeiten 56
Zugriffssteuerung für Datenbankadministratoren (DBAs) 64
Indirekte Methoden des Zugriffs auf Daten . . . 66
Datenverschlüsselung 68
Konfigurieren der SSL-Unterstützung in einer DB2-Instanz 70
IBM Database Encryption Expert für die Verschlüsselung von ruhenden Daten (Data At Rest, DAR) 100
Datenbankverschlüsselung mit AIX EFS . . . 103
Prüfen von DB2-Aktivitäten 106
Einführung in die DB2-Prüffunktion 106
Verwalten der Prüffunktion 130
Sicherheitsmodell für den Befehl 'db2cluster' . . . 134

Kapitel 2. Rollen 137
Erstellen von Rollen und Erteilen der Zugehörigkeit 138
Rollenhierarchien 140
Wirkung des Widerrufens von Zugriffsrechten für Rollen 141
Delegieren der Rollenverwaltung mit der Klausel WITH ADMIN OPTION 142
Rollen im Vergleich zu Gruppen 143
Verwenden von Rollen nach der Migration von IBM Informix Dynamic Server 145

Kapitel 3. Verwenden gesicherter Kontexte und gesicherter Verbindungen . 147
Gesicherte Kontexte und Verbindungen 149
Übernahme der Rollenzugehörigkeit über einen gesicherten Kontext 153
Regeln zum Wechseln der Benutzer-ID bei einer expliziten gesicherten Verbindung 154
Fehlerbestimmung für gesicherte Kontexte . . . 156

Kapitel 4. Zeilen- und Spaltenzugriffssteuerung (RCAC) - Übersicht 159
Zeilen- und Spaltenzugriffssteuerung (RCAC) - Regeln 160
SQL-Anweisungen zur Verwaltung von RCAC-Regeln 160
Integrierte Funktionen zum Verwalten von RCAC-Berechtigungen und -Masken 160
Szenario: ExampleHMO als Benutzer der Zeilen- und Spaltenzugriffssteuerung 161
Szenario: ExampleHMO als Benutzer der Zeilen- und Spaltenzugriffssteuerung - Sicherheitsrichtlinien 161
Szenario: ExampleHMO als Benutzer der Zeilen- und Spaltenzugriffssteuerung - Datenbankbenutzer und -rollen 162
Szenario: ExampleHMO als Benutzer der Zeilen- und Spaltenzugriffssteuerung - Datenbanktabellen 163
Szenario: ExampleHMO als Benutzer der Zeilen- und Spaltenzugriffssteuerung - Verwaltung von Sicherheitsfunktionen 166
Szenario: ExampleHMO als Benutzer der Zeilen- und Spaltenzugriffssteuerung - Zeilenberechtigungen 166
Szenario: ExampleHMO als Benutzer der Zeilen- und Spaltenzugriffssteuerung - Spaltenmasken . 167
Szenario: ExampleHMO als Benutzer der Zeilen- und Spaltenzugriffssteuerung - Einfügen von Daten 169
Szenario: ExampleHMO als Benutzer der Zeilen- und Spaltenzugriffssteuerung - Datenaktualisierungen 169
Szenario: ExampleHMO als Benutzer der Zeilen- und Spaltenzugriffssteuerung - Datenabfragen . 170
Szenario: ExampleHMO als Benutzer der Zeilen- und Spaltenzugriffssteuerung - Ansichtserstellung 172
Szenario: ExampleHMO als Benutzer der Zeilen- und Spaltenzugriffssteuerung - Sichere Funktionen 173
Szenario: ExampleHMO als Benutzer der Zeilen- und Spaltenzugriffssteuerung - Sichere Trigger . 175
Szenario: ExampleHMO als Benutzer der Zeilen- und Spaltenzugriffssteuerung - Entziehen von Berechtigungen 176

Szenario: ExampleBANK als Benutzer der Zeilen- und Spaltenzugriffssteuerung 176
Szenario: ExampleBANK als Benutzer der Zeilen- und Spaltenzugriffssteuerung - Sicherheitsrichtlinien 176
Szenario: ExampleBANK als Benutzer der Zeilen- und Spaltenzugriffssteuerung - Datenbankbenutzer und -rollen 177
Szenario: ExampleBANK als Benutzer der Zeilen- und Spaltenzugriffssteuerung - Datenbanktabellen 178
Szenario: ExampleBANK als Benutzer der Zeilen- und Spaltenzugriffssteuerung - Zeilenberechtigungen 179
Szenario: ExampleBANK als Benutzer der Zeilen- und Spaltenzugriffssteuerung - Spaltenmasken 180
Szenario: ExampleBANK als Benutzer der Zeilen- und Spaltenzugriffssteuerung - Datenabfragen 181

Kapitel 5. Kennsatzbasierte Zugriffssteuerung (LBAC) 183
LBAC-Sicherheitsrichtlinien 185
Komponenten von LBAC-Sicherheitskennsätzen - Übersicht 186
Typ der LBAC-Sicherheitskennsatzkomponente: SET 188
Typ der LBAC-Sicherheitskennsatzkomponente: ARRAY 188
Typ der LBAC-Sicherheitskennsatzkomponente: TREE 189
LBAC-Sicherheitskennsätze 192
Format für Werte von Sicherheitskennsätzen . . . 194
Vergleichen von LBAC-Sicherheitskennsätzen. . . 195
LBAC-Regelsätze - Übersicht 196
LBAC-Regelsatz: DB2LBACRULES 196
Freistellungen von LBAC-Regeln 201
Integrierte Funktionen zum Verwalten von LBAC-Sicherheitskennsätzen 202
Schützen von Daten mit LBAC 204
Lesen von LBAC-geschützten Daten 205
Einfügen von LBAC-geschützten Daten. 208
Aktualisieren von LBAC-geschützten Daten . . . 211
Löschen oder Entfernen von LBAC-geschützten Daten 216
Entfernen des LBAC-Schutzes von Daten 219

Kapitel 6. Verwenden des Systemkatalogs für Sicherheitsinformationen . . 221
Abrufen von Berechtigungsnamen mit erteilten Zugriffsrechten 222
Abrufen aller Namen mit der Berechtigung DBADM 223
Abrufen der Namen mit Zugriffsberechtigung für eine Tabelle 223
Abrufen aller Benutzern erteilter Zugriffsrechte . . 224
Schützen der Systemkatalogsicht 224

Kapitel 7. Firewallunterstützung . . . 229
Screening-Router-Firewalls 229
Proxy-Firewalls der Anwendungsebene. 229
Circuit-Level-Firewalls 230
SMLI-Firewalls (Stateful Multi-Layer Inspection) 230

Kapitel 8. Sicherheits-Plug-ins 231
Speicherpositionen für Sicherheits-Plug-in-Bibliotheken. 236
Namenskonventionen für Sicherheits-Plug-ins . . 237
Unterstützung zweiteiliger Benutzer-IDs in Sicherheits-Plug-ins 238
Versionssteuerung für Sicherheits-Plug-in-APIs . . 240
Hinweise zu 32- und 64-Bit-Sicherheits-Plug-ins 240
Fehlerbestimmung für Sicherheits-Plug-ins . . . 241
Aktivieren von Plug-ins 242
Implementieren eines Plug-ins zum Abrufen von Gruppen 242
Implementieren eines Benutzer-ID/Kennwort-Plug-ins 243
Implementieren eines GSS-API-Plug-ins . . . 244
Implementieren eines Kerberos-Plug-ins . . . 246
Unterstützung für LDAP-basierte Authentifizierung und Gruppensuchfunktion 247
Konfigurieren von transparentem LDAP für Authentifizierung und Gruppensuche (AIX) . . . 249
Konfigurieren von transparentem LDAP für Authentifizierung und Gruppensuche (Linux) . . 252
Konfigurieren von transparentem LDAP für Authentifizierung und Gruppensuche (HP-UX) . . 254
Konfigurieren von transparentem LDAP für Authentifizierung und Gruppensuche (Solaris) . . 256
Konfigurieren der LDAP-Plug-in-Module . . . 257
Aktivieren der LDAP-Plug-in-Module 260
Verbindungsaufbau mit einer LDAP-Benutzer-ID 261
Hinweise zur Gruppensuchfunktion. 262
Fehlerbehebung beim Authentifizieren von LDAP-Benutzern oder beim Abrufen von Gruppen. 263
Schreiben von Sicherheits-Plug-ins 263
Laden von Sicherheits-Plug-ins in DB2 263
Einschränkungen für die Entwicklung von Sicherheits-Plug-in-Bibliotheken 265
Einschränkungen für Sicherheits-Plug-ins . . . 267
Rückkehrcodes für Sicherheits-Plug-ins. . . . 269
Fehlernachrichtenbehandlung für Sicherheits-Plug-ins 272
Aufrufreihenfolgen für die APIs der Sicherheits-Plug-ins 273

Kapitel 9. Sicherheits-Plug-in-APIs 277
APIs für Plug-ins zum Abrufen von Gruppen . . 278
db2secDoesGroupExist (API) - Vorhandensein einer Gruppe überprüfen 280
db2secFreeErrormsg (API) - Speicher für Fehlernachricht freigeben 280
db2secFreeGroupListMemory (API) - Speicher für Gruppenliste freigeben 281
db2secGetGroupsForUser (API) - Liste von Gruppen für Benutzer abrufen 281

db2secGroupPluginInit (API) - Gruppen-Plug-in initialisieren 285
db2secPluginTerm - Gruppen-Plug-in-Ressourcen bereinigen 286
APIs für Plug-ins zur Benutzer-ID/Kennwort-Authentifizierung 286
db2secClientAuthPluginInit (API) - Plug-in zur Clientauthentifizierung initialisieren 292
db2secClientAuthPluginTerm (API) - Ressourcen für Plug-in zur Clientauthentifizierung bereinigen 294
db2secDoesAuthIDExist - Vorhandensein der Berechtigungs-ID überprüfen 294
db2secFreeInitInfo (API) - Von 'db2secGenerateInitialCred' genutzte Ressourcen bereinigen 295
db2secFreeToken (API) - Vom Token belegten Speicher freigeben 295
db2secGenerateInitialCred (API) - Anfangsberechtigungsnachweise generieren 296
db2secGetAuthIDs (API) - Berechtigungs-IDs abrufen 298
db2secGetDefaultLoginContext (API) - Standardanmeldekontext abrufen 300
db2secProcessServerPrincipalName (API) - Vom Server zurückgegebenen Service-Principal-Namen verarbeiten 301
db2secRemapUserid (API) - Zuordnung von Benutzer-ID und Kennwort ändern 302
db2secServerAuthPluginInit - Plug-in zur Serverauthentifizierung initialisieren 304
db2secServerAuthPluginTerm (API) - Ressourcen für Plug-in zur Serverauthentifizierung bereinigen 307
db2secValidatePassword (API) - Kennwort prüfen 307
Erforderliche APIs und Definitionen für Plug-ins zur GSS-API-Authentifizierung 310
Einschränkungen für Plug-ins zur GSS-API-Authentifizierung 311

Kapitel 10. Exitbibliotheken für Kommunikationspuffer 313

Exitbibliothek für Kommunikationspuffer - Implementierung 314
Exitbibliothek für Kommunikationspuffer - Position. 314
Exitbibliothek für Kommunikationspuffer - Namenskonventionen und Berechtigungen . . . 314
Aktivieren von Exitbibliotheken für Kommunikationspuffer außerhalb von DB2 pureScale-Umgebungen. 315
Aktivieren von Exitbibliotheken für Kommunikationspuffer in DB2 pureScale-Umgebungen. . 316
Exitbibliothek für Kommunikationspuffer - Fehlerbestimmung 317
Exitbibliothek für Kommunikationspuffer - Entwicklung 317
Vorgehensweise beim Laden einer Exitbibliothek für Kommunikationspuffer 318
APIs der Exitbibliothek für Kommunikationspuffer 318
Exitbibliothek für Kommunikationspuffer - Funktionsstruktur 327
Exitbibliothek für Kommunikationspuffer - Informationsstruktur 328
Pufferstruktur der Exitbibliothek für Kommunikationspuffer 329
Exitbibliothek für Kommunikationspuffer - Verbindungssteuerung 330
Exitbibliothek für Kommunikationspuffer - API-Versionen. 330
Exitbibliothek für Kommunikationspuffer - Fehlerbehandlung und Rückkehrcodes 330
Exitbibliothek für Kommunikationspuffer - Einschränkungen bei der Entwicklung 331
Exitbibliothek für Kommunikationspuffer - API-Aufrufreihenfolgen 333

Kapitel 11. Aufbau der Datensätze der Prüffunktion 339

Prüfsatzobjekttypen 339
Prüfsatzaufbau für AUDIT-Ereignisse 341
Prüfsatzaufbau für CHECKING-Ereignisse . . . 344
Gründe für die Zugriffsgewährung bei CHECKING-Ereignissen 347
Typen von Zugriffsversuchen bei CHECKING-Ereignissen 349
Prüfsatzaufbau für OBJMAINT-Ereignisse 353
Prüfsatzaufbau für SECMAINT-Ereignisse 356
SECMAINT-Zugriffsrechte bzw. -Berechtigungen 360
Prüfsatzaufbau für SYSADMIN-Ereignisse. . . . 364
Prüfsatzaufbau für VALIDATE-Ereignisse 366
Prüfsatzaufbau für CONTEXT-Ereignisse 368
Prüfsatzaufbau für EXECUTE-Ereignisse 370
Prüfereignisse 376

Kapitel 12. Arbeiten mit der Betriebssystemsicherheit 383

DB2- und Windows-Sicherheit. 383
Authentifizierungsszenarios 384
Unterstützung globaler Gruppen (Windows) . . 385
Benutzerauthentifizierung und Gruppeninformationen mit DB2 unter Windows 386
Definieren von Benutzern mit der Berechtigung SYSADM (Windows) 392
Unterstützung für das lokale Systemkonto (LocalSystem) unter Windows 393
Erweiterte Windows-Sicherheit mithilfe der Gruppen DB2ADMNS und DB2USERS 394
Aspekte zu Windows 2008 und Windows Vista oder höher: Benutzerzugriffssteuerungsfunktion (User Access Control, UAC) 398
DB2- und UNIX-Sicherheit 399
DB2- und Linux-Sicherheit 399
Unterstützung für Kennwortänderung (Linux) 399
Implementieren eines Plug-ins zur Kennwortänderung (Linux) 400

Anhang A. Übersicht über technische Informationen zu DB2 403
Bibliothek mit technischen Informationen zu DB2 im Hardcopy- oder PDF-Format 404
Aufrufen der Hilfe für den SQL-Status über den Befehlszeilenprozessor 406
Zugriff auf verschiedene Versionen des DB2 Information Center 406
Aktualisieren des auf Ihrem Computer oder Intranet-Server installierten DB2 Information Center . . 407
Manuelles Aktualisieren des auf Ihrem Computer oder Intranet-Server installierten DB2 Information Center 408
DB2-Lernprogramme 411
Informationen zur Fehlerbehebung in DB2. 411
Bedingungen 412

Anhang B. Bemerkungen 413

Index 417

Zu diesem Handbuch

Das Handbuch *Datenbanksicherheit* beschreibt, wie DB2-Sicherheitsfunktionen zur Implementierung und Verwaltung der für Ihre Datenbankinstallation erforderlichen Sicherheitsstufe eingesetzt werden.

Das Handbuch *Datenbanksicherheit* stellt ausführliche Informationen zu folgenden Themen bereit:

- Verwalten der Authentifizierung von Benutzern, die auf DB2-Datenbank zugreifen können
- Einrichten der Berechtigungsfunktion zur Steuerung des Benutzerzugriffs auf Objekte und Daten in Datenbanken

Kapitel 1. DB2-Sicherheitsmodell

Zwei Sicherheitsmodi steuern den Zugriff auf die Daten und Funktionen des DB2-Datenbanksystems. Der Zugriff auf das DB2-Datenbanksystem wird durch Einrichtungen verwaltet, die sich außerhalb des DB2-Datenbanksystems (Authentifizierung) befinden, während der Zugriff innerhalb des DB2-Datenbanksystems durch den Datenbankmanager (Berechtigungen) verwaltet wird.

Authentifizierung

Die Authentifizierung ist ein Prozess, bei dem ein System die Identität eines Benutzers prüft. Die Benutzerauthentifizierung wird durch eine Sicherheitseinrichtung außerhalb des DB2-Datenbanksystems mithilfe eines Sicherheits-Plug-in-Moduls zur Authentifizierung ausgeführt. Ein Standard-Sicherheits-Plug-in-Modul zur Authentifizierung, das auf die betriebssystembasierte Authentifizierung zurückgreift, wird bereitgestellt, wenn Sie das DB2-Datenbanksystem installieren. Um Ihnen die Arbeit zu erleichtern, wird der DB2-Datenbankmanager auch mit den Authentifizierungs-Plug-in-Modulen für Kerberos und LDAP (Lightweight Directory Access Protocol) geliefert. Zur zusätzlichen Erhöhung der Flexibilität bei der Umsetzung Ihrer speziellen Authentifizierungsanforderungen können Sie ein eigenes Sicherheits-Plug-in-Modul zur Authentifizierung erstellen.

Der Authentifizierungsprozess generiert eine DB2-Berechtigungs-ID. Informationen zur Gruppenzugehörigkeit des Benutzers werden ebenfalls bei der Authentifizierung abgerufen. Das Standardverfahren zum Abrufen von Gruppeninformationen greift auf ein Plug-in-Modul zum Abrufen von Gruppenzugehörigkeiten auf Betriebssystemebene zurück, das zur Verfügung gestellt wird, wenn Sie das DB2-Datenbanksystem installieren. Wenn Sie dies bevorzugen, können Sie Informationen zur Gruppenzugehörigkeit auch abrufen, indem Sie ein spezielles Plug-in-Modul für Gruppenzugehörigkeiten verwenden, wie zum Beispiel ein LDAP-Modul.

Berechtigung

Wenn ein Benutzer authentifiziert wurde, stellt der Datenbankmanager fest, ob diesem Benutzer Zugriff auf Daten oder Ressourcen von DB2 gewährt wird. Die Berechtigung ist ein Prozess, bei dem der DB2-Datenbankmanager Informationen zum authentifizierten Benutzer abruft, die angeben, welche Datenbankoperationen der Benutzer ausführen und auf welche Datenobjekte er zugreifen darf.

Die folgenden unterschiedlichen Quellen von Berechtigungen sind für eine Berechtigungs-ID verfügbar:

1. Primäre Berechtigungen: die Berechtigungen, die der Berechtigungs-ID direkt erteilt werden.
2. Sekundäre Berechtigungen: die Berechtigungen, die den Gruppen und Rollen erteilt sind, zu denen die Berechtigungs-ID gehört.
3. Allgemein zugängliche Berechtigungen: die Berechtigungen, die der speziellen Gruppe PUBLIC erteilt werden.
4. Kontextabhängige Berechtigungen: die Berechtigungen, die einer Rolle in einem gesicherten Kontext erteilt werden.

Berechtigungen können Benutzern in den folgenden Kategorien erteilt werden:

- Berechtigungen der Systemebene

Die Berechtigungen Systemadministrator (SYSADM), Systemsteuerung (SYSCTRL), Systempflege (SYSMAINT) und Systemmonitor (SYSMON) stellen verschiedene Grade an Steuerungsmöglichkeiten für Funktionen auf Instanzebene zur Verfügung. Berechtigungen bieten die Möglichkeit, Zugriffsrechte zu gruppieren und Verwaltungs- und Dienstprogrammoperationen für Instanzen, Datenbanken und Datenbankobjekte zu steuern.

- Berechtigungen der Datenbankebene

  Die Berechtigungen SECADM (Sicherheitsadministrator), DBADM (Datenbankadministrator), ACCESSCTRL (Zugriffssteuerung), DATAACCESS (Datenzugriff), SQLADM (SQL-Administrator), WLMADM (Administrator für Workload-Management) und EXPLAIN (Erläutern) ermöglichen eine Steuerung in der Datenbank. Weitere Datenbankberechtigungen sind LOAD (die Möglichkeit zum Laden von Daten in eine Tabelle) und CONNECT (die Möglichkeit, eine Verbindung zu einer Datenbank herzustellen).
- Berechtigungen der Objektebene

  Zu den Berechtigungen der Objektebene gehört das Überprüfen von Zugriffsrechten, wenn eine Operation an einem Objekt ausgeführt wird. Zum Beispiel muss ein Benutzer (mindestens) das Zugriffsrecht SELECT für eine Tabelle besitzen, um Daten aus dieser Tabelle auswählen zu können.
- Kontextabhängige Berechtigungen

  Mithilfe von Sichten lässt sich steuern, welche Spalten oder Zeilen einer Tabelle bestimmte Benutzer lesen können. Die kennsatzbasierte Zugriffssteuerung (LBAC, Label-Based Access Control) bestimmt, welche Benutzer Lese- und Schreibzugriff auf einzelne Zeilen und einzelne Spalten haben.

Sie können diese Funktionen in Verbindung mit der DB2-Prüffunktion zur Zugriffsüberwachung verwenden, um die Sicherheitsstufe zu definieren und zu verwalten, die für Ihre Datenbankinstallation erforderlich ist.

Authentifizierung

Die Authentifizierung eines Benutzers wird mithilfe einer Sicherheitseinrichtung außerhalb des DB2-Datenbanksystems ausgeführt. Die Sicherheitseinrichtung kann Teil des Betriebssystems oder ein separates Produkt sein.

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.

Anmerkung: In Nichtrootinstallationen muss die betriebssystembasierte Authentifizierung durch Ausführen des Befehls **db2rfe** aktiviert werden.

Nach erfolgreicher Authentifizierung:

- Der Benutzer muss für DB2 mithilfe 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 unter UNIX-Betriebssystemen bei Verwendung des Standardberechtigungs-Plug-in-Moduls eine DB2-Berechtigungs-ID (*authid*) abgeleitet, indem eine UNIX-Benutzer-ID, die den DB2-Namenskonventionen entspricht, in Großbuchstaben umgesetzt wird.

- Eine Liste von Gruppen, zu denen der Benutzer gehört, wird abgerufen. Die Gruppenzugehörigkeit kann bei der Berechtigung des Benutzers verwendet werden. Gruppen sind Entitäten von Sicherheitseinrichtungen, die auch DB2-Berechtigungsnamen zugeordnet sein müssen. Diese Zuordnung erfolgt mit einer ähnlichen Methode wie für Benutzer-IDs.

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

- Eine erfolgreiche Anmeldung am Sicherheitssystem dient als Nachweis der Identität und lässt Folgendes zu:
  - Verwendung lokaler Befehle zum Zugriff auf lokale Daten
  - Verwendung von Remoteverbindungen, wenn der Server die Clientauthentifizierung akzeptiert
- Eine erfolgreiche Prüfung einer Benutzer-ID mit Kennwort durch die Sicherheitseinrichtung dient als Nachweis der Identität und lässt Folgendes zu:
  - Verwendung von Remoteverbindungen, bei denen der Server einen Nachweis der Authentifizierung verlangt
  - Verwendung von Operationen, bei denen der Benutzer einen Befehl unter einer anderen als der zur Anmeldung verwendeten Identität ausführen will

Anmerkung: Auf einigen UNIX-Systemen kann der DB2-Datenbankmanager fehlgeschlagene Versuche der Kennworteingabe über das Betriebssystem protokollieren und erkennen, wann ein Client die Anzahl zulässiger Anmeldeversuche überschritten hat, die mit dem Parameter LOGINRETRIES festgelegt wird.

Berechtigung

Die Berechtigung wird mithilfe von DB2-Funktionen durchgeführt. DB2-Tabellen und -Konfigurationsdateien werden zum Aufzeichnen der Berechtigungen verwendet, die jedem Berechtigungsnamen zugeordnet sind.

Versucht ein authentifizierter Benutzer auf Daten zuzugreifen, werden die aufgezeichneten Berechtigungen mit den folgenden Berechtigungen verglichen:

- Berechtigungen des Berechtigungsnamens des Benutzers
- Berechtigungen der Gruppen, denen der Benutzer angehört
- Berechtigungen der Rollen, die dem Benutzer direkt oder indirekt über eine Gruppe oder eine Rolle erteilt wurden
- Berechtigungen, die über einen gesicherten Kontext erworben wurden

Anhand dieses Vergleichs bestimmt der DB2-Server, ob der angeforderte Zugriff gewährt wird.

Die Typen von Berechtigungen, die aufgezeichnet werden, sind Zugriffsrechte, Berechtigungsstufen und LBAC-Berechtigungsnachweise.

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 zusammenzufassen und Operationen des Datenbankmanagers zu steuern. Datenbankspezifische Berechtigungen werden in den Datenbankkatalogen gespeichert, während Systemberechtigungen durch Gruppenzugehörigkeit zugeordnet und die den Berechtigungsstufen

zugeordneten Gruppennamen in der Konfigurationsdatei des Datenbankmanagers für die jeweilige Instanz gespeichert werden.

LBAC-Berechtigungsnachweise sind LBAC-Sicherheitskennsätze und Freistellungen von LBAC-Regeln, die ein Zugreifen auf Daten zulassen, die durch LBAC (Label-Based Access Control) geschützt sind. LBAC-Berechtigungsnachweise werden in den Datenbankkatalogen gespeichert.

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 Instanzebene und Dienstprogramme), jedoch nicht für statisches SQL berücksichtigt. Eine Ausnahme von diesem allgemeinen Fall ist das Erteilen von Zugriffsrechten für PUBLIC: Diese werden bei der Verarbeitung von statischem SQL berücksichtigt. Bestimmte Fälle, in denen die Gruppenzugehörigkeit nicht gültig ist, werden in der DB2-Dokumentation entsprechend vermerkt.

Eine Rolle ist ein Datenbankobjekt, das ein oder mehrere Zugriffsrechte zusammenfasst und Benutzern, Gruppen, der Gruppe PUBLIC oder anderen Rollen durch eine Anweisung GRANT oder einem gesicherten Kontext durch eine Anweisung CREATE TRUSTED CONTEXT oder ALTER TRUSTED CONTEXT erteilt werden kann. Eine Rolle kann für das Verbindungsattribut SESSION_USER ROLE in einer Auslastungsdefinition (Workloaddefinition) angegeben werden. Bei der Verwendung von Rollen ordnen Sie Zugriffsberechtigungen für Datenbankobjekte über die Rollen zu. Benutzer, die diese Rollen haben, verfügen über die Zugriffsrechte, die für die Rolle definiert sind und mit denen sie auf Datenbankobjekte zugreifen können.

Rollen stellen eine ähnliche Funktionalität wie Gruppen zur Verfügung. Sie führen die Berechtigung für einen Benutzerverbund aus, ohne dass Zugriffsrechte jedem Benutzer einzeln erteilt oder entzogen werden müssen. Ein Vorteil von Rollen besteht darin, dass sie vom DB2-Datenbanksystem verwaltet werden. Die den Rollen erteilten Berechtigungen werden während des Berechtigungsprozesses für Ansichten, Trigger, MQTs (Materialized Query Tables), Pakete und SQL-Routinen geprüft, während die Berechtigungen, die Gruppen erteilt wurden, nicht geprüft werden. Die Berechtigungen, die Gruppen erteilt sind, werden während des Berechtigungsprozesses für Ansichten, Trigger, MQTs, Pakete und SQL-Routinen deshalb nicht geprüft, weil das DB2-Datenbanksystem nicht erkennen kann, wenn sich die Zugehörigkeit zu einer Gruppe ändert. Daher kann es die genannten Objekte bei Bedarf nicht ungültig machen.

Anmerkung: Berechtigungen, die Rollen erteilt sind, die wiederum Gruppen erteilt sind, werden während des Berechtigungsprozesses für Ansichten, Trigger, MQTs, Pakete und SQL-Routinen nicht geprüft.

Während der Verarbeitung von SQL-Anweisungen werden vom DB2-Berechtigungsmodell alle folgenden Berechtigungen geprüft:

1. Die Berechtigungen, die der primären Berechtigungs-ID der SQL-Anweisung erteilt sind
2. Die Berechtigungen, die den sekundären Berechtigungs-IDs (Gruppen oder Rollen) der SQL-Anweisung erteilt sind

3. Die Berechtigungen, die der Gruppe PUBLIC erteilt sind, einschließlich der Rollen, die der Gruppe PUBLIC direkt oder indirekt durch andere Rollen erteilt sind
4. Die Berechtigungen, die der Rolle des gesicherten Kontexts erteilt sind, falls zutreffend.

Sicherheitsaspekte bei Installation und Verwendung des DB2-Datenbankmanagers

Sicherheitsaspekte spielen für den DB2-Administrator von dem Moment der Installation des Produkts an eine wichtige Rolle.

Zur Ausführung der Installation des DB2-Datenbankmanagers sind eine Benutzer-ID, ein Gruppenname und ein Kennwort erforderlich. Das grafisch orientierte Installationsprogramm des DB2-Datenbankmanagers erstellt Standardwerte für verschiedene Benutzer-IDs und die Gruppe. Abhängig davon, ob Sie unter Linux und UNIX oder Windows installieren, werden unterschiedliche Standardwerte erstellt:

- Unter UNIX- und Linux-Betriebssystemen erstellt das DB2-Datenbankinstallationsprogramm, falls Sie eine DB2-Instanz im Fenster für die Instanzinstallation erstellen möchten, standardmäßig verschiedene Benutzer für den DB2-Verwaltungsserver (`dasusr`), den Instanzeigner (`db2inst`) und den abgeschirmten Benutzer (`db2fenc`). Sie können optional verschiedene Benutzernamen angeben.

  Das DB2-Datenbankinstallationsprogramm hängt eine Nummer von 1 bis 99 an den Standardbenutzernamen an, bis eine Benutzer-ID erstellt wird, die nicht bereits vorhanden ist. Wenn zum Beispiel die Benutzer `db2inst1` und `db2inst2` bereits vorhanden sind, erstellt das DB2-Datenbankinstallationsprogramm den Benutzer `db2inst3`. Wenn eine Nummer über 10 verwendet wird, wird der Buchstabenteil des Namens in der Standardbenutzer-ID abgeschnitten. Wenn zum Beispiel die Benutzer-ID `db2fenc9` bereits vorhanden ist, schneidet das DB2-Datenbankinstallationsprogramm das `c` der Benutzer-ID ab und hängt dann den Wert `10` an (also `db2fen10`). Dieses Abschneiden findet nicht statt, wenn der numerische Wert an die Standard-DAS-Benutzer-ID angehängt wird (z. B. `dasusr24`).
- Unter Windows-Betriebssystemen erstellt das DB2-Datenbankinstallationsprogramm standardmäßig die Benutzer-ID `db2admin` für den DAS-Benutzer, den Instanzeigner und abgeschirmte Benutzer. (Falls erwünscht, können Sie bei der Konfiguration einen anderen Benutzernamen angeben.) Im Unterschied zu Linux- und UNIX-Betriebssystemen wird kein numerischer Wert an die Benutzer-ID angehängt.

Zur Minimierung des Risikos, dass ein anderer Benutzer als der Administrator von den Standardwerten erfährt und sie in unlauterer Weise in Datenbanken und Instanzen verwendet, sollten Sie die Standardwerte bei der Installation in eine neue oder vorhandene Benutzer-ID Ihrer Wahl ändern.

Anmerkung: Installationen mit Antwortdateien arbeiten nicht mit Standardwerten für Benutzer-IDs oder Gruppennamen. Diese Werte müssen in der Antwortdatei angegeben werden.

Kennwörter spielen bei der Authentifizierung von Benutzern eine wichtige Rolle. Wenn auf der Ebene des Betriebssystems keine Anforderungen an die Authentifizierung festgelegt wurden und die Datenbank das Betriebssystem zur Authentifizierung von Benutzern verwendet, wird den Benutzern die Berechtigung zur Verbindung erteilt. Unter Linux- und UNIX-Betriebssystemen werden nicht definierte

Kennwörter beispielsweise wie NULL behandelt. In diesem Fall wird jeder Benutzer ohne definiertes Kennwort wie ein Benutzer mit dem Kennwort NULL behandelt. Aus der Perspektive des Betriebssystems ist dies eine Übereinstimmung, und der Benutzer erhält eine gültige Berechtigung und kann eine Verbindung zur Datenbank herstellen. Verwenden Sie Kennwörter auf Betriebssystemebene, wenn das Betriebssystem für die Authentifizierung von Benutzern Ihrer Datenbank zuständig sein soll.

Bei der Verwendung von Umgebungen mit partitionierten Datenbanken unter Linux- und UNIX-Betriebssystemen nutzt der DB2-Datenbankmanager standardmäßig das Dienstprogramm 'rsh' ('remsh' unter HP-UX) für die Ausführung bestimmter Befehle auf fernen Membern. Das Dienstprogramm 'rsh' überträgt Kennwörter unverschlüsselt über das Netz, was ein Sicherheitsrisiko darstellen kann, wenn sich der DB2-Server nicht in einem gesicherten Netz befindet. Sie können mithilfe der Registrierdatenbankvariablen **DB2RSHCMD** eine sicherere Alternative für das Programm für die ferne Shell definieren, durch die dieses Sicherheitsrisiko vermieden wird. Ein Beispiel für eine sicherere Alternative ist 'ssh'. Informationen zu den Einschränkungen bei der Konfiguration ferner Shells finden Sie in der Dokumentation zur Registrierdatenbankvariablen **DB2RSHCMD**.

Nach der Installation des DB2-Datenbankmanagers, sollten Sie darüber hinaus die Standardzugriffsrechte, die Benutzern erteilt wurden, prüfen und (falls erforderlich) ändern. Standardmäßig erteilt der Installationsprozess den folgenden Benutzern unter den einzelnen Betriebssystemen die Zugriffsrechte eines Systemadministrators (SYSADM):

Linux - und UNIX -Betriebssysteme
Einem gültigen DB2-Datenbankbenutzernamen, der zur Primärgruppe des Instanzeigners gehört

Windows-Umgebungen
- Mitgliedern der lokalen Administratorgruppe
- Mitgliedern der Gruppe 'Administratoren' auf dem Domänencontroller, wenn der DB2-Datenbankmanager so konfiguriert ist, dass er Gruppen für Benutzer an der Position aufzählt, an der die Benutzer definiert sind. Zum Konfigurieren der Gruppenaufzählung unter Windows-Betriebssystemen wird die Umgebungsvariable **DB2_GRP_LOOKUP** verwendet.
- Mitgliedern der Gruppe DB2ADMNS, wenn die erweiterte Sicherheit unter Windows aktiviert ist. Die Position der Gruppe DB2ADMNS wird bei der Installation festgelegt.
- Dem lokalen Systemkonto (LocalSystem)

Durch Aktualisieren des Konfigurationsparameters **sysadm_group** des Datenbankmanagers kann der Administrator steuern, welche Gruppe von Benutzern SYSADM-Zugriffsrechte besitzt. Sie müssen die folgenden Richtlinien befolgen, um die Sicherheitsanforderungen für die DB2-Datenbankinstallation sowie die nachfolgende Erstellung von Instanzen und Datenbanken zu erfüllen.

Jede Gruppe, die (durch Aktualisieren des Parameters **sysadm_group**) als Systemadministratorgruppe definiert wird, muss vorhanden sein. Der Name sollte diese Gruppe leicht als die Gruppe zu erkennen geben, die für Instanzeigner erstellt wurde. Die Benutzer-IDs und Gruppen, die dieser Gruppe angehören, verfügen über die Systemadministratorberechtigung für die entsprechenden Instanzen.

Der Administrator sollte in Betracht ziehen, eine Benutzer-ID für den Instanzeigner zu erstellen, die sich leicht als zu einer bestimmten Instanz gehörig erkennen lässt.

Diese Benutzer-ID sollte als eine ihrer Gruppen den Namen der zuvor erstellten SYSADM-Gruppe enthalten. Eine weitere Empfehlung ist, diese Benutzer-ID des Instanzeigners nur als Mitglied der Instanzeignergruppe und in keiner anderen Gruppe zu verwenden. Dadurch wird die Verbreitung von Benutzer-IDs und Gruppen begrenzt, die die Instanz modifizieren können.

Der erstellten Benutzer-ID muss ein Kennwort zugewiesen werden, um eine Authentifizierung vor dem Zugriff auf die Daten und Datenbanken der Instanz zu ermöglichen. Es wird empfohlen, bei der Erstellung des Kennworts die Kennwortrichtlinien des jeweiligen Unternehmens einzuhalten.

Anmerkung: Zur Vermeidung eines unbeabsichtigten Löschens oder Überschreibens von Instanzkonfigurationsdateien und anderen Dateien, sollten Administratoren die Verwendung eines anderen Benutzerkontos, das nicht zur Primärgruppe als Instanzeigner gehört, für die täglichen, direkt auf dem Server auszuführenden Verwaltungsaufgaben in Betracht ziehen.

Vorausgesetzte Dateiberechtigungen für die Instanz- und Datenbankverzeichnisse

Das DB2-Datenbanksystem setzt voraus, dass Ihr Instanzverzeichnis und Ihr Datenbankverzeichnis über ein Mindestmaß an Berechtigungen verfügen.

Anmerkung: Wenn das Instanzverzeichnis und das Datenbankverzeichnis durch den DB2-Datenbankmanager erstellt werden, sind die Berechtigungen korrekt und sollten nicht geändert werden.

Auf UNIX- und Linux-Systemen müssen die folgenden Mindestberechtigungen für das Instanzverzeichnis und das Verzeichnis `NODE000x/sqldbdir` festgelegt werden: `u=rwx` und `go=rx`. Die Bedeutung der Buchstaben wird in folgender Tabelle erläutert:

Zeichen	Stellt dar:
u	Benutzer (Eigner)
g	Gruppe
o	Andere Benutzer
r	Lesen
w	Schreiben
x	Ausführen

Zum Beispiel gelten für die Instanz `db2inst1` in `/home` folgende Berechtigungen:

```
drwxr-xr-x  36 db2inst1  db2grp1         4096 Jun 15 11:13 db2inst1
```

Verzeichnisse, die die Datenbanken enthalten, benötigen auf allen Verzeichnisebenen bis einschließlich `NODE000x` die folgenden Berechtigungen:

```
drwxrwxr-x  11 db2inst1  db2grp1         4096 Jun 14 15:53 NODE0000/
```

Wenn sich eine Datenbank in `/db2/data/db2inst1/db2inst1/NODE0000` befindet, müssen die Verzeichnisse `/db2`, `/db2/data`, `/db2/data/db2inst1`, `/db2/data/db2inst1/db2inst1` und `/db2/data/db2inst1/db2inst1/NODE0000` die Berechtigungen `drwxrwxr-x` haben.

Innerhalb des Verzeichnisses NODE000x erfordert das Verzeichnis sqldbdir die Berechtigungen drwxrwxr-x. Beispiel:

```
drwx------   5 db2inst1  db2grp1         256 Jun 14 14:17 SAMPLE/
drwxr-x---   7 db2inst1  db2grp1        4096 Jun 14 13:26 SQL00001/
drwxrwxr-x   2 db2inst1  db2grp1         256 Jun 14 13:02 sqldbdir/
```

Vorsicht:
Zur Aufrechterhaltung der Sicherheit Ihrer Dateien sollten Sie die Berechtigungen für die Verzeichnisse in *DBNAME* (z. B. SAMPLE) und die Verzeichnisse SQL*xxxx*, die vom DB2-Datenbankmanager bei der Erstellung der Verzeichnisse zugeordnet werden, nicht ändern.

Details zur Authentifizierung

Authentifizierungsmethoden für den Server

Der Zugriff auf eine Instanz oder eine Datenbank erfordert zunächst, dass der Benutzer *authentifiziert* wird. Der *Authentifizierungstyp* für die jeweilige Instanz bestimmt, wie und wo der Benutzer überprüft wird.

Der Authentifizierungstyp wird in der Konfigurationsdatei auf dem Server gespeichert. Sie wird erstmalig bei der Erstellung der Instanz definiert. Es gibt einen Authentifizierungstyp pro Instanz, der für den Zugriff auf den zugehörigen Datenbankserver und alle Datenbanken unter seiner Steuerung gilt.

Wenn Sie von einer föderierten Datenbank aus auf Datenquellen zugreifen wollen, müssen Sie die Authentifizierungsverarbeitung der Datenquellen und die Definitionen für die Authentifizierungstypen föderierter Datenbanken beachten.

Anmerkung: Auf der folgenden Website können Sie die Zertifizierungsinformationen zu den kryptografischen Routinen nachlesen, die vom DB2-Datenbankverwaltungssystem verwendet werden, um beim Authentifizierungstyp SERVER_ENCRYPT Benutzer-IDs und Kennwörter bzw. beim Authentifizierungstyp DATA_ENCRYPT Benutzer-IDs, Kennwörter und Benutzerdaten zu verschlüsseln: http://www.ibm.com/security/standards/st_evaluations.shtml.

Wechseln des Benutzers bei einer expliziten gesicherten Verbindung

Bei CLI/ODBC- und XA CLI/ODBC-Anwendungen stimmt das Authentifizierungsverfahren, das bei der Verarbeitung einer Anforderung zum Wechseln des Benutzers mit Authentifizierung verwendet wird, mit dem Verfahren überein, das auch bei der Herstellung der gesicherten Verbindung selbst angewendet wird. Aus diesem Grund wird davon ausgegangen, dass alle anderen variablen Sicherheitsattribute (z. B. Verschlüsselungsalgorithmus, Verschlüsselungsschlüssel und Plug-in-Namen), die während der Herstellung der expliziten gesicherten Verbindung verwendet wurden, bei allen Authentifizierungen gleich sind, die für eine Benutzerwechselanforderung auf dieser gesicherten Verbindung erforderlich sind. Bei Java-Anwendungen kann das Authentifizierungsverfahren bei einer Benutzerwechselanforderung (mithilfe einer Datenquelleneigenschaft) geändert werden.

Da ein Objekt eines gesicherten Kontextes so definiert werden kann, dass beim Benutzerwechsel auf einer gesicherten Verbindung *keine* Authentifizierung erforderlich ist, müssen die benutzerdefinierten Sicherheits-Plug-ins folgende Bedingungen erfüllen, um die Vorteile der Funktion für den Benutzerwechsel auf einer expliziten gesicherten Verbindung voll nutzen zu können:

- Sie müssen ein reines Benutzer-ID-Token akzeptieren.
- Sie müssen eine gültige DB2-Berechtigungs-ID für diese Benutzer-ID zurückgeben.

Anmerkung: Eine explizite gesicherte Verbindung kann nicht hergestellt werden, wenn der Authentifizierungstyp CLIENT aktiv ist.

Bereitgestellte Authentifizierungstypen

Die folgenden Authentifizierungstypen stehen zur Verfügung:

SERVER
Gibt an, dass die Authentifizierung auf dem Server durch den Sicherheitsmechanismus erfolgt, der für diese Konfiguration eingerichtet ist, zum Beispiel durch ein Sicherheits-Plug-in-Modul. Der Standardsicherheitsmechanismus besteht darin, dass bei Angabe einer Benutzer-ID und eines Kennworts beim Versuch, eine Verbindung herzustellen, diese Kombination aus ID und Kennwort an den Server gesendet und dort mit gültigen Kombinationen aus Benutzer-ID und Kennwort verglichen wird, um festzustellen, ob der Zugriff des Benutzers auf die Instanz zulässig ist.

Anmerkung: Der Server-Code erkennt, ob eine Verbindung lokal oder fern ist. Für lokale Verbindungen sind beim Authentifizierungstyp SERVER zur erfolgreichen Authentifizierung keine Benutzer-ID und kein Kennwort erforderlich.

SERVER_ENCRYPT
Gibt an, dass der Server verschlüsselte SERVER-Authentifizierungsschemata akzeptiert. Wenn die Clientauthentifizierung nicht angegeben ist, wird der Client mit der Methode überprüft, die auf dem Server ausgewählt ist. Benutzer-ID und Kennwort werden verschlüsselt, wenn sie über das Netz vom Client an den Server gesendet werden.

Wenn zwischen Client und Server die Authentifizierungsmethode SERVER_ENCRYPT ausgehandelt wird, können Sie Benutzer-ID und Kennwort über einen AES-Algorithmus (Advanced Encryption Standard) mit 256-Bit-Schlüssel verschlüsseln. Definieren Sie dazu den Konfigurationsparameter **alternate_auth_enc** des Datenbankmanagers. Mögliche Werte für diesen Konfigurationsparameter:

- `NOT_SPECIFIED` (Standardwert): Der Server akzeptiert den Verschlüsselungsalgorithmus, den der Client vorschlägt, den AES-Algorithmus mit 256-Bit-Schlüssel eingeschlossen.
- `AES_CMP`: Wenn von dem Client, der die Verbindung herstellt, DES vorgeschlagen wird, versucht der Server, die AES-Verschlüsselung auszuhandeln, wenn der Client die AES-Verschlüsselung unterstützt.
- `AES_ONLY`: Der Server akzeptiert ausschließlich die AES-Verschlüsselung. Unterstützt der Client diese Verschlüsselung nicht, wird die Verbindung zurückgewiesen.

Eine Verwendung der AES-Verschlüsselung ist nur möglich, wenn zwischen Client und Server die Authentifizierungsmethode SERVER_ENCRYPT ausgehandelt wurde.

CLIENT
Gibt an, dass die Authentifizierung mithilfe der Sicherheitseinrichtungen des Betriebssystems in der Datenbankpartition erfolgt, in der die Anwendung aufgerufen wird. Die bei der Herstellung einer CONNECT- oder AT-

TACH-Verbindung angegebene Benutzer-ID und das zugehörige Kennwort werden mit den zulässigen Kombinationen aus Benutzer-ID und Kennwort verglichen, die auf dem Clientknoten hinterlegt sind. Auf diese Weise wird festgestellt, ob die Benutzer-ID über die Zugriffsberechtigung für die Instanz verfügt. Es findet keine weitere Authentifizierung auf dem Datenbankserver statt. Diese Vorgehensweise wird auch als einmalige Anmeldung oder Einzelanmeldung (Single Sign-on) bezeichnet.

Wenn der Benutzer eine lokale Anmeldung bzw. Anmeldung auf dem Client vornimmt, ist der Benutzer nur dieser lokalen Client-Workstation bekannt.

Wenn für die ferne Instanz der Authentifizierungstyp CLIENT definiert ist, bestimmen zwei weitere Parameter den endgültigen Authentifizierungstyp: **trust_allclnts** und **trust_clntauth**.

Sicherheit auf CLIENT-Ebene nur für gesicherte Clients (TRUSTED):

Gesicherte Clients sind Clients, die über ein zuverlässiges lokales Sicherheitssystem verfügen.

Wenn der Authentifizierungstyp CLIENT ausgewählt wurde, kann eine zusätzliche Option zum Schutz vor Clients ausgewählt werden, deren Betriebsumgebung keine eigenen Sicherheitssysteme hat.

Zum Schutz gegen ungesicherte Clients kann der Administrator die Authentifizierung für gesicherte Clients auswählen, indem er den Parameter **trust_allclnts** auf den Wert NO setzt. Dies impliziert, dass alle gesicherten Plattformen den Benutzer im Namen des Servers authentifizieren können. Nicht gesicherte Clients werden auf dem Server authentifiziert und müssen eine Benutzer-ID und ein Kennwort bereitstellen. Der Konfigurationsparameter **trust_allclnts** wird verwendet, um anzugeben, ob Sie Clients vertrauen. Der Standardwert für diesen Parameter ist YES.

Anmerkung: Es ist möglich, alle Clients zu sichern (in dem Sie **trust_allclnts** auf YES setzen), auch wenn Sie einige solcher Clients haben, die über kein eigenes Sicherheitssystem zur Authentifizierung verfügen.

Vielleicht erscheint es Ihnen außerdem wünschenswert, die Authentifizierung auch für gesicherte Clients auf dem Server durchzuführen. Zur Festlegung, wo gesicherte Clients überprüft werden sollen, verwenden Sie den Konfigurationsparameter **trust_clntauth**. Der Standardwert für diesen Parameter ist CLIENT.

Anmerkung: Nur für gesicherte Clients gilt, dass wenn bei einem Verbindungsversuch mit CONNECT oder **ATTACH** keine Benutzer-ID oder kein Kennwort explizit angegeben wird, die Gültigkeitsprüfung auf dem Client stattfindet. Der Parameter **trust_clntauth** wird nur dazu verwendet, festzulegen, wo die Informationen, die in den Klauseln USER bzw. USING angegeben werden, zu überprüfen sind.

Definieren Sie den Parameter **trust_allclnts** mit DRDAONLY, um einen Schutz gegen alle Clients einschließlich der Clients für z/OS und System i, jedoch ausschließlich der nativen DB2-Clients für z/OS, OS/390, VM, VSE und System i einzurichten. Nur diese Clients dürfen die Authentifizierung auf der Clientseite ausführen.

Alle anderen Clients müssen eine Benutzer-ID und ein Kennwort bereitstellen, wobei die Authentifizierung vom Server durchgeführt wird.

Der Parameter **trust_clntauth** wird verwendet, um zu bestimmen, wo die zuvor genannten Clients authentifiziert werden: Wenn **trust_clntauth** auf den Wert CLIENT gesetzt ist, findet die Authentifizierung auf dem Client statt. Wenn **trust_clntauth** auf den Wert SERVER gesetzt ist, wird die Authentifizierung auf dem Client ausgeführt, wenn keine Benutzer-ID und kein Kennwort angegeben werden, und auf dem Server, wenn eine Benutzer-ID und ein Kennwort angegeben werden.

Tabelle 1. Authentifizierungsmodi mit einer Kombination der Parameter TRUST_ALLCLNTS und TRUST_CLNTAUTH

trust_ allclnts	trust_ clntauth	Ungesicherte Nicht-DRDA-Client-authentifizierung (ohne Benutzer-ID & ohne Kennwort)	Ungesicherte Nicht-DRDA-Client-authentifizierung (mit Benutzer-ID & ohne Kennwort)	Gesicherte Nicht-DRDA-Client-authentifizierung (ohne Benutzer-ID & ohne Kennwort)	Gesicherte Nicht-DRDA-Client-authentifizierung (mit Benutzer-ID & ohne Kennwort)	DRDA-Client-authentifizierung (ohne Benutzer-ID & ohne Kennwort)	DRDA-Client-authentifizierung (mit Benutzer-ID & ohne Kennwort)
YES	CLIENT	CLIENT	CLIENT	CLIENT	CLIENT	CLIENT	CLIENT
YES	SERVER	CLIENT	SERVER	CLIENT	SERVER	CLIENT	SERVER
NO	CLIENT	SERVER	SERVER	CLIENT	CLIENT	CLIENT	CLIENT
NO	SERVER	SERVER	SERVER	CLIENT	SERVER	CLIENT	SERVER
DRDAONLY	CLIENT	SERVER	SERVER	SERVER	SERVER	CLIENT	CLIENT
DRDAONLY	SERVER	SERVER	SERVER	SERVER	SERVER	CLIENT	SERVER

DATA_ENCRYPT

Der Server akzeptiert verschlüsselte SERVER-Authentifizierungsschemata und die Verschlüsselung von Benutzerdaten. Die Authentifizierung funktioniert in gleicher Weise wie für den Authentifizierungstyp SERVER_ENCRYPT erläutert. Benutzer-ID und Kennwort werden verschlüsselt, wenn sie über das Netz vom Client an den Server gesendet werden.

Die folgenden Benutzerdaten werden bei Verwendung dieses Authentifizierungstyps verschlüsselt:

- SQL- und XQuery-Anweisungen
- Daten von SQL-Programmvariablen
- Ausgabedaten aus der Serververarbeitung einer SQL- oder XQuery-Anweisung, einschließlich einer Beschreibung der Daten
- Einige oder alle Antwortgruppendaten, die aus einer Abfrage resultieren
- LOB-Datenströme (große Objekte)
- SQLDA-Deskriptoren

DATA_ENCRYPT_CMP

Der Server akzeptiert verschlüsselte SERVER-Authentifizierungsschemata und die Verschlüsselung von Benutzerdaten. Darüber hinaus bietet dieser Authentifizierungstyp Kompatibilität mit Produkten früherer Versionen, die den Authentifizierungstyp DATA_ENCRYPT nicht unterstützen. Diese Produkte erhalten die Möglichkeit, die Verbindung mit dem Authentifizierungstyp SERVER_ENCRYPT und ohne Verschlüsselung von Benutzerda-

ten herzustellen. Produkte, die den neuen Authentifizierungstyp unterstützen, müssen ihn verwenden. Dieser Authentifizierungstyp ist nur in der Konfigurationsdatei des Datenbankmanagers des Servers, jedoch nicht im Befehl **CATALOG DATABASE** gültig.

KERBEROS
Wird verwendet, wenn sowohl der DB2-Client als auch der Server unter Betriebssystemen arbeiten, die das Kerberos-Sicherheitsprotokoll unterstützen. Das Kerberos-Sicherheitsprotokoll führt die Authentifizierung mithilfe von Services anderer Hersteller aus und verwendet ein konventionelles Verschlüsselungsverfahren, um einen gemeinsam genutzten, geheimen Schlüssel zu erstellen. Dieser Schlüssel wird als Identitätsnachweis für den Benutzer definiert und zur Prüfung der Identität von Benutzern während aller Anforderungen für lokale Services oder Netzservices verwendet. Der Schlüssel beseitigt die Notwendigkeit, den Benutzernamen sowie das zugehörige Kennwort unverschlüsselt über das Netz zu übertragen. Mit dem Kerberos-Sicherheitsprotokoll kann eine einmalige Anmeldung (Single Sign-on) an einem fernen DB2-Datenbankserver durchgeführt werden. Der Kerberos-Authentifizierungstyp wird unter verschiedenen Betriebssystemen unterstützt; weitere Informationen finden Sie im Abschnitt mit den zugehörigen Informationen.

Die Kerberos-Authentifizierung funktioniert folgendermaßen:

1. Ein Benutzer, der sich an der Clientmaschine mit einem Domänenkonto anmeldet, wird in der Kerberos-Schlüsselverteilungszentrale (KDC - Key Distribution Center) auf dem Domänencontroller authentifiziert. Die Schlüsselverteilungszentrale gibt ein so genanntes Ticket-granting Ticket (TGT) an den Client aus.
2. Während der ersten Phase des Verbindungsaufbaus sendet der Server den Ziel-Principal-Namen, bei dem es sich um den Servicekontonamen für den Service des DB2-Datenbankservers handelt, an den Client. Mithilfe des Ziel-Principal-Namens des Servers und dem TGT fordert der Client ein Service-Ticket vom TGS (Ticket erteilenden Service, Ticket-Granting Service) an, der sich auf dem Domänencontroller befindet. Wenn sowohl das Ticket-granting Ticket des Clients als auch der Ziel-Principal-Name des Servers gültig sind, gibt der TGS ein Service-Ticket an den Client aus. Der Name des Principals, der im Datenbankverzeichnis aufgezeichnet wird, kann im Format 'name/instanz@REALM' angegeben werden. (Dieses Format ist zusätzlich zu den Formaten 'DOMÄNE\benutzerID' und 'benutzerID@xxx.xxx.xxx.com' möglich, die unter Windows akzeptiert werden.)
3. Der Client sendet dieses Service-Ticket über den Kommunikationskanal (bei dem es sich beispielsweise um TCP/IP handeln kann) an den Server.
4. Der Server prüft die Gültigkeit des Server-Tickets des Clients. Wenn das Service-Ticket des Clients gültig ist, wird die Authentifizierung abgeschlossen.

Es ist möglich, die Datenbanken auf der Clientmaschine zu katalogisieren und den Authentifizierungstyp Kerberos explizit zusammen mit dem Ziel-Principal-Namen des Servers anzugeben. Auf diese Weise lässt sich die erste Phase des Verbindungsaufbaus überspringen.

Wenn eine Benutzer-ID und ein Kennwort angegeben werden, fordert der Client das Ticket-granting Ticket für dieses Benutzerkonto an und verwendet es zur Authentifizierung.

KRB_SERVER_ENCRYPT
Gibt an, dass der Server Schemata der KERBEROS-Authentifizierung oder einer verschlüsselten SERVER-Authentifizierung akzeptiert. Wenn als Clientauthentifizierung KERBEROS angegeben wurde, wird der Client mithilfe des Kerberos-Sicherheitssystems authentifiziert. Wenn als Clientauthentifizierung SERVER_ENCRYPT angegeben ist, wird der Client mithilfe einer Benutzer-ID und eines verschlüsselten Kennworts authentifiziert. Wenn die Clientauthentifizierung nicht angegeben ist, arbeitet der Client mit Kerberos, falls verfügbar, oder mit Kennwortverschlüsselung. Für andere Clientauthentifizierungstypen wird ein Authentifizierungsfehler zurückgegeben. Als Authentifizierungstyp des Clients darf nicht KRB_SERVER_ENCRYPT angegeben werden.

Anmerkung: Die Kerberos-Authentifizierungstypen werden auf Clients und Servern unter verschiedenen Betriebssystemen unterstützt; weitere Informationen finden Sie im Abschnitt mit den zugehörigen Informationen. Bei Windows-Betriebssystemen müssen sowohl die Client- als auch die Servermaschine entweder zur gleichen Windows-Domäne oder zu gesicherten Domänen gehören. Dieser Authentifizierungstyp sollte verwendet werden, wenn der Server Kerberos unterstützt und einige, jedoch nicht alle, Clientmaschinen die Kerberos-Authentifizierung unterstützen.

GSSPLUGIN
Gibt an, dass der Server ein GSS-API-Plug-in zur Ausführung der Authentifizierung verwendet. Wenn die Clientauthentifizierung nicht angegeben ist, gibt der Server eine Liste der vom Server unterstützten Plug-ins, einschließlich aller Kerberos-Plug-ins, die im Konfigurationsparameter **srvcon_gssplugin_list** des Datenbankmanagers aufgelistet sind, an den Client zurück. Der Client wählt das erste Plug-in in der Liste aus, das im Plug-in-Verzeichnis des Clients zu finden ist. Wenn der Client kein Plug-in der Liste unterstützt, wird der Client mit dem Kerberos-Authentifizierungsschema authentifiziert (falls dies zurückgegeben wird). Wenn der Clientauthentifizierungstyp das Authentifizierungsschema GSSPLUGIN angibt, wird der Client mit dem ersten unterstützten Plug-in der Liste authentifiziert.

GSS_SERVER_ENCRYPT
Gibt an, dass der Server Plug-in-Authentifizierungsschemata oder verschlüsselte SERVER-Authentifizierungsschemata akzeptiert. Wenn die Clientauthentifizierung durch ein Plug-in erfolgt, wird der Client mit dem ersten vom Client unterstützten Plug-in in der Liste der vom Server unterstützten Plug-ins authentifiziert.

Wenn die Clientauthentifizierung nicht angegeben ist und eine implizite Verbindung hergestellt wird (d. h., der Client gibt bei der Herstellung der Verbindung keine Benutzer-ID und kein Kennwort an), gibt der Server eine Liste der vom Server unterstützten Plug-ins, das Kerberos-Authentifizierungsschema (wenn eines der Plug-ins in der Liste auf Kerberos basiert) und das verschlüsselte Serverauthentifizierungsschema zurück. Der Client wird mit dem ersten unterstützten Plug-in in der Liste authentifiziert, das im Plug-in-Verzeichnis des Clients zu finden ist. Wenn der Client keines der Plug-ins der Liste unterstützt, wird der Client mit dem Kerberos-Authentifizierungsschema authentifiziert. Wenn der Client das Kerberos-Authentifizierungsschema nicht unterstützt, wird der Client mit dem verschlüsselten SERVER-Authentifizierungsschema authentifiziert und die Verbindung schlägt fehl, weil das Kennwort fehlt. Ein Client unterstützt das Kerberos-Authentifizierungsschema, wenn für das Betriebssystem ein

von DB2 bereitgestelltes Kerberos-Plug-in vorhanden ist oder ein Kerberos-basiertes Plug-in im Konfigurationsparameter **srvcon_gssplugin_list** des Datenbankmanagers angegeben ist.

Wenn die Clientauthentifizierung nicht angegeben ist und eine explizite Verbindung hergestellt wird (d. h., die Benutzer-ID und das Kennwort werden angegeben), ist der Authentifizierungstyp mit SERVER_ENCRYPT äquivalent. In diesem Fall hängt die Auswahl des Verschlüsselungsalgorithmus für Benutzer-ID und Kennwort von der Einstellung des Konfigurationsparameters **alternate_auth_enc** des Datenbankmanagers ab.

Anmerkung:

1. Sperren Sie sich nicht versehentlich von Ihrer Instanz aus, wenn Sie Berechtigungsinformationen ändern, da der Zugriff auf die Konfigurationsdatei selbst durch Informationen in der Konfigurationsdatei geschützt wird. Mit den folgenden Parametern in der Konfigurationsdatei des Datenbankmanagers wird der Zugriff auf die Instanz gesteuert:
   - **authentication** *
   - **sysadm_group** *
   - **trust_allclnts**
   - **trust_clntauth**
   - **sysctrl_group**
   - **sysmaint_group**

   * kennzeichnet die beiden wichtigsten Parameter.

   Es gibt einige Maßnahmen, um dies zu verhindern: Wenn Sie sich versehentlich aus dem DB2-Datenbanksystem aussperren, gibt es auf allen Plattformen eine Sicherheitsoption, die Ihnen ermöglicht, die normalen Sicherheitsprüfungen des DB2-Datenbanksystems außer Kraft zu setzen, um die Konfigurationsdatei des Datenbankmanagers über einen Sicherheitsbenutzer des lokalen Betriebssystems, der über Zugriffsrechte einer hohen Berechtigungsstufe verfügt, zu aktualisieren. Dieser Benutzer hat *immer* das Zugriffsrecht zur Aktualisierung der Konfigurationsdatei des Datenbankmanagers und kann daher das Problem beheben. Diese Umgehung der Sicherheit ist jedoch auf eine lokale Aktualisierung der Konfigurationsdatei des Datenbankmanagers beschränkt. Sie können einen Sicherheitsbenutzer nicht über Remotezugriff oder für irgendeinen anderen DB2-Datenbankbefehl verwenden. Dieser Benutzer mit Sonderberechtigung wird folgendermaßen identifiziert:
   - UNIX-Plattformen: der Instanzeigner.
   - Windows-Plattform: Eine zur lokalen Gruppe der „Administratoren" gehörende Person.
   - Andere Plattformen: Da es auf anderen Plattformen keine lokalen Sicherheitseinrichtungen gibt, bestehen alle Benutzer lokale Sicherheitsprüfungen ohnehin.

Authentifizierungsaspekte für ferne Clients

Bei der Katalogisierung einer Datenbank für den fernen Zugriff können Sie den Authentifizierungstyp im Datenbankverzeichniseintrag angeben.

Der Authentifizierungstyp ist nicht erforderlich. Ist kein Typ angegeben, wird der Standardtyp SERVER_ENCRYPT verwendet. Wenn der Server jedoch den Authentifizierungstyp SERVER_ENCRYPT nicht unterstützt, versucht es der Client mit einem vom Server unterstützten Wert erneut. Wenn der Server mehrere Authentifizierungstypen unterstützt, trifft der Client keine Auswahl unter diesen, sondern

gibt stattdessen einen Fehler zurück. Die Rückgabe eines Fehlers soll sicherstellen, dass der korrekte Authentifizierungstyp verwendet wird. In diesem Fall muss der Client die Datenbank mit einem unterstützten Authentifizierungstyp katalogisieren. Wenn ein Authentifizierungstyp angegeben wird, kann die Authentifizierung unverzüglich beginnen, sofern der angegebene Wert mit dem auf dem Server übereinstimmt. Wenn keine Übereinstimmung erkannt wird, unternimmt die DB2-Datenbank einen Behebungsversuch. Dieser Behebungsversuch kann zum Ausgleich des Unterschieds zu mehr Übertragungen führen oder einen Fehler verursachen, wenn keine Behebung durch die DB2-Datenbank möglich ist. Im Fall einer fehlenden Übereinstimmung wird angenommen, dass der Wert auf dem Server korrekt ist.

Der Authentifizierungstyp DATA_ENCRYPT_CMP ermöglicht Clients früherer Versionen, die keine Datenverschlüsselung unterstützen, die Verbindung zu einem Server mit der Authentifizierung SERVER_ENCRYPT anstelle von DATA_ENCRYPT. Diese Authentifizierung funktioniert nicht, wenn Folgendes gilt:

- Die Version des Clients ist Version 7.2.
- Die Version des Gateways ist Version 8 FixPak 7 oder höher.
- Der Server ist Version 8 FixPak 7 oder höher.

Wenn diese Bedingungen alle zutreffen, kann der Client keine Verbindung zum Server herstellen. Damit die Verbindung hergestellt werden kann, müssen Sie entweder für den Client ein Upgrade auf Version 8 oder höher ausführen oder ein Gateway mit Version 8 FixPak 6 oder einer früheren Version verwenden.

Die Festlegung des zur Herstellung einer Verbindung verwendeten Authentifizierungstyps erfolgt durch die Angabe des geeigneten Authentifizierungstyps in Form eines Datenbankkatalogeintrags auf dem Gateway. Dies gilt sowohl für DB2 Connect-Szenarios als auch für Clients und Server in einer Umgebung mit partitionierten Datenbanken, in der für den Client die Registrierdatenbankvariable **DB2NODE** definiert ist. Sie katalogisieren den Authentifizierungstyp in der Katalogpartition mit der Absicht, zur entsprechenden Partition zu „springen" (engl. „hop"). In einem solchen Szenario wird der auf dem Gateway katalogisierte Authentifizierungstyp nicht verwendet, weil die Vereinbarung ausschließlich zwischen dem Client und dem Server stattfindet.

Sie müssen eventuell mehrere Datenbankaliasnamen auf dem Gateway mit unterschiedlichen Authentifizierungstypen katalogisieren, wenn Clients betrieben werden müssen, die unterschiedliche Authentifizierungstypen verwenden. Bei der Entscheidung, welcher Authentifizierungstyp auf einem Gateway zu katalogisieren ist, können Sie den gleichen Authentifizierungstyp verwenden wie auf dem Client und dem Server. Alternativ können Sie auch den Authentifizierungstyp NOTSPEC verwenden, wobei dieser Wert standardmäßig als SERVER interpretiert wird.

Authentifizierungsaspekte bei partitionierten Datenbanken

In einer partitionierten Datenbank muss jede Partition der Datenbank über dieselbe Menge definierter Benutzer und Gruppen verfügen. Wenn die Definitionen nicht übereinstimmen, kann der Benutzer in verschiedenen Partitionen zu verschiedenen Operationen berechtigt sein.

Konsistenz über alle Partitionen hinweg ist zu empfehlen.

Kerberos-Authentifizierung

Kerberos ist ein Netzauthentifizierungsprotokoll eines Fremdanbieters, das mit einem System gemeinsamer geheimer Schlüssel (Shared Secret Keys) zur sicheren

Authentifizierung eines Benutzers in einer ungesicherten Netzumgebung arbeitet. Das DB2-Datenbanksystem bietet unter den Betriebssystemen AIX, HP-UX, Solaris, Linux IA32 und AMD64 sowie unter Windows Unterstützung für das Kerberos-Authentifizierungsprotokoll.

Einführung

Die Kerberos-Authentifizierung wird über ein dreistufiges System verwaltet, in dem verschlüsselte Service-Tickets anstelle von unverschlüsselten Paaren aus Benutzer-ID und Kennwort zwischen dem Anwendungsserver und dem zugehörigen Client ausgetauscht werden. Diese verschlüsselten Service-Tickets (sog. *Berechtigungsnachweise*) werden von einem separaten Server bereitgestellt, der als Kerberos Key Distribution Center (KDC) bezeichnet wird. Berechtigungsnachweise haben eine endliche Gültigkeitsdauer und können nur vom jeweiligen Client und Server entschlüsselt werden. Diese Features verringern das Sicherheitsrisiko, selbst dann, wenn das Ticket im Netz abgefangen wird. Jeder Benutzer, bzw. in der Kerberos-Terminologie *Principal*, besitzt einen privaten Chiffrierschlüssel, der mit dem KDC gemeinsam verwendet wird. Zusammengenommen werden die Principals und Computer, die bei einem KDC registriert sind, als *Realm* bezeichnet.

Eine Schlüsselfunktion von Kerberos besteht in der Bereitstellung einer Umgebung für die einmalige Anmeldung (Single Sign-on), in der ein Benutzer seine Identität nur einmal nachweisen muss, um auf die Ressourcen innerhalb des Kerberos-Realms zuzugreifen. Diese Single Sign-on-Umgebung ermöglicht dem Benutzer die Herstellung einer Verbindung zu einem DB2-Datenbankserver, ohne dass er hierzu eine Benutzer-ID und ein Kennwort angeben muss. Ein weiterer Vorteil besteht darin, dass die Verwaltung der Benutzeridentifikation vereinfacht wird, da Kerberos für Principals ein zentrales Repository verwendet. Und schließlich unterstützt Kerberos eine gegenseitige Authentifizierung, mit deren Hilfe der Client auch die Identität des Servers überprüfen kann.

Einrichtung

Bevor Sie Kerberos für ein DB2-Datenbanksystem verwenden können, müssen Sie die Kerberos-Schicht auf allen Computern installieren und konfigurieren. Bei einer Standardkonfiguration müssen Sie dazu die folgenden Voraussetzungen erfüllen:

- Erstellen Sie die entsprechenden Principals.
- Stellen Sie sicher, dass der Client- und der Server-Computer und die Principals zum selben Realm oder zu vertrauenswürdigen Realms gehören. Vertrauenswürdige Realms werden in der Windows-Terminologie als vertrauenswürdige Domänen bezeichnet.
- Erstellen Sie gegebenenfalls Serverchiffrierschlüsseldateien.
- Synchronisieren Sie die Uhren auf allen Computern. Kerberos toleriert normalerweise eine Zeitabweichung von bis zu fünf Minuten. Wenn die Zeitabweichung größer ist, wird ein Vorabauthentifizierungsfehler ausgegeben, wenn Sie versuchen, Berechtigungsnachweise abzurufen.

Einrichten von Kerberos für einen DB2-Server

Bevor Sie die Kerberos-Authentifizierung für ein DB2-Datenbanksystem verwenden können, müssen Sie die Kerberos-Schicht auf allen Computern installieren und konfigurieren. Bei einer Standardkonfiguration müssen Sie dazu die auf dieser Seite aufgeführten Anweisungen befolgen:

Vorbereitende Schritte

Wenn Sie ein Linux-, Sun Solaris- oder HP-UX-Betriebssystem verwenden, dann müssen Sie sicherstellen, dass keine anderen Kerberos-Bibliotheken außer der Bibliothek `krb5` auf Ihrem System installiert sind. Andernfalls schlägt die Kerberos-Authentifizierung fehl und in den `db2diag`-Protokolldateien wird eine Nachricht protokolliert.

Wenn Sie mit einem Linux- oder Sun Solaris-Betriebssystem arbeiten, dann deinstallieren Sie alle Instanzen von IBM® Network Authentication Service (NAS) Toolkit und entfernen Sie alle Referenzen auf die NAS-Installationspfadpositionen aus der Systemvariablen **PATH**.

Informationen zu diesem Vorgang

Die Verwendung der Kerberos-Authentifizierung durch eine DB2-Datenbank hängt davon ab, ob die Sicherheitsauthentifizierung mithilfe der Berechtigungsnachweise, die von der Anwendung bereitgestellt werden, die die Verbindung herstellt, erfolgreich durchgeführt werden konnte. Wann immer sie verfügbar ist, wird außerdem die gegenseitige Kerberos-Authentifizierung unterstützt, bei der sowohl Client als auch Server ihre Identität für die Verwendung von Kerberos nachweisen müssen. Andere Kerberos-Funktionen, wie zum Beispiel das Signieren oder die Verschlüsselung von Nachrichten, sind jedoch nicht verfügbar.

Weitere Details zur Installation und Konfiguration von Kerberos-Produkten auf Ihren Systemen finden Sie unter http://www.ibm.com/developerworks/data/library/techarticle/dm-0603see/index.html oder in der Dokumentation des installierten Kerberos-Produkts.

Die Kerberos-Unterstützung für ein DB2-Datenbanksystem wird über das GSS-API-Sicherheits-Plug-in 'IBMkrb5' bereitgestellt. Dieses Plug-in wird sowohl für die Server- als auch für die Clientauthentifizierung verwendet. Die Plug-in-Bibliothek wird während der DB2-Installation an den folgenden Speicherpositionen installiert:

- Unter UNIX- und Linux-32-Bit-Betriebssystemen: Verzeichnisse `sqllib/security32/plugin/IBM/client` und `sqllib/security32/plugin/IBM/server`.
- Unter UNIX- und Linux-64-Bit-Betriebssystemen: Verzeichnisse `sqllib/security64/plugin/IBM/client` und `sqllib/security64/plugin/IBM/server`.
- Unter Windows-Betriebssystemen: Verzeichnisse `sqllib\security\plugin\IBM\client` und `sqllib\security\plugin\IBM\server`.

Der Quellcode für das UNIX- und Linux-Plug-in `IBMkrb5.C` ist im Verzeichnis `sqllib/samples/security/plugins` verfügbar. Bei 64-Bit-Windows-Betriebssystemen hat die Plug-in-Bibliothek den Namen `IBMkrb564.dll`.

Kerberos und Gruppen

Unter Kerberos ist das Konzept von Gruppen nicht definiert. Infolgedessen greift die DB2-Datenbankinstanz auf das lokale Betriebssystem zurück, um eine Gruppenliste für einen Kerberos-Principal abzurufen. Unter UNIX- und Linux-Betriebssystemen ist dazu auf allen Principals ein gleichwertiges Systemkonto erforderlich. Für den Principal *name@REALM* sammelt das DB2-Datenbankprodukt Gruppeninformationen, indem das lokale Betriebssystem nach den Namen aller Gruppen ab-

gefragt wird, zu denen der Betriebssystembenutzer *name* gehört. Wenn der Betriebssystembenutzer *name* nicht vorhanden ist, gehört die Berechtigungs-ID (AUTHID) nur zur Gruppe PUBLIC.

Unter Windows-Betriebssystemen wird einem Kerberos-Principal automatisch ein Domänenkonto zugeordnet. Der zusätzliche Schritt zur Erstellung eines separaten Betriebssystemkontos ist nicht erforderlich.

Kerberos-Chiffrierschlüsseldateien

Um Sicherheitskontextanforderungen akzeptieren zu können, muss jeder Kerberos-Service unter einem UNIX- oder Linux-Betriebssystem seine Berechtigungsnachweise in einer Chiffrierschlüsseldatei (*keytab*) hinterlegen. Diese Anforderung gilt für diejenigen Principals, die die DB2-Datenbankinstanz als Server-Principals verwendet. Der Serverschlüssel wird nur in der Standardchiffrierschlüsseldatei gesucht. Anweisungen zum Hinzufügen eines Schlüssels zur Chiffrierschlüsseldatei finden Sie in der Dokumentation zu Ihrem Kerberos-Produkt.

Das Konzept einer Chiffrierschlüsseldatei ist unter Windows-Betriebssystemen unbekannt. Das System sorgt automatisch für das Speichern und Abrufen der Berechtigungsnachweise für einen Principal.

Sie können den Namen der Standardchiffrierschlüsseldatei angeben, indem Sie die Umgebungsvariable **KRB5_KTNAME** verwenden. Da das Server-Plug-in jedoch innerhalb eines Prozesses der DB2-Datenbanksteuerkomponente ausgeführt wird, ist möglicherweise kein Zugriff auf diese Umgebungsvariable möglich. Zur Vermeidung dieser Situation fügen Sie die Umgebungsvariable **KRB5_KTNAME** mit dem Befehl **db2set** zur Registrierdatenbankvariablen **DB2ENVLIST** hinzu:

```
db2set DB2ENVLIST=KRB5_KTNAME
```

Da Kerberos für Windows keine Chiffrierschlüsseldateien verwendet, ist diese Option nur für einen Linux- oder UNIX-Server verfügbar.

Vorgehensweise

Gehen Sie wie folgt vor, um Kerberos für einen DB2-Server einzurichten:

1. Installieren Sie Kerberos, indem Sie einen der folgenden Schritte ausführen:
   - Installieren Sie unter AIX-Betriebssystemen NAS (Network Authentication Services) Toolkit for DB2 on AIX ab Version 1.4. Sie können das NAS-Paket unter der Adresse https://www.ibm.com/services/forms/preLogin.do?source=dm-nas herunterladen.
   - Installieren Sie unter Linux- und HP-UX-Betriebssystemen (nur 64-Bit-Systeme) das Kerberos-Paket `krb5`, das auf den Installationsmedien für Ihr Betriebssystem verfügbar ist.
   - Bei Sun Solaris-Betriebssystemen ist der Kerberos-Service im Release Solaris 10 enthalten. Es ist keine weitere Installation erforderlich.
   - Für Windows-Betriebssysteme müssen Sie auf Ihrem Domänencontroller Active Directory aktivieren.
2. Konfigurieren Sie das DB2-Produkt zur Verwendung des Kerberos-Plug-ins. Siehe hierzu „Implementieren eines Kerberos-Plug-ins" auf Seite 246.
3. Führen Sie für den DB2-Server einen Neustart durch.

Benennung und Zuordnung für Kerberos

Bevor Sie Kerberos für ein DB2-Datenbanksystem verwenden können, müssen Sie sicherstellen, dass der Client- und der Servercomputer und die Principals zum selben Realm oder zu vertrauenswürdigen Realms gehören.

Client-Principals

Alle eindeutigen Identitäten, die Kerberos-Tickets zur Authentifizierung empfangen können, werden als *Principal* bezeichnet. Eine Kerberos-Principalidentität wird entweder durch ein zweiteiliges oder ein mehrteiliges Format (entweder *name@REALM* oder *name/instanz@REALM*) definiert. Da die Komponente *name* für die Zuordnung der Berechtigungs-ID (AUTHID) verwendet wird, muss der Name (*name*) den Namenskonventionen der DB2-Datenbank entsprechen. Diese Regeln begrenzen den Namen auf eine Länge von 128 Zeichen und schränken außerdem die Zeichen ein, die ausgewählt werden können.

Anmerkung: Unter Windows-Betriebssystemen wird eine Kerberos-Principalidentität direkt einem Domänenbenutzer zugeordnet. Daraus folgt, dass die Kerberos-Authentifizierung für Windows-Betriebssysteme, die nicht zu einer Domäne oder einem Realm gehören, nicht zur Verfügung steht. Darüber hinaus unterstützen Windows-Betriebssysteme nur das zweiteilige Format zur Definition von Principalidentitäten, also das Format *name@domain*.

Zuordnung von Berechtigungs-IDs

Im Unterschied zu Betriebssystembenutzer-IDs, deren Existenzbereich normalerweise auf einen einzigen Computer beschränkt ist, können Kerberos-Principals auch in anderen Realms als dem eigenen authentifiziert werden. Sie können das potenzielle Problem doppelter Principalnamen dadurch vermeiden, dass der Principalname mit dem Realmnamen vollständig qualifiziert wird. In Kerberos hat ein vollständig qualifizierter Principalname das folgende Format:

name/instanz@REALM

Hierbei kann *instanz* für mehrere Instanznamen stehen, die durch einen Schrägstrich (/) voneinander getrennt sind. Beispiel: *name/instanz1/instanz2@REALM*. Alternativ hierzu können Sie das Feld *instanz* auch übergehen.

Der Realmname muss innerhalb aller Realms, die in einem Netz definiert sind, eindeutig sein. Zwischen der Berechtigungs-ID und dem Principalnamen (d. h. dem Wert im Feld *name*) muss eine Eins-zu-eins-Zuordnung bestehen. Diese einfache Zuordnung ist erforderlich, da die Berechtigungs-ID vom DB2-Datenbankmanager als Standardschema verwendet wird und daher einfach und logisch abgeleitet werden können sollte. Beachten Sie die potenziellen Probleme, die durch die folgenden Zuordnungen hervorgerufen werden:

- Principals mit dem gleichen Namen, die zu unterschiedlichen Realms gehören, werden derselben Berechtigungs-ID zugeordnet. Die beiden folgenden Principalnamen werden z. B. beide der Berechtigungs-ID 'gregor1x' zugeordnet:
  - gregor1x@EXAMPLE.COM
  - gregor1x@WWW.COM
- Principals mit dem gleichen Namen, jedoch auf verschiedenen Instanzen, werden derselben Berechtigungs-ID zugeordnet. Die beiden folgenden Principalnamen werden z. B. beide der Berechtigungs-ID 'gregor1x' zugeordnet:
  - gregor1x/bigmachine@EXAMPLE.COM

– gregor1x/littlemachine@EXAMPLE.COM

Aus diesem Grund müssen die folgenden Richtlinien befolgt werden:

- Verwenden Sie einen eindeutigen Namensbereich für einen Namen innerhalb aller vertrauenswürdigen Realms, über die auf den DB2-Datenbankserver zugegriffen wird.
- Alle Principals mit dem gleichen Wert im Feld *name* müssen unabhängig von der Instanz demselben Benutzer gehören.

Server-Principals

Unter UNIX- und Linux-Betriebssystemen wird als Name des Server-Principals für die DB2-Datenbankinstanz ein Name im Format *instanzname/vollständig qualifizierter hostname@REALM* verwendet. Dieser Principal muss in der Lage sein, Kerberos-Sicherheitskontexte zu akzeptieren. Außerdem muss er vorhanden sein, bevor die DB2-Datenbankinstanz gestartet wird, da der Servername bei der Initialisierung durch das Plug-in an die DB2-Datenbankinstanz gemeldet wird.

Unter Windows-Betriebssystemen wird der Server-Principal normalerweise durch das Domänenkonto angegeben, unter dem der DB2-Datenbankservice gestartet wurde. Eine Ausnahme ist, wenn die Instanz von dem Konto 'LocalSystem' gestartet wird. In diesem Fall wird der Name des Server-Principals als 'host/*hostname*' angegeben. Diese Identität ist nur dann gültig, wenn sowohl Client als auch Server Windows-Domänen angehören.

Windows-Betriebssysteme bieten keine Unterstützung für Namen, die mehr als zwei Teile umfassen. Beispiel: *komponente/komponente@REALM*. Dies verursacht ein Problem, wenn ein Windows-Client versucht, eine Verbindung zu einem UNIX-Server aufzubauen. Wenn Interoperabilität mit UNIX-Kerberos erforderlich ist, muss deshalb eine Zuordnung zwischen dem Kerberos-Principal und einem Windows-Konto in der Windows-Domäne definiert werden. Anweisungen hierzu finden Sie in der entsprechenden Windows-Dokumentation.

Sie können den Namen des Kerberos-Server-Principals, der vom DB2-Server unter UNIX- und Linux-Betriebssystemen verwendet wird, überschreiben. Geben Sie hierzu für die Umgebungsvariable **DB2_KRB5_PRINCIPAL** den vollständig qualifizierten Namen des Server-Principals an. Der Ersatzname für den Server-Principal wird vom DB2-Datenbanksystem erst erkannt, nachdem Sie für die Instanz mithilfe des Befehls **db2start** einen Neustart durchgeführt haben.

Aktivierung der Kerberos-Authentifizierung

Bevor Sie Kerberos für ein DB2-Datenbanksystem verwenden können, müssen Sie die Kerberos-Authentifizierung aktivieren.

Aktivieren der Kerberos-Authentifizierung auf dem Client

Zum Aktivieren der Kerberos-Authentifizierung auf dem Client müssen Sie den Konfigurationsparameter **clnt_krb_plugin** des Datenbankmanagers auf den Namen des Kerberos-Plug-ins setzen, das von Ihnen verwendet wird.

Für lokale Berechtigungen verwendet der Client Kerberos, wenn der Konfigurationsparameter **authentication** auf den Wert KERBEROS oder KRB_SERVER_ENCRYPT gesetzt ist. Ansonsten wird keine clientseitige Kerberos-Unterstützung angenommen.

Wichtig: Es werden keine Prüfungen durchgeführt, um festzustellen, ob die Kerberos-Unterstützung verfügbar ist.

Zur Aktivierung der Kerberos-Authentifizierung für abgehende Verbindungen zu einem DB2-Server müssen Sie stattdessen Kerberos als Authentifizierungstyp angeben, wenn Sie die Datenbank katalogisieren. Diese Operation ist im folgenden Beispiel dargestellt:

```
CATALOG DATABASE testdb AT NODE testknoten
   AUTHENTICATION KERBEROS TARGET PRINCIPAL
   service/host@REALM
```

Wenn die Authentifizierungsinformationen jedoch nicht angegeben werden, sendet der Server den Namen des Server-Principals an den Client.

Aktivieren der Kerberos-Authentifizierung auf dem Server

Um die Kerberos-Authentifizierung auf dem Server zu aktivieren, müssen Sie den gewünschten Kerberos-Plug-in-Namen in der Liste der Plug-ins angeben, die Sie im Konfigurationsparameter **srvcon_gssplugin_list** des Datenbankmanagers auf dem Server angeben. Wenn der Name des Kerberos-Plug-ins in dieser Liste enthalten ist, dann kann der Client den Server durchsuchen und die Kerberos-Authentifizierungsmethode auswählen, wenn eine Verbindung hergestellt wird.

Wenn für diesen Konfigurationsparameter kein Wert angegeben wird und wenn Sie für den Konfigurationsparameter **authentication** den Wert KERBEROS oder KRB_SERVER_ENCRYPT angeben, wird stattdessen das Standard-Kerberos-Plug-in IBMkrb5 verwendet. Sie können nur ein Kerberos-Plug-in angeben.

Um Kerberos ausschließlich zur Berechtigung eingehender Verbindungen zu verwenden, müssen Sie den Parameter **svrcon_auth** auf eine der beiden folgenden Optionen setzen:

- KERBEROS: Zur ausschließlichen Verwendung der Kerberos-Authentifizierung.
- KRB_SERVER_ENCRYPT: Zur Verwendung von Kerberos und der Berechtigung SERVER_ENCRYPT.

Wenn Sie Kerberos für eingehende Verbindungen und lokale Berechtigungen verwenden wollen, dann geben Sie für den Konfigurationsparameter **svrcon_auth** keinen Wert an und setzen Sie den Wert für den Konfigurationsparameter **authentication** auf eine der Kerberos-Optionen.

Kerberos-Plug-in-Erstellung

Um das Verhalten der Kerberos-Authentifizierung auf einem DB2-Datenbanksystem anzupassen, können Sie eigene Kerberos-Authentifizierungs-Plug-ins entwickeln.

Beim Erstellen eines Kerberos-Plug-ins sind die folgenden Aspekte zu berücksichtigen:

- Schreiben Sie das Kerberos-Plug-in als GSS-API-Plug-in. Setzen Sie jedoch in der Initialisierungsfunktion die Variable *plugintype* für den Funktionszeigerarray, der an die DB2-Datenbankinstanz zurückgegeben wird, auf den Wert DB2SEC_PLUGIN_TYPE_KERBEROS.
- Unter bestimmten Bedingungen meldet der Server den Namen des Server-Principals an den Client. Das Kerberos-Plug-in muss Principals im Format GSS_C_NT_USER_NAME (d. h. *server/host@REALM*) angeben. Das Format GSS_C_NT_HOSTBASED_SERVICE (d. h. *service@host*) wird nicht unterstützt.

Kerberos-Kompatibilität

Die DB2-Kerberos-Authentifizierung ist kompatibel mit IBM System z-, IBM i- und Windows-Systemen.

IBM System z- und IBM i-Kompatibilität

Um auf einem IBM System z- oder IBM i-System eine Verbindung zu einer Datenbank herzustellen, müssen Sie die Datenbank mithilfe der Parameter **AUTHENTICATION** und **KERBEROS TARGET PRINCIPAL** des Befehls **CATALOG DATABASE** katalogisieren.

Weder IBM System z- noch IBM i-Betriebssysteme unterstützen die Sicherheitsfunktion für die gegenseitige Authentifizierung von Kerberos.

Windows-Aspekte

Wenn Sie Kerberos unter Windows-Betriebssystemen verwenden, müssen Sie Folgendes beachten:

- Aufgrund der Art und Weise, wie Windows-Betriebssysteme bestimmte Fehler erkennen und melden, können die folgenden Bedingungen zu einem Fehler beim Sicherheits-Plug-in des Clients führen:
  - Abgelaufenes Konto
  - Ungültiges Kennwort
  - Abgelaufenes Kennwort
  - Vom Administrator erzwungene Kennwortänderung
  - Inaktiviertes Konto

  Darüber hinaus finden Sie in allen Fällen im DB2-Verwaltungsprotokoll oder in den **db2diag**-Protokolldateien die Fehlernachrichten `Anmeldung fehlgeschlagen` oder `Anmeldung verweigert`.
- Wenn ein Domänenkontoname auch lokal definiert ist, schlagen Verbindungsanforderungen, die den Domänennamen und das Kennwort explizit angeben, mit dem folgenden Fehler fehl: `Die lokale Sicherheitsautorität (LSA) ist nicht erreichbar`. Der Fehler wird dadurch verursacht, dass das Windows-Betriebssystem den lokalen Benutzer zuerst lokalisiert. Die Lösung besteht darin, den Benutzer in der Verbindungszeichenfolge vollständig qualifiziert anzugeben. Beispiel: `name@DOMAIN.IBM.COM`.
- Windows-Konten dürfen in ihren Namen nicht das Zeichen @ (kommerzielles A) enthalten, da das Kerberos-Plug-in von DB2 annimmt, dass es sich bei diesem Zeichen um das Domänennamentrennzeichen handelt.
- Wenn sowohl der Client als auch der Server unter einem Windows-Betriebssystem arbeiten, können Sie den DB2-Service mithilfe des lokalen Systemkontos (`LocalSystem`) starten. Wenn sich der Client und der Server jedoch in unterschiedlichen Domänen befinden, kann die Herstellung der Verbindung mit einem Fehler wegen eines ungültigen Namens für den Zielprincipal fehlschlagen. Um diesen Fehler zu vermeiden, müssen Sie den Zielprincipal explizit auf dem Client katalogisieren. Verwenden Sie hierzu den Befehl **CATALOG DATABASE** und den vollständig qualifizierten Hostnamen des Servers sowie den vollständig qualifizierten Domänennamen. Benutzen Sie das folgende Format: *host/server_hostname@serverdomänenname*. Beispiel: host248/server34.toronto.ibm.com@TORONTO.IBM.COM. Alternativ zur Verwendung des Kontos `LocalSystem` können Sie auch ein gültiges Domänenkonto verwenden.

Kennwortverwaltung auf Servern

Möglicherweise müssen Sie Tasks für die Kennwortverwaltung ausführen. Da derartige Tasks im Allgemeinen auf dem Server ausgeführt werden, viele Benutzer jedoch mit der Arbeit in der Serverumgebung nicht vertraut sind, stellt das Ausfüh-

ren dieser Tasks teilweise eine schwer zu bewältigende Herausforderung dar. Das DB2-Datenbanksystem stellt eine Möglichkeit zur Verfügung, mit der Sie Kennwörter aktualisieren und überprüfen können, auch wenn Sie sich nicht am Serversystem befinden.

Sie können neue Kennwörter zuordnen, wenn Sie eine Verbindung zu Datenbanken auf den folgenden Servern für die angegebenen (und neuere) Releases herstellen: DB2 Universal Database Version 8 unter AIX- und Windows-Betriebssystemen, DB2 Version 9.1 Fix Pack 3 oder höher unter Linux-Betriebssystemen, DB2 for z/OS Version 7 und DB2 for i V6R1.

Wird z. B. die Fehlernachricht SQL1404N „Das Kennwort ist abgelaufen" oder die Fehlernachricht SQL30082N „Die Sicherheitsverarbeitung ist mit Ursachencode 1 (PASSWORD EXPIRED) fehlgeschlagen" empfangen, können Sie das Kennwort wie folgt mit der Anweisung CONNECT ändern:

```
CONNECT TO datenbank USER benutzer-id USING kennwort NEW neues_kennwort CONFIRM
neues_kennwort
```

Berechtigungen, Zugriffsrechte und Objekteigentumsrecht

Benutzer (angegeben durch eine Berechtigungs-ID) können Operationen nur erfolgreich ausführen, wenn sie über die Berechtigung zum Ausführen der angegebenen Funktion verfügen. Zur Erstellung einer Tabelle muss ein Benutzer berechtigt sein, Tabellen zu erstellen, zur Änderung einer Tabelle muss ein Benutzer berechtigt sein, die Tabelle zu ändern, usw.

Der Datenbankmanager setzt voraus, dass jeder Benutzer zur Verwendung der jeweiligen Datenbankfunktion berechtigt wird, die er für bestimmte Tasks benötigt. Benutzern können die erforderlichen Berechtigungen erteilt werden, indem die Berechtigungen den jeweiligen Benutzer-IDs zugeordnet werden oder indem die Benutzer einer Rolle oder einer Gruppe zugeordnet werden, die über diese Berechtigung verfügt.

Es gibt drei Formen von Berechtigungen: *Verwaltungsberechtigung*, *Zugriffsrechte* und *LBAC-Berechtigungsnachweise*. Darüber hinaus ergeben sich aus den Objekteigentumsrechten bestimmte Berechtigungen für die erstellten Objekte. Diese Berechtigungsformen werden im folgenden Abschnitt erläutert.

Verwaltungsberechtigung

Die Person bzw. die Personen mit Verwaltungsberechtigung muss/müssen den Datenbankmanager steuern und ist/sind für die Sicherheit und Integrität der Daten verantwortlich.

Berechtigungen der Systemebene

Die Berechtigungen auf Systemebene ermöglichen eine unterschiedliche Steuerung über Funktionen auf Instanzebene:

- SYSADM (Systemadministrator)

  Die Berechtigung SYSADM für Systemadministratoren ermöglicht eine Steuerung über alle vom Datenbankmanager erstellten und verwalteten Ressourcen. Der Systemadministrator besitzt die Berechtigungen SYSCTRL, SYSMAINT und SYSMON. Der Benutzer, der die Berechtigung SYSADM besitzt, ist sowohl für die Steuerung des Datenbankmanagers als auch für die Gewährleistung der Sicherheit und Integrität der Daten zuständig.

- SYSCTRL

  Die Berechtigung SYSCTRL berechtigt zur Steuerung von Operationen, die sich auf Systemressourcen beziehen. Zum Beispiel kann ein Benutzer mit der Berechtigung SYSCTRL eine Datenbank erstellen, aktualisieren, starten, stoppen oder löschen. Ein solcher Benutzer kann darüber hinaus eine Instanz starten oder stoppen, jedoch nicht auf Tabellendaten zugreifen. Benutzer mit der Berechtigung SYSCTRL verfügen auch über die Berechtigung SYSMON.

- SYSMAINT

  Die Berechtigung SYSMAINT erteilt die Berechtigung, die zur Ausführung von Pflegeoperationen an allen zu einer Instanz gehörenden Datenbanken erforderlich ist. Ein Benutzer mit der Berechtigung SYSMAINT kann die Datenbankkonfiguration aktualisieren, eine Datenbank oder einen Tabellenbereich sichern, eine vorhandene Datenbank wiederherstellen und eine Datenbank überwachen. Ebenso wie SYSCTRL erteilt SYSMAINT keinen Zugriff auf Tabellendaten. Benutzer mit der Berechtigung SYSMAINT verfügen auch über die Berechtigung SYSMON.

- SYSMON (Systemmonitor)

  Die Berechtigung SYSMON für den Systemmonitor ist für die Verwendung des Datenbanksystemmonitors erforderlich.

Berechtigungen der Datenbankebene

Die Berechtigungen auf Datenbankebene ermöglichen eine Steuerung innerhalb der Datenbank:

- DBADM (Datenbankadministrator)

  Die Berechtigungsebene DBADM stellt die Verwaltungsberechtigung über eine einzige Datenbank bereit. Der Datenbankadministrator besitzt die Zugriffsrechte, die zum Erstellen von Objekten und zum Ausführen von Datenbankbefehlen erforderlich sind.

  Die Berechtigung DBADM kann nur von einem Benutzer mit der Berechtigung SECADM erteilt werden. Der Gruppe PUBLIC kann die Berechtigung DBADM nicht erteilt werden.

- SECADM (Sicherheitsadministrator)

  Die Berechtigungsebene SECADM stellt die Verwaltungsberechtigung für Sicherheitseinrichtungen für eine einzige Datenbank bereit. Die Verwaltungsberechtigung für Sicherheitseinrichtungen ermöglicht das Verwalten von Objekten für die Datenbanksicherheit (Datenbankrollen, Prüfrichtlinien, gesicherte Kontexte, Sicherheitskennsatzkomponenten und Sicherheitskennsätze) und das Erteilen und Entziehen sämtlicher Datenbankzugriffsrechte und -berechtigungen. Ein Benutzer mit der Berechtigung SECADM kann das Eigentumsrecht von Objekten übertragen, die ihm nicht gehören. Er kann darüber hinaus mit der Anweisung AUDIT einer bestimmten Datenbank bzw. einem bestimmten Datenbankobjekt auf dem Server eine Prüfrichtlinie zuordnen.

  Die Berechtigung SECADM hat kein eigenes Zugriffsrecht zum Zugreifen auf Daten, die in Tabellen gespeichert werden. Diese Berechtigung kann nur von einem Benutzer mit der Berechtigung SECADM erteilt werden. Der Gruppe PUBLIC kann die Berechtigung SECADM nicht erteilt werden.

- SQLADM (SQL-Administrator)

  Die Berechtigungsebene SQLADM stellt die Verwaltungsberechtigung zum Überwachen und Optimieren von SQL-Anweisungen für eine ein-

zelne Datenbank bereit. Sie kann von einem Benutzer mit der Berechtigung ACCESSCTRL oder SECADM erteilt werden.

- WLMADM (Administrator für Workload-Management)

  Die Verwaltungsberechtigung WLMADM ermöglicht das Verwalten von Objekten des Workload-Managements, z. B. Serviceklassen, Arbeitsaktionssets, Arbeitsklassensets und Auslastungen. Sie kann von einem Benutzer mit der Berechtigung ACCESSCTRL oder SECADM erteilt werden.

- EXPLAIN (Erläuterungsberechtigung)

  Die Berechtigungsebene EXPLAIN stellt die Verwaltungsberechtigung für das Erläutern von Abfrageplänen bereit, ohne jedoch zum Zugriff auf Daten zu berechtigen. Sie kann nur von einem Benutzer mit der Berechtigung ACCESSCTRL oder SECADM erteilt werden.

- ACCESSCTRL (Zugriffssteuerungsberechtigung)

  Die Berechtigungsebene ACCESSCTRL stellt die Verwaltungsberechtigung für die Ausgabe der folgenden Anweisungen GRANT (und REVOKE) bereit.

  – GRANT (Datenbankberechtigungen)

    Benutzer mit der Berechtigung ACCESSCTRL können nicht die Berechtigungen ACCESSCTRL, DATAACCESS, DBADM oder SECADM erteilen. Diese Berechtigungen können nur von einem Benutzer erteilt werden, der über die Berechtigung SECADM verfügt.
  – GRANT (Zugriffsrechte für globale Variablen)
  – GRANT (Indexzugriffsrechte)
  – GRANT (Modulzugriffsrechte)
  – GRANT (Paketzugriffsrechte)
  – GRANT (Routinenzugriffsrechte)
  – GRANT (Schemazugriffsrechte)
  – GRANT (Sequenzzugriffsrechte)
  – GRANT (Serverzugriffsrechte)
  – GRANT (Zugriffsrechte für Tabellen, Sichten oder Kurznamen)
  – GRANT (Tabellenbereichszugriffsrechte)
  – GRANT (Zugriffsrechte für Auslastungen)
  – GRANT (Zugriffsrechte für XSR-Objekte)

  Die Berechtigung ACCESSCTRL kann nur von einem Benutzer mit der Berechtigung SECADM erteilt werden. Der Gruppe PUBLIC kann die Berechtigung ACCESSCTRL nicht erteilt werden.

- DATAACCESS (Datenzugriffsberechtigung)

  Die Berechtigungsebene DATAACCESS umfasst die folgenden Zugriffsrechte und Berechtigungen.

  – Berechtigung LOAD
  – Zugriffsrechte SELECT, INSERT, UPDATE, DELETE für Tabellen, Sichten, Kurznamen und MQTs (MQT - Materialized Query Table, gespeicherte Abfragetabelle)
  – Zugriffsrecht EXECUTE für Pakete
  – Zugriffsrecht EXECUTE für Pakete
  – Zugriffsrecht EXECUTE für Routinen

Ausgenommen sind die Prüfroutinen AUDIT_ARCHIVE, AUDIT-_LIST_LOGS und AUDIT_DELIM_EXTRACT.

- Zugriffsrecht READ für alle globalen Variablen und Zugriffsrecht WRITE für alle globalen Variablen außer Variablen, die schreibgeschützt sind
- Zugriffsrecht USAGE für alle XSR-Objekte
- Zugriffsrecht USAGE für alle Sequenzen

Sie kann nur von einem Benutzer mit der Berechtigung SECADM erteilt werden. Der Gruppe PUBLIC kann die Berechtigung DATAACCESS nicht erteilt werden.

- Nicht auf die Verwaltungsebene bezogene Datenbankberechtigungen

  Zur Ausführung solcher Aktivitäten wie das Erstellen einer Tabelle oder einer Routine oder das Laden von Daten in eine Tabelle sind bestimmte Datenbankberechtigungen erforderlich. Die Datenbankberechtigung LOAD ist z. B. erforderlich, wenn mit dem Dienstprogramm **LOAD** Daten in Tabellen geladen werden sollen. Der Benutzer muss in diesem Fall auch über das Zugriffsrecht INSERT für die betreffende Tabelle verfügen.

Zugriffsrechte

Bei einem Zugriffsrecht handelt es sich um das Recht, eine Aktion oder Task auszuführen. Berechtigte Benutzer können Objekte erstellen, auf ihre eigenen Objekte zugreifen und Zugriffsrechte für ihre eigenen Objekte an andere Benutzer mithilfe der Anweisung GRANT weitergeben.

Zugriffsrechte können einzelnen Benutzern, Gruppen oder PUBLIC erteilt werden. PUBLIC ist eine spezielle Gruppe, die aus allen Benutzern, einschließlich zukünftigen Benutzern, besteht. Benutzer, die Mitglieder einer Gruppe sind, können die der Gruppe erteilten Zugriffsrechte indirekt an den Stellen nutzen, an denen Gruppen unterstützt werden.

Zugriffsrecht CONTROL: Der Besitz des Zugriffsrechts CONTROL für ein Objekt berechtigt einen Benutzer zum Zugriff auf dieses Datenbankobjekt sowie zum Erteilen und Entziehen von Zugriffsrechten für andere Benutzer in Bezug auf dieses Objekt.

Anmerkung: Das Zugriffsrecht CONTROL gilt nur für Tabellen, Sichten, Kurznamen und Pakete.

Wenn ein anderer Benutzer das Zugriffsrecht CONTROL für dieses Objekt benötigt, kann ein Benutzer mit der Berechtigung SECADM oder ACCESSCTRL das Zugriffsrecht CONTROL für dieses Objekt erteilen. Das Zugriffsrecht CONTROL kann dem Objekteigner nicht entzogen werden; allerdings kann der Objekteigner mit der Anweisung TRANSFER OWNERSHIP geändert werden.

Einzelzugriffsrecht: Einzelzugriffsrechte können erteilt werden, um einem Benutzer die Ausführung bestimmter Tasks für bestimmte Objekte zu erlauben. Benutzer mit der Verwaltungsberechtigung SECADM oder ACCESSCTRL oder dem Zugriffsrecht CONTROL können anderen Benutzern Zugriffsrechte erteilen und entziehen.

Einzelne Zugriffsrechte und Datenbankberechtigungen erlauben eine bestimmte Funktion, jedoch beinhalten sie nicht das Recht, die gleichen Zugriffsrechte oder Berechtigungen anderen Benutzern zu erteilen. Das Recht, Zugriffsrechte für Tabellen, Sichten, Schemata, Pakete, Routinen und Sequenzen zu erteilen, kann durch

die Klausel WITH GRANT OPTION in der Anweisung GRANT auf andere Benutzer übertragen werden. Jedoch erlaubt die Klausel WITH GRANT OPTION der Person, die das Zugriffsrecht erteilt, nicht, das erteilte Zugriffsrecht zu entziehen. Sie müssen über die Berechtigung SECADM bzw. ACCESSCTRL oder das Zugriffsrecht CONTROL verfügen, um das Zugriffsrecht entziehen zu können.

Zugriffsrechte für Objekte in einem Paket oder einer Routine: Wenn ein Benutzer über das Zugriffsrecht zum Ausführen eines Pakets oder einer Routine verfügt, sind nicht notwendigerweise bestimmte Zugriffsrechte für die Objekte erforderlich, die in dem Paket oder der Routine verwendet werden. Wenn das Paket oder die Routine statische SQL- oder XQuery-Anweisungen enthält, werden die Zugriffsrechte des Eigners des Pakets für diese Anweisungen verwendet. Wenn das Paket oder die Routine dynamische SQL- oder XQuery-Anweisungen enthält, ist die Berechtigungs-ID für die Zugriffsrechtsüberprüfung von der Einstellung der Option **DYNAMICRULES** des Befehls **BIND** desjenigen Pakets abhängig, das die dynamischen Abfrageanweisungen absetzt; außerdem ist sie davon abhängig, ob diese Anweisungen abgesetzt werden, wenn das Paket im Kontext einer Routine verwendet wird (mit Ausnahme der Prüfroutinen: AUDIT_ARCHIVE, AUDIT_LIST_LOGS, AUDIT_DELIM_EXTRACT).

Ein Benutzer oder eine Gruppe kann für jede Kombination individueller Zugriffsrechte oder Berechtigungen berechtigt werden. Wenn ein Zugriffsrecht einem Objekt zugeordnet wird, muss dieses Objekt vorhanden sein. Zum Beispiel kann einem Benutzer nur dann das Zugriffsrecht SELECT für eine Tabelle erteilt werden, wenn die Tabelle zuvor erstellt wurde.

Anmerkung: Wenn einem Berechtigungsnamen, der einen Benutzer oder eine Gruppe darstellt, Berechtigungen und Zugriffsrechte erteilt werden, und es keinen Benutzer oder keine Gruppe gibt, die mit diesem Namen erstellt wurden, ist Vorsicht geboten. Später kann ein Benutzer oder eine Gruppe mit diesem Namen erstellt werden und automatisch alle diesem Berechtigungsnamen zugeordneten Berechtigungen und Zugriffsrechte erhalten.

Die Anweisung REVOKE dient zum Entziehen zuvor erteilter Zugriffsrechte. Durch das Entziehen eines Zugriffsrechts für einen Berechtigungsnamen wird das von allen Berechtigungsnamen erteilte Zugriffsrecht widerrufen.

Durch das Entziehen eines Zugriffsrechts für einen Berechtigungsnamen wird das gleiche Zugriffsrecht nicht für andere Berechtigungsnamen widerrufen, die das Zugriffsrecht von diesem Berechtigungsnamen erhalten haben. Nehmen Sie zum Beispiel an, dass die Benutzerin CLAIRE dem Benutzer RICK das Zugriffsrecht SELECT WITH GRANT OPTION erteilt, und anschließend der Benutzer RICK das Zugriffsrecht SELECT den Benutzern BOBBY und CHRIS erteilt. Wenn nun die Benutzerin CLAIRE das Zugriffsrecht SELECT für RICK entzieht, behalten die Benutzer BOBBY und CHRIS das Zugriffsrecht SELECT.

LBAC-Berechtigungsnachweise

Mit der kennsatzbasierten Zugriffssteuerung (LBAC - Label-Based Access Control) kann der Sicherheitsadministrator exakt festlegen, wer Schreibzugriff und wer Lesezugriff auf einzelne Zeilen und einzelne Spalten erhält. Der Sicherheitsadministrator konfiguriert das LBAC-System, indem er Sicherheitsrichtlinien erstellt. Eine Sicherheitsrichtlinie (Security Policy) beschreibt die Bedingungen, die bestimmen, wer auf welche Daten Zugriff besitzt. Eine Tabelle kann nur durch eine Sicherheitsrichtlinie geschützt werden, jedoch können verschiedene Tabellen durch verschiedene Sicherheitsrichtlinien geschützt werden.

Nach der Erstellung einer Sicherheitsrichtlinie erstellt der Sicherheitsadministrator Datenbankobjekte, die als Sicherheitskennsätze (Security Labels) und Freistellungen (Exemptions) bezeichnet werden und die Teil der jeweiligen Richtlinie sind. Ein Sicherheitskennsatz beschreibt eine bestimmte Gruppe von Sicherheitsbedingungen. Eine Freistellung bietet die Möglichkeit, dass eine Regel zum Vergleich von Sicherheitskennsätzen für den Benutzer nicht angewendet wird, der die Freistellung besitzt, wenn er auf Daten zugreift, die durch die entsprechende Sicherheitsrichtlinie geschützt werden.

Nach der Erstellung kann ein Sicherheitskennsatz einzelnen Spalten und Zeilen in einer Tabelle zugeordnet werden, um die dort gespeicherten Daten zu schützen. Daten, die durch einen Sicherheitskennsatz geschützt sind, werden als geschützte Daten bezeichnet. Ein Sicherheitsadministrator berechtigt Benutzer zum Zugriff auf geschützte Daten, indem er ihnen Sicherheitskennsätze erteilt. Wenn ein Benutzer versucht, auf geschützte Daten zuzugreifen, wird sein Sicherheitskennsatz mit dem Sicherheitskennsatz verglichen, der die Daten schützt. Der schützende Kennsatz blockiert einige Sicherheitskennsätze und andere nicht.

Objekteigentumsrecht

Wenn ein Objekt erstellt wird, wird einer Berechtigungs-ID das *Eigentumsrecht* an diesem Objekt zugeordnet. Eigentumsrecht bedeutet, dass der Benutzer berechtigt ist, in jeder beliebigen gültigen SQL- oder XQuery-Anweisung auf das Objekt zu verweisen.

Wenn ein Objekt innerhalb eines Schemas erstellt wird, muss die Berechtigungs-ID der Anweisung über das erforderliche Zugriffsrecht verfügen, um Objekte in dem implizit oder explizit angegebenen Schema erstellen zu können. Das bedeutet, der Berechtigungsname muss entweder der Eigner des Schemas sein oder das Zugriffsrecht CREATEIN für das Schema besitzen.

Anmerkung: Diese Voraussetzung gilt nicht für die Erstellung von Tabellenbereichen, Pufferpools oder Datenbankpartitionsgruppen. Solche Objekte werden nicht in Schemata erstellt.

Wenn ein Objekt erstellt wird, ist die Berechtigungs-ID der Anweisung der definierende Benutzer dieses Objekts und wird standardmäßig zum Eigner des Objekts, wenn es erstellt ist.

Anmerkung: Es gibt eine Ausnahme. Wenn die Option AUTHORIZATION in der Anweisung CREATE SCHEMA angegeben wird, ist jedes andere Objekt, das im Rahmen der CREATE SCHEMA-Operation erstellt wird, der Berechtigungs-ID eigen, die in der Option AUTHORIZATION angegeben wird. Alle Objekte, die in dem Schema nach der ersten CREATE SCHEMA-Operation erstellt werden, sind jedoch der Berechtigungs-ID eigen, die der bestimmten CREATE-Anweisung zugeordnet wird.

Zum Beispiel erstellt die Anweisung `CREATE SCHEMA SCOTTSTUFF AUTHORIZATION SCOTT CREATE TABLE T1 (C1 INT)` das Schema `SCOTTSTUFF` und die Tabelle `SCOTTSTUFF.T1`, deren beider Eigner SCOTT ist. Nehmen Sie an, der Benutzer BOBBY erhält das Zugriffsrecht CREATEIN für das Schema `SCOTTSTUFF` und erstellt einen Index für die Tabelle `SCOTTSTUFF.T1`. Da der Index nach dem Schema erstellt wird, ist BOBBY der Eigner des Index für die Tabelle `SCOTTSTUFF.T1`.

Zugriffsrechte werden dem Objekteigner auf der Grundlage des Typs des erstellten Objekts erteilt:

- Das Zugriffsrecht CONTROL wird für neu erstellte Tabellen, Indizes und Pakete implizit erteilt. Dieses Zugriffsrecht berechtigt den Objektersteller zum Zugriff auf das Datenbankobjekt sowie zum Erteilen und Entziehen von Zugriffsrechten für andere Benutzer in Bezug auf dieses Objekt. Wenn ein anderer Benutzer das Zugriffsrecht CONTROL für dieses Objekt benötigt, muss ein Benutzer mit der Berechtigung SECADM oder ACCESSCTRL das Zugriffsrecht CONTROL für dieses Objekt erteilen. Das Zugriffsrecht CONTROL kann nicht durch den Objekteigner entzogen werden.
- Das Zugriffsrecht CONTROL wird implizit für neu erstellte Sichten erteilt, wenn der Objekteigner über das Zugriffsrecht CONTROL für alle Tabellen, Sichten und Kurznamen verfügt, die in der Sichtdefinition angegeben sind.
- Anderen Objekten wie Triggern, Routinen, Sequenzen, Tabellenbereichen und Pufferpools ist kein Zugriffsrecht CONTROL zugeordnet. Der Objekteigner erhält jedoch automatisch alle einem Objekt zugeordneten Zugriffsrechte. Diese Zugriffsrechte beinhalten die Klausel WITH GRANT OPTION, soweit dies unterstützt wird. Der Objekteigner kann diese Zugriffsrechte anderen Benutzern somit mithilfe der Anweisung GRANT erteilen. Erstellt BENUTZER1 einen Tabellenbereich, verfügt BENUTZER1 automatisch über das Zugriffsrecht USEAUTH mit WITH GRANT OPTION für diesen Tabellenbereich und kann anderen Benutzern das Zugriffsrecht USEAUTH erteilen. Darüber hinaus kann der Objekteigner das Objekt ändern und löschen oder ihm einen Kommentar hinzufügen. Diese Berechtigungen gelten für den Objekteigner implizit und können nicht entzogen werden.

Bestimmte Zugriffsrechte für das Objekt, z. B. das Ändern einer Tabelle, können vom Eigner erteilt werden und können dem Eigner von einem Benutzer mit der Berechtigung ACCESSCTRL oder SECADM entzogen werden. Bestimmte Zugriffsrechte für das Objekt, z. B. zum Kommentieren einer Tabelle, können durch den Eigner nicht erteilt und dem Eigner auch nicht entzogen werden. Verwenden Sie die Anweisung TRANSFER OWNERSHIP, um diese Zugriffsrechte einem anderen Benutzer zu erteilen. Wenn ein Objekt erstellt wird, ist die Berechtigungs-ID der Anweisung der definierende Benutzer dieses Objekts und wird standardmäßig zum Eigner des Objekts, wenn es erstellt ist. Wenn allerdings ein Paket mit dem Befehl **BIND** erstellt und die Option **OWNER** *berechtigungs-id* angegeben wird, ergibt sich der Eigner der Objekte, die mit den statischen SQL-Anweisungen in dem Paket erstellt werden, aus dem Wert für *berechtigungs-id*. Außerdem handelt es sich, wenn die Klausel AUTHORIZATION in einer CREATE SCHEMA-Anweisung angegeben wird, bei dem Berechtigungsnamen, der nach dem Schlüsselwort AUTHORIZATION angegeben wird, um den Eigner des Schemas.

Ein Sicherheitsadministrator oder der Objekteigner kann zum Ändern des Eigentumsrechts eines Datenbankobjekts die Anweisung TRANSFER OWNERSHIP verwenden. Daher kann ein Administrator im Namen einer Berechtigungs-ID ein Objekt erstellen, indem das Objekt mit der Berechtigungs-ID als Qualifikationsmerkmal erstellt und anschließend die Anweisung TRANSFER OWNERSHIP zum Übertragen des Eigentumsrechts, über das der Administrator für das Objekt verfügt, an die Berechtigungs-ID verwendet wird.

Berechtigungen - Übersicht

Auf Instanzebene und auf Datenbankebene gibt es verschiedene Verwaltungsberechtigungen. Diese Verwaltungsberechtigung beinhalten bestimmte Zugriffsrechte und Berechtigungen, die so zusammengestellt sind, dass Sie sie Benutzern, die bei Ihrer Datenbankinstallation für die entsprechenden Aufgaben zuständig sind, erteilen können.

Berechtigungen der Instanzebene

Die Berechtigungen der Instanzebene ermöglichen es Ihnen, Funktionen auszuführen, die die gesamte Instanz betreffen, z. B. Datenbanken erstellen und aufrüsten, Tabellenbereiche verwalten und die Aktivitäts- und Leistungsdaten der Instanz überwachen. Auf Instanzebene gibt es keine Berechtigung, die den Zugriff auf Daten in Datenbanktabellen ermöglicht. Das folgende Diagramm enthält eine Zusammenfassung der Möglichkeiten, die die Verwaltungsberechtigungen der Instanzebene beinhalten:

- SYSADM - für Benutzer, die die Instanz insgesamt verwalten
- SYSCTRL - für Benutzer, die eine Datenbankmanagerinstanz verwalten
- SYSMAINT - für Benutzer, die Datenbanken in einer Instanz pflegen
- SYSMON - für Benutzer, die die Instanz und die zugehörigen Datenbanken überwachen

Benutzer mit einer übergeordneten Berechtigung verfügen über alle Möglichkeiten, die die untergeordneten Berechtigungen beinhalten: Ein Benutzer mit der Berechtigung SYSCTRL kann z. B. auch die Funktionen ausführen, zu deren Ausführung Benutzer mit der Berechtigung SYSMAINT und SYSMON berechtigt sind.

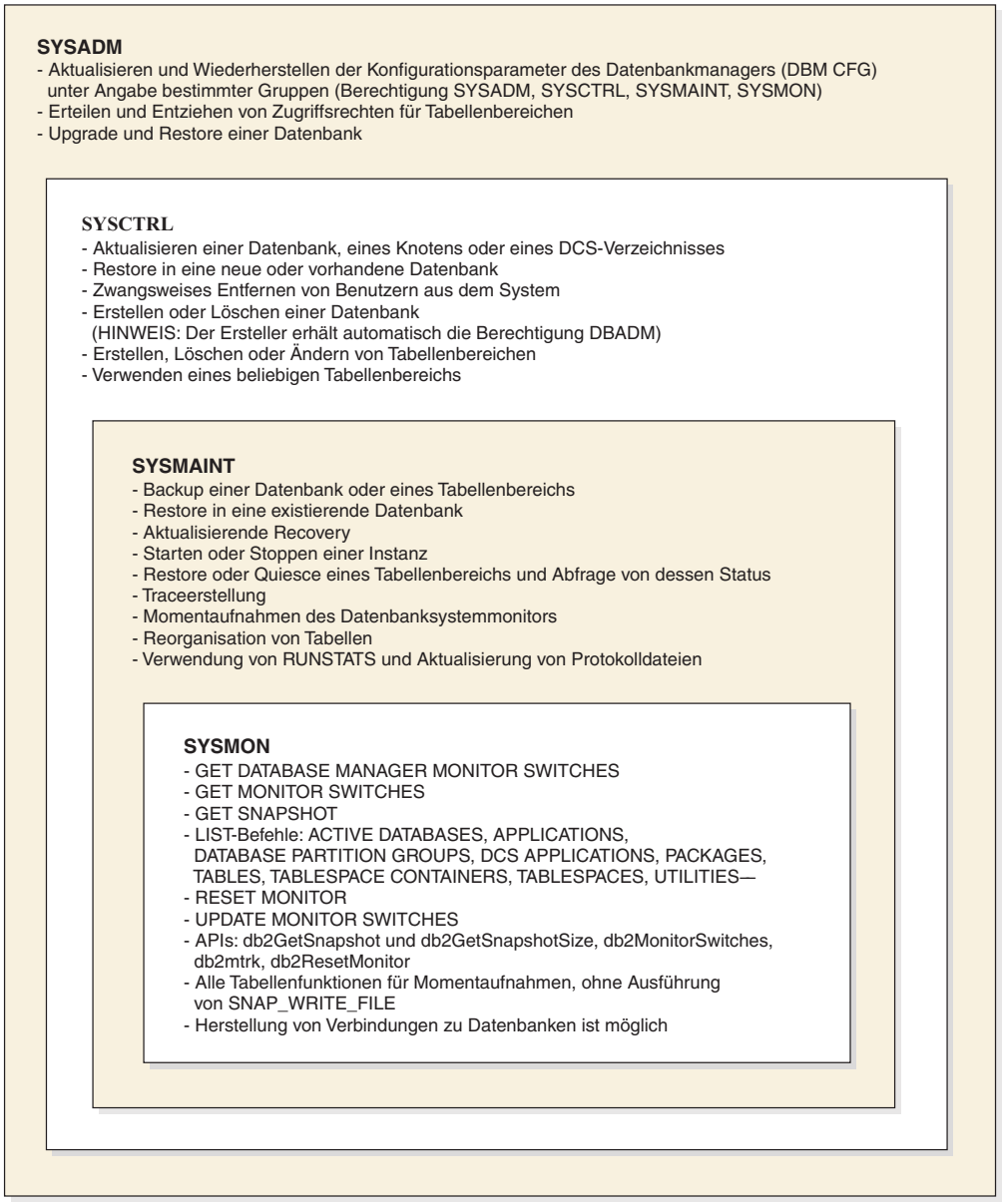


Abbildung 1. Berechtigungen der Instanzebene

Berechtigungen der Datenbankebene

Die Berechtigungen der Datenbankebene ermöglichen es Ihnen, Funktionen wie das Erteilen und Entziehen von Zugriffsrechten, das Einfügen, Auswählen, Löschen und Aktualisieren von Daten sowie das Verwalten von Auslastungen für eine bestimmte Datenbank auszuführen. Das folgende Diagramm enthält eine Zusammenfassung zu den Möglichkeiten, die die einzelnen Berechtigungen der Datenbankebene eröffnen. Folgende Verwaltungsberechtigungen für Datenbanken sind verfügbar:

- SECADM - für Benutzer, die die Sicherheitseinrichtungen für eine Datenbank verwalten
- DBADM - für Benutzer, die eine Datenbank verwalten

- ACCESSCTRL - für Benutzer, die Berechtigungen und Zugriffsrechte (SECADM, DBADM, ACCESSCTRL und DATAACCESS ausgenommen) erteilen und entziehen. Zum Erteilen und Erziehen der Berechtigungen SECADM, DBADM, ACCESSCTRL und DATAACCESS ist die Berechtigung SECADM erforderlich.
- DATAACCESS - für Benutzer, die auf Daten zugreifen können müssen
- SQLADM - für Benutzer, die SQL-Abfragen überwachen und optimieren
- WLMADM - für Benutzer, die Auslastungen verwalten
- EXPLAIN - für Benutzer, die Abfragepläne erläutern müssen. Diese Berechtigung ermöglicht nicht den Zugriff auf die Daten selbst.

Das folgende Diagramm verdeutlicht, welche übergeordneten Berechtigungen die Möglichkeiten beinhalten, die einer untergeordneten Berechtigung zugeordnet sind. Ein Benutzer mit der Berechtigung DBADM kann z. B. die Funktionen ausführen, zu denen Benutzer mit der Berechtigung SQLADM und EXPLAIN berechtigt sind, sowie alle Funktionen, zu denen Benutzer mit der Berechtigung WLMADM berechtigt sind (das Erteilen des Zugriffsrechts USAGE für Auslastungen ausgenommen).

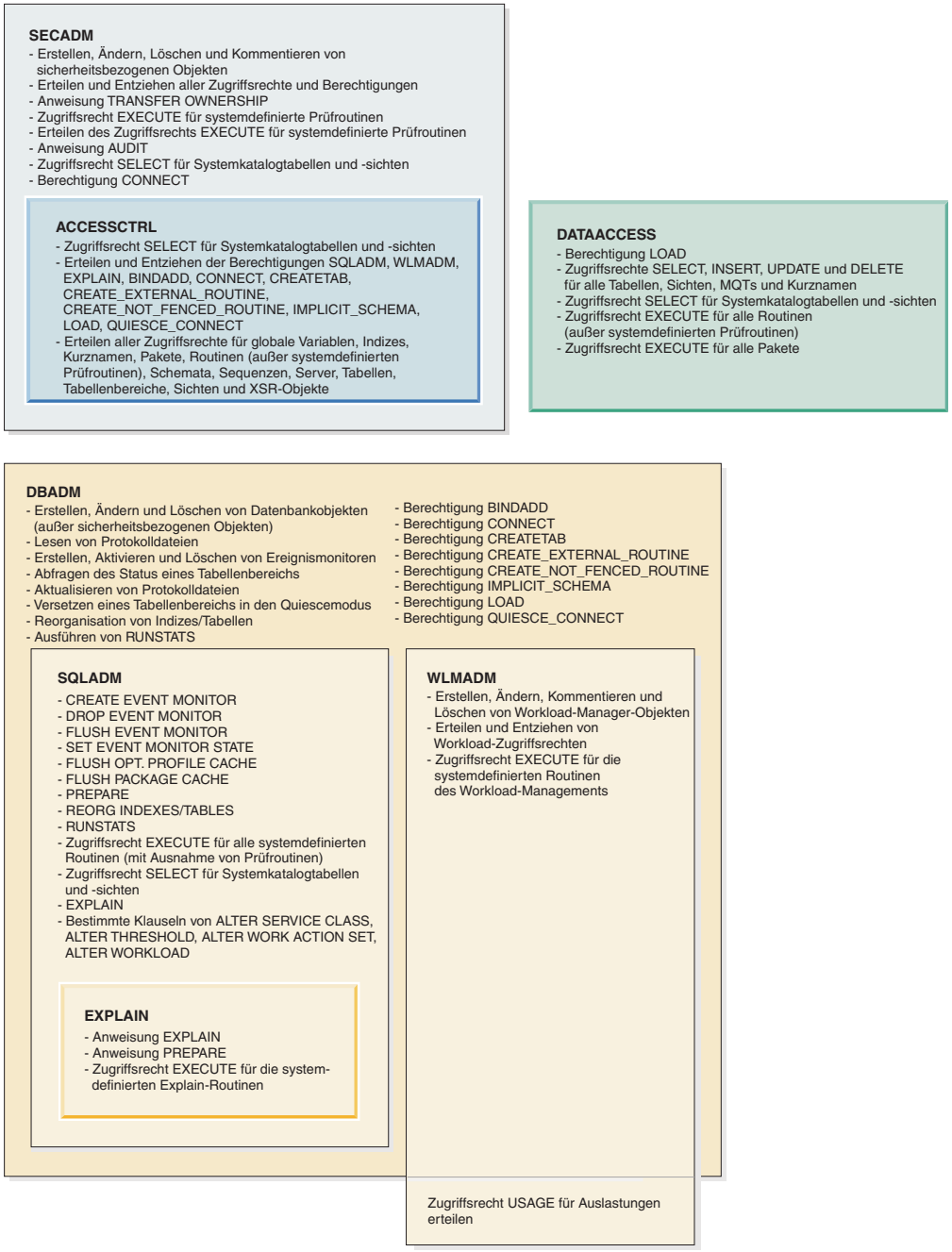


Abbildung 2. Berechtigungen der Datenbankebene

Berechtigungen der Instanzebene

Systemverwaltungsberechtigung (SYSADM)

Die Berechtigung SYSADM ist die höchste Stufe der Verwaltungsberechtigung auf Instanzebene. Benutzer mit der Berechtigung SYSADM können einige Dienstprogramme ausführen und verschiedene Datenbank- und Datenbankmanagerbefehle innerhalb der Instanz ausgeben.

Die Berechtigung SYSADM wird der Gruppe zugeordnet, die im Konfigurationsparameter **sysadm_group** angegeben ist. Die Zugehörigkeit zu dieser Gruppe wird außerhalb des Datenbankmanagers über die auf Ihrer Plattform verwendete Sicherheitseinrichtung gesteuert.

Nur ein Benutzer mit der Berechtigung SYSADM kann folgende Funktionen ausführen:

- Aufrüsten einer Datenbank
- Wiederherstellen einer Datenbank
- Ändern der Konfigurationsdatei des Datenbankmanagers (einschließlich Angabe der Gruppen, die über die Berechtigung SYSADM, SYSCTRL, SYSMAINT oder SYSMON verfügen)

Ein Benutzer mit der Berechtigung SYSADM kann die Zugriffsrechte für Tabellenbereiche erteilen oder entziehen und beliebige Tabellenbereiche verwenden.

Anmerkung: Wenn ein Benutzer mit der Berechtigung SYSADM eine Datenbank erstellt, werden diesem Benutzer automatisch die Berechtigungen ACCESSCTRL, DATAACCESS, DBADM und SECADM für die Datenbank erteilt. Wenn der Benutzer an einem Zugriff auf die Datenbank als Datenbank- oder Sicherheitsadministrator gehindert werden soll, müssen Sie diese Datenbankberechtigungen des Benutzers explizit entziehen.

In Releases vor Version 9.7 beinhaltet die Berechtigung SYSADM implizit die Berechtigung DBADM und die Möglichkeit, sämtliche Berechtigungen und Zugriffsrechte zu erteilen und zu entziehen. Version 9.7 verfügt über ein aktualisiertes DB2-Berechtigungsmodell, bei dem die Aufgaben von Systemadministrator, Datenbankadministrator und Sicherheitsadministrator klar voneinander getrennt sind. Teil dieser Neuerung ist eine Einschränkung der Möglichkeiten, die der Berechtigung SYSADM zuvor zugeordnet waren.

In Version 9.7 schließt nur die Berechtigung SECADM die Möglichkeit zum Erteilen und Entziehen sämtlicher Berechtigungen und Zugriffsrechte ein.

Benutzern mit der Berechtigung SYSADM, die über dieselben Möglichkeiten wie in Version 9.5 verfügen sollen (abgesehen von der Möglichkeit, die Berechtigung SECADM zu erteilen), muss der Sicherheitsadministrator explizit die Benutzerberechtigung DBADM sowie die neuen Berechtigungen DATAACCESS und ACCESSCTRL erteilen. Diese neuen Berechtigungen können mithilfe der Anweisung GRANT DBADM ON DATABASE mit den Standardoptionen WITH DATAACCESS und WITH ACCESSCTRL erteilt werden. Die Berechtigung DATAACCESS ermöglicht den Zugriff auf Daten in einer bestimmten Datenbank, während die Berechtigung ACCESSCTRL das Erteilen und Entziehen von Zugriffsrechten und Berechtigungen, die nicht auf die Verwaltungsebene bezogen sind, für eine bestimmte Datenbank ermöglicht.

Aspektes des Windows-Kontos LocalSystem

Auf Windows-Systemen gilt das lokale Systemkonto LocalSystem als Systemadministratorkonto mit der Berechtigung SYSADM, wenn der Konfigurationsparameter **sysadm_group** des Datenbankmanagers nicht angegeben ist. Die Änderungen in Version 9.7 in Bezug auf die Berechtigung SYSADM betreffen alle DB2-Anwendungen, die über das lokale Systemkonto ausgeführt werden. Diese Anwendungen werden im Allgemeinen in Form von Windows-Diensten geschrieben und unter dem lokalen Systemkonto als Anmeldekonto für den Service ausgeführt. Wenn

über diese Anwendungen Datenbankaktionen ausgeführt werden sollen, die nicht mehr in den Geltungsbereich der Berechtigung SYSADM fallen, müssen Sie das lokale Systemkonto mit den erforderlichen Datenbankzugriffsrechten und -berechtigungen ausstatten. Erteilen Sie dem lokalen Systemkonto z. B. mit der Anweisung GRANT (Datenbankberechtigungen) die Berechtigung DBADM, wenn eine Anwendung Datenbankadministratorberechtigungen erfordert. Dabei ist zu beachten, dass die Berechtigungs-ID für das lokale Systemkonto SYSTEM lautet.

Systemsteuerungsberechtigung (SYSCTRL)

Die Berechtigung SYSCTRL ist die höchste Stufe der Systemsteuerungsberechtigung. Diese Berechtigung bietet die Möglichkeit, Pflege- und Dienstprogrammoperationen für die Datenbankmanagerinstanz und ihre Datenbanken auszuführen. Diese Operationen können die Systemressourcen beeinflussen, aber sie ermöglichen keinen direkten Zugriff auf Daten in den Datenbanken.

Die Systemsteuerungsberechtigung ist für Benutzer gedacht, die eine Datenbankmanagerinstanz mit sensiblen Daten verwalten.

Die Berechtigung SYSCTRL wird der Gruppe zugeordnet, die im Konfigurationsparameter **sysctrl_group** angegeben ist. Wenn eine Gruppe angegeben wird, wird die Zugehörigkeit zu dieser Gruppe außerhalb des Datenbankmanagers durch die auf Ihrer Plattform verwendete Sicherheitseinrichtung gesteuert.

Nur ein Benutzer, der mindestens über die Berechtigung SYSCTRL verfügt, kann folgende Aktionen ausführen:

- Aktualisieren einer Datenbank, eines Knotens oder eines DCS-Verzeichnisses (Distributed Connection Services)
- Zwangsweises Entfernen von Benutzern aus dem System
- Erstellen oder Löschen einer Datenbank
- Löschen, Erstellen oder Ändern eines Tabellenbereichs
- Verwendung eines beliebigen Tabellenbereichs
- Restore in eine neue oder vorhandene Datenbank

Zusätzlich kann ein Benutzer mit der Berechtigung SYSCTRL die Funktionen von Benutzern mit Systempflegeberechtigung (SYSMAINT) und Systemmonitorberechtigung (SYSMON) ausführen.

Benutzer mit der Berechtigung SYSCTRL verfügen auch über das implizite Zugriffsrecht, eine Verbindung zu einer Datenbank herzustellen.

Anmerkung: Wenn Benutzer mit der Berechtigung SYSCTRL Datenbanken erstellen, werden ihnen automatisch die expliziten Berechtigungen ACCESSCTRL, DATAACCESS, DBADM und SECADM für die Datenbank erteilt. Wenn der Datenbankersteller aus der SYSCTRL-Gruppe gelöscht wird und Sie ihn auch daran hindern möchten, als Administrator auf die Datenbank zuzugreifen, müssen Sie diese vier Administratorberechtigungen explizit entziehen.

Systempflegeberechtigung (SYSMAINT)

Die Berechtigung SYSMAINT ist die zweite Stufe der Systemsteuerungsberechtigung. Diese Berechtigung bietet die Möglichkeit, Pflege- und Dienstprogrammoperationen für die Datenbankmanagerinstanz und ihre Datenbanken auszuführen. Diese Operationen können die Systemressourcen beeinflussen, aber sie ermöglichen keinen direkten Zugriff auf Daten in den Datenbanken.

Die Systempflegeberechtigung ist für Benutzer konzipiert, die Datenbanken innerhalb einer Datenbankmanagerinstanz pflegen, die sensible Daten enthält.

Die Berechtigung SYSMAINT wird der Gruppe zugeordnet, die im Konfigurationsparameter **sysmaint_group** angegeben ist. Wenn eine Gruppe angegeben wird, wird die Zugehörigkeit zu dieser Gruppe außerhalb des Datenbankmanagers durch die auf Ihrer Plattform verwendete Sicherheitseinrichtung gesteuert.

Nur ein Benutzer mit der Berechtigung SYSMAINT oder einer höheren Systemberechtigung kann folgende Aktionen ausführen:

- Backup einer Datenbank oder eines Tabellenbereichs
- Restore in eine existierende Datenbank
- Ausführen einer aktualisierenden Recovery
- Starten oder Stoppen einer Instanz
- Restore eines Tabellenbereichs
- Durchführen eines Trace mit dem Befehl **db2trc**
- Erstellen von Momentaufnahmen einer Datenbankmanagerinstanz oder der zugehörigen Datenbanken

Ein Benutzer mit der Berechtigung SYSMAINT kann folgende Aktionen ausführen:

- Abfragen des Status eines Tabellenbereichs
- Aktualisieren von Protokolldateien
- Versetzen eines Tabellenbereichs in den Quiescemodus
- Neuorganisieren einer Tabelle
- Sammeln von Katalogstatistiken mit dem Dienstprogramm **RUNSTATS**

Benutzer mit der Berechtigung SYSMAINT verfügen auch über das implizite Zugriffsrecht, eine Verbindung zu einer Datenbank herzustellen, und sie können die Funktionen von Benutzern mit Systemmonitorberechtigung (SYSMON) ausführen.

Systemmonitorberechtigung (SYSMON)

Die Berechtigung SYSMON ermöglicht die Verwendung des Datenbanksystemmonitors zur Erstellung von Momentaufnahmen (Snapshots) einer Datenbankmanagerinstanz oder seiner Datenbanken.

Die Berechtigung SYSMON wird der Gruppe zugeordnet, die im Konfigurationsparameter **sysmon_group** angegeben ist. Wenn eine Gruppe angegeben wird, wird die Zugehörigkeit zu dieser Gruppe außerhalb des Datenbankmanagers durch die auf Ihrer Plattform verwendete Sicherheitseinrichtung gesteuert.

Die Berechtigung SYSMON gibt dem Benutzer die Möglichkeit, die folgenden Befehle auszuführen:

- **GET DATABASE MANAGER MONITOR SWITCHES**
- **GET MONITOR SWITCHES**
- **GET SNAPSHOT**
- LIST (verschiedene Befehle):
  - **LIST ACTIVE DATABASES**
  - **LIST APPLICATIONS**
  - **LIST DATABASE PARTITION GROUPS**
  - **LIST DCS APPLICATIONS**
  - **LIST PACKAGES**
  - **LIST TABLES**
  - **LIST TABLESPACE CONTAINERS**

  - **LIST TABLESPACES**
  - **LIST UTILITIES**
- **RESET MONITOR**
- **UPDATE MONITOR SWITCHES**

Die Berechtigung SYSMON gibt dem Benutzer die Möglichkeit, die folgenden APIs zu verwenden:
- db2GetSnapshot - Momentaufnahme erstellen
- db2GetSnapshotSize - Größe des erforderlichen Ausgabepuffers für db2GetSnapshot() schätzen
- db2MonitorSwitches - Monitorschalter abrufen/aktualisieren
- db2mtrk - Speichertracker
- db2ResetMonitor - Monitor zurücksetzen

Die Berechtigung SYSMON gibt dem Benutzer die Möglichkeit, die folgenden SQL-Tabellenfunktionen zu verwenden:

- Alle Tabellenfunktionen zur Erstellung von Momentaufnahmen, ohne zuvor SYSPROC.SNAP_WRITE_FILE auszuführen.

  SYSPROC.SNAP_WRITE_FILE erfasst eine Momentaufnahme und speichert ihren Inhalt in einer Datei. Wenn eine der Tabellenfunktionen für Momentaufnahmen mit null Eingabeparametern aufgerufen wird, wird der Inhalt der Datei anstelle einer Echtzeitmomentaufnahme des Systems zurückgegeben.

Datenbankberechtigungen

Jede Datenbankberechtigung verleiht der Berechtigungs-ID, der sie erteilt ist, das Recht, einen bestimmten Typ von Aktion an der Datenbank als Ganzer auszuführen. Datenbankberechtigungen unterscheiden sich von Zugriffsrechten, die zur Ausführung einer bestimmten Aktion an einem bestimmten Datenbankobjekt, zum Beispiel einer Tabelle oder einem Index, berechtigen.

Die folgenden Datenbankberechtigungen sind verfügbar:

ACCESSCTRL
Berechtigt den Inhaber zum Erteilen und Entziehen aller Objektzugriffsrechte und Datenbankberechtigungen mit Ausnahme der Zugriffsrechte für Prüfroutinen und der Berechtigungen ACCESSCTRL, DATAACCESS, DBADM und SECADM.

BINDADD
Berechtigt den Inhaber zur Erstellung neuer Pakete in der Datenbank.

CONNECT
Berechtigt den Inhaber zur Herstellung einer Verbindung zur Datenbank.

CREATETAB
Berechtigt den Inhaber zur Erstellung neuer Tabellen in der Datenbank.

CREATE_EXTERNAL_ROUTINE
Berechtigt den Inhaber zur Erstellung einer Prozedur zur Verwendung durch Anwendungen und andere Benutzer der Datenbank.

CREATE_NOT_FENCED_ROUTINE
Berechtigt den Inhaber zum Erstellen einer benutzerdefinierten Funktion (UDF) oder Prozedur, die nicht abgeschirmt (*Not Fenced*) ist. CREATE_EXTERNAL_ROUTINE wird jedem Benutzer automatisch erteilt, dem CREATE_NOT_FENCED_ROUTINE erteilt wird.

Achtung: Der Datenbankmanager schützt seinen Speicher und seine Steuerblöcke nicht vor UDFs oder Prozeduren, die nicht abgeschirmt sind. Benutzer mit dieser Berechtigung müssen ihre UDFs deshalb sorgfältig testen, bevor sie sie als nicht abgeschirmt registrieren.

DATAACCESS
Berechtigt den Inhaber zum Zugriff auf Daten, die in Datenbanktabellen gespeichert sind.

DBADM
Mit dieser Berechtigung wird der Inhaber zum Datenbankadministrator. Als Datenbankadministrator verfügt der Inhaber gleichzeitig über alle übrigen Datenbankberechtigungen mit Ausnahme von ACCESSCTRL, DATAACCESS und SECADM.

EXPLAIN
Berechtigt den Inhaber dazu, Abfragepläne zu erläutern, ohne dabei über die Zugriffsrechte für die Daten in den Tabellen zu verfügen, auf die sich die Abfragepläne beziehen.

IMPLICIT_SCHEMA
Berechtigt jeden Benutzer zur impliziten Erstellung eines Schemas, indem er ein Objekt mit der Anweisung CREATE und einem Schemanamen, der noch nicht existiert, erstellt. SYSIBM wird zum Eigner des implizit erstellten Schemas, und die Gruppe PUBLIC erhält das Zugriffsrecht zur Erstellung von Objekten in diesem Schema.

LOAD
Berechtigt den Inhaber zum Laden von Daten in eine Tabelle.

QUIESCE_CONNECT
Berechtigt den Inhaber zum Zugriff auf die Datenbank, während sie im Quiescemodus ist.

SECADM
Mit dieser Berechtigung wird der Inhaber zum Sicherheitsadministrator der Datenbank.

SQLADM
Berechtigt den Inhaber zum Überwachen und Optimieren von SQL-Anweisungen.

WLMADM
Mit dieser Berechtigung wird der Inhaber zum Workloadadministrator. Inhaber der Berechtigung WLMADM können Workload-Manager-Objekte erstellen und löschen, Workload-Manager-Zugriffsrechte erteilen und entziehen und Workload-Manager-Routinen ausführen.

Nur Berechtigungs-IDs mit der Berechtigung SECADM können die Berechtigungen ACCESSCTRL, DATAACCESS, DBADM und SECADM erteilen. Alle anderen Berechtigungen können mit Berechtigungs-IDs erteilt werden, die über die Berechtigungen ACCESSCTRL oder SECADM verfügen.

Wenn der Gruppe PUBLIC eine Datenbankberechtigung entzogen werden soll, muss eine Berechtigungs-ID mit der Berechtigung ACCESSCTRL oder SECADM diese Berechtigung explizit entziehen (REVOKE).

Sicherheitsverwaltungsberechtigung (SECADM)

Bei der Berechtigung SECADM handelt es sich um eine Sicherheitsverwaltungsberechtigung für eine bestimmte Datenbank. Diese Berechtigung ermöglicht es Ihnen, sicherheitsbezogene Datenbankobjekte zu erstellen und zu verwalten sowie Daten-

bankberechtigungen und -zugriffsrechte zu erteilen und zu entziehen. Darüber hinaus kann der Sicherheitsadministrator Prüfsystemroutinen ausführen und die Benutzer verwalten, die diese Routinen ebenfalls ausführen können.

Die Berechtigung SECADM ermöglicht den Zugriff auf Daten (mit SELECT) in Katalogtabellen und -sichten, lässt jedoch keinen Zugriff auf Daten zu, die in Benutzertabellen gespeichert sind.

Die Berechtigung SECADM kann nur durch den Sicherheitsadministrator (mit der Berechtigung SECADM) erteilt und Benutzern, Gruppen oder Rollen zugeordnet werden. Die Gruppe PUBLIC kann weder direkt noch indirekt über die Berechtigung SECADM verfügen.

Die Berechtigung SECADM ermöglicht Benutzern das Ausführen der folgenden Operationen:

- Erstellen, Ändern, Kommentieren und Löschen folgender Elemente:
  - Prüfrichtlinien
  - Sicherheitkennsatzkomponenten
  - Sicherheitsrichtlinien
  - Gesicherte Kontexte
- Erstellen, Kommentieren und Löschen folgender Elemente:
  - Rollen
  - Sicherheitskennsätze
- Erteilen und Entziehen von Datenbankzugriffsrechten und -berechtigungen
- Führen Sie die angegebenen Tasks mit den folgenden Prüfroutinen aus:
  - Die gespeicherte Prozedur und die Tabellenfunktion SYSPROC.AUDIT_ARCHIVE archivieren Prüfprotokolle.
  - Die Tabellenfunktion SYSPROC.AUDIT_LIST_LOGS ermöglicht es Ihnen, für Ihre Zwecke interessante Protokolle zu ermitteln.
  - Die gespeicherte Prozedur SYSPROC.AUDIT_DELIM_EXTRACT extrahiert Daten zu Analysezwecken in Dateien mit begrenzter Satzlänge.

  Der Sicherheitsadministrator kann anderen Benutzern das Zugriffsrecht EXECUTE für diese Routinen erteilen und entziehen und die in seinen Bereich fallenden Aufgaben auf diese Weise bei Bedarf delegieren. Das Zugriffsrecht EXECUTE für diese Routinen kann nur vom Sicherheitsadministrator erteilt werden. Das Zugriffsrecht EXECUTE mit WITH GRANT OPTION kann für diese Routinen nicht erteilt werden (SQLSTATE 42501).
- Verwenden der Anweisung AUDIT, um einer bestimmten Datenbank bzw. einem Datenbankobjekt auf dem Server eine Prüfrichtlinie zuzuordnen.
- Ausführen der SQL-Anweisung TRANSFER OWNERSHIP zum Übertragen von Objekten, deren Eigner nicht die Berechtigungs-ID der Anweisung ist

Keine andere Berechtigung berechtigt zu diesen Operationen.

Nur der Sicherheitsadministrator (SECADM) kann anderen Benutzern die Berechtigungen ACCESSCTRL, DATAACCESS, DBADM und SECADM mit Grant erteilen.

Version 9.7 verfügt über ein aktualisiertes DB2-Berechtigungsmodell, bei dem die Aufgaben von Systemadministrator, Datenbankadministrator und Sicherheitsadministrator klar voneinander getrennt sind. Teil dieser Neuerung ist eine Erweiterung der Möglichkeiten, die der Berechtigung SECADM zuvor zugeordnet waren. In Releases vor Version 9.7 beinhaltet die Berechtigung SECADM nicht die Möglichkeit,

sämtliche Berechtigungen und Zugriffsrechte zu erteilen und zu entziehen. Die Berechtigung SECADM kann in diesen Releases darüber hinaus nur einem Benutzer, nicht einer Rolle oder Gruppe erteilt werden. Außerdem bot die Berechtigung SECADM nicht die Möglichkeit, anderen Benutzern das Zugriffsrecht EXECUTE für integrierte Prüfprozeduren und Tabellenfunktionen zu erteilen.

Datenbankadministratorberechtigung (DBADM)

Die Berechtigung DBADM ist eine Verwaltungsberechtigung für eine bestimmte Datenbank. Der Datenbankadministrator besitzt die Zugriffsrechte, die zum Erstellen von Objekten und zum Ausführen von Datenbankbefehlen erforderlich sind. Darüber hinaus verfügen Benutzer mit der Berechtigung DBADM über das Zugriffsrecht SELECT für die Systemkatalogtabellen und -sichten und können alle integrierten DB2-Routinen (Prüfroutinen ausgenommen) ausführen.

Die Berechtigung DBADM kann nur durch den Sicherheitsadministrator (mit der Berechtigung SECADM) erteilt und entzogen werden. Die Berechtigung kann Benutzern, Gruppen oder Rollen zugeordnet werden. Die Gruppe PUBLIC kann weder direkt noch indirekt über die Berechtigung DBADM verfügen.

Mit der Berechtigung DBADM für eine Datenbank kann ein Benutzer die folgenden Aktionen für diese Datenbank durchführen:

- Erstellen, Ändern und Löschen von Datenbankobjekten, die nicht zu Sicherheitseinrichtungen gehören
- Lesen von Protokolldateien
- Erstellen, Aktivieren und Löschen von Ereignismonitoren
- Abfragen des Status eines Tabellenbereichs
- Aktualisieren von Protokolldateien
- Versetzen eines Tabellenbereichs in den Quiescemodus
- Neuorganisieren einer Tabelle
- Sammeln von Katalogstatistiken mit dem Dienstprogramm **RUNSTATS**

Die Berechtigung SQLADM und die Berechtigung WLMADM umfassen Teile der Berechtigung DBADM. Die Berechtigung WLMADM ermöglicht zusätzlich das Erteilen des Zugriffrechts USAGE für Auslastungen.

Erteilen der Berechtigung DATAACCESS mit der Berechtigung DBADM

Der Sicherheitsadministrator kann festlegen, ob ein Datenbankadministrator auf Daten in der Datenbank zugreifen kann. Für den Zugriff auf Daten einer Datenbank ist die Berechtigung DATAACCESS für die betreffende Datenbank erforderlich. Der Sicherheitsadministrator kann einem Datenbankadministrator diese Berechtigung mit der Option WITH DATAACCESS der Anweisung GRANT DBADM ON DATABASE erteilen. Standardmäßig wird die Berechtigung DATAACCESS erteilt, wenn weder die Option WITH DATAACCCESS noch die Option WITHOUT DATAACCCESS angegeben ist.

Mit der Angabe GRANT DBADM WITHOUT ACCESSCTRL in der SQL-Anweisung können Sie die Datenbankadministratorberechtigung ohne die Berechtigung DATAACCESS erteilen.

Erteilen der Berechtigung ACCESSCTRL mit der Berechtigung DBADM

Der Sicherheitsadministrator kann festlegen, ob ein Datenbankadministrator Zugriffsrechte für die Datenbank erteilen und entziehen kann. Die Berechtigung AC-

CESSCTRL ermöglicht einem Benutzer das Erteilen und Entziehen von Zugriffsrechten und Berechtigungen, die nicht auf die Verwaltungsebene bezogen sind, für eine bestimmte Datenbank. Der Sicherheitsadministrator kann einem Datenbankadministrator diese Berechtigung mit der Option WITH ACCESSCTRL der Anweisung GRANT DBADM ON DATABASE erteilen. Standardmäßig wird die Berechtigung ACCESSCTRL erteilt, wenn weder die Option WITH ACCESSCTRL noch die Option WITHOUT ACCESSCTRL angegeben ist.

Mit der Angabe GRANT DBADM WITHOUT ACCESSCTRL in der SQL-Anweisung können Sie die Datenbankadministratorberechtigung ohne die Berechtigung ACCESSCRTL erteilen.

Entziehen der Berechtigung DBADM

Hat ein Sicherheitsadministrator die Berechtigung DBADM einschließlich der Berechtigung DATAACCESS oder ACCESSCTRL erteilt, können die Berechtigungen DATAACCESS und ACCESSCTRL nur entzogen werden, indem sie explizit vom Sicherheitsadministrator entzogen werden. Angenommen z. B. der Sicherheitsadministrator erteilt einem Benutzer die Berechtigung DBADM:

```
GRANT DBADM ON DATABASE TO benutzer1
```

Standardmäßig ist definiert, dass die Berechtigungen DATAACCESS und ACCESSCTRL *benutzer1* ebenfalls erteilt werden.

Zu einem späteren Zeitpunkt kann der Sicherheitsadministrator *benutzer1* die Berechtigung DBADM wie folgt entziehen:

```
REVOKE DBADM ON DATABASE FROM benutzer1
```

Der Benutzer *benutzer1* verfügt nun zwar nicht mehr über die Berechtigung DBADM, weiterhin jedoch über die Berechtigungen DATAACCESS und ACCESSCTRL.

Diese verbleibenden Berechtigungen können nur entzogen werden, indem sie explizit vom Sicherheitsadministrator entzogen werden:

```
REVOKE ACCESSCTRL, DATAACCESS ON DATABASE FROM benutzer1
```

Abweichungen bei der Berechtigung DBADM in früheren Releases

Version 9.7 verfügt über ein aktualisiertes DB2-Berechtigungsmodell, bei dem die Aufgaben von Systemadministrator, Datenbankadministrator und Sicherheitsadministrator klar voneinander getrennt sind. Teil dieser Neuerung ist eine Änderung der Möglichkeiten, die der Berechtigung DBADM zuvor zugeordnet waren. In Releases vor Version 9.7 beinhaltet die Berechtigung DBADM automatisch die Möglichkeit, auf die Daten einer Datenbank zuzugreifen und Zugriffsrechte für eine Datenbank zu erteilen und zu entziehen. In Version 9.7 sind diese Möglichkeiten, wie zuvor beschrieben, den neuen Berechtigungen DATAACCESS und ACCESSCTRL zugeordnet.

In Releases vor Version 9.7 beinhaltet die Berechtigung DBADM darüber hinaus automatisch die folgenden Berechtigungen:

- BINDADD
- CONNECT
- CREATETAB
- CREATE_EXTERNAL_ROUTINE

- CREATE_NOT_FENCED_ROUTINE
- IMPLICIT_SCHEMA
- QUIESCE_CONNECT
- LOAD

In Releases vor 9.7 werden diese Berechtigungen bei einem Widerruf der Berechtigung DBADM nicht entzogen.

In Version 9.7 gehören diese Berechtigungen nun zu der Berechtigung DBADM. Wird die Berechtigung DBADM in Version 9.7 entzogen, werden auch diese Berechtigungen entzogen.

Verfügt ein Benutzer bei einem Upgrade auf Version 9.7 jedoch bereits über die Berechtigung DBADM, werden diese Berechtigungen anschließend bei einem Widerruf der Berechtigung DBADM nicht entzogen. Ein Widerruf der Berechtigung DBADM in Version 9.7 führt nur dazu, dass einem Benutzer diese Berechtigungen entzogen werden, wenn die Berechtigungen erst in Version 9.7 in Zusammenhang mit der Berechtigung DBADM erteilt wurden.

Verwaltungsberechtigung für Zugriffssteuerung (ACCESSCTRL)

Die Berechtigung ACCESSCTRL ermöglicht das Erteilen und Entziehen von Zugriffsrechten und Berechtigungen für Objekte einer bestimmten Datenbank. Die Berechtigung ACCESSCTRL beinhaltet kein eigenes Zugriffsrecht zum Zugreifen auf Daten, die in Tabellen gespeichert werden, die Katalogtabellen und -sichten ausgenommen.

Die Berechtigung ACCESSCTRL kann nur vom Sicherheitsadministrator (dem Benutzer mit der Berechtigung SECADM) erteilt werden. Die Berechtigung kann an einen Benutzer, eine Gruppe oder eine Rolle vergeben werden. Die Gruppe PUBLIC kann weder direkt noch indirekt über die Berechtigung ACCESSCTRL verfügen. Die Berechtigung ACCESSCTRL ermöglicht Benutzern das Ausführen der folgenden Operationen:

- Erteilen und Entziehen der folgenden Verwaltungsberechtigungen:
  - EXPLAIN
  - SQLADM
  - WLMADM
- Erteilen und Entziehen der folgenden Datenbankberechtigungen:
  - BINDADD
  - CONNECT
  - CREATETAB
  - CREATE_EXTERNAL_ROUTINE
  - CREATE_NOT_FENCED_ROUTINE
  - IMPLICIT_SCHEMA
  - LOAD
  - QUIESCE_CONNECT
- Erteilen und Entziehen aller Zugriffsrechte für die folgenden Objekte, unabhängig davon, von wem die Zugriffsrechte erteilt wurden:
  - Globale Variable
  - Index
  - Kurzname
  - Paket

  - Routine (Prüfroutinen ausgenommen)
  - Schema.
  - Sequenz
  - Server
  - Tabelle
  - Tabellenbereich
  - Sicht
  - XSR-Objekte
- Zugriffsrecht SELECT für die Systemkatalogtabellen und -sichten

Diese Berechtigung umfasst einen Teil der Sicherheitsadministratorberechtigung SECAM.

Datenzugriffsverwaltungsberechtigung (DATAACCESS)

Die Berechtigung DATAACCESS ermöglicht den Zugriff auf Daten einer bestimmten Datenbank.

Die Berechtigung DATAACCESS kann nur vom Sicherheitsadministrator (dem Benutzer mit der Berechtigung SECADM) erteilt werden. Die Berechtigung kann an einen Benutzer, eine Gruppe oder eine Rolle vergeben werden. Die Gruppe PUBLIC kann weder direkt noch indirekt über die Berechtigung DATAACCESS verfügen.

Für alle Tabellen, Sichten, MQTs und Kurznamen ergeben sich folgende Berechtigungen und Zugriffsrechte:

- Berechtigung LOAD für die Datenbank
- Zugriffsrecht SELECT (Systemkatalogtabellen und -sichten eingeschlossen)
- Zugriffsrecht INSERT
- Zugriffsrecht UPDATE
- Zugriffsrecht DELETE

Zusätzlich werden über die Berechtigung DATAACCESS die folgenden Zugriffsrechte bereitgestellt:

- Zugriffsrecht EXECUTE für Pakete
- Zugriffsrecht EXECUTE für alle Routinen (Prüfroutinen ausgenommen)
- Zugriffsrecht EXECUTE für alle Module
- Zugriffsrecht READ für alle globalen Variablen und Zugriffsrecht WRITE für alle globalen Variablen außer Variablen, die schreibgeschützt sind
- Zugriffsrecht USAGE für alle XSR-Objekte
- Zugriffsrecht USAGE für alle Sequenzen

SQL-Verwaltungsberechtigung (SQLADM)

Die Berechtigung SQLADM ist zum Überwachen und Optimieren von SQL-Anweisungen erforderlich.

Die Berechtigung SQLADM kann vom Sicherheitsadministrator (dem Benutzer mit der Berechtigung SECADM) oder einem Benutzer mit der Berechtigung ACCESSCTRL erteilt werden. Die Berechtigung SQLADM kann einem Benutzer, einer Gruppe, einer Rolle oder der Gruppe PUBLIC erteilt werden. Die Berechtigung SQLADM ermöglicht Benutzern das Ausführen der folgenden Funktionen:

- Ausführung der folgenden SQL-Anweisungen:

  - CREATE EVENT MONITOR
  - DROP EVENT MONITOR
  - EXPLAIN
  - FLUSH EVENT MONITOR
  - FLUSH OPTIMIZATION PROFILE CACHE
  - FLUSH PACKAGE CACHE
  - PREPARE
  - REORG INDEXES/TABLE
  - RUNSTATS
  - SET EVENT MONITOR STATE

  Anmerkung: Wenn die Registry-Variable **DB2AUTH** auf`SQLADM_NO_RUNSTATS_REORG` gesetzt ist, können Benutzer mit SQLADM-Berechtigung keine Reorganisations- oder Statistikoperationen ausführen.
- Die Ausführung bestimmter Klauseln der folgenden SQL-Anweisungen des Workload-Managers:
  - Die folgenden Klauseln der Anweisung ALTER SERVICE CLASS:
    - COLLECT AGGREGATE ACTIVITY DATA
    - COLLECT AGGREGATE REQUEST DATA
    - COLLECT REQUEST METRICS
  - Die folgende Klausel der Anweisung ALTER THRESHOLD:
    - WHEN EXCEEDED COLLECT ACTIVITY DATA

    .
  - Die folgenden Klauseln der Anweisung ALTER WORK ACTION SET, mit denen Sie eine Arbeitsaktion ändern können:
    - ALTER WORK ACTION ... COLLECT ACTIVITY DATA
    - ALTER WORK ACTION ... COLLECT AGGREGATE ACTIVITY DATA
    - ALTER WORK ACTION ... WHEN EXCEEDED COLLECT ACTIVITY DATA
  - Die folgenden Klauseln der Anweisung ALTER WORKLOAD:
    - COLLECT ACTIVITY METRICS
    - COLLECT AGGREGATE ACTIVITY DATA
    - COLLECT LOCK TIMEOUT DATA
    - COLLECT LOCK WAIT DATA
    - COLLECT UNIT OF WORK DATA
- Zugriffsrecht SELECT für die Systemkatalogtabellen und -sichten
- Zugriffsrecht EXECUTE für alle integrierten DB2-Routinen (Prüfroutinen ausgenommen)

Die Berechtigung SQLADM umfasst einen Teil der Datenbankadministratorberechtigung DBADM.

Die Berechtigung EXPLAIN umfasst einen Teil der Berechtigung SQLADM.

Workload-Verwaltungsberechtigung (WLMADM)

Die Berechtigung WLMADM ist für das Verwalten von Objekten des Workload-Managers für eine bestimmte Datenbank erforderlich. Diese Berechtigung ermöglicht Ihnen das Erstellen, Ändern, Löschen von Objekten des Workload-Managers

sowie das Erstellen von Kommentaren zu diesen Objekten und das Erteilen und Entziehen von Berechtigungen für die Objekte.

Die Berechtigung WLMADM kann vom Sicherheitsadministrator (dem Benutzer mit der Berechtigung SECADM) oder einem Benutzer mit der Berechtigung ACCESSCTRL erteilt werden. Die Berechtigung WLMADM kann einem Benutzer, einer Gruppe, einer Rolle oder der Gruppe PUBLIC erteilt werden. Die Berechtigung WLMADM ermöglicht Benutzern das Ausführen der folgenden Operationen:

- Erstellen, Ändern, Kommentieren und Löschen der folgenden Objekte des Workload-Managers:
  - Histogrammschablonen
  - Serviceklassen
  - Schwellenwerte
  - Arbeitsaktionssets
  - Arbeitsklassensets
  - Workloads
- Erteilen und Entziehen der Workloadzugriffsrechte
- Ausführen der integrierten Routinen des Workload-Managements

Die Berechtigung WLMADM umfasst einen Teil der Datenbankadministratorberechtigung DBADM.

Verwaltungsberechtigung EXPLAIN

Die Berechtigung EXPLAIN ist für das Erläutern von Abfrageplänen erforderlich, berechtigt jedoch nicht zum Zugriff auf Daten einer bestimmten Datenbank. Diese Berechtigung umfasst einen Teil der Datenbankadministratorberechtigung und verfügt über kein inhärentes Zugriffsrecht auf Daten, die in Tabellen gespeichert sind.

Die Berechtigung EXPLAIN kann vom Sicherheitsadministrator (dem Benutzer mit der Berechtigung SECADM) oder einem Benutzer mit der Berechtigung ACCESSCTRL erteilt werden. Die Berechtigung EXPLAIN kann einem Benutzer, einer Gruppe, einer Rolle oder der Gruppe PUBLIC erteilt werden. Sie ermöglicht das Ausführen der folgenden SQL-Anweisungen:

- EXPLAIN
- PREPARE
- DESCRIBE für die Ausgabe einer Anweisung SELECT oder einer XQuery-Anweisung

Die Berechtigung EXPLAIN beinhaltet auch die Berechtigung EXECUTE für die integrierten EXPLAIN-Routinen.

Die Berechtigung EXPLAIN umfasst einen Teil der Berechtigung SQLADM.

Berechtigung LOAD

Benutzer, die über die Berechtigung LOAD auf Datenbankebene sowie über das Zugriffsrecht INSERT für eine Tabelle verfügen, können den Befehl **LOAD** verwenden, um Daten in eine Tabelle zu laden.

Anmerkung: Benutzer mit der Berechtigung DATAACCESS verfügen über die volle Berechtigung für die Verwendung des Befehls LOAD.

Benutzer, die über die Berechtigung LOAD auf Datenbankebene sowie über das Zugriffsrecht INSERT auf eine Tabelle verfügen, können den Befehl **LOAD RESTART**

oder **LOAD TERMINATE** verwenden, wenn bei der vorangegangenen Ladeoperation Daten eingefügt (LOAD INSERT) wurden.

Benutzer, die über die Berechtigung LOAD auf Datenbankebene sowie über die Zugriffsrechte INSERT und DELETE für eine Tabelle verfügen, können den Befehl **LOAD REPLACE** verwenden.

Wenn bei der vorangegangenen Ladeoperation Daten ersetzt (LOAD REPLACE) wurden, muss diesem Benutzer auch das Zugriffsrecht DELETE erteilt werden, damit dieser den Befehl **LOAD RESTART** oder **LOAD TERMINATE** verwenden kann.

Wenn die Ausnahmetabellen im Rahmen der LOAD-Operation verwendet werden, muss der Benutzer über das Zugriffsrechte INSERT für die Ausnahmetabellen verfügen.

Der Benutzer mit dieser Berechtigung kann die Befehle **QUIESCE TABLESPACES FOR TABLE**, **RUNSTATS** und **LIST TABLESPACES** ausführen.

Hinweise zur Berechtigung IMPLICIT_SCHEMA

Wenn eine neue Datenbank erstellt wird, erhält die Gruppe PUBLIC die Datenbankberechtigung IMPLICIT_SCHEMA, sofern beim Befehl **CREATE DATABASE** zum Erstellen der Datenbank nicht die Option **RESTRICTIVE** angegeben wurde.

Mit der Berechtigung IMPLICIT_SCHEMA kann jeder Benutzer ein Schema erstellen, indem er ein Objekt erstellt und dabei einen Schemanamen angibt, der noch nicht existiert. SYSIBM wird zum Eigner des implizit erstellten Schemas, und die Gruppe PUBLIC erhält das Zugriffsrecht zur Erstellung von Objekten in diesem Schema.

Wenn für die Datenbank eine Kontrolle darüber erforderlich ist, wer implizit Schemaobjekte erstellen kann, sollte die Berechtigung IMPLICIT_SCHEMA der Gruppe PUBLIC entzogen werden. Wenn dies geschehen ist, gibt es nur drei Möglichkeiten zur Erstellung eines Schemenobjekts:

- Jeder Benutzer kann ein Schema erstellen, indem er seinen eigenen Berechtigungsnamen in der Anweisung CREATE SCHEMA verwendet.
- Jeder Benutzer mit der Berechtigung DBADM kann ein beliebiges, noch nicht vorhandenes Schema explizit erstellen und optional einen anderen Benutzer als Eigner des Schemas angeben.
- Jeder Benutzer mit der Berechtigung DBADM hat die Berechtigung IMPLICIT_SCHEMA, sodass er implizit ein Schema mit einem beliebigen Namen bei der Erstellung anderer Datenbankobjekte erstellen kann. SYSIBM wird zum Eigner des implizit erstellten Schemas, und die Gruppe PUBLIC erhält das Zugriffsrecht zur Erstellung von Objekten in dem Schema.

Zugriffsrechte

Zugriffsrechte der Berechtigungs-IDs: SETSESSIONUSER

Zugriffsrechte der Berechtigungs-IDs beinhalten Aktionen, die an Berechtigungs-IDs ausgeführt werden können. Gegenwärtig ist nur ein solches Zugriffsrecht vorhanden: SETSESSIONUSER.

Das Zugriffsrecht SETSESSIONUSER kann einem Benutzer oder einer Gruppe erteilt werden und gibt seinem Besitzer die Möglichkeit, seine Identität in eine beliebige der Berechtigungs-IDs zu wechseln, für die das Zugriffsrecht erteilt wurde. Der Identitätswechsel wird mithilfe der SQL-Anweisung SET SESSION AUTHORIZATION durchgeführt. Das Zugriffsrecht SETSESSIONUSER kann nur einem Benutzer mit der Berechtigung SECADM erteilt werden.

Anmerkung: Wenn Sie eine Datenbank der Version 8 auf Version 9.1 (oder eine neuere Version) aufrüsten, wird den Berechtigungs-IDs mit expliziter DBADM-Berechtigung für diese Datenbank automatisch das Zugriffsrecht SETSESSIONUSER für die Gruppe PUBLIC erteilt. Dadurch wird verhindert, dass Anwendungen fehlschlagen, die davon abhängig sind, dass Berechtigungs-IDs mit DBADM-Berechtigung die Sitzungsberechtigungs-ID auf eine beliebige Berechtigungs-ID setzen zu können. Dies geschieht nicht, wenn die Berechtigungs-ID die Berechtigung SYSADM besitzt, ihr jedoch die Berechtigung DBADM nicht explizit erteilt wurde.

Schemazugriffsrechte

Zugriffsrechte für Schemata gehören zur Kategorie der Zugriffsrechte für Objekte.

Die Zugriffsrechte für Objekte sind in Abb. 3 auf Seite 48 dargestellt.

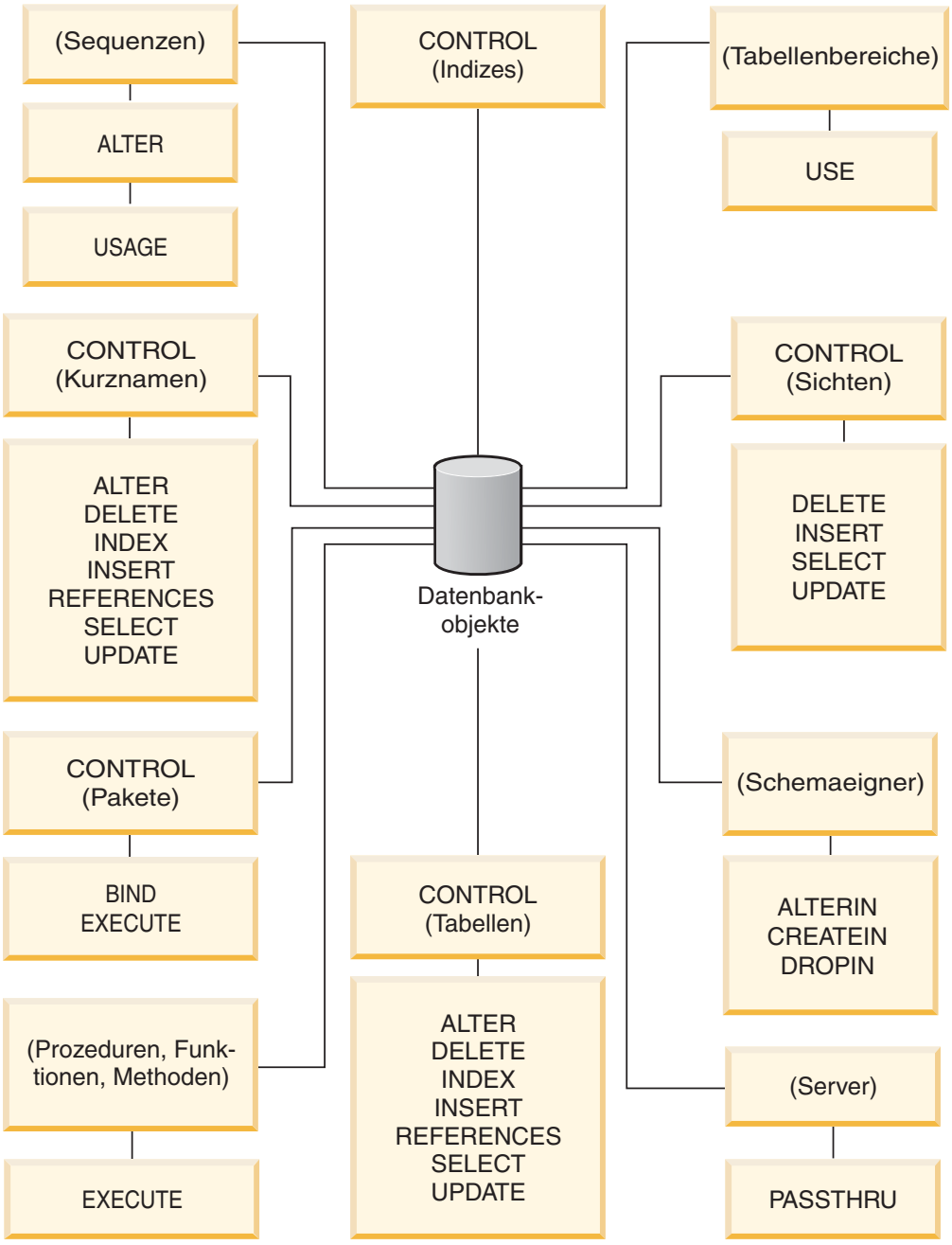


Abbildung 3. Zugriffsrechte für Objekte

Zugriffsrechte für Schemata berechtigen zu Aktionen an Schemata in einer Datenbank. Benutzern, Gruppen, Rollen und der Gruppe PUBLIC kann jedes der folgenden Zugriffsrechte erteilt werden:

- CREATEIN erlaubt dem Benutzer, Objekte innerhalb des Schemas zu erstellen.
- ALTERIN erlaubt dem Benutzer, Objekte innerhalb des Schemas zu ändern.
- DROPIN erlaubt dem Benutzer, Objekte aus dem Schema zu löschen.

Der Eigner des Schemas besitzt alle diese Zugriffsrechte und die Möglichkeit, diese Zugriffsrechte anderen Benutzern zu erteilen. Zu den Objekten, die innerhalb des Schemenobjekts bearbeitet werden, gehören Tabellen, Sichten, Indizes, Pakete, Datentypen, Funktionen, Trigger, Prozeduren und Aliasnamen.

Zugriffsrechte für Tabellenbereiche

Mit den Zugriffsrechten für Tabellenbereiche können Aktionen für die Tabellenbereiche einer Datenbank ausgeführt werden. Wenn einem Benutzer das Zugriffsrecht USE für einen Tabellenbereich zugeordnet wird, ist er zum Erstellen von Tabellen innerhalb des Tabellenbereichs berechtigt.

Der Eigner des Tabellenbereichs erhält das Zugriffsrecht USE. Zusätzlich können Benutzer, die über die Berechtigung SECADM oder ACCESSCTRL verfügen, das Zugriffsrecht USE für den Tabellenbereich erteilen.

Benutzer mit der Berechtigung SYSADM oder SYSCTRL können einen beliebigen Tabellenbereich verwenden.

Standardmäßig wird beim Erstellen der Datenbank das Zugriffsrecht USE für den Tabellenbereich USERSPACE1 der Gruppe PUBLIC erteilt. Dieses Zugriffsrecht kann jedoch auch entzogen werden.

Das Zugriffsrecht USE kann bei SYSCATSPACE oder anderen temporären Systemtabellenbereichen nicht verwendet werden.

Zugriffsrechte für Tabellen und Sichten

Zugriffsrechte für Tabellen und Sichten umfassen Aktionen für Tabellen oder Sichten in einer Datenbank.

Ein Benutzer muss über die Berechtigung CONNECT für die Datenbank verfügen, um eines der folgenden Zugriffsrechte verwenden zu können:

- CONTROL erteilt dem Benutzer alle Zugriffsrechte für eine Tabelle oder Sicht, einschließlich der Erlaubnis, sie zu löschen und einzelne Zugriffsrechte für Tabellen zu erteilen oder zu entziehen. Sie müssen über die Berechtigung ACCESSCTRL oder SECADM verfügen, um das Zugriffsrecht CONTROL erteilen zu können. Der Ersteller einer Tabelle erhält automatisch das Zugriffsrecht CONTROL für die Tabelle. Der Ersteller einer Sicht erhält nur dann automatisch das Zugriffsrecht CONTROL, wenn er über das Zugriffsrecht CONTROL für alle Tabellen, Sichten und Kurznamen verfügt, auf die in der Sichtdefinition verwiesen wird.
- Das Zugriffsrecht ALTER ermöglicht dem Benutzer, eine Tabelle zu modifizieren, zum Beispiel, der Tabelle Spalten oder eine eindeutige Integritätsbedingung hinzuzufügen. Ein Benutzer mit dem Zugriffsrecht ALTER kann außerdem die Anweisung COMMENT ON für eine Tabelle bzw. für Spalten der Tabelle angeben. Informationen zu den möglichen Modifikationen, die an einer Tabelle vorgenommen werden können, finden Sie in den Beschreibungen der Anweisungen ALTER TABLE und COMMENT.
- DELETE erlaubt dem Benutzer das Löschen von Zeilen aus einer Tabelle oder Sicht.
- INDEX erlaubt dem Benutzer das Erstellen eines Index für eine Tabelle. Die Ersteller von Indizes verfügen automatisch über das Zugriffsrecht CONTROL für den Index.
- INSERT erlaubt dem Benutzer das Einfügen einer Zeile in eine Tabelle oder Sicht und das Ausführen des Dienstprogramms **IMPORT**.
- REFERENCES erlaubt dem Benutzer das Erstellen und Löschen eines Fremdschlüssels unter Angabe der Tabelle als die übergeordnete Tabelle in einer Abhängigkeitsbeziehung. Der Benutzer hat dieses Zugriffsrecht möglicherweise nur für bestimmte Spalten.

- SELECT erlaubt dem Benutzer das Abrufen von Zeilen aus einer Tabelle oder Sicht, das Erstellen einer Sicht für eine Tabelle und das Ausführen des Dienstprogramms **EXPORT**.
- UPDATE erlaubt dem Benutzer das Ändern eines Eintrags in einer Tabelle oder Sicht bzw. für eine oder mehrere Spalten in einer Tabelle oder Sicht. Der Benutzer hat dieses Zugriffsrecht eventuell nur für bestimmte Spalten.

Das Zugriffsrecht zum Erteilen dieser Zugriffsrechte an andere Benutzer kann auch mithilfe der Klausel WITH GRANT OPTION in der Anweisung GRANT erteilt werden.

Anmerkung: Wenn einem Benutzer oder einer Gruppe das Zugriffsrecht CONTROL für eine Tabelle erteilt wird, werden alle anderen Zugriffsrechte für diese Tabelle automatisch mit der Klausel WITH GRANT OPTION erteilt. Wenn Sie anschließend das Zugriffsrecht CONTROL eines Benutzers für die Tabelle entziehen, behält der Benutzer weiterhin die anderen Zugriffsrechte, die ihm automatisch erteilt wurden. Wenn Sie alle Zugriffsrechte entziehen wollen, die zusammen mit dem Zugriffsrecht CONTROL erteilt wurden, müssen Sie entweder jedes einzelne Zugriffsrecht explizit entziehen oder das Schlüsselwort ALL in der Anweisung REVOKE angeben wie im folgenden Beispiel:

```
REVOKE ALL
  ON EMPLOYEE FROM USER HERON
```

Bei der Arbeit mit typisierten Tabellen sind folgende Punkte für die Tabellen- und Sichtzugriffsrechte zu berücksichtigen.

Anmerkung: Zugriffsrechte können auf jeder Stufe einer Tabellenhierarchie unabhängig erteilt werden. Daher kann ein Benutzer, dem ein Zugriffsrecht für eine übergeordnete Tabelle innerhalb einer Hierarchie typisierter Tabellen erteilt wurde, indirekt auch beliebige untergeordnete Tabellen beeinflussen. Ein Benutzer kann jedoch nur direkt auf eine untergeordnete Tabelle einwirken, wenn er über das nötige Zugriffsrecht für diese untergeordnete Tabelle verfügt.

Die Abhängigkeiten von über- und untergeordneten Tabellen in einer Tabellenhierarchie bedeuten, dass Operationen wie SELECT, UPDATE und DELETE die Zeilen der Zieltabelle für die Operation und alle ihre untergeordneten Tabellen (sofern vorhanden) beeinflussen. Dieses Verhalten wird als *Substitutionsfähigkeit* bezeichnet. Nehmen Sie zum Beispiel an, Sie haben eine Mitarbeitertabelle des Typs Employee_t erstellt, zu der eine untergeordnete Tabelle des Typs Manager_t gehört. Ein Manager ist eine (besondere) Art von Mitarbeiter, wie durch die Abhängigkeit von Typ und untergeordnetem Typ für die strukturierten Typen Employee_t und Manager_t und durch die entsprechende Abhängigkeit von Tabelle und untergeordneter Tabelle für die Tabellen Employee und Manager angegeben wird. Die SQL-Abfrage

```
SELECT * FROM Employee
```

gibt daher als Ergebnis dieser Abhängigkeit die Objektkennung und die Employee_t-Attribute für Mitarbeiter und Manager zurück. Gleichermaßen definiert die Aktualisierungsoperation

```
UPDATE Employee SET Salary = Salary + 1000
```

eine Gehaltserhöhung von 1000 € sowohl für Manager als auch für reguläre Mitarbeiter.

Ein Benutzer mit dem Zugriffsrecht SELECT für die Tabelle Employee kann diese SELECT-Operation ausführen, selbst wenn er kein explizites Zugriffsrecht SELECT

für die Tabelle Manager hat. Einem Benutzer wird jedoch nicht erlaubt, eine SELECT-Operation direkt für die untergeordnete Tabelle Manager auszuführen. Daher kann er nicht auf die nicht übernommenen Spalten der Tabelle Manager zugreifen.

Gleichermaßen kann ein Benutzer mit dem Zugriffsrecht UPDATE für die Tabelle Employee eine UPDATE-Operation für die Tabelle Manager ausführen, was sich auf gewöhnliche Mitarbeiter und Manager auswirkt, selbst wenn er nicht über das explizite Zugriffsrecht UPDATE für die Tabelle Manager verfügt. Einem Benutzer wird jedoch nicht erlaubt, UPDATE-Operationen direkt für die untergeordnete Tabelle Manager auszuführen, und daher kann er die nicht übernommenen Spalten der Tabelle Manager nicht aktualisieren.

Paketzugriffsrechte

Ein Paket ist ein Datenbankobjekt, das die Informationen enthält, die vom Datenbankmanager zum effizientesten Zugriff auf Daten für ein bestimmtes Anwendungsprogramm benötigt werden. Zugriffsrechte für Pakete erlauben einem Benutzer das Erstellen und Bearbeiten von Paketen.

Der Benutzer muss über die Berechtigung CONNECT für die Datenbank verfügen, um eines der folgenden Zugriffsrechte verwenden zu können:

- CONTROL gibt dem Benutzer die Möglichkeit, ein Paket erneut zu binden, zu löschen oder auszuführen sowie die Möglichkeit, die Zugriffsrechte auf andere Benutzer zu erweitern. Der Ersteller eines Pakets erhält dieses Zugriffsrecht automatisch. Einem Benutzer mit dem Zugriffsrecht CONTROL werden die Zugriffsrechte BIND und EXECUTE erteilt, und er kann auch anderen Benutzern diese Zugriffsrechte mithilfe der Anweisung GRANT erteilen. (Wenn ein Zugriffsrecht mit der Klausel WITH GRANT OPTION erteilt wird, kann ein Benutzer, der das Zugriffsrecht BIND oder EXECUTE erhält, seinerseits dieses Zugriffsrecht anderen Benutzern erteilen.) Um das Zugriffsrecht CONTROL erteilen zu können, muss der Benutzer über die Berechtigung ACCESSCTRL oder SECADM verfügen.
- Das Zugriffsrecht BIND für ein Paket ermöglicht dem Benutzer, dieses Paket zu binden oder erneut zu binden sowie neue Paketversionen des gleichen Paketnamens mit dem gleichen Ersteller hinzuzufügen.
- EXECUTE ermöglicht dem Benutzer, ein Paket auszuführen.

Anmerkung: Alle Paketzugriffsrechte gelten für sämtliche VERSIONen, die den gleichen Paketnamen und den gleichen Ersteller haben.

Zusätzlich zu diesen Zugriffsrechten für Pakete erlaubt die Berechtigung BINDADD für Datenbanken Benutzern das Erstellen neuer Pakete oder das Durchführen eines Rebinds für ein vorhandenes Paket in der Datenbank.

Objekte, auf die über Kurznamen verwiesen wird, müssen Authentifizierungsprüfungen an den Datenquellen durchlaufen, die die Objekte enthalten. Außerdem müssen Paketbenutzer über geeignete Zugriffsrechte oder eine geeignete Berechtigungsstufe für die Datenquellenobjekte an der Datenquelle verfügen.

Für Pakete mit Kurznamen sind möglicherweise weitere Berechtigungsschritte erforderlich, da die DB2-Datenbank dynamische Abfragen verwendet, um mit Datenquellen der DB2-Produktfamilie Daten auszutauschen. Die Berechtigungs-ID, die das Paket an der Datenquelle ausführt, muss mit der entsprechenden Berechtigung ausgestattet sein, um das Paket an dieser Datenquelle dynamisch ausführen zu können.

Indexzugriffsrechte

Der Ersteller eines Index oder einer Indexspezifikation erhält automatisch das Zugriffsrecht CONTROL für den Index. Das Zugriffsrecht CONTROL für einen Index entspricht effektiv der Berechtigung zum Löschen des Index. Zum Erteilen eines Zugriffsrechts CONTROL für einen Index muss ein Benutzer über die Berechtigung ACCESSCTRL oder SECADM verfügen.

Das Zugriffsrecht INDEX auf Tabellenebene ermöglicht einem Benutzer das Erstellen eines Index für diese Tabelle.

Das Zugriffsrecht INDEX auf Kurznamenebene ermöglicht einem Benutzer das Erstellen eines Index für diesen Kurznamen.

Zugriffsrechte für Sequenzen

Der Ersteller einer Sequenz erhält automatisch die Zugriffsrechte USAGE und ALTER für die Sequenz. Das Zugriffsrecht USAGE ist zur Verwendung der Ausdrücke NEXT VALUE und PREVIOUS VALUE für die Sequenz erforderlich.

Um anderen Benutzern die Verwendung der Ausdrücke NEXT VALUE und PREVIOUS VALUE zu erlauben, müssen die Sequenzzugriffsrechte der Gruppe PUBLIC erteilt werden. Dadurch erhalten alle Benutzer die Erlaubnis die Ausdrücke mit der angegebenen Sequenz zu verwenden.

Das Zugriffsrecht ALTER für die Sequenz erlaubt dem Benutzer solche Tasks wie das erneute Starten der Sequenz oder das Ändern des Inkrements für zukünftige Sequenzwerte auszuführen. Der Ersteller der Sequenz kann das Zugriffsrecht ALTER anderen Benutzern erteilen. Wenn dazu die Klausel WITH GRANT OPTION verwendet wird, können diese Benutzer selbst diese Zugriffsrechte wieder anderen Benutzern erteilen.

Zugriffsrechte für Routinen

Zugriffsrechte zum Ausführen (EXECUTE) beinhalten Aktionen an allen Arten von Routinen, wie zum Beispiel Funktionen, Prozeduren und Methoden, innerhalb einer Datenbank. Wenn ein Benutzer das Zugriffsrecht EXECUTE besitzt, kann er die entsprechende Routine aufrufen, eine Quellenfunktion erstellen, die von dieser Routine abgeleitet ist (gilt nur für Funktionen), und auf die Routine in einer beliebigen DDL-Anweisung wie CREATE VIEW oder CREATE TRIGGER verweisen.

Der Benutzer, der die extern gespeicherte Prozedur, Funktion oder Methode definiert, erhält das Zugriffsrecht EXECUTE WITH GRANT. Wenn das Zugriffsrecht EXECUTE einem anderen Benutzer über die Klausel WITH GRANT OPTION erteilt wird, kann dieser Benutzer selbst das Zugriffsrecht EXECUTE wieder einem anderen Benutzer erteilen.

Zugriffsrecht USAGE für Auslastungen

Zur Verwendung einer Auslastung (Workload) kann ein Benutzer, der über die Berechtigung ACCESSCTRL, SECADM oder WLMADM verfügt, einem Benutzer, einer Gruppe oder einer Rolle das Zugriffsrecht USAGE für die betreffende Auslastung mithilfe der Anweisung GRANT USAGE ON WORKLOAD erteilen.

Wenn das DB2-Datenbanksystem eine entsprechende Auslastung findet, überprüft es, ob der Sitzungsbenutzer das Zugriffsrecht USAGE für diese Auslastung besitzt. Wenn der Sitzungsbenutzer das Zugriffsrecht USAGE für diese Auslastung nicht besitzt, sucht das DB2-Datenbanksystem nach der nächsten entsprechenden Auslas-

tung in der geordneten Liste. Das heißt, die Auslastungen, für die der Sitzungsbenutzer das Zugriffsrecht USAGE nicht hat, werden so behandelt, als wären sie nicht vorhanden.

Die Informationen zum Zugriffsrecht USAGE werden in den Katalogen gespeichert und können über die Sicht SYSCAT.WORKLOADAUTH angezeigt werden.

Das Zugriffsrecht USAGE kann mithilfe der Anweisung REVOKE USAGE ON WORKLOAD entzogen werden.

Benutzer mit der Berechtigung ACCESSCTRL, DATAACCESS, DBADM, SECADM oder WLMADM verfügen automatisch auch über das Zugriffsrecht USAGE für alle Auslastungen.

Die Auslastung SYSDEFAULTUSERWORKLOAD und das Zugriffsrecht USAGE

Das Zugriffsrecht USAGE für die Auslastung SYSDEFAULTUSERWORKLOAD wird der speziellen Gruppe PUBLIC bei der Erstellung einer Datenbank erteilt, wenn die Datenbank ohne die Option RESTRICT erstellt wird. Anderenfalls muss das Zugriffsrecht USAGE durch einen Benutzer mit der Berechtigung ACCESSCTRL, WLMADM oder SECADM explizit erteilt werden.

Wenn der Sitzungsbenutzer das Zugriffsrecht USAGE für keine der Auslastungen (einschließlich SYSDEFAULTUSERWORKLOAD) besitzt, wird ein SQL-Fehler zurückgegeben.

Die Auslastung SYSDEFAULTADMWORKLOAD und das Zugriffsrecht USAGE

Das Zugriffsrecht USAGE für die Auslastung SYSDEFAULTADMWORKLOAD kann keinem Benutzer explizit erteilt werden. Nur Benutzer, die den Befehl **SET WORKLOAD TO SYSDEFAULTADMWORKLOAD** ausgeben und deren Sitzungsberechtigungs-ID über die Berechtigung ACCESSCTRL, DATAACCESS, DBADM, WLMADM oder SECADM verfügt, sind zum Verwenden der Auslastung berechtigt.

Die Anweisungen GRANT USAGE ON WORKLOAD und REVOKE USAGE ON WORKLOAD haben auf die Auslastung SYSDEFAULTADMWORKLOAD keine Wirkung.

Berechtigungs-IDs in verschiedenen Kontexten

Eine Berechtigungs-ID dient zwei Zwecken: Identifikation und Berechtigungsprüfung. Zum Beispiel wird die Sitzungsberechtigungs-ID für die einleitende Berechtigungsprüfung verwendet.

Wenn auf die Verwendung einer Berechtigungs-ID in einem bestimmten Kontext Bezug genommen wird, ist der Verweis auf die Berechtigung dazu geeignet, den Kontext zu identifizieren, wie im folgenden Abschnitt gezeigt.

Kontextbezogener Verweis auf die Berechtigungs-ID
Definition

Systemberechtigungs-ID
Die Berechtigungs-ID, die für jede einleitende Berechtigungsprüfung verwendet wird, zum Beispiel zum Überprüfen des Zugriffsrechts CONNECT bei der CONNECT-Verarbeitung. Als Teil des Authentifizierungsprozesses

während der CONNECT-Verarbeitung wird eine Berechtigungs-ID generiert, die mit den DB2-Namenskonventionen kompatibel ist und die externe Benutzer-ID innerhalb des DB2-Datenbanksystems repräsentiert. Die Systemberechtigungs-ID stellt den Benutzer dar, der die Verbindung hergestellt hat. Mithilfe des Sonderregisters SYSTEM_USER können Sie den aktuellen Wert der Systemberechtigungs-ID ermitteln. Die Systemberechtigungs-ID kann für eine Verbindung nicht geändert werden.

Sitzungsberechtigungs-ID
Die Berechtigungs-ID, die für jede Sitzungsberechtigungsprüfung nach den einleitenden Überprüfungen der CONNECT-Verarbeitung verwendet wird. Der Standardwert der Sitzungsberechtigungs-ID ist der Wert der Systemberechtigungs-ID. Mithilfe des Sonderregisters SESSION_USER können Sie den aktuellen Wert der Sitzungsberechtigungs-ID feststellen. Das Sonderregister USER ist ein Synonym für das Sonderregister SESSION_USER. Die Sitzungsberechtigungs-ID kann mithilfe der Anweisung SET SESSION AUTHORIZATION geändert werden.

Paketberechtigungs-ID
Die Berechtigungs-ID, die zum Binden (BIND) eines Pakets an die Datenbank verwendet wird. Die Berechtigungs-ID ergibt sich aus dem Wert für die Option **OWNER** *berechtigungs-id* des Befehls **BIND**. Die Paketberechtigungs-ID wird manchmal als Paketbinder oder Paketeigner bezeichnet.

Routineneignerberechtigungs-ID
Die Berechtigungs-ID, die in den Systemkatalogen als Eigner der SQL-Routine eingetragen ist, die aufgerufen wurde.

Routinenaufruferberechtigungs-ID
Die Berechtigungs-ID, die die Anweisungsberechtigungs-ID für die Anweisung ist, die eine SQL-Routine aufgerufen hat.

Anweisungsberechtigungs-ID
Die Berechtigungs-ID, die einer bestimmten SQL-Anweisung zugeordnet ist und die für alle Berechtigungsanforderungen sowie zur Bestimmung des Objekteigentumsrechts (sofern zutreffend) zu verwenden ist. Ihr Wert leitet sich abhängig vom Typ der SQL-Anweisung aus der entsprechenden Quellenberechtigungs-ID ab:

- Statisches SQL

  Die Paketberechtigungs-ID wird verwendet.
- Dynamisches SQL (aus Nichtroutinenkontext)

  Die Tabelle zeigt, welche Berechtigungs-ID in den einzelnen Fällen verwendet wird:

Wert der Option DYNAMICRULES zum Ausführen des Pakets	**Verwendete Berechtigungs-ID**
RUN	Sitzungsberechtigungs-ID
BIND	Paketberechtigungs-ID
DEFINERUN, INVOKERUN	Sitzungsberechtigungs-ID
DEFINEBIND, INVOKEBIND	Paketberechtigungs-ID

- Dynamisches SQL (aus Routinenkontext)

  Die Tabelle zeigt, welche Berechtigungs-ID in den einzelnen Fällen verwendet wird:

Wert der Option DYNAMICRULES zum Ausführen des Pakets	Verwendete Berechtigungs-ID
DEFINERUN, DEFINEBIND	Routineneignerberechtigungs-ID
INVOKERUN, INVOKEBIND	Routinenaufruferberechtigungs-ID

Mithilfe des Sonderregisters CURRENT_USER können Sie den aktuellen Wert der Anweisungsberechtigungs-ID ermitteln. Die Anweisungsberechtigungs-ID kann nicht direkt geändert werden. Sie wird vom DB2-Datenbanksystem abhängig vom Typ der jeweiligen SQL-Anweisung automatisch geändert.

Beim Erstellen einer Datenbank erteilte Standardzugriffsrechte

Wenn Sie eine Datenbank erstellen, werden Ihnen auf die Datenbankebene bezogene Standardberechtigungen und auf die Objektebene bezogene Standardzugriffsrechte für die Datenbank erteilt.

Es werden die folgenden Berechtigungen und Zugriffsrechte erteilt (geordnet nach den Systemkatalogsichten, in denen sie aufgezeichnet werden):

1. SYSCAT.DBAUTH
   - Dem Datenbankersteller werden die folgenden Berechtigungen erteilt:
     - ACCESSCTRL
     - DATAACCESS
     - DBADM
     - SECADM
   - Bei einer nicht-restriktiven Datenbank werden der speziellen Gruppe PUBLIC die folgenden Berechtigungen erteilt:
     - CREATETAB
     - BINDADD
     - CONNECT
     - IMPLICIT_SCHEMA
2. SYSCAT.TABAUTH

   Bei einer nicht-restriktiven Datenbank werden der speziellen Gruppe PUBLIC die folgenden Zugriffsrechte erteilt:
   - SELECT für alle SYSCAT- und SYSIBM-Tabellen
   - SELECT und UPDATE für alle SYSSTAT-Tabellen
   - SELECT für die folgenden Sichten im Schema SYSIBMADM:
     - ALL_*
     - USER_*
     - ROLE_*
     - SESSION_*
     - DICTIONARY
     - TAB
3. SYSCAT.ROUTINEAUTH

   Bei einer nicht-restriktiven Datenbank werden der speziellen Gruppe PUBLIC die folgenden Zugriffsrechte erteilt:
   - EXECUTE with GRANT für alle Prozeduren im Schema SQLJ
   - EXECUTE with GRANT für alle Funktionen und Prozeduren im Schema SYSFUN

- EXECUTE with GRANT für alle Funktionen und Prozeduren im Schema SYSPROC (Prüfroutinen ausgenommen)
- EXECUTE für alle Tabellenfunktionen im Schema SYSIBM
- EXECUTE für alle anderen Prozeduren im Schema SYSIBM
- EXECUTE für die folgenden Module im Schema SYSIBMADM:
  - DBMS_JOB
  - DBMS_LOB
  - DBMS_OUTPUT
  - DBMS_SQL
  - DBMS_UTILITY

4. SYSCAT.PACKAGEAUTH
   - Dem Datenbankersteller werden die folgenden Zugriffsrechte erteilt:
     - CONTROL für alle im Schema NULLID erstellten Pakete
     - BIND with GRANT für alle im Schema NULLID erstellten Pakete
     - EXECUTE with GRANT für alle im Schema NULLID erstellten Pakete
   - Bei einer nicht-restriktiven Datenbank werden der speziellen Gruppe PUBLIC die folgenden Zugriffsrechte erteilt:
     - BIND für alle im Schema NULLID erstellten Pakete
     - EXECUTE für alle im Schema NULLID erstellten Pakete
5. SYSCAT.SCHEMAAUTH

   Bei einer nicht-restriktiven Datenbank werden der speziellen Gruppe PUBLIC die folgenden Zugriffsrechte erteilt:
   - CREATEIN im Schema SQLJ
   - CREATEIN im Schema NULLID
6. SYSCAT.TBSPACEAUTH

   Bei einer nicht-restriktiven Datenbank erhält die spezielle Gruppe PUBLIC das Zugriffsrecht USE für die Tabelle USERSPACE1.
7. SYSCAT.WORKLOADAUTH

   Bei einer nicht-restriktiven Datenbank erhält die spezielle Gruppe PUBLIC das Zugriffsrecht USAGE für SYSDEFAULTUSERWORKLOAD.
8. SYSCAT.VARIABLEAUTH

   Bei einer nicht-restriktiven Datenbank erhält die spezielle Gruppe PUBLIC das Zugriffsrecht READ für globale Schemavariablen im Schema SYSIBM mit Ausnahme der folgenden Variablen:
   - SYSIBM.CLIENT_ORIGUSERID
   - SYSIBM.CLIENT_USRSECTOKEN

Bei einer nicht-restriktiven Datenbank handelt es sich um eine Datenbank, für deren Erstellung der Befehl CREATE DATABASE ohne die Option RESTRICTIVE verwendet wurde.

Erteilen und Entziehen von Zugriffsmöglichkeiten

Erteilen von Zugriffsrechten

Damit Sie Zugriffsrechte für die meisten Datenbankobjekte erteilen können, müssen Sie über die Berechtigung ACCESSCTRL, die Berechtigung SECADM oder das Zugriffsrecht CONTROL für das betreffende Objekt verfügen. Oder Sie müssen das jeweilige Zugriffsrecht über die Klausel WITH GRANT OPTION besitzen. Darüber

hinaus können Benutzer mit der Berechtigung SYSADM oder SYSCTRL Zugriffsrechte für Tabellenbereiche erteilen. Zugriffsrechte können von Ihnen nur für existierende Objekte erteilt werden.

Informationen zu diesem Vorgang

Damit Sie das Zugriffsrecht CONTROL einem anderen Benutzer erteilen können, müssen Sie über die Berechtigung ACCESSCTRL oder SECADM verfügen. Zum Erteilen der Berechtigung ACCESSCTRL, DATAACCESS, DBADM oder SECADM ist die Berechtigung SECADM erforderlich.

Mit der Anweisung GRANT kann ein berechtigter Benutzer Zugriffsrechte erteilen. Ein Zugriffsrecht kann einem oder mehreren Berechtigungsnamen in einer Anweisung erteilt werden. Das Zugriffsrecht kann auch der Gruppe PUBLIC erteilt werden, wodurch es allen Benutzern zur Verfügung gestellt wird. Bei einem Berechtigungsnamen kann es sich um einen Einzelbenutzer oder um eine Gruppe handeln.

Bei Betriebssystemen, unter denen Benutzer denselben Namen wie Gruppen haben können, sollten Sie angeben, ob Sie das Zugriffsrecht dem Benutzer oder der Gruppe erteilen. Die Anweisungen GRANT und REVOKE unterstützen beide die Schlüsselwörter USER, GROUP und ROLE. Wenn diese optionalen Schlüsselwörter nicht verwendet werden, überprüft der Datenbankmanager die Sicherheitseinrichtung des Betriebssystems, um zu ermitteln, ob der Berechtigungsname einen Benutzer oder eine Gruppe bezeichnet. Es wird ebenfalls überprüft, ob eine Berechtigungs-ID des Rollentyps mit demselben Namen vorhanden ist. Kann vom Datenbankmanager nicht ermittelt werden, ob der Berechtigungsname sich auf einen Benutzer, eine Gruppe oder eine Rolle bezieht, wird ein Fehler zurückgegeben. Im folgenden Beispiel wird das Zugriffsrecht SELECT für die Tabelle EMPLOYEE dem Benutzer HERON erteilt:

```
GRANT SELECT
       ON EMPLOYEE TO USER HERON
```

Im folgenden Beispiel wird das Zugriffsrecht SELECT für die Tabelle EMPLOYEE der Gruppe HERON erteilt:

```
GRANT SELECT
        ON EMPLOYEE TO GROUP HERON
```

Widerrufen von Zugriffsrechten

Die Anweisung REVOKE erlaubt es berechtigten Benutzern, Zugriffsrechte zu entziehen, die anderen Benutzern zuvor erteilt wurden.

Informationen zu diesem Vorgang

Zum Entziehen von Zugriffsrechten für Datenbankobjekte müssen Sie über die Berechtigung ACCESSCTRL, die Berechtigung SECADM oder das Zugriffsrecht CONTROL für das betreffende Objekt verfügen. Berechtigungen für Tabellenbereiche können ebenfalls von Benutzern mit der Berechtigung SYSADM oder SYSCTRL entzogen werden. Beachten Sie, dass es nicht ausreicht, ein Zugriffsrecht mit der Klausel WITH GRANT OPTION zu besitzen, um es entziehen zu können. Zum Entziehen des Zugriffsrechts CONTROL eines anderen Benutzers müssen Sie die Berechtigung ACCESSCTRL oder SECADM haben. Zum Entziehen der Berechtigung ACCESSCTRL, DATAACCESS, DBADM oder SECADM ist die Berechtigung SECADM erforderlich. Berechtigungen für Tabellenbereiche können nur von einem Benutzer mit der Berechtigung SYSADM oder SYSCTRL entzogen werden. Zugriffsrechte können nur für existierende Objekte entzogen werden.

Anmerkung: Ein Benutzer ohne die Berechtigung ACCESSCTRL oder SECADM und ohne das Zugriffsrecht CONTROL kann kein Zugriffsrecht entziehen, das er durch Verwendung der Klausel WITH GRANT OPTION erteilt hat. Auch gibt es kein benutzerübergreifendes Entziehen für die Benutzer, die Zugriffsrechte besitzen, die ihnen von der Person, deren Zugriffsrechte entzogen werden, erteilt wurden.
Wenn ein explizit erteiltes Zugriffsrecht auf eine Tabelle (oder eine Sicht) für einen Benutzer mit der Berechtigung DBADM entzogen wird, werden die Zugriffsrechte für andere Sichten, die für diese Tabelle definiert sind, nicht entziehen. Der Grund hierfür ist, dass die Zugriffsrechte für die Sichten auf der Berechtigung DBADM basieren und nicht von expliziten Zugriffsrechten auf die entsprechenden Tabellen abhängig sind.

Wenn ein Zugriffsrecht einem Benutzer, einer Gruppe und einer Rolle mit demselben Namen erteilt wurde, müssen Sie beim Entziehen des Zugriffsrechts das Schlüsselwort GROUP, USER oder ROLE angeben. Im folgenden Beispiel wird das Zugriffsrecht SELECT für die Tabelle EMPLOYEE des Benutzers HERON entzogen:

```
REVOKE SELECT
   ON EMPLOYEE FROM USER HERON
```

Im folgenden Beispiel wird das Zugriffsrecht SELECT für die Tabelle EMPLOYEE der Gruppe HERON entzogen:

```
REVOKE SELECT
   ON EMPLOYEE FROM GROUP HERON
```

Zu beachten ist, dass beim Entziehen eines Zugriffsrechts für eine Gruppe nicht notwendigerweise das Zugriffsrecht jedes einzelnen Angehörigen dieser Gruppe entzogen wird. Wenn einem einzelnen Namen ein Zugriffsrecht direkt erteilt wurde, bleibt dieses Zugriffsrecht erhalten, bis es direkt entzogen wird.

Wenn ein Zugriffsrecht auf eine Tabelle für einen Benutzer entzogen wird, werden auch Zugriffsrechte für alle Sichten entzogen, die von dem betreffenden Benutzer erstellt wurden und die von dem widerrufenen Zugriffsrecht auf die Tabelle abhängen. Es werden jedoch nur die vom System implizit erteilten Zugriffsrechte entzogen. Wenn ein Zugriffsrecht für die Sicht direkt von einem anderen Benutzer erteilt wurde, bleibt dieses erhalten.

Wenn ein Zugriffsrecht auf eine Tabelle für einen Benutzer entzogen wird, werden auch Zugriffsrechte für alle Sichten widerrufen, die von dem betreffenden Benutzer erstellt wurden und die von dem widerrufenen Zugriffsrecht auf die Tabelle abhängen. Es werden jedoch nur die vom System implizit erteilten Zugriffsrechte entzogen. Wenn ein Zugriffsrecht für die Sicht direkt von einem anderen Benutzer erteilt wurde, bleibt dieses erhalten.

Es kann zu Situationen kommen, in denen Sie einer Gruppe ein Zugriffsrecht erteilen (GRANT) und nur für ein Mitglied dieser Gruppe entziehen (REVOKE) möchten. Es gibt nur wenige Möglichkeiten, dies zu tun, ohne die Fehlernachricht SQL0556N zu erhalten:

- Sie können das Mitglied aus der Gruppe entfernen oder eine neue Gruppe mit weniger Mitgliedern erstellen und dieser neuen Gruppe das Zugriffsrecht erteilen (GRANT).
- Sie können das Zugriffsrecht für die Gruppe entziehen (REVOKE) und es anschließend einzelnen Benutzern (Berechtigungs-IDs) erteilen (GRANT).

Anmerkung: Wenn einem Benutzer das Zugriffsrecht CONTROL für eine Tabelle oder Sicht entzogen wird, behält der Benutzer die Möglichkeit, Zugriffsrechte anderen Benutzern zu erteilen. Durch Erhalt des Zugriffsrechts CONTROL empfängt der Benutzer auch alle anderen Zugriffsrechte mit der Klausel WITH GRANT OPTION. Wird das Zugriffsrecht CONTROL entzogen, bleiben alle anderen Zugriffsrechte mit der Klausel WITH GRANT OPTION erhalten, bis sie explizit entzogen werden.

Alle Pakete, die von entzogenen Zugriffsrechten abhängig sind, werden als ungültig markiert, können jedoch wieder zu gültigen Paketen werden, wenn ein Benutzer mit der entsprechenden Berechtigung einen Rebind für sie durchführt. Pakete können auch wiederhergestellt werden, wenn die Zugriffsrechte später dem Benutzer, der die Anwendung gebunden hat, wieder erteilt werden. Durch Ausführen der Anwendung wird ein erfolgreicher impliziter Rebind ausgelöst. Wenn Zugriffsrechte der Gruppe PUBLIC entzogen werden, werden alle Pakete ungültig gemacht, die von Benutzern gebunden wurden, die dies nur aufgrund der Zugriffsrechte der Gruppe PUBLIC tun konnten. Wenn die Berechtigung DBADM einem Benutzer entzogen wird, werden alle von diesem Benutzer gebundenen Pakete ungültig gemacht, einschließlich der Pakete, die Dienstprogrammen für Datenbanken zugeordnet sind. Wenn versucht wird, ein Paket zu verwenden, das als ungültig markiert wurde, versucht das System, einen Rebind für dieses Paket auszuführen. Wenn dieser Rebind fehlschlägt, tritt ein Fehler auf (SQLCODE -727). In diesem Fall müssen die Pakete von einem Benutzer explizit erneut gebunden werden, der über Folgendes verfügt:

- Die Berechtigung für den Rebind der Pakete
- Entsprechende Berechtigungen für die Objekte innerhalb der Pakete

Für diese Pakete sollte zu dem Zeitpunkt, an dem die Zugriffsrechte widerrufen werden, ein Rebind durchgeführt werden.

Wenn Sie einen Trigger oder eine SQL-Funktion in Abhängigkeit von einem oder mehreren Zugriffsrechten definieren und Sie eines oder mehrere dieser Zugriffsrechte verlieren, kann der Trigger bzw. die SQL-Funktion nicht verwendet werden.

Verwalten impliziter Berechtigungen durch Erstellen und Löschen von Objekten

Der Datenbankmanager erteilt einem Benutzer, der ein Datenbankobjekt, wie zum Beispiel eine Tabelle oder ein Paket, erstellt, implizit bestimmte Zugriffsrechte. Es werden auch Zugriffsrechte erteilt, wenn Objekte von Benutzern mit der Berechtigung DBADM erstellt werden. In ähnlicher Weise werden Zugriffsrechte entfernt, wenn ein Objekt gelöscht wird.

Informationen zu diesem Vorgang

Wenn das erstellte Objekt eine Tabelle, ein Kurzname, ein Index oder ein Paket ist, erhält der Benutzer das Zugriffsrecht CONTROL für das Objekt. Wenn das Objekt eine Sicht ist, wird das Zugriffsrecht CONTROL für die Sicht nur dann implizit erteilt, wenn der Benutzer über das Zugriffsrecht CONTROL für alle Tabellen, Sichten und Kurznamen verfügt, auf die in der Sichtdefinition verwiesen wird.

Wenn das explizit erstellte Objekt ein Schema ist, erhält der Schemaeigner die Zugriffsrechte ALTERIN, CREATEIN und DROPIN mit der Klausel WITH GRANT OPTION. Für ein implizit erstelltes Schema wird das Zugriffsrecht CREATEIN der Gruppe PUBLIC erteilt.

Einrichten des Eigentumsrechts für ein Paket

Mit den Befehlen **BIND** und **PRECOMPILE** kann ein Anwendungspaket erstellt oder geändert werden. Für jeden der Befehle können Sie die Option **OWNER** angeben, um den Eigner des generierten Pakets zu benennen.

Informationen zu diesem Vorgang

Für die Benennung des Paketeigners gibt es die folgenden einfachen Regeln:

- Jeder Benutzer kann sich selbst als Eigner benennen. Dies ist die Standardeinstellung, falls die Option **OWNER** nicht angegeben wird.
- Eine Benutzer-ID mit der Berechtigung DBADM darf eine beliebige Berechtigungs-ID mit der Option **OWNER** als Eigner benennen.

Nicht alle Betriebssysteme, die mit DB2-Datenbankprodukten ein Paket binden können, unterstützen die Option **OWNER**.

Implizite Zugriffsrechte durch ein Paket

Der Zugriff auf Daten in einer Datenbank kann durch Anwendungsprogramme sowie durch Personen, die an einer interaktiven Workstation-Sitzung teilnehmen, angefordert werden. Ein Paket enthält Anweisungen, die es Benutzern ermöglichen, eine Vielzahl von Aktionen für viele Datenbankobjekte auszuführen. Für jede dieser Aktionen ist mindestens ein Zugriffsrecht erforderlich.

Zugriffsrechte, die Einzelbenutzern, die das Paket binden, und der Gruppe PUBLIC erteilt werden, sowie die Rollen, die den Einzelbenutzern und der Gruppe PUBLIC erteilt werden, werden beim Binden von statischen SQL- und XQuery-Anweisungen zur Berechtigungsprüfung verwendet. Zugriffsrechte, die über Gruppen erteilt wurden, und die Rollen, die Gruppen erteilt wurden, werden beim Binden von statischen SQL- und XQuery-Anweisungen nicht zur Berechtigungsprüfung verwendet.

Wird beim Binden des Pakets nicht die Option VALIDATE RUN angegeben, muss für den Benutzer mit einer gültigen Berechtigungs-ID, der das Paket bindet, eine der folgenden Aussagen zutreffen:

- Der Benutzer verfügt über alle Berechtigungen, die zum Ausführen der statischen SQL- und Xquery-Anweisungen im Paket erforderlich sind.
- Der Benutzer hat die erforderlichen Zugriffsrechte über mindestens eine der folgenden Gruppen oder Rollen erworben:
  - PUBLIC
  - PUBLIC erteilte Rollen
  - Dem Benutzer erteilte Rollen

Wurde bei der Ausführung von BIND die Option VALIDATE RUN angegeben, schlägt BIND beim Auftreten von Berechtigungsfehlern für statische SQL- oder XQuery-Anweisungen innerhalb des Pakets nicht fehl. Die entsprechenden SQL- oder XQuery-Anweisungen werden dann während der Laufzeit nochmals geprüft. Alle Zugriffsrechte über PUBLIC sowie Gruppen-, Rollen- und Einzelbenutzerzugriffsrechte werden bei der Überprüfung verwendet, um sicherzustellen, dass der Benutzer die nötige Berechtigung (Zugriffsrecht BIND oder BINDADD) zum Binden des Pakets besitzt.

Pakete können sowohl statische als auch dynamische SQL- und XQuery-Anweisungen umfassen. Zum Verarbeiten eines Pakets mit statischen Abfragen muss ein Benutzer lediglich über das Zugriffsrecht EXECUTE für das Paket verfügen. Der Be-

nutzer kann in diesem Fall implizit die Zugriffsrechte des Paketbinders für statische Abfragen im Paket erhalten. Dies ist allerdings nur im Rahmen der durch das Paket festgelegten Einschränkungen möglich.

Wenn das Paket dynamische SQL- oder XQuery-Anweisungen enthält, hängen die erforderlichen Zugriffsrechte von dem Wert ab, der beim Vorkompilieren (PRECOMPILE) oder Binden (BIND) des Pakets für **DYNAMICRULES** angegeben wurde. Weitere Informationen finden Sie im Abschnitt zu den Auswirkungen von **DYNAMICRULES** auf dynamische Abfragen.

Indirekte Zugriffsrechte durch ein Paket mit Kurznamen

Wenn ein Paket Verweise auf Kurznamen enthält, ist die Berechtigungsverarbeitung für Ersteller und Benutzer von Paketen etwas komplexer.

Wenn ein Paketersteller Pakete, die Kurznamen enthalten, erfolgreich bindet, muss der Paketersteller keine Authentifizierung oder Überprüfung von Zugriffsrechten für Tabellen und Sichten durchlaufen, auf die an der Datenquelle mit den Kurznamen verwiesen wird. Wer jedoch das Paket ausführt, muss eine Authentifizierung und eine Berechtigungsprüfung an den Datenquellen durchlaufen.

Nehmen Sie z. B. an, dass eine Datei mit der Endung .SQC eines Paketerstellers mehrere SQL- oder XQuery-Anweisungen enthält. Eine statische Anweisung verweist auf eine lokale Tabelle. Eine weitere dynamische Anweisung verweist auf einen Kurznamen. Wenn das Paket gebunden wird, wird die Berechtigungs-ID des Paketerstellers verwendet, um Zugriffsrechte für die lokale Tabelle und den Kurznamen zu überprüfen. Allerdings findet für die Datenquellenobjekte, die durch den Kurznamen angegeben werden, keine Überprüfung statt. Wenn ein anderer Benutzer das Paket ausführt und das Zugriffsrecht EXECUTE für dieses Paket gegeben ist, muss dieser Benutzer für die Anweisung, die auf die Tabelle verweist, keinerlei weitere Überprüfung seiner Zugriffsrechte durchlaufen. Allerdings muss der Benutzer, der das Paket ausführt, für die Anweisung mit dem Kurznamen eine Authentifizierung und eine Überprüfung der Zugriffsrechte an der Datenquelle durchlaufen.

Wenn die Datei mit der Endung .SQC nur dynamische SQL- und XQuery-Anweisungen und eine Kombination aus Tabellen- und Kurznamenverweisen enthält, ist die Berechtigungsprüfung durch die DB2-Datenbank für lokale Objekte und Kurznamen ähnlich. Paketbenutzer müssen die Überprüfung der Zugriffsrechte für beliebige lokale Objekte (Tabellen, Sichten), die sich in der Anweisung befinden, durchlaufen. Außerdem werden die Zugriffsrechte für Kurznamenobjekte überprüft. (Paketbenutzer müssen eine Authentifizierung und eine Überprüfung der Zugriffsrechte an der Datenquelle durchlaufen, die die durch die Kurznamen angegebenen Objekte enthält.) In beiden Fällen müssen die Benutzer des Pakets über das Zugriffsrecht EXECUTE verfügen.

Die Berechtigungs-ID und das Kennwort des Benutzers, der das Paket ausführt, werden für die gesamte Verarbeitung der Authentifizierungen und Zugriffsrechte verwendet. Diese Daten können geändert werden, indem eine Benutzerzuordnung erstellt wird.

Anmerkung: In statischen SQL- und XQuery-Anweisungen dürfen Kurznamen nicht angegeben werden. Verwenden Sie mit Paketen, die Kurznamen enthalten, nicht die Option **DYNAMICRULES** (auf BIND gesetzt).

Für Pakete mit Kurznamen sind möglicherweise weitere Berechtigungsschritte erforderlich, da die DB2-Datenbank dynamisches SQL verwendet, um mit Datenquel-

len der DB2-Produktfamilie Daten auszutauschen. Die Berechtigungs-ID, die das Paket an der Datenquelle ausführt, muss mit der entsprechenden Berechtigung ausgestattet sein, um das Paket an dieser Datenquelle dynamisch ausführen zu können.

Steuern des Zugriffs auf Daten mit Sichten

Eine Sicht bietet eine Möglichkeit, den Zugriff auf eine Tabelle zu beschränken bzw. die Zugriffsrechte für eine Tabelle zu erweitern.

Die Verwendung einer Sicht bietet die folgenden Steuerungsmöglichkeiten für den Zugriff auf eine Tabelle:

- Es wird der Zugriff nur auf bestimmte Spalten der Tabelle ermöglicht.

  Für Benutzer und Anwendungsprogramme, die nur einen Zugriff auf bestimmte Spalten einer Tabelle benötigen, kann ein berechtigter Benutzer eine Sicht erstellen, um den Zugriff auf die benötigten Spalten zu beschränken.
- Es wird der Zugriff nur auf eine Teilmenge der Tabellenzeilen ermöglicht.

  Durch Angabe einer Klausel WHERE in der Unterabfrage einer Sichtdefinition kann ein berechtigter Benutzer die Zeilen beschränken, auf die über eine Sicht zugegriffen wird.
- Es wird der Zugriff nur auf eine Teilmenge der Tabellenzeilen oder -spalten an der Datenquelle ermöglicht. Wenn Sie durch Kurznamen auf Datenquellen zugreifen, können Sie lokale DB2-Datenbanksichten erstellen, die auf Kurznamen verweisen. Diese Sichten können auf Kurznamen aus einer oder mehreren Datenquellen verweisen.

  Anmerkung: Da Sie eine Sicht erstellen können, die Verweise auf Kurznamen aus mehreren Datenquellen enthalten kann, können die Benutzer von einer Sicht aus auf Daten mehrerer Datenquellen zugreifen. Diese Sichten sind *Sichten mehrerer Speicherpositionen*. Solche Sichten sind nützlich, wenn Sie Daten von Spalten mit sensiblen Tabellendaten in einer verteilten Umgebung verknüpfen oder wenn einzelne Benutzer für spezifische Objekte nicht über die erforderlichen Zugriffsrechte auf Datenquellen verfügen.

Zum Erstellen einer Sicht muss ein Benutzer über die Berechtigung DATAACCESS oder das Zugriffsrecht CONTROL oder SELECT für alle Tabellen, Sichten und Kurznamen verfügen, auf die in der Sichtdefinition verwiesen wird. Der Benutzer muss darüber hinaus in der Lage sein, ein Objekt in dem für die Sicht angegebenen Schema zu erstellen. Das heißt, er muss über die Berechtigung DBADM und das Zugriffsrecht CREATEIN für ein vorhandenes Schema oder die Berechtigung IMPLICIT_SCHEMA für die Datenbank verfügen, wenn das Schema noch nicht existiert.

Wenn Sie Sichten mit Verweisen auf Kurznamen erstellen, müssen Sie über keine weitere Berechtigung auf die Datenquellen (Tabellen oder Sichten) verfügen, auf die mit den Kurznamen in der Sicht verwiesen wird. Allerdings müssen die Benutzer der Sicht die Berechtigung SELECT oder eine gleichwertige Berechtigungsstufe für die zugrunde liegenden Datenquellenobjekte haben, wenn sie auf die Sicht zugreifen.

Wenn die Benutzer an der Datenquelle nicht die entsprechende Berechtigung für die zugrunde liegenden Objekte (Tabellen und Sichten) haben, können Sie wie folgt vorgehen:

1. Erstellen Sie eine Sicht auf die Datenquelle für diejenigen Spalten in der Tabelle der Datenquelle, auf die der Benutzer zureifen darf.

2. Erteilen Sie den Benutzern das Zugriffsrecht SELECT auf diese Sicht.
3. Erstellen Sie einen Kurznamen, der auf diese Sicht verweist.

Anschließend können die Benutzer auf die Spalten zugreifen, indem Sie eine SELECT-Anweisung absetzen, die den neuen Kurznamen als Verweis enthält.

Im folgenden Szenario erhalten Sie ein genaueres Beispiel, wie Sichten zur Einschränkung des Datenzugriffs verwendet werden können.

Viele Benutzer benötigen vielleicht aus unterschiedlichen Gründen Zugriff auf Informationen der Tabelle STAFF. Zum Beispiel:

- Die Personalabteilung muss in der Lage sein, die gesamte Tabelle anzuzeigen und zu aktualisieren.

  Diese Anforderung kann leicht dadurch erfüllt werden, dass der Gruppe PERSONNL die Zugriffsrechte SELECT und UPDATE für die Tabelle STAFF erteilt werden:

  ```
  GRANT SELECT,UPDATE ON TABLE STAFF TO GROUP PERSONNL
  ```

- Die einzelnen Abteilungsleiter müssen in der Lage sein, Gehaltsdaten für ihre Mitarbeiter anzuzeigen.

  Diese Anforderung kann erfüllt werden, indem eine Sicht für jeden Abteilungsleiter erstellt wird. Zum Beispiel kann folgende Sicht für den Leiter der Abteilung Nr. 51 erstellt werden:

  ```
  CREATE VIEW EMP051 AS
          SELECT NAME,SALARY,JOB FROM STAFF
          WHERE DEPT=51
     GRANT SELECT ON TABLE EMP051 TO JANE
  ```

  Der Abteilungsleiter mit dem Berechtigungsnamen JANE würde die Sicht EMP051 genauso wie die Tabelle STAFF abfragen. Wenn er auf die Sicht EMP051 der Tabelle STAFF zugreift, werden dem Abteilungsleiter folgende Informationen angezeigt:

NAME	SALARY	JOB
Fraye	45150.0	Mgr
Williams	37156.5	Sales
Smith	35654.5	Sales
Lundquist	26369.8	Clerk
Wheeler	22460.0	Clerk

- Alle Benutzer müssen in der Lage sein, andere Mitarbeiter zu finden. Diese Anforderung kann erfüllt werden, indem eine Sicht auf die Spalte NAME der Tabelle STAFF und auf die Spalte LOCATION der Tabelle ORG erstellt wird und die beiden Tabellen über die entsprechenden Spalten DEPT und DEPTNUMB verknüpft werden:

  ```
  CREATE VIEW EMPLOCS AS
          SELECT NAME, LOCATION FROM STAFF, ORG
          WHERE STAFF.DEPT=ORG.DEPTNUMB
     GRANT SELECT ON TABLE EMPLOCS TO PUBLIC
  ```

  Wenn Benutzer auf die Sicht EMPLOCS über die Standorte der Mitarbeiter zugreifen, werden folgende Informationen angezeigt:

NAME	LOCATION
Molinare	New York
Lu	New York

NAME	LOCATION
Daniels	New York
Jones	New York
Hanes	Boston
Rothman	Boston
Ngan	Boston
Kermisch	Boston
Sanders	Washington
Pernal	Washington
James	Washington
Sneider	Washington
Marenghi	Atlanta
O'Brien	Atlanta
Quigley	Atlanta
Naughton	Atlanta
Abrahams	Atlanta
Koonitz	Chicago
Plotz	Chicago
Yamaguchi	Chicago
Scoutten	Chicago
Fraye	Dallas
Williams	Dallas
Smith	Dallas
Lundquist	Dallas
Wheeler	Dallas
Lea	San Francisco
Wilson	San Francisco
Graham	San Francisco
Gonzales	San Francisco
Burke	San Francisco
Quill	Denver
Davis	Denver
Edwards	Denver
Gafney	Denver

Zugriffssteuerung für Datenbankadministratoren (DBAs)

Teilweise ist es sinnvoll, den Datenzugriff von Datenbankadministratoren (Benutzern mit der Berechtigung DBADM) zu überwachen, steuern oder zu verhindern.

Überwachen des Datenzugriffs

Den Datenzugriff von Datenbankadministratoren können Sie mithilfe der DB2-Prüffunktion überwachen. Führen Sie dazu die folgenden Schritte aus:

1. Erstellen Sie eine Prüfrichtlinie zum Überwachen der Ereignisse, die Sie bei Benutzern mit der Berechtigung DBADM erfassen möchten.
2. Ordnen Sie dieser Prüfrichtlinie die Berechtigung DBADM zu.

Steuern des Datenzugriffs

Sie können den Zugriff von Datenbankadministratoren mithilfe einer Kombination von gesicherten Kontexten und Rollen steuern. Führen Sie dazu die folgenden Schritte aus:

1. Erstellen Sie eine Rolle, und erteilen Sie dieser Rolle die Berechtigung DBADM.
2. Definieren Sie einen gesicherten Kontext, und machen Sie die Rolle zur Standardrolle für diesen gesicherten Kontext.

   Erteilen Sie die Zugehörigkeit zu dieser Rolle keiner Berechtigungs-ID explizit. Auf diese Weise ist die Rolle nur über diesen gesicherten Kontext verfügbar, und Benutzer erhalten DBADM-Berechtigungen nur, wenn sie sich innerhalb der Grenzen des gesicherten Kontexts befinden.
3. Es gibt zwei Verfahren zur Steuerung, wie Benutzer auf den gesicherten Kontext zugreifen:
   - Impliziter Zugriff: Erstellen Sie einen eindeutigen gesicherten Kontext für jeden Benutzer. Wenn der Benutzer eine reguläre Verbindung herstellt, die den Attributen des gesicherten Kontexts entspricht, wird er als implizit gesicherter Benutzer erkannt und erhält Zugriff auf die Rolle.
   - Expliziter Zugriff: Erstellen Sie einen gesicherten Kontext mit der Klausel WITH USE FOR, indem Sie alle Benutzer definieren, die Zugriff haben sollen. Erstellen Sie eine Anwendung, über die diese Benutzer Datenbankanforderungen absetzen können. Die Anwendung stellt eine explizite gesicherte Verbindung her. Wenn ein Benutzer eine Anforderung absetzt, wechselt die Anwendung zur entsprechenden Benutzer-ID und führt die Anforderung als dieser Benutzer in der Datenbank aus.

Wenn Sie die Verwendung dieses gesicherten Kontexts überwachen möchten, können Sie dazu eine Prüffunktion erstellen, die die für Sie interessanten Ereignisse für Benutzer dieses gesicherten Kontexts erfasst. Ordnen Sie diese Prüfrichtlinie dem gesicherten Kontext zu.

Verhindern von Datenzugriffen

Wählen Sie eine der folgenden Optionen aus, wenn Sie den Zugriff auf Daten in Tabellen verhindern möchten:

- Wenn Sie den Zugriff auf Daten in allen Tabellen verhindern möchten, entziehen Sie dem Benutzer, der Rolle oder der Gruppe mit der Berechtigung DBADM die Berechtigung DATAACCESS. Stattdessen können Sie dem betreffenden Benutzer bzw. der betreffenden Rolle oder Gruppe auch die Berechtigung DBADM ohne die Option DATAACCESS erteilen.
- Gehen Sie wie folgt vor, wenn Sie den Zugriff auf Daten in einer bestimmten Tabelle verhindern möchten:
  - Ordnen Sie jeder Spalte der Tabelle einen Sicherheitskennsatz zu.
  - Erteilen Sie diesen Sicherheitskennsatz einer Rolle.
  - Erteilen Sie allen Benutzern (oder Rollen), die den Zugriff auf die Tabelle für ihre Arbeit benötigen, diese Rolle.

  Kein Benutzer, der nicht zu dieser Rolle gehört, kann anschließend auf die Daten in der betreffenden Tabelle zugreifen, unabhängig davon, über welche Berechtigungen er verfügt.

Indirekte Methoden des Zugriffs auf Daten

Für eine erfolgreiche Verwaltung der Sicherheit müssen Sie mit den indirekten Methoden vertraut sein, mit denen sich Benutzer Zugriff auf Daten verschaffen können.

Im Folgenden sind die indirekten Methoden aufgelistet, durch die sich Benutzer Zugriff auf Daten verschaffen können, für die sie möglicherweise nicht berechtigt sind:

- **Katalogsichten:** In den Katalogsichten des DB2-Datenbanksystems werden Metadaten und Statistiken über Datenbankobjekte gespeichert. Benutzer mit dem Zugriffsrecht SELECT für die Katalogsichten können Erkenntnisse über Daten gewinnen, für die sie möglicherweise nicht qualifiziert sind. Zur besseren Sicherheit müssen Sie sicherstellen, dass nur qualifizierte Benutzer Zugriff auf die Katalogsichten haben.

  Anmerkung: In DB2 Universal Database Version 8 und früheren Versionen wurde das Zugriffsrecht SELECT standardmäßig der Gruppe PUBLIC erteilt. In Datenbanksystemen von DB2 ab Version 9.1 können Benutzer wählen, ob das Zugriffsrecht SELECT für die Katalogtabellen der Gruppe PUBLIC erteilt werden soll oder nicht, indem sie die neue Option **RESTRICTIVE** im Befehl **CREATE DATABASE** verwenden.
- **EXPLAIN-Momentaufnahme:** Die EXPLAIN-Momentaufnahme enthält komprimierte Informationen, die erfasst werden, wenn eine SQL- oder XQuery-Anweisung mit EXPLAIN bearbeitet wird. Sie wird als großes Binärobjekt (BLOB) in der Tabelle EXPLAIN_STATEMENT gespeichert und enthält Spaltenstatistiken, die Aufschluss über Tabellendaten geben können. Zur besseren Sicherheit sollte der Zugriff auf die EXPLAIN-Tabellen nur qualifizierten Benutzern erteilt werden.
- **Section Explain:** Die Prozeduren SECTION EXPLAIN (EXPLAIN_FROM_SECTION, EXPLAIN_FROM_CATALOG, EXPLAIN_FROM_ACTIVITY und EXPLAIN_FROM_DATA) können EXPLAIN-Tabellen mit Daten aus beliebigen Abschnitten füllen, die sich im Paketcache befinden. Zu diesen Daten gehören auch Anweisungstexte, die möglicherweise Eingabedatenwerte enthalten. Zur besseren Sicherheit sollte der Zugriff auf die SECTION EXPLAIN-Prozeduren und EXPLAIN-Tabellen nur qualifizierten Benutzern erteilt werden.
- **Protokolllesefunktionen:** Ein Benutzer, der zur Ausführung einer Funktion berechtigt ist, die die Protokolle liest, kann sich Zugriff auf Daten verschaffen, für die er nicht berechtigt ist, wenn er das Format eines Protokollsatzes versteht. Die folgenden Funktionen lesen die Protokolle:

Funktion	Zum Ausführen der Funktion erforderliche Berechtigung
db2ReadLog	SYSADM oder DBADM
db2ReadLogNoConn	Keine

- **Replikation:** Wenn Sie Daten replizieren, werden auch die geschützten Daten an der Zielposition reproduziert. Zur besseren Sicherheit müssen Sie sicherstellen, dass die Zielposition mindestens ebenso sicher ist wie die Quellenposition.
- **Ausnahmetabellen:** Wenn Sie eine Ausnahmetabelle angeben, wenn Sie Daten in eine Tabelle laden, können Benutzer mit Zugriff auf die Ausnahmetabelle Informationen erlangen, für die sie möglicherweise nicht berechtigt sind. Zur besseren Sicherheit sollten Sie den Zugriff auf die Ausnahmetabelle nur berechtigten Benutzern erteilen und die Ausnahmetabelle sofort löschen, wenn Sie sie nicht mehr benötigen.

- **Backups von Tabellenbereichen oder Datenbanken:** Benutzer mit der Berechtigung zur Ausführung des Befehls **BACKUP DATABASE** können ein Backup einer Datenbank oder eines Tabellenbereichs, einschließlich aller geschützten Daten, erstellen und die Daten an einer beliebigen anderen Position wiederherstellen. Das Backup kann Daten umfassen, auf die der Benutzer ansonsten möglicherweise keinen Zugriff hat.

  Der Befehl **BACKUP DATABASE** kann von Benutzern mit der Berechtigung SYSADM, SYSCTRL oder SYSMAINT ausgeführt werden.
- **SQL-Anweisung SET SESSION AUTHORIZATION:** In DB2 Universal Database Version 8 und früheren Versionen konnte ein Benutzer mit der Berechtigung DBADM mithilfe der SQL-Anweisung SET SESSION AUTHORIZATION die Sitzungsberechtigungs-ID auf einen beliebigen Datenbankbenutzer setzen. In Datenbanksystemen von DB2 ab Version 9.1 muss ein Benutzer explizit durch die Anweisung GRANT SETSESSIONUSER berechtigt werden, bevor er die Sitzungsberechtigungs-ID festlegen kann.

  Bei einem Upgrade einer vorhandenen Datenbank der Version 8 auf ein Datenbanksystem von DB2 Version 9.1 (oder höhere Version) behält ein Benutzer mit expliziter Berechtigung DBADM (z. B. in SYSCAT.DBAUTH erteilt) die Möglichkeit, die Sitzungsberechtigungs-ID auf einen beliebigen Datenbankbenutzer zu setzen. Dies wird zugelassen, damit vorhandene Anwendungen auch weiterhin funktionieren. Die Fähigkeit, die Sitzungsberechtigung festzulegen, bietet potenziell die Möglichkeit, auf alle geschützten Daten zuzugreifen. Zur Implementierung einer restriktiveren Sicherheitsregelung können Sie diese Einstellung überschreiben, indem Sie die SQL-Anweisung REVOKE SETSESSIONUSER ausführen.
- **Lock Monitoring:** Im Rahmen der Aktivitäten zur Sperrenüberwachung von DB2-Datenbankmanagementsystemen werden Werte mit Parametermarken in die Überwachungsausgabe geschrieben, wenn die Aufzeichnungsstufe HIST_AND-_VALUES angegeben wird. Die Werte können auch in den Anweisungstext eingebettet sein, der vom Sperrereignismonitor aufgezeichnet wird. Ein Benutzer mit Zugriff auf die Überwachungsausgabe kann möglicherweise Zugriff auf Informationen erhalten, für die er nicht berechtigt ist.
- **Activity Monitoring:** Im Rahmen der Überwachungsaktivitäten in einem DB2-Datenbankmanagementsystem unter Verwendung eines Aktivitätsereignismonitors werden Werte mit Parametermarken in die Überwachungsausgabe geschrieben, wenn die Klausel VALUES angegeben ist, und der Anweisungstext (der Eingabedatenwerte enthalten kann) wird in die Überwachungsausgabe geschrieben, wenn die Klausel WITH DETAILS angegeben wurde. Ein Benutzer mit Zugriff auf die Überwachungsausgabe kann möglicherweise Zugriff auf Informationen erhalten, für die er nicht berechtigt ist. Zur besseren Sicherheit sollte der Zugriff auf die Anweisung CREATE EVENT MONITOR und sonstige Ereignismonitortabellen nur qualifizierten Benutzern erteilt werden.
- **Package Cache Monitoring:** Im Rahmen der Überwachung des Paketcache in einem DB2-Datenbankmanagementsystem unter Verwendung eines Ereignismonitors für den Paketcache wird der Anweisungstext (der Eingabedatenwerte enthalten kann) in die Überwachungsausgabe geschrieben, sobald ein Abschnitt aus dem Paketcache ausgegeben wird. Zur besseren Sicherheit sollte der Zugriff auf die Anweisung CREATE EVENT MONITOR und sonstige Ereignismonitortabellen nur qualifizierten Benutzern erteilt werden.
- **Monitor Table Functions, Views And Reports:** Die folgenden Überwachungstabellenfunktionen, Sichten und Berichte zeigen Anweisungstext für entweder gerade ausgeführte Anweisungen oder für Anweisungen im Paketcache:
  - SYSPROC.MON_GET_ACTIVITY_DETALS
  - SYSPROC.MON_GET_PKG_CACHE_STMT

- SYSPROC.MON_GET_PKG_CACHE_STMT_DETALS
- SYSIBMADM.MON_PKG_CACHE_SUMMARY
- SYSIBMADM.MON_CURRENT_SQL
- SYSIBMADM.MON_LOCKWAITS
- SYSIBMADM.MONREPORT.LOCKWAIT
- SYSIBMADM.MONREPORT.CURRENTSQL
- SYSIBMADM.MONREPORT.PKGCACHE

Der Anweisungstext kann Eingabedatenwerte enthalten. Zur besseren Sicherheit sollte das Zugriffsrecht EXECUTE für diese Tabellenfunktionen und Berichte sowie das Zugriffsrecht SELECT für diese Sichten nur qualifizierten Benutzern erteilt werden.

- **Traces:** Ein Trace kann Tabellendaten enthalten. Ein Benutzer mit Zugriff auf einen solchen Trace kann Zugriff auf Informationen erhalten, für die er möglicherweise nicht berechtigt ist.
- **Speicherauszugsdateien:** Als Debughilfe für bestimmte Probleme generieren DB2-Datenbankprodukte möglicherweise Speicherauszugsdateien im Verzeichnis `sqllib\db2dump`. Diese Speicherauszugsdateien können Tabellendaten enthalten. Wenn dies der Fall ist, können Benutzer mit Zugriff auf die Dateien Zugriff auf Informationen erhalten, für die sie möglicherweise nicht berechtigt sind. Zur besseren Sicherheit sollten Sie den Zugriff auf das Verzeichnis `sqllib\db2dump` einschränken.
- **db2dart:** Das Tool **db2dart** untersucht eine Datenbank und meldet alle von ihm festgestellten architekturbezogenen Fehler. Das Tool kann auf Tabellendaten zugreifen, und DB2 schränkt diesen Zugriff durch keinerlei Zugriffssteuerung ein. Ein Benutzer mit der Berechtigung zur Ausführung des Tools **db2dart** bzw. mit Zugriff auf die Ausgabe von **db2dart** kann Zugriff auf Informationen erhalten, für die er möglicherweise nicht berechtigt ist.
- **Bindeoption REOPT:** Wenn die Bindeoption **REOPT** angegeben ist, werden die EXPLAIN-Momentaufnahmeninformationen für jede SQL-Anweisung zum reoptimierbaren inkrementellen Binden bei der Ausführung in die EXPLAIN-Tabellen eingefügt. Die EXPLAIN-Informationen zeigen auch Eingabedatenwerte.
- **db2cat:** Das Tool **db2cat** dient zum Extrahieren des gepackten Deskriptors einer Tabelle. Der gepackte Deskriptor einer Tabelle enthält Statistiken, die Informationen zum Inhalt der Tabelle enthüllen können. Ein Benutzer, der das Tool **db2cat** ausführt oder Zugriff auf die Ausgabe hat, kann Zugriff auf Informationen erhalten, für die er möglicherweise nicht berechtigt ist.

Datenverschlüsselung

Daten können beim DB2-Datenbanksystem auf unterschiedliche Weise verschlüsselt werden, sowohl beim Speichern als auch bei der Übertragung über das Netz.

Verschlüsseln von gespeicherten Daten

Sie können gespeicherte Daten auf unterschiedliche Weise verschlüsseln:

- Mit den integrierten Ver- und Entschlüsselungsfunktionen ENCRYPT, DECRYPT_BIN, DECRYPT_CHAR und GETHINT können Sie Ihre in Datenbanktabellen enthaltenen Daten verschlüsseln.
- Mit IBM Database Encryption Expert können Sie die zugrunde liegenden Betriebssystemdaten und Sicherungsdateien verschlüsseln.

- Wenn Sie ein DB2 Enterprise Server Edition-System unter dem Betriebssystem AIX ausführen und die Verschlüsselung nur auf Dateiebene erfolgen soll, können Sie die Betriebssystemdaten und Sicherungsdateien mit EFS (Encrypted File System) verschlüsseln.

Verschlüsseln von Daten bei der Übertragung

Zur Verschlüsselung von Daten bei der Übertragung zwischen Clients und DB2-Datenbanken können Sie den Authentifizierungstyp DATA_ENCRYPT oder die Unterstützung des DB2-Datenbanksystems für Secure Sockets Layer (SSL) verwenden.

Verwendung der Funktionen ENCRYPT, DECRYPT_BIN, DECRYPT_CHAR und GETHINT

Die integrierte Funktion ENCRYPT verschlüsselt Daten mithilfe eines kennwortbasierten Verschlüsselungsverfahrens. Diese Funktionen ermöglichen ferner das Einbinden eines Kennworthinweises. Der Kennworthinweis ist in die verschlüsselten Daten eingebunden. Nach der Verschlüsselung können die Daten nur unter Verwendung des richtigen Kennworts wieder entschlüsselt werden. Anwendungsentwickler, die diese Funktionen verwenden möchten, sollten die Verwaltung vergessener Kennwörter und nicht nutzbarer Daten in Ihre Planung mit einbeziehen.

Das Ergebnis der ENCRYPT-Funktionen sind Daten des Typs VARCHAR FOR BIT DATA (mit einer Begrenzung von 32.631).

Nur Daten der Typen CHAR, VARCHAR und FOR BIT DATA können verschlüsselt werden.

Mit den Funktionen DECRYPT_BIN und DECRYPT_CHAR werden Daten mit der Entschlüsselung auf Kennwortbasis entschlüsselt.

DECRYPT_BIN gibt immer Daten des Typs VARCHAR FOR BIT DATA zurück, während DECRYPT_CHAR immer Daten des Typs VARCHAR zurückgibt. Da das erste Argument CHAR FOR BIT DATA oder VARCHAR FOR BIT DATA sein kann, gibt es Fälle, in denen das Ergebnis nicht mit dem ersten Argument übereinstimmt.

Die Länge des Ergebnisses hängt von den Byte bis zur nächsten 8-Byte-Grenze ab. Die Länge des Ergebnisses könnte die Länge des Datenarguments plus 40 plus die Anzahl von Byte bis zur nächsten 8-Byte-Grenze sein, wenn der optionale Hinweisparameter angegeben wird. Oder die Länge des Ergebnisses könnte die Länge des Datenarguments plus 8 plus die Anzahl von Byte bis zur nächsten 8-Byte-Grenze sein, wenn der optionale Hinweisparameter nicht angegeben wird.

Mit der Funktion GETHINT wird ein eingebundener Kennworthinweis wieder zurückgegeben. Ein Kennworthinweis ist ein Ausdruck, mit dessen Hilfe die Eigner von Daten sich wieder an die Kennwörter erinnern können. Das Wort „Ozean" kann zum Beispiel als Hinweis für das Kennwort "Pazifik" verwendet werden.

Das Kennwort, das zum Verschlüsseln der Daten verwendet wird, wird mit einer der folgenden Methoden ermittelt:

- Kennwortargument. Das Kennwort ist eine Zeichenfolge, die beim Aufruf der Funktion ENCRYPT explizit übergeben wird. Die Daten werden mit dem angegebenen Kennwort verschlüsselt und entschlüsselt.
- Sonderregister für Verschlüsselungskennwort. Die Anweisung SET ENCRYPTION PASSWORD verschlüsselt den Kennwortwert und sendet das verschlüsselte

Kennwort an den Datenbankmanager, der das Kennwort in einem Sonderregister speichert. Werden die Funktionen ENCRYPT, DECRYPT_BIN und DECRYPT_CHAR ohne Kennwortparameter aufgerufen, wird der Wert im Sonderregister ENCRYPTION PASSWORD verwendet. Das Sonderregister ENCRYPTION PASSWORD wird nur in verschlüsselter Form gespeichert.

Der Anfangs- oder Standardwert für das Sonderregister ist eine leere Zeichenfolge.

Gültige Längen für Kennwörter liegen zwischen 6 und 127 einschließlich dieser Längen selbst. Gültige Längen für Hinweise liegen zwischen 0 und 32 einschließlich.

Konfigurieren der SSL-Unterstützung in einer DB2-Instanz

Das DB2-Datenbanksystem unterstützt SSL. Dies bedeutet, dass eine DB2-Clientanwendung, die ebenfalls SSL unterstützt, über ein SSL-Socket eine Verbindung zu einer DB2-Datenbank herstellen kann. CLI-, CLP- und .Net Data Provider-Clientanwendungen und Anwendungen, die IBM Data Server Driver for JDBC and SQLJ (Verbindungen des Typs 4) verwenden, unterstützen SSL.

Vorbereitende Schritte

Führen Sie die folgenden Schritte aus, bevor Sie die SSL-Unterstützung konfigurieren:

- Stellen Sie sicher, dass der Pfad zu den IBM Global Security Kit-Bibliotheken (GSKit-Bibliotheken) in der Umgebungsvariablen **PATH** auf Windows-Plattformen und den Umgebungsvariablen **LIBPATH**, **SHLIB_PATH** oder **LD_LIBRARY_PATH** auf Linux- und UNIX-Plattformen angegeben ist. GSKit wird beim Installieren des DB2-Datenbanksystems automatisch eingeschlossen.

  Auf Windows-32-Bit-Plattformen befinden sich die GSKit-Bibliotheken im Verzeichnis `C:\Programme\IBM\GSK8\lib`. In diesem Fall muss die Umgebungsvariable **PATH** des Systems das Verzeichnis `C:\Programme\IBM\GSK8\lib` einschließen. Bei Windows-64-Bit-Plattformen befinden sich die GSKit-Bibliotheken für 64 Bit im Verzeichnis `C:\Programme\IBM\GSK8\lib64` und die GSKit-Bibliotheken für 32 Bit im Verzeichnis `C:\Programme (x86)\IBM\GSK8\lib`.

  Auf UNIX- und Linux-Plattformen befinden sich die GSKit-Bibliotheken im Verzeichnis `sqllib/lib/gskit`.

  Auf Nicht-Windows-Plattformen wird GSKit vom DB2-Datenbankmanager lokal installiert, sodass sich die GSKit-Bibliotheken einer Instanz entweder im Verzeichnis `sqllib/lib/gskit` oder `sqllib/lib64/gskit` befinden. Um die Instanz aufzurufen, ist es nicht erforderlich, eine weitere Kopie von GSKit an einer globalen Position zu installieren. Ist eine globale Kopie von GSKit vorhanden, ist darauf zu achten, dass die globale Version von GSKit stets dem Versionsstand der lokalen GSKit-Kopie entspricht.
- Stellen Sie sicher, dass der Verbindungskonzentrator nicht aktiviert ist. Die SSL-Unterstützung wird in der DB2-Instanz nicht aktiviert, wenn der Verbindungskonzentrator aktiv ist.

  Wenn Sie feststellen wollen, ob der Verbindungskonzentrator aktiviert ist, führen Sie den Befehl **GET DATABASE MANAGER CONFIGURATION** aus. Wenn der Konfigurationsparameter **max_connections** einen größeren Wert als der Konfigurationsparameter **max_coordagents** aufweist, ist der Verbindungskonzentrator aktiviert.

Informationen zu diesem Vorgang

Die SSL-Kommunikation findet immer im FIPS-Modus statt.

SSL-Unterstützung für DB2 Connect

Bei Verwendung von DB2 Connect for System i, DB2 Connect for System z oder DB2 Enterprise Server Edition auf einem zwischengeschalteten Server-Computer zur Verbindung von DB2-Clients mit einer Host- oder System i-Datenbank ist SSL-Unterstützung in den folgenden Konfigurationen verfügbar:

- Zwischen dem Client und dem DB2 Connect-Server
- Zwischen dem DB2 Connect-Server und dem Server
- Sowohl zwischen dem Client und dem DB2 Connect-Server als auch zwischen dem DB2 Connect-Server und dem Server

Anmerkung: Damit die SSL-Unterstützung für alle Pfade in der Konfiguration aktiviert werden kann, muss jeder Client und jeder Server sämtliche Voraussetzungen für die SSL-Unterstützung erfüllen. Wenn zum Beispiel der DB2 Connect-Verbindungskonzentrator aktiviert ist, können eingehende Anforderungen an den DB2 Connect-Server SSL nicht verwenden. Die abgehende Anforderung an den Zielserver kann jedoch SSL verwenden.

SSL-Unterstützung für HADR-Systeme

SSL wird für Verbindungen zwischen Clients und dem primären HADR-Server unterstützt. Clients, die über SSL eine Verbindung zum primären HADR-Server herstellen, können mithilfe von SSL an die HADR-Bereitschaftsdatenbank weitergeleitet werden. SSL wird jedoch nicht für Verbindungen zwischen primären HADR-Servern und Bereitschaftsservern unterstützt.

Dokumentation zum GSKit-Tool: GSKCapiCmd

Informationen zum GSKit-Tool GSKCapiCmd finden Sie im Handbuch *GSKCapiCmd User's Guide* unter ftp://ftp.software.ibm.com/software/webserver/appserv/library/v80/GSK_CapiCmd_UserGuide.pdf.

Konfigurieren der SSL-Unterstützung

Zum Konfigurieren der SSL-Unterstützung müssen Sie zunächst eine Schlüsseldatenbank zwecks Verwaltung Ihrer digitalen Zertifikate erstellen. Diese Zertifikate und Verschlüsselungsschlüssel werden zur Herstellung der SSL-Verbindungen verwendet. Als nächsten Schritt muss der DB2-Instanzeigner die DB2-Instanz für SSL-Unterstützung konfigurieren.

Vorgehensweise

1. Erstellen Sie eine Schlüsseldatenbank und richten Sie Ihre digitalen Zertifikate ein.
   a. Verwenden Sie das Tool GSKCapiCmd zum Erstellen der Schlüsseldatenbank. Die Schlüsseldatenbank muss vom Typ CMS (Certificate Management System, Zertifikatsmanagementsystem) sein. Das Tool GSKCapiCmd ist ein Befehlszeilentool, das nicht auf Java basiert. Um dieses Tool verwenden zu können, ist es somit nicht erforderlich, dass Java in Ihrem System installiert ist.

      Das Tool GSKCapiCmd wird mithilfe des Befehls **gskcapicmd** aufgerufen. Eine Beschreibung hierzu finden Sie im Handbuch *GSKCapiCmd User's Guide*. Der Pfad für den Befehl lautet `sqllib/gskit/bin` auf Linux- und UNIX-Plattformen und `C:\Programme\IBM\GSK8\bin` auf Windows-Plattformen (32 Bit und 64 Bit). (Auf 64-Bit-Plattformen sind die ausführbaren GSKit-Dateien und -Bibliotheken für 32-Bit-Systeme ebenfalls vorhanden. In diesem Fall lautet der Pfad für den Befehl `C:\Programme (x86)\IBM\GSK8\bin`.) Stellen Sie sicher, dass PATH (auf Windows-Plattformen) bzw. LIBPATH, SHLIB-

_PATH oder LD_LIBRARY_PATH (auf UNIX- oder Linux-Plattformen) den richtigen GSKit-Bibliothekspfad enthält (z. B. sqllib/lib64/gskit).

Mit dem folgenden Befehl werden z. B. die Schlüsseldatenbank `mydbserver.kdb` und die Stashdatei `mydbserver.sth` erstellt:

```
gsk8capicmd_64 -keydb -create -db "mydbserver.kdb" -pw "myServerPassw0rdpw0"
      -stash
```

Bei Verwendung der Option **-stash** wird eine Stashdatei (Dateierweiterung `.sth`) in demselben Verzeichnis wie die Schlüsseldatenbank erstellt. Beim Start der Instanz verwendet GSKit die Stashdatei zum Abrufen des Kennworts für die Schlüsseldatenbank.

Anmerkung: Für die Stashdatei sollte ein wirkungsvoller Schutz des Dateisystems verwendet werden. Standardmäßig hat nur der Instanzeigner Zugriff auf diese Datei (Lese- und Schreibzugriff).

Wenn Sie eine Schlüsseldatenbank erstellen, wird diese automatisch mit Unterzeichnerzertifikaten einiger wenigen Zertifizierungsstellen (CAs) wie beispielsweise Verisign gefüllt.

b. Fügen Sie der Schlüsseldatenbank ein Zertifikat für Ihren Server hinzu. Der Server sendet dieses Zertifikat während des SSL-Handshakes an die Clients, um Authentifizierungsinformationen für den Server bereitzustellen. Zum Abrufen eines Zertifikats können Sie entweder GSKCapiCmd verwenden, um eine neue Zertifikatsanforderung zu erstellen und zwecks Unterzeichnung an eine CA zu übergeben, oder Sie können zu Testzwecken ein selbst signiertes Zertifikat erstellen.

   Erstellen Sie z. B. mit dem Befehl **GSKCapiCmd** ein selbst signiertes Zertifikat mit der Bezeichnung "myselfsigned" wie im folgenden Beispiel:

```
gsk8capicmd_64 -cert -create -db "mydbserver.kdb" -pw "myServerPassw0rdpw0"
       -label "myselfsigned" -dn "CN=myhost.mycompany.com,O=myOrganization,
       OU=myOrganizationUnit,L=myLocation,ST=ON,C=CA"
```

c. Extrahieren Sie das soeben erstellte Zertifikat in eine Datei, damit Sie das Zertifikat an Computer mit Clients verteilen können, die SSL-Verbindungen zu Ihrem DB2-Server herstellen sollen.

   Mit dem folgenden Befehl GSKCapiCmd wird das Zertifikat beispielsweise in eine Datei mit der Bezeichnung `mydbserver.arm` extrahiert:

```
gsk8capicmd_64 -cert -extract -db "mydbserver.kdb" -pw "myServerPassw0rdpw0"
      -label "myselfsigned" -target "mydbserver.arm" -format ascii -fips
```

2. Um Ihren DB2-Server für SSL-Unterstützung einzurichten, melden Sie sich als DB2-Instanzeigner an und setzen Sie die folgenden Konfigurationsparameter sowie die Registrierdatenbankvariable **DB2COMM**.

   a. Setzen Sie den Konfigurationsparameter **ssl_svr_keydb** auf den vollständig qualifizierten Pfad zur Schlüsseldatenbankdatei. Beispiel:

```
db2 update dbm cfg using SSL_SVR_KEYDB /home/test/sqllib/security/keystore/key.kdb
```

   Wird für **ssl_svr_keydb** kein Wert gesetzt, ist die SSL-Unterstützung nicht aktiviert.

   b. Setzen Sie den Konfigurationsparameter **ssl_svr_stash** auf den vollständig qualifizierten Pfad zur Stashdatei. Beispiel:

```
db2 update dbm cfg using SSL_SVR_STASH /home/test/sqllib/security/keystore/mydbserver.sth
```

   Wird für **ssl_svr_stash** kein Wert gesetzt, ist die SSL-Unterstützung nicht aktiviert.

   c. Setzen Sie den Konfigurationsparameter **ssl_svr_label** auf die Bezeichnung des digitalen Zertifikats des Servers, das Sie in Schritt 1 hinzugefügt haben.

Wird für **ssl_svr_label** kein Wert gesetzt, wird das Standardzertifikat in der Schlüsseldatenbank verwendet. Ist in der Schlüsseldatenbank kein Standardzertifikat vorhanden, ist SSL nicht aktiviert. Beispiel: `db2 update dbm cfg using SSL_SVR_LABEL myselfsigned`, wobei es sich bei *myselfsigned* um eine Beispielbezeichnung handelt.

d. Setzen Sie den Konfigurationsparameter **ssl_svcename** auf den Port, an dem das DB2-Datenbanksystem für SSL-Verbindungen empfangsbereit sein soll. Sind sowohl TCP/IP als auch SSL aktiviert (die Registrierdatenbankvariable **DB2COMM** ist auf 'TCPIP, SSL' gesetzt), müssen Sie **ssl_svcename** auf einen anderen Port setzen als den, der für **svcename** festgelegt wurde. Über den Konfigurationsparameter **svcename** wird der Port angegeben, an dem das DB2-Datenbanksystem für TCP/IP-Verbindungen empfangsbereit sein soll. Wenn Sie **ssl_svcename** auf denselben Port setzen wie **svcename**, ist weder TCP/IP noch SSL aktiviert. Wird für **ssl_svcename** kein Wert angegeben, ist die SSL-Unterstützung nicht aktiviert.

   Anmerkung: In HADR-Umgebungen dürfen Sie den Konfigurationsparameter **hadr_local_svc** auf dem primären Datenbanksystem oder Bereitschaftsdatenbanksystem nicht auf denselben Wert setzen wie für den Konfigurationsparameter **ssl_svcename**. Ebenso dürfen Sie den Parameter **hadr_local_svc** nicht auf denselben Wert setzen wie für **svcename** oder **svcename** plus eins.

   Anmerkung: Wenn die Registrierdatenbankvariable **DB2COMM** auf 'TCPIP,SSL' gesetzt ist und die TCP/IP-Unterstützung nicht ordnungsgemäß aktiviert ist (weil beispielsweise kein Wert für den Konfigurationsparameter **svcename** angegeben wurde), wird die Fehlermeldung SQL5043N zurückgegeben und ist die SSL-Unterstützung nicht aktiviert.

e. Optional: Wenn Sie angeben wollen, welche Cipher Suites der Server verwenden kann, setzen Sie den Konfigurationsparameter **ssl_cipherspecs** entsprechend. Wenn für **ssl_cipherspecs** kein Wert angegeben wird, ist GSKit in der Lage, aus den verfügbaren Cipher Suites die wirkungsvollste auszuwählen, die sowohl vom Client als auch vom Server unterstützt wird. Informationen zu den verfügbaren Cipher Suites finden Sie im Abschnitt „Unterstützte Cipher Suites” auf Seite 84.

f. Fügen Sie den Wert SSL der Registrierdatenbankvariablen **DB2COMM** hinzu. Beispiel:

   ```
   db2set -i db2inst1 DB2COMM=SSL
   ```

   Dabei ist *db2inst1* der Name der DB2-Instanz. Der Datenbankmanager kann mehrere Protokolle gleichzeitig unterstützen. Wenn Sie zum Beispiel die beiden Kommunikationsprotokolle TCP/IP und SSL aktivieren wollen, geben Sie den folgenden Befehl ein:

   ```
   db2set -i db2inst1 DB2COMM=SSL,TCPIP
   ```

g. Starten Sie die DB2-Instanz erneut. Beispiel:

   ```
   db2stop
   db2start
   ```

Beispiel

Das folgende Beispiel veranschaulicht das Anzeigen eines Zertifikats. Dieses Beispiel verwendet das selbst signierte Zertifikat, das mit dem folgenden Befehl erstellt wird:

```
gsk8capicmd_64 -cert -create -db "mydbserver.kdb" -pw "mydbserverpw0"
  -label "myselfsigned" -dn "CN=myhost.mycompany.com,O=myOrganization,
        OU=myOrganizationUnit,L=myLocation,ST=ON,C=CA"
```

Geben Sie den folgenden Befehl ein, um das Zertifikat anzuzeigen:

```
gsk8capicmd_64 -cert -details -db "mydbserver.kdb" -pw "mydbserverpw0"
  -label "myselfsigned"
```

Die folgende Ausgabe wird angezeigt:

```
label : myselfsigned
key size : 1024
version : X509 V3
serial : 96c2db8fa769a09d
issue:CN=myhost.mycompany.com,O=myOrganization,OU=myOrganizationUnit,
   L=myLocation,ST=ON,C=CA
subject:CN=myhost.mycompany.com,O=myOrganization,OU=myOrganizationUnit,
   L=myLocation,ST=ON,C=CA
not before : Tuesday, 24 February 2009 17:11:50 PM
not after : Thursday, 25 February 2010 17:11:50 PM
public Key
    30 81 9F 30 0D 06 09 2A 86 48 86 F7 0D 01 01 01
    05 00 03 81 8D 00 30 81 89 02 81 81 00 B6 B8 DC
    79 69 62 C9 A5 C1 5C 38 31 53 AB 27 BE 63 C0 DB
    DE C6 BC 2E A4 0D 37 45 95 22 0E 83 32 FE 67 A9
    2F D7 51 FF 40 A3 76 68 B9 E3 34 CB 7D 4A D8 38
    CA B1 6B 32 66 74 8F E2 B8 DA 8F D0 F3 62 04 BE
    C4 FE 80 2A D0 FF 27 72 37 9A 36 1D DB D3 A1 33
    A1 A6 48 33 E9 64 B9 9B 6B DB 08 60 7D 5E 0E 20
    0A 26 AA 62 3A DF D3 78 56 DC 15 DE 9F 0B 91 DD
    3B 1B 2B E2 82 FA 24 FF 81 A3 F7 3F C1 02 03 01
    00 01
public key type : RSA : 1.2.840.113549.1.1.1
finger print : SHA1 :
    2D C1 93 F8 AC A0 8F E2 C2 05 D8 23 D7 5D 87 E6
    82 3C 47 EC
signature algorithm : SHA1WithRSASignature : 1.2.840.113549.1.1.5
value
    0E 80 24 98 F6 6E 89 43 76 57 76 7F 82 95 18 6A
    43 A5 81 EC F4 82 1F 1F F2 3F E5 61 67 48 C0 59
    94 17 8E 8F DE 4F 7C 35 0C 5D A7 98 73 2A 34 7D
    1E BA 53 78 A5 E4 31 45 D1 08 86 BE 5E 57 C6 9D
    B5 E7 A7 01 3F 54 01 5E 8F 8B 2F 66 19 24 1E A4
    94 58 B0 D4 40 95 AB 98 C2 EF 1C 5C 4A 29 48 EC
    8C C0 A2 B1 AC 2A E9 3C 14 E5 77 B2 A6 55 A8 21
    CB 59 81 86 79 F0 46 35 F8 FC 99 2D EC D4 B9 EB
Trusted : enabled
```

Zum Abrufen eines von einer CA signierten Zertifikats (anstelle eines selbst signierten Zertifikats) müssen Sie eine Anforderung zum Signieren eines Zertifikats generieren und bei der anerkannten CA, zum Beispiel VeriSign, eine entsprechende Gebühr für das Signieren des Zertifikats entrichten. Nachdem Sie das signierte Zertifikat zurückerhalten haben, müssen Sie es empfangen und in die Schlüsseldatenbank des Servers einfügen. Das folgende Beispiel veranschaulicht das Anfordern und Empfangen eines Zertifikats. Dabei wird eine Testversion eines Zertifikats verwendet.

1. Erstellen Sie zuerst eine CSR (Certificate Signing Request, Anforderung zum Signieren eines Zertifikats) für 'mydbserver.kdb'. Mit dem folgenden Befehl wird ein neues Paar aus privatem und öffentlichem RSA-Schlüssel erstellt und eine PKCS10-Zertifikatsanforderung wird in der angegebenen Schlüsseldatenbank generiert. Für CMS-Schlüsseldatenbanken werden die Zertifikatsanforderungs-

informationen in der Datei mit der Erweiterung '.rdb' gespeichert. Die mit der Option '**-file**' angegebene Datei ist die Datei, die an eine CA gesendet werden muss.

```
gsk8capicmd_64 -certreq -create -db "mydbserver.kdb" -pw "mydbserverpw0"
   -label "mycert" -dn "CN=myhost.mycompany.com,
    O=myOrganization,OU=myOrganizationUnit,L=myLocation,ST=ON,C=CA",
   -file "mycertRequestNew"
```

Mit dem folgenden Befehl werden detaillierte Informationen zur Zertifikatsanforderung für 'my db server' aufgeführt:

```
gsk8capicmd_64 -certreq -details -showOID -db "mydbserver.kdb"
   -pw "mydbserverpw0" -label "mycert"
```

Die folgende Ausgabe wird angezeigt:

```
label : mycert
key size : 1024
subject :    Common Name (CN):
    Type : 2.5.4.3
    Value: myhost.mycompany.com
   Organization (O):
    Type : 2.5.4.10
    Value: myOrganization
   Organizational Unit (OU):
    Type : 2.5.4.11
    Value: myOrganizationUnit
   Locality (L):
    Type : 2.5.6.3
    Value: myLocation
   State (ST):
    Type : ?
    Value: Ontario
   Country or region (C):
    Type : 2.5.4.6
    Value: CA
public Key
    30 81 9F 30 0D 06 09 2A 86 48 86 F7 0D 01 01 01
    05 00 03 81 8D 00 30 81 89 02 81 81 00 9C B4 62
    3C 89 02 4E B0 D8 EA 0B B8 CC 70 63 4A 59 1F 0F
    FD 98 9A 1A 39 94 E3 43 C1 63 7A CD 21 47 57 D9
    86 6F 11 B8 91 08 AC E3 E2 21 32 FE 43 1F 07 C9
    F5 40 6B 3E 4D 56 35 05 62 D6 78 0B E3 97 28 F7
    27 31 A4 05 BE F2 3A 44 6B D8 D1 FF 1E DA 59 63
    E6 49 52 39 45 9C 1E 8E CC DA A1 D9 0F 3A 96 09
    66 5C 89 23 2E EE 31 65 8D 87 8E B9 61 C6 69 BC
    A5 DB EB 03 16 E6 33 85 14 68 BC DD F1 02 03 01
    00 01
finger print :
e0dcde10ded3a46a53c0190e84cc994e
5d7e4bad
attributes
signature algorithm1.2.840.113549.1.1.5
value
    4F 06 B4 E3 1F 00 B4 81 90 CC A2 99 4A 02 68 D0
    84 B5 7F 33 0B F0 04 D5 7D 4C 5C CB 5C D3 37 77
    E2 6D 10 17 50 19 D0 7F 61 C7 C8 54 7B DB CD 6F
    47 9F 7E 7E 5A CC 64 20 85 95 A8 5E C7 7D FB F4
    8A 7F 4B 74 6F 0A C6 EF 09 E7 0A 15 17 CC 1D D2
    5D ED 02 A1 BE 1D FC F2 65 EB 0D E2 93 BC 88 4C
    4C 73 76 16 9F 1B 12 3B 7A 01 CF E0 63 97 E8 38
    02 FB 47 EE F2 17 54 66 4D F7 7F 9E 13 DA 76 A2
```

Anzeigen der Zertifikatanforderungsdatei:

```
$ cat mycertRequestNew

-----BEGIN NEW CERTIFICATE REQUEST-----
MIIBrjCCARcCAQAwbjELMAkGA1UEBhMCQ0ExEDAOBgNVBAgTB09udGFyaW8xEDAO
BgNVBAcTB01hcmtoYW0xDDAKBgNVBAoTA0lCTTEMMAoGA1UECxMDREIyMR8wHQYD
VQQDExZnaWxlcmEudG9yb2xhYi5pYm0uY29tMIGfMA0GCSqGSIb3DQEBAQUAA4GN
ADCBiQKBgQCctGI8iQJOsNjqC7jMcGNKWR8P/ZiaGjmU40PBY3rNIUdX2YZvEbiR
CKzj4iEy/kMfB8n1QGs+TVY1BWLWeAvjlyj3JzGkBb7yOkRr2NH/HtpZY+ZJUjlF
nB6OzNqh2Q86lglmXIkjLu4xZY2Hjrlhxmm8pdvrAxbmM4UUaLzd8QIDAQABoAAw
DQYJKoZIhvcNAQEFBQADgYEATwa04x8AtIGQzKKZSgJo0IS1fzML8ATVfUxcy1zT
N3fibRAXUBnQf2HHyFR7281vR59+flrMZCCFlahex3379Ip/S3RvCsbvCecKFRfM
HdJd7QKhvh388mXrDeKTvIhMTHN2Fp8bEjt6Ac/gY5foOAL7R+7yF1RmTfd/nhPa
dqI=
-----END NEW CERTIFICATE REQUEST-----
```

Falls die Zertifikatsanforderung gelöscht werden muss, verwenden Sie einen Befehl ähnlich dem folgenden Beispiel:

```
gsk8capicmd_64 -certreq -delete -db "mydbserver.kdb" -pw "mydbserverpw0"
   -label "mycert"
```

2. Rufen Sie die VeriSign-Website auf und führen Sie eine Registrierung durch. Sie werden dazu aufgefordert, die Anforderungsdatei auszuschneiden und einzufügen, um die Anforderung zu übergeben. Sie erhalten für die Testversion eine E-Mail mit dem signierten Zertifikat. Die E-Mail enthält darüber hinaus Links zum Download des CA-Teststammzertifikats und des temporären CA-Testzertifikats. Verwenden Sie Notepad oder vi, um alle drei Zertifikate in Dateien zu speichern.
   - RootCert.arm
   - IntermediateCert.arm
   - MyCertificate.arm

   Diese drei Zertifikate sind Teil einer Trust-Kette.

   Fügen Sie das CA-Teststammzertifikat mit dem folgenden Befehl in 'mydbserver.kdb' ein:

   ```
   gsk8capicmd_64 -cert -add -db "mydbserver.kdb" -pw "mydbserverpw0"
      -label "trialRootCACert"  -file RootCert.arm -format ascii
   ```

   Fügen Sie das temporäre CA-Testzertifikat mit dem folgenden Befehl in 'mydbserver.kdb' ein:

   ```
   gsk8capicmd_64 -cert -add -db "mydbserver.kdb" -pw "mydbserverpw0"
      -label "trialIntermediateCACert" -file IntermediateCert.arm -format ascii
   ```

   Empfangen Sie das Testzertifikat und fügen Sie es in 'mydbserver.kdb' ein, indem Sie den folgenden Befehl verwenden:

   ```
   $ cat SSLCertificate.cer2

   -----BEGIN CERTIFICATE-----
   MIIFVjCCBD6gAwIBAgIQdOydrySM+J4uUPNzbPHhVjANBgkqhkiG9w0BAQUFADCB
   yzELMAkGA1UEBhMCVVMxFzAVBgNVBAoTDlZlcmlTaWduLCBJbmMuMTAwLgYDVQQL
   EydGb3IgVGVzdCBQdXJwb3NlcyBPbmx5LiAgTm8gYXNzdXJhbmNlcy4xQjBABgNV
   BAsTOVRlcm1zIG9mIHVzZSBhdCBodHRwczovL3d3dy52ZXJpc2lnbi5jb20vY3Bz
   L3Rlc3RjYSAoYykwNTEtMCsGA1UEAxMkVmVyaVNpZ24gVHJpYWwgU2VjdXJlIFNl
   cnZlciBUZXN0IENBMB4XDTA5MDIyMzAwMDAwMFoXDTA5MDMwOTIzNTk1OVowgaox
   CzAJBgNVBAYTAkNBMRAwDgYDVQQIEwdPbnRhcmlvMRAwDgYDVQQHFAdNYXJraGFt
   MQwwCgYDVQQKFANJQk0xDDAKBgNVBAsUA0RCMjE6MDgGA1UECxQxVGVybXMgb2Yg
   dXNlIGF0IHd3dy52ZXJpc2lnbi5jb20vY3BzL3Rlc3RjYSAoYykwNTEfMB0GA1UE
   AxQWZ2lsZXJhLnRvcm9sYWIuaWJtLmNvbTCBnzANBgkqhkiG9w0BAQEFAAOBjQAw
   gYkCgYEAnLRiPIkCTrDY6gu4zHBjSlkfD/2Ymho5lONDwWN6zSFHV9mGbxG4kQis
   4+IhMv5DHwfJ9UBrPk1WNQVi1ngL45co9ycxpAW+8jpEa9jR/x7aWWPmSVI5RZwe
   jszaodkPOpYJZlyJIy7uMWWNh465YcZpvKXb6wMW5jOFFGi83fECAwEAAaOCAdcw
   ggHTMAkGA1UdEwQCMAAwCwYDVR0PBAQDAgWgMEMGA1UdHwQ8MDowOKA2oDSGMmh0
   ```

```
dHA6Ly9TVlJTZWN1cmUtY3JsLnZlcmlzaWduLmNvbS9TVlJUcmlhbDIwMDUuY3Js
MEoGA1UdIARDMEEwPwYKYIZIAYb4RQEHFTAxMC8GCCsGAQUFBwIBFiNodHRwczov
L3d3dy52ZXJpc2lnbi5jb20vY3BzL3Rlc3RjYTAdBgNVHSUEFjAUBggrBgEFBQcD
AQYIKwYBBQUHAwIwHwYDVR0jBBgwFoAUZiKOgeAxWd0qf6tGxTYCBnAnh1oweAYI
KwYBBQUHAQEEbDBqMCQGCCsGAQUFBzABhhhodHRwOi8vb2NzcC52ZXJpc2lnbi5j
b20wQgYIKwYBBQUHMAKGNmh0dHA6Ly9TVlJTZWN1cmUtYWlhLnZlcmlzaWduLmNv
bS9TVlJUcmlhbDIwMDUtYWlhLmNlcjBuBggrBgEFBQcBDARiMGChXqBcMFowWDBW
FglpbWFnZS9naWYwITAfMAcGBSsOAwIaBBRLa7kolgYMu9BSOJsprEsHiyEFGDAm
FiRodHRwOi8vbG9nby52ZXJpc2lnbi5jb20vdnNsb2dvMS5naWYwDQYJKoZIhvcN
AQEFBQADggEBAKs1YpIeOAL6mTryIXpYfokkzRdwP5ooDutHhVbRYcPwq9ynOrHM
3gZolv8th5PpSkZAGTPr3HJZG6HnxRiQjPT88PAADR3SEzVMzQEESHfYToF1qBPZ
svigphI9eIHcg5IWwv7dyuXtkFGbTCqcvEqJiT3UHhubgMfoTuTGayhNoGt75FGU
h4kSJz3af6MNuGmQLs4wzJTepU7srlhGV1C1ujTCydax2BiWfWwO4YaFcckvHxbR
6I7vVj1PTC2RO8n5qcWJYmGU0PG3d58hJETD4E8tAReh21ShBWDgn4+e0k1XtQ8K
lB66QpsFYGTLtGyd/4w4BAgq/QLmcs+mpjc=
-----END CERTIFICATE-----

gsk8capicmd_64 -cert -receive -file MyCertificate.arm -db "mydbserver.kdb"
   -pw "mydbserverp -format ascii
```

Auflisten aller Zertifikate in 'mydbserver.kdb' mit dem folgenden Befehl:

```
gsk8capicmd_64 -cert -list all -db "mydbserver.kdb" -pw "mydbserverpw0"

certificates found
* default, - personal, ! trusted
-!      mycert
!       trialIntermediateCACert
!       trialRootCACert
-!      myselfsigned
db2 update dbm cfg using SSL_SVR_LABEL mycert
```

Konfigurieren der SSL-Unterstützung in nicht auf Java basierten DB2-Clients

Sie können DB2-Datenbankclients wie beispielsweise CLI-, CLP- und .Net Data Provider-Clients zur Unterstützung von SSL (Secure Sockets Layer) für die Kommunikation mit dem DB2-Server konfigurieren.

Vorbereitende Schritte

Anmerkung: Wenn Ihr DB2-Client der Version 9.7 oder DB2 Connect-Server eine SSL-Verbindung zu einem DB2 for z/OS-Server auf einem System mit z/OS Version 1.8, 1.9 oder 1.10 herstellt, muss die entsprechende vorläufige Programmkorrektur für APAR PK72201 auf die IP Services von Communication Server für z/OS angewendet werden.

Anmerkung: Aufgrund einer Inkompatibilität zwischen GSKit Version 8 und GSKit-7d-Versionen vor Version 7.0.4.20 schlagen CLI-Anwendungen, die eine Verbindung zu einem IDS-Datenserver unter Verwendung von GSKit-7d-Versionen vor Version 7.0.4.20 herstellen, fehl. Führen Sie zur Behebung dieses Fehlers ein Upgrade für die GSKit-Bibliothek auf dem IDS-Datenserver auf GSKit Version 7.0.4.20 oder eine spätere Version durch.

Führen Sie die folgenden Schritte aus, bevor Sie die SSL-Unterstützung für einen Client konfigurieren:

- Wenn sich der Client und der Server auf demselben physischen Computer befinden, brauchen Sie GSKit nicht zu installieren, da GSKit automatisch mit dem DB2-Server installiert wird.

  Ab Version 9.7 Fixpack 1 werden bei Installation der 64-Bit-Version des DB2-Servers die 32-Bit-GSKit-Bibliotheken automatisch mit installiert. Bei Verwendung dieser Bibliotheken unter Linux- und UNIX-Betriebssystemen müssen Sie sicher-

stellen, dass die Umgebungsvariable **LD_LIBRARY_PATH**, **LIBPATH** oder **SHLIB_PATH** richtig eingestellt ist. Unter Windows-Betriebssystemen muss sichergestellt werden, dass die Umgebungsvariable **PATH** richtig eingestellt ist, wie aus folgender Tabelle hervorgeht.

Anwendung	Betriebssystem	Position der GSKit-Bibliotheken	Umgebungsvariableneinstellung
32-Bit	Linux und UNIX (64-Bit)	*$INSTHOME*/sqllib/lib32/gskit	*$INSTHOME*/sqllib/lib32/gskit muss in die Umgebungsvariable *LD_LIBRARY_PATH*, *LIBPATH* oder *SHLIB_PATH* aufgenommen werden.
64-Bit	Linux und UNIX (64-Bit)	*$INSTHOME*/sqllib/lib64/gskit	*$INSTHOME*/sqllib/lib64/gskit muss in die Umgebungsvariable *LD_LIBRARY_PATH*, *LIBPATH* oder *SHLIB_PATH* aufgenommen werden.
32-Bit	Windows (64-Bit)	C:\Programme (x86)\IBM\GSK8\lib	C:\Programme (x86)\IBM\GSK8\lib muss in die Umgebungsvariable *PATH* aufgenommen werden.
64-Bit	Windows (64-Bit)	C:\Programme\IBM\GSK8\lib64	C:\Programme\IBM\GSK8\lib64 muss in die Umgebungsvariable *PATH* aufgenommen werden.

Die SSL-Kommunikation findet immer im FIPS-Modus statt.

Auf Nicht-Windows-Plattformen wird GSKit vom DB2-Datenbankmanager lokal installiert, sodass sich die GSKit-Bibliotheken einer Instanz entweder im Verzeichnis sqllib/lib/gskit oder sqllib/lib64/gskit befinden. Es ist nicht erforderlich, eine weitere Kopie von GSKit an einer globalen Position zu installieren. Ist eine globale Kopie von GSKit vorhanden, ist darauf zu achten, dass die globale Version von GSKit stets dem Versionsstand der lokalen GSKit-Kopie entspricht.

- Wenn bei der Verwendung von C-basierten Clients der Client auf einem separaten Computer installiert wird, muss GSKit installiert werden, wenn die Clients SSL für die Kommunikation mit den Servern verwenden. Sie können GSKit-Bibliotheken mithilfe der DVD mit IBM DB2-Dateien zur Unterstützung der SSL-Funktionalität installieren. Stattdessen können Sie auch ein Image installieren, das Sie von Passport Advantage heruntergeladen haben.
  - Stellen Sie sicher, dass der Pfad zu den IBM Global Security Kit-Bibliotheken (GSKit-Bibliotheken) in der Umgebungsvariablen **PATH** unter Windows und in den Umgebungsvariablen **LIBPATH**, **SHLIB_PATH** oder **LD_LIBRARY_PATH** unter Linux und UNIX angegeben ist. Beispiel: Fügen Sie unter Windows der Umgebungsvariablen **PATH** wie folgt die GSKit-Verzeichnisse bin und lib hinzu:

    ```
    set PATH="C:\Programme\ibm\gsk8\bin";%PATH%
    set PATH="C:\Programme\ibm\gsk8\lib";%PATH%
    ```

Dokumentation zum GSKit-Tool: GSKCapiCmd

Informationen zum GSKit-Tool GSKCapiCmd finden Sie im Handbuch *GSKCapiCmd User's Guide* unter ftp://ftp.software.ibm.com/software/webserver/appserv/library/v80/GSK_CapiCmd_UserGuide.pdf.

Informationen zu diesem Vorgang

Die SSL-Kommunikation findet immer im FIPS-Modus statt.

Vorgehensweise

Gehen Sie wie folgt vor, um die SSL-Unterstützung in einem DB2-Client zu konfigurieren:

1. Verschaffen Sie sich das Unterzeichnerzertifikat des digitalen Serverzertifikats auf dem Client. Bei dem Serverzertifikat kann es sich um ein selbst signiertes Zertifikat oder ein von einer Zertifizierungsstelle (CA) unterzeichnetes Zertifikat handeln.
   - Wenn es sich bei dem Serverzertifikat um ein selbst signiertes Zertifikat handelt, müssen Sie das zugehörige Unterzeichnerzertifikat zunächst in eine Datei auf dem Server-Computer extrahieren und es anschließend an die Computer mit Clients verteilen, die SSL-Verbindungen zum betreffenden Server herstellen sollen. Informationen zum Extrahieren des Zertifikats in eine Datei finden Sie im Abschnitt „Konfigurieren der SSL-Unterstützung in einer DB2-Instanz" auf Seite 70.
   - Wenn das Serverzertifikat von einer anerkannten Zertifizierungsstelle unterzeichnet ist, ist das CA-Zertifikat, mit dem das Serverzertifikat unterzeichnet wurde, möglicherweise bereits in der Schlüsseldatenbank des Clients vorhanden. Ist dies nicht der Fall, müssen Sie das CA-Zertifikat abrufen. Wenden Sie sich an die Website der Zertifizierungsstelle.
2. Verwenden Sie auf dem DB2-Client-Computer das Tool GSKCapiCmd, um eine Schlüsseldatenbank vom Typ CMS zu erstellen. Das Tool GSKCapiCmd ist ein Befehlszeilentool, das nicht auf Java basiert. Um dieses Tool verwenden zu können, ist es somit nicht erforderlich, dass Java in Ihrem System installiert ist.

   Das Tool GSKCapiCmd wird mithilfe des Befehls **gskcapicmd** aufgerufen. Eine Beschreibung hierzu finden Sie im Handbuch *GSKCapiCmd User's Guide*. Der Pfad für den Befehl lautet `sqllib/gskit/bin` unter Linux und UNIX und `C:\Programm\IBM\GSK8\bin` unter Windows (32-Bit- und 64-Bit-Systeme; unter 64-Bit-Systemen sind die ausführbaren GSKit-Dateien und -Bibliotheken für 32-Bit-Systeme ebenfalls vorhanden. In diesem Fall lautet der Pfad für den Befehl `C:\Programme (x86)\IBM\GSK8\bin`.)

   Mit dem folgenden Befehl werden z. B. eine Schlüsseldatenbank mit der Bezeichnung `mydbclient.kdb` und eine Stashdatei mit der Bezeichnung `mydbclient.sth` erstellt:

   ```
   gsk8capicmd_64 -keydb -create -db "mydbclient.kdb" -pw "myClientPassw0rdpw0"
         -stash
   ```

   Bei Verwendung der Option **-stash** wird eine Stashdatei (Dateierweiterung `.sth`) in demselben Verzeichnis wie die Schlüsseldatenbank erstellt. Beim Herstellen der Verbindung verwendet GSKit die Stashdatei zum Abrufen des Kennworts für die Schlüsseldatenbank.
3. Fügen Sie das Unterzeichnerzertifikat zur Client-Schlüsseldatenbank hinzu.

   Mit dem Befehl **gsk8capicmd** wird das Zertifikat z. B. aus der Datei `mydbserver.arm` in die Schlüsseldatenbank `mydbclient.kdb` kopiert:

   ```
   gsk8capicmd_64 -cert -add -db "mydbclient.kdb" -pw "myClientPassw0rdpw0"
         -label "dbselfsigned" -file "mydbserver.arm" -format ascii -fips
   ```
4. Setzen Sie für Ihre Clientanwendung die entsprechende Verbindungszeichenfolge oder die entsprechenden Konfigurationsparameter anhand des jeweiligen Beispiels für Ihren Client.

Beispiel

CLP- und eingebettete SQL-Clients

CLP-Clients und eingebettete SQL-Clients können eine Verbindung zu einer Datenbank auf einem fernen Host herstellen, sofern mithilfe des Befehls **CATALOG TCPIP NODE** ein entsprechender Eintrag in den Knotenkatalog eingefügt wurde. Setzen Sie den Befehl **CATALOG TCPIP NODE** mit dem auf SSL gesetzten Schlüsselwort **SECURITY** ab, um SSL für die betreffende Verbindung anzugeben.

Das folgende Beispiel zeigt, wie ein Knoten und eine Datenbank so katalogisiert werden, dass ein CLP-Client über SSL eine Verbindung zu ihnen herstellen kann.

Zunächst müssen Sie den Knoten und die Datenbank so katalogisieren, dass Clientanwendungen SSL-Verbindungen zu ihnen herstellen können:

```
catalog TCPIP NODE mynode  REMOTE  127.0.0.1  SERVER 50001  SECURITY SSL

catalog DATABASE sample  AS myssldb  AT NODE mynode  AUTHENTICATION SERVER
```

Als Nächstes müssen Sie mithilfe der Konfigurationsparameter **ssl_clnt_keydb** und **ssl_clnt_stash** die Clientschlüsseldatenbank und die Stashdatei angeben. Setzen Sie den Konfigurationsparameter **ssl_clnt_keydb** auf den vollständig qualifizierten Pfad der Schlüsseldatenbankdatei (.kdb) und den Konfigurationsparameter **ssl_clnt_stash** auf den vollständig qualifizierten Pfad der Stashdatei:

```
db2 update dbm cfg using SSL_CLNT_KEYDB /home/test1/sqllib/security/keystore/
clientkey.kdb
      SSL_CLNT_STASH /home/test1/sqllib/security/keystore/clientstore.sth
```

Ist der Konfigurationsparameter **ssl_clnt_keydb** oder der Konfigurationsparameter **ssl_clnt_stash** nicht definiert (auf null gesetzt), schlägt die Verbindung fehl, und es wird der Fehler SQL10013N mit dem Token GSKit-Fehler: *GSKit-Rückkehrcode* zurückgegeben.

Stellen Sie anschließend über den CLP-Client eine Verbindung zum Server her:

```
db2 connect to myssldb user user1 using password
```

Alternativ kann eine eingebettete SQL-Anwendung zum Herstellen der Verbindung auch die folgende Anweisung verwenden:

```
Strcpy(dbAlias,"myssldb");
 EXEC SQL CONNECT TO :dbAlias USER :user USING :pswd;
```

CLI/ODBC-Clientanwendungen

In Abhängigkeit von der Umgebung, in der Sie Ihre CLI-Anwendung ausführen, verwenden Sie entweder Verbindungszeichenfolgeparameter (ssl_client_keystoredb und ssl_client_keystash) oder DB2-Konfigurationsparameter (**ssl_clnt_keydb** und **ssl_clnt_stash**), um den Pfad für die Clientschlüsseldatenbank und die Stashdatei anzugeben.

- Wenn Sie IBM Data Server Driver for ODBC and CLI einsetzen, verwenden Sie Verbindungszeichenfolgeparameter entsprechend dem folgenden Beispiel:

  Rufen Sie SQLDriverConnect mit einer Verbindungszeichenfolge auf, die das Schlüsselwort SECURITY=SSL enthält. Beispiel:

  ```
  "Database=sampledb; Protocol=tcpip; Hostname= myhost; Servicename=50001;
  Security=ssl; Ssl_client_keystoredb=/home/test1/keystore/clientstore.kdb;
  Ssl_client_keystash=/home/test1/keystore/clientstore.sth;"
  ```

  Da Security=ssl angegeben ist, müssen in diesem Fall die Verbindungszeichenfolgeparameter ssl_client_keystoredb und ssl_client_keystash gesetzt werden. Andernfalls schlägt die Verbindung fehl.

- Wenn Sie IBM Data Server-Client oder IBM Data Server Runtime Client verwenden, können Sie entweder Verbindungszeichenfolgeparameter oder DB2-Konfigurationsparameter verwenden, um den Pfad für die Clientschlüsseldatenbank und die Stashdatei zu definieren. Werden die Verbindungszeichenfolgeparameter **ssl_client_keystoredb** und **ssl_client_keystash** gesetzt, überschreiben diese Einstellungen sämtliche Werte der Konfigurationsparameter **ssl_clnt_keydb** und **ssl_clnt_stash**.

  Im folgenden Beispiel wird die Datei `db2cli.ini` verwendet, um die Verbindungszeichenfolgeparameter zu definieren:

  ```
  [sampledb]
  Database=sampledb
  Protocol=tcpip
  Hostname=myhost
  Servicename=50001
  Security=ssl

  SSL_client_keystoredb=/home/test1/keystore/clientstore.kdb
  SSL_client_keystash=/home/test1/keystore/clientstore.sth
  ```

  Im nachstehenden Beispiel wird das CLI/PDBC-Schlüsselwort **FileDSN** verwendet, um eine DSN-Datei zu identifizieren, die die Datenbankkonnektivitätsinformationen zum Definieren der Verbindungszeichenfolgeparameter enthält. Die DSN-Datei könnte beispielsweise wie folgt aussehen:

  ```
  [ODBC]
  DRIVER=IBM DB2 ODBC DRIVER - DB2COPY1
  UID=user1
  AUTHENTICATION=SERVER
  PORT=50001
  HOSTNAME=myhost
  PROTOCOL=TCPIP
  DATABASE=SAMPLEDB
  SECURITY=SSL
  SSL_client_keystoredb=/home/test1/keystore/clientstore.kdb
  SSL_client_keystash=/home/test1/keystore/clientstore.sth
  ```

  Da `Security=ssl` angegeben ist schlägt in diesen Fällen die Verbindung fehl, wenn die Verbindungszeichenfolgeparameter **ssl_client_keystoredb** und **ssl_client_keystash** nicht gesetzt sind und auch die Konfigurationsparameter **ssl_clnt_keydb** und **ssl_clnt_stash** nicht gesetzt sind.

DB2 .Net Data Provider-Anwendungen

Eine DB2 .Net Data Provider-Anwendung kann eine SSL-Verbindung zu einer Datenbank herstellen, indem der Pfad für die Clientschlüsseldatenbank und die Stashdatei mithilfe der Verbindungszeichenfolgeparameter SSLClientKeystoredb und SSLClientKeystash angegeben wird. Darüber hinaus muss die Verbindungszeichenfolge `Security=SSL` enthalten. Beispiel:

```
String connectString = "Server=myhost:50001;Database=sampledb;Security=ssl;
SSLClientKeystoredb=/home/test1/keystore/clientstore.kdb;
SSLClientKeystash=/home/test1/keystore/clientstore.sth";
```

Anschließend müssen Sie die Zeichenfolge **connectString** an den **DB2Connection**-Konstruktor übergeben und die Methode **Open** des **DB2Connection**-Objekts verwenden, um eine Verbindung zu der in **connectString** angegebenen Datenbank herzustellen. Dies wird anhand des folgenden C#-Codefragments gezeigt:

```
DB2Connection conn = new DB2Connection(connectString);
Conn.Open();
Return conn;
```

Ist der Verbindungszeichenfolgeparameter **SSLClientKeystoredb** oder der Verbindungszeichenfolgeparameter **SSLClientKeystash** nicht definiert (auf null gesetzt), schlägt die Verbindung fehl, und es wird der Fehler SQL10013N mit dem Token `GSKit-Fehler: GSKit-Rückkehrcode` zurückgegeben.

Secure Sockets Layer (SSL)

Das DB2-Datenbanksystem unterstützt den Einsatz von Secure Sockets Layer (SSL) und dem Nachfolgeprodukt Transport Layer Security (TLS), um einen Client in die Lage zu versetzen, einen Server zu authentifizieren, und um mithilfe von Verschlüsselung eine private Kommunikation zwischen dem Client und dem Server zu ermöglichen. Die Authentifizierung erfolgt durch den Austausch von digitalen Zertifikaten.

Anmerkung: Wenn in diesem Abschnitt SSL erwähnt wird, gelten dieselben Informationen auch für TLS, sofern nicht explizit etwas anderes angegeben wird.

Ohne Verschlüsselung können Informationspakete, die in einem Netz übertragen werden, von jedem eingesehen werden, der Zugriff auf das Netz hat. Mithilfe von SSL können Sie Daten, die gerade übertragen werden, in allen Netzen schützen, die TCP/IP verwenden (man kann sich eine SSL-Verbindung demnach als gesicherte TCP/IP-Verbindung vorstellen).

Ein Client und ein Server stellen eine sichere SSL-Verbindung her, indem Sie einen SSL-Handshake durchführen.

Übersicht über den SSL-Handshake

Bei einem *SSL-Handshake* wird der Algorithmus eines öffentlichen Schlüssels (Public Key) verwendet (normalerweise RSA), um in einer sicheren Umgebung digitale Signaturen und Verschlüsselungsschlüssel zwischen einem Server und einem Client auszutauschen. Anhand dieser Identitäts- und Schlüsselinformationen wird eine sichere Verbindung für die Sitzung zwischen dem Client und dem Server hergestellt. Nach Herstellung der sicheren Sitzung wird die Datenübertragung zwischen dem Client und dem Server mittels eines symmetrischen Algorithmus (beispielsweise AES) verschlüsselt.

Während des SSL-Handshakes führen Client und Server die folgenden Schritte aus:

1. Der Client fordert eine SSL-Verbindung an und listet seine unterstützten Cipher Suites auf.
2. Der Server antwortet mit einer ausgewählten Cipher Suite.
3. Der Server sendet sein digitales Zertifikat an den Client.
4. Der Client prüft zwecks Authentifizierung die Gültigkeit des Serverzertifikats. Dies kann durch Überprüfung des Serverzertifikats bei der anerkannten Zertifizierungsstelle (CA) oder in der eigenen Schlüsseldatenbank erfolgen.
5. Der Client und der Server ermitteln in einer sicheren Umgebung einen Sitzungsschlüssel und einen Nachrichtenauthentifizierungscode (MAC).
6. Der Client und der Server tauschen in einer sicheren Umgebung Informationen mithilfe des ausgewählten Schlüssels und MACs aus.

Anmerkung: Das DB2-Datenbanksystem unterstützt nicht die (optionale) Authentifizierung des Clients während des SSL-Handshakes.

Verwendung von SSL-Verschlüsselung mit DB2-Authentifizierung

Sie können die SSL-Verschlüsselung zusammen mit allen vorhandenen DB2-Authentifizierungsmethoden (wie beispielsweise KERBEROS oder SERVER) verwenden. Dabei gehen Sie wie gewohnt vor, indem Sie den Authentifizierungstyp für die Instanz in den Konfigurationsparametern des Datenbankmanagers auf die gewünschte Authentifizierungsmethode setzen.

Digitale Zertifikate und Zertifizierungsstellen (CAs)

Digitale Zertifikate werden von anerkannten Parteien, den sogenannten Zertifizierungsstellen (Certificate Authorities, CAs), ausgegeben, um die Identität einer Entität wie beispielsweise eines Clients oder Servers zu bestätigen.

Digitale Zertifikate dienen zwei Zwecken: Sie bestätigen die Identität des Eigners und machen den öffentlichen Schlüssel des Eigners verfügbar. Jedes digitale Zertifikat wird mit einem Ablaufdatum ausgestellt. Nach diesem Datum wird von der Zertifizierungsstelle (CA) keine Garantie mehr für das Zertifikat übernommen.

Um ein digitales Zertifikat abzurufen, senden Sie eine Anforderung an die gewünschte CA wie beispielsweise Verisign oder RSA. Die Anforderung muss Ihren definierten Namen, Ihren öffentlichen Schlüssel und Ihre Signatur enthalten. Ein definierter Name (DN) ist eine eindeutige Kennung für jeden Benutzer oder Host, für den Sie ein Zertifikat beantragen. Die CA prüft Ihre Signatur anhand Ihres öffentlichen Schlüssels und führt eine gewisse Überprüfung Ihrer Identität durch, wobei der Grad dieser Überprüfung von der jeweiligen CA abhängt. Im Anschluss daran erhalten Sie von der CA ein unterzeichnetes digitales Zertifikat, das Ihren definierten Namen, Ihren öffentlichen Schlüssel sowie den definierten Namen und die Signatur der CA enthält. Dieses unterzeichnete Zertifikat speichern Sie in Ihrer Schlüsseldatenbank.

Wenn Sie dieses Zertifikat an einen Empfänger senden, führt der Empfänger die folgenden beiden Schritte aus, um Ihre Identität zu prüfen:

1. Er prüft Ihre digitale Signatur anhand Ihres mit dem Zertifikat gesendeten öffentlichen Schlüssels.
2. Er überprüft, ob die CA, von der Ihr Zertifikat stammt, legitimiert und vertrauenswürdig ist. Hierfür benötigt der Empfänger den öffentlichen Schlüssel der CA. Es kann sein, dass der Empfänger in seiner Schlüsseldatenbank bereits über eine garantierte Kopie des öffentlichen Schlüssels der CA verfügt. Ist dies nicht der Fall, muss der Empfänger ein zusätzliches digitales Zertifikat beschaffen, um den öffentlichen Schlüssel der CA abzurufen. Dieses Zertifikat wiederum hängt eventuell vom digitalen Zertifikat einer anderen CA ab. Unter Umständen gibt es eine Hierarchie von Zertifikaten, die von mehreren CAs ausgestellt wurden und jeweils von der Gültigkeit des nächsten Zertifikats abhängen. Letztlich benötigt der Empfänger jedoch den öffentlichen Schlüssel der *Root-CA*. Die Root-CA ist die CA, die in der Hierarchie an erster Stelle steht. Um auf die Gültigkeit des digitalen Zertifikats der Root-CA vertrauen zu können, muss der Benutzer des öffentlichen Schlüssels das betreffende digitale Zertifikat in einer sicheren Umgebung empfangen, beispielsweise über ein Download von einem authentifizierten Server, mithilfe einer vorinstallierten Software von einer zuverlässigen Quelle oder auf einer sicher bereitgestellten Diskette.

Viele Anwendungen, die ein digitales Zertifikat an einen Empfänger senden, übertragen nicht nur ihr eigenes Zertifikat, sondern auch alle digitalen CA-Zertifikate, die erforderlich sind, um die Zertifikatshierarchie bis zum Zertifikat der Root-CA prüfen zu können.

Damit einem digitalen Zertifikat vollständig vertraut werden kann, muss der Eigner des digitalen Zertifikats seinen privaten Schlüssel sorgfältig geschützt haben, beispielsweise durch Verschlüsselung auf der Festplatte seines Computers. Wurde die Sicherheit seines privaten Schlüssels beeinträchtigt, könnte dieser Schlüssel für betrügerische Machenschaften missbraucht werden.

Zu Testzwecken können Sie ein selbst signiertes digitales Zertifikat verwenden. Solch ein selbst signiertes digitales Zertifikat enthält Ihren definierten Namen, Ihren öffentlichen Schlüssel und Ihre Signatur.

Verschlüsselung mit öffentlichem Schlüssel

SSL verwendet Algorithmen öffentlicher Schlüssel, um Informationen zu Verschlüsselungsschlüsseln und digitalen Zertifikaten zwecks Authentifizierung auszutauschen. Bei der Verschlüsselung mit einem öffentlichen Schlüssel (auch als asymmetrische Verschlüsselung bezeichnet) werden zwei unterschiedliche Verschlüsselungsschlüssel verwendet: ein öffentlicher Schlüssel zum Verschlüsseln von Daten und ein zugehöriger privater Schlüssel zum Entschlüsseln dieser Daten.

Im Gegensatz dazu wird bei der symmetrischen Verschlüsselung lediglich ein Schlüssel verwendet, der von allen beteiligten Parteien im Rahmen einer sicheren Kommunikation gemeinsam genutzt wird. Dieser geheime Schlüssel wird sowohl zum Verschlüsseln als auch zum Entschlüsseln von Informationen verwendet. Der Schlüssel muss an alle Parteien sicher verteilt und von diesen sicher gespeichert werden, was nur schwer zu gewährleisten ist. Bei der Verschlüsselung mit einem öffentlichen Schlüssel ist der öffentliche Schlüssel nicht geheim, doch können die Nachrichten, die mit seiner Hilfe verschlüsselt werden, nur mithilfe des zugehörigen privaten Schlüssels entschlüsselt werden. Der private Schlüssel muss sicher gespeichert werden (beispielsweise in Ihrer Schlüsseldatenbank) oder er muss auf der Festplatte Ihres Computers verschlüsselt werden.

Die Algorithmen der öffentlichen Schlüssel allein gewährleisten noch keine sichere Kommunikation. Sie müssen die Identität aller Ihrer Kommunikationspartner auch weiterhin überprüfen. Für diese Authentifizierung verwendet SSL digitale Zertifikate. Wenn Sie jemandem Ihr digitales Zertifikat senden, erhält der Empfänger mit dem Zertifikat auch Ihren öffentlichen Schlüssel. Da Sie Ihr Zertifikat anhand Ihres privaten Schlüssels digital signiert haben, kann der Empfänger der Kommunikation Ihre Signatur nun mithilfe Ihres öffentlichen Schlüssels überprüfen. Die Gültigkeit des digitalen Zertifikats selbst wird von der entsprechenden Zertifizierungsstelle (CA) garantiert.

Unterstützte Cipher Suites

Während eines SSL-Handshakes ermitteln der Client und der Server, welche Cipher Suite für den Austausch von Daten verwendet werden soll. Bei einer Cipher Suite handelt es sich um eine Gruppe von Algorithmen, die zwecks Authentifizierung, Verschlüsselung und Datenintegrität verwendet werden.

Das DB2-Datenbanksystem verwendet GSKit im FIPS-Modus, um SSL-Unterstützung bereitzustellen. GSKit unterstützt die folgenden Cipher Suites:

- TLS_RSA_WITH_AES_256_CBC_SHA
- TLS_RSA_WITH_AES_128_CBC_SHA

- TLS_RSA_WITH_3DES_EDE_CBC_SHA

Die Namen der Cipher Suites geben jeweils den für die Authentifizierung, Verschlüsselung und Datenintegritätsprüfung verwendeten Algorithmus an. Die Cipher Suite TLS_RSA_WITH_AES_256_CBC_SHA beispielsweise verwendet RSA für die Authentifizierung, AES 256-Bit und CBC als Verschlüsselungsalgorithmen und SHA-1 als Hashfunktion für die Datenintegrität.

Während eines SSL-Handshakes wählt das DB2-Datenbanksystem automatisch die leistungsfähigste Cipher Suite aus, die sowohl vom Client als auch vom Server unterstützt wird. Wenn Sie wollen, dass der Server nur bestimmte Cipher Suites akzeptieren darf, können Sie den Konfigurationsparameter **ssl_cipherspecs** auf einen oder mehrere der folgenden Werte setzen:

- TLS_RSA_WITH_AES_256_CBC_SHA
- TLS_RSA_WITH_AES_128_CBC_SHA
- TLS_RSA_WITH_3DES_EDE_CBC_SHA
- Hierbei können Sie aus diesen drei Werten beliebige Kombinationen bilden. Um mehrere Werte zu definieren, trennen Sie die einzelnen Werte durch ein Komma, aber ohne Leerzeichen.
- Keine Angabe. In diesem Fall wird automatisch der leistungsfähigste der verfügbaren Algorithmen ausgewählt.

Für die Auswahl von Cipher Suites kann keine Rangfolge nach Priorität angegeben werden. Wird der Konfigurationsparameter **ssl_cipherspecs** definiert, wählt das DB2-Datenbanksystem die leistungsfähigste der verfügbaren Cipher Suites aus, und zwar unabhängig von der Reihenfolge, in der die Cipher Suites beim Definieren von **ssl_cipherspecs** angegeben werden.

Regeln für gebündelte Bibliotheken und Prozesse

Wenn DB2 for Linux, UNIX and Windows Software eines anderen Anbieters bündelt, für die GSKit erforderlich ist, oder wenn DB2 for Linux, UNIX and Windows von dieser Art von Software gebündelt wird, müssen bestimmte Regeln beachtet werden.

Bibliotheksregel

Wenn DB2 for Linux, UNIX and Windows Software eines anderen Anbieters bündelt, für die GSKit erforderlich ist, stellt diese Software Bibliotheken bereit, zu denen DB2 for Linux, UNIX and Windows eine Programmverbindung (Link) herstellt. Diese Bibliotheken müssen eine bestimmte Regel befolgen. Diese Regel wird als Bibliotheksregel bezeichnet.

Bibliotheksregel: Kurznamen verwenden

Wenn eine GSKit-Bibliothek dynamisch geladen wird, darf das aufrufende Programm der Ladefunktion nur den Basisdateinamen der GSKit-Bibliothek übergeben und nicht den Pfad.

Beispiel: `dlopen("libgsk8ssl_64.so", RTLD_NOW | RTLD_GLOBAL)` ist korrekt, während `dlopen("/usr/opt/ibm/gsk8_64/lib/libgsk8ssl_64.so", RTLD_NOW | RTLD_GLOBAL)` nicht korrekt ist.

Prozessregeln

Wenn die Software eines anderen Anbieters, für die GSKit erforderlich ist, DB2 for Linux, UNIX and Windows bündelt, stellt die Software eine Programmverbindung (Link) zum IBM Data Server-Client her. Die Software des anderen Anbieters muss eine bestimmte Regel befolgen. Diese Regel wird als Prozessregel bezeichnet.

Prozessregel: Umgebungssuchpfad definieren

Ein Prozess muss den Umgebungssuchpfad definieren, unter dem die GSKit-Bibliotheken zu finden sind. Der Prozess muss diesen Pfad definieren, damit die enthaltenen Bibliotheken die GSKit-Bibliotheken aus derselben Position laden können.

Unter AIX kann ein Prozess LIBPATH oder RPATH der ausführbaren Datei auf den Pfad der GSKit-Bibliotheken setzen. Im Fall von **setuid** und **setgid** kann der Prozess den Suchpfad von GSKit mithilfe von 'db2chglibpath' in RPATH der ausführbaren Datei einfügen. Nur dann können GSKit-Bibliotheken in dieser Position verwendet werden. Unter Linux, Sun und HP-UX kann ein Prozess LD_LIBRARY_PATH auf den Pfad der GSKit-Bibliotheken setzen. Im Fall von **setuid** und **setgid** kann der Prozess den Suchpfad von GSKit mithilfe von 'db2chglibpath' in RPATH der IBM Data Server-Client-Bibliothek einfügen. Nur dann können GSKit-Bibliotheken in dieser Position verwendet werden. Wenn ein Prozess beispielsweise in Serverinstanzen eine globale Version von GSKit oder in Client- oder Serverinstanzen eine eigene lokale Version von GSKit verwenden muss, kann RPATH mithilfe von 'db2chglibpath' geändert werden.

Symbolischer Link - Methode und Einschränkungen

Bei der Installation von DB2 for Linux, UNIX and Windows auf UNIX- und Linux-Plattformen werden auch lokale GSKit-Bibliotheken installiert. Diese Bibliotheken befinden sich in `<db2-installationspfad>/lib64/gskit_db2` oder `<db2-installationspfad>/lib32/gskit_db2`.

Während der Installation weiterer IBM Produkte wird möglicherweise eine weitere Kopie der GSKit-Bibliotheken installiert. Bei diesen Bibliotheken kann es sich je nach Produkt um lokale oder globale GSKit-Bibliotheken handeln. Werden DB2 for Linux, UNIX and Windows und ein weiteres IBM Produkt mit GSKit-Bibliotheken auf derselben Maschine installiert, kann es zu Interoperabilitätsproblemen kommen. Diese Interoperabilitätsprobleme können auftreten, weil GSKit in einem Prozess nur Bibliotheken aus einer einzigen GSKit-Quelle zulässt. Die Interoperabilitätsprobleme können zu unvorhersehbarem Verhalten und Laufzeitfehlern führen.

Wenn sichergestellt werden soll, dass in Situationen, in denen mehrere GSKit-Installationen auf derselben Maschine vorhanden sind, eine einzige Quelle von GSKit-Bibliotheken verwendet wird, kann ein symbolischer Link verwendet werden. Während der Erstinstallation von DB2 for Linux, UNIX and Windows erstellt das Installationsprogramm einen symbolischen Link zwischen `<db2-installationspfad>/lib64/gskit` oder `<db2-installationspfad>/lib32/gskit` und`<db2-installationspfad>/lib64/gskit_db2` oder `<db2-installationspfad>/lib32/gskit_db2`. Diese Position ist die Standardposition, aus der GSKit-Bibliotheken geladen werden. Produkte, die DB2 for Linux, UNIX and Windows bündeln und den symbolischen Link vom angegebenen Standardverzeichnis in das Bibliotheksverzeichnis einer anderen Kopie von GSKit ändern, müssen sicherstellen, dass das neu installierte GSKit dieselbe oder eine höhere Version aufweist. Diese Einschränkung gilt sowohl für globale als auch für lokale Bibliotheken. Bei einem Upgrade oder einer Aktualisierung von DB2 for Linux, UNIX and Windows wird der

symbolische Link beibehalten. Wenn die neu installierte Kopie über einen symbolischen Link verfügt, der auf die Standardposition verweist, wird der der älteren Installationskopie zugeordnete symbolische Link beibehalten. Wenn die neu installierte Kopie über einen symbolischen Link verfügt, der nicht auf die Standardposition verweist, wird der der neueren Installationskopie zugeordnete symbolische Link für die neuere Installationskopie verwendet. In diesem Zusammenhang gibt es einige Einschränkungen, da sich der symbolische Link `<db2-installationspfad>/lib64/gskit` oder `<db2-installationspfad>/lib32/gskit` im Pfad der DB2 for Linux, UNIX and Windows-Installationskopie befindet. Wenn beispielsweise zwei oder mehr Instanzen für eine DB2-Kopie erstellt werden, wirken sich die Änderungen an dem symbolischen Link auf alle Instanzen aus.

Im Lieferumfang von DB2 for Linux, UNIX and Windows ist GSKit Version 8.0.14.14 (8.0.15.1 unter Solaris x64) enthalten.

Beispiele

DB2 for Linux, UNIX and Windows bündelt den LDAP-Client. Die DB2 for Linux, UNIX and Windows-Prozesse befolgen die Prozessregel. Dazu wird der Umgebungssuchpfad über RPATH auf die lokale GSKit-Kopie gesetzt. LDAP-Clientbibliotheken laden GSKit-Bibliotheken aus derselben Position. LDAP-Clientbibliotheken, die Bibliotheksregeln befolgen, laden GSKit-Bibliotheken nach den entsprechenden Basisdateinamen, wenn GSKIT_LOCAL_INSTALL_MODE gesetzt ist.

Ein LDAP-Server bündelt DB2 for Linux, UNIX and Windows. LDAP-Prozesse befolgen die Prozessregel. Der Umgebungssuchpfad wird auf die globale Kopie von GSKit gesetzt und IBM Data Server-Client-Bibliotheken laden GSKit-Bibliotheken aus derselben Position. IBM Data Server-Client-Bibliotheken, die Bibliotheksregeln befolgen, laden GSKit-Bibliotheken nach den entsprechenden Basisdateinamen.

GSKit-Rückkehrcodes

In einigen Nachrichten des DB2-Datenbankmanagers wird möglicherweise ein Rückkehrcode aus IBM GSKit (Global Security Kit) angegeben.

Allgemeine GSKit-Rückkehrcodes

Tabelle 2. Allgemeine GSKit-Rückkehrcodes

Rückkehrcode (hexadezimal)	Rückkehrcode (dezimal)	Konstante	Erläuterung
0x00000000	0	GSK_OK	Die Task wurde erfolgreich ausgeführt. Dieser Rückkehrcode wird von jedem erfolgreich ausgeführten Funktionsaufruf ausgegeben.
0x00000001	1	GSK_INVALID_HANDLE	Die Umgebungs- oder SSL-Kennung ist nicht gültig. Die angegebene Kennung war nicht das Ergebnis eines erfolgreichen Funktionsaufrufs zum Öffnen (OPEN).
0x00000002	2	GSK_API_NOT_AVAILABLE	Die DDL (Dynamic Link Library) wurde entladen und ist nicht verfügbar. (Nur Windows)

Tabelle 2. Allgemeine GSKit-Rückkehrcodes (Forts.)

Rückkehrcode (hexadezimal)	Rückkehrcode (dezimal)	Konstante	Erläuterung
0x00000003	3	GSK_INTERNAL_ERROR	Interner Fehler. Melden Sie diesen Fehler an den Service.
0x00000004	4	GSK_INSUFFICIENT_STORAGE	Für die Ausführung der Operation steht kein ausreichender Speicher zur Verfügung.
0x00000005	5	GSK_INVALID_STATE	Die Kennung hat keinen gültigen Status für die Operation wie beispielsweise das zweimalige Ausführen einer Initialisierungsoperation für eine Kennung.
0x00000006	6	GSK_KEY_LABEL_NOT_FOUND	Der angegebene Schlüsselkennsatz wurde in der Schlüsseldatei nicht gefunden.
0x00000007	7	GSK_CERTIFICATE_NOT_AVAILABLE	Es wurde kein Zertifikat vom Partner empfangen.
0x00000008	8	GSK_ERROR_CERT_VALIDATION	Gültigkeitsfehler bei Zertifikat.
0x00000009	9	GSK_ERROR_CRYPTO	Fehler bei der Verarbeitung der Verschlüsselung.
0x0000000a	10	GSK_ERROR_ASN	Fehler bei der Prüfung von ASN-Feldern im Zertifikat.
0x0000000b	11	GSK_ERROR_LDAP	Fehler beim Herstellen der Verbindung zum LDAP-Server.
0x0000000c	12	GSK_ERROR_UNKNOWN_ERROR	Interner Fehler. Melden Sie diesen Fehler an den Service.
0x00000065	101	GSK_OPEN_CIPHER_ERROR	Interner Fehler. Melden Sie diesen Fehler an den Service.
0x00000066	102	GSK_KEYFILE_IO_ERROR	E/A-Fehler beim Lesen der Schlüsseldatei.
0x00000067	103	GSK_KEYFILE_INVALID_FORMAT	Die Schlüsseldatei hat ein ungültiges internes Format. Erstellen Sie die Schlüsseldatei erneut.
0x00000068	104	GSK_KEYFILE_DUPLICATE_KEY	Die Schlüsseldatei enthält zwei Einträge mit demselben Schlüssel. Entfernen Sie den doppelten Schlüssel mithilfe des Dienstprogramms 'iKeyman'.
0x00000069	105	GSK_KEYFILE_DUPLICATE_LABEL	Die Schlüsseldatei enthält zwei Einträge mit demselben Kennsatz. Entfernen Sie den doppelten Kennsatz mithilfe des Dienstprogramms 'iKeyman'.
0x0000006a	106	GSK_BAD_FORMAT_OR_ INVALID_PASSWORD	Das Schlüsseldateikennwort wird zur Integritätsprüfung verwendet. Entweder wurde die Schlüsseldatei beschädigt oder die Kennwort-ID ist falsch.

Tabelle 2. Allgemeine GSKit-Rückkehrcodes (Forts.)

Rückkehrcode (hexadezimal)	Rückkehrcode (dezimal)	Konstante	Erläuterung
0x0000006b	107	GSK_KEYFILE_CERT_EXPIRED	Der Standardschlüssel in der Schlüsseldatei hat ein abgelaufenes Zertifikat. Entfernen Sie abgelaufene Zertifikate mithilfe des Dienstprogramms 'iKeyman'.
0x0000006c	108	GSK_ERROR_LOAD_GSKLIB	Beim Laden einer der DDL-Dateien (Dynamic Link Library) von GSKit trat ein Fehler auf. Vergewissern Sie sich, dass GSKit ordnungsgemäß installiert wurde.
0x0000006d	109	GSK_PENDING_CLOSE_ERROR	Dieser Rückkehrcode gibt an, dass in einer GSKit-Umgebung versucht wird, eine Verbindung herzustellen, nachdem GSK_ENVIRONMENT_CLOSE_OPTIONS auf GSK_DELAYED_ENVIRONMENT_CLOSE gesetzt und die Funktion gsk_environment_close() aufgerufen wurde.
0x000000c9	201	GSK_NO_KEYFILE_PASSWORD	Weder das Kennwort noch der Name der Stashdatei wurde angegeben, sodass die Schlüsseldatei nicht initialisiert werden konnte.
0x000000ca	202	GSK_KEYRING_OPEN_ERROR	Die Schlüsseldatei oder der Microsoft-Zertifikatsspeicher kann nicht geöffnet werden. Der Pfad wurde nicht ordnungsgemäß angegeben oder die Dateiberechtigungen erlaubten nicht das Öffnen der Datei oder das Dateiformat ist falsch.
0x000000cb	203	GSK_RSA_TEMP_KEY_PAIR	Es kann kein temporäres Schlüsselpaar generiert werden. Melden Sie diesen Fehler an den Service.
0x000000cc	204	GSK_ERROR_LDAP_NO_SUCH_OBJECT	Ein Benutzernamensobjekt wurde angegeben, das nicht gefunden werden kann.
0x000000cd	205	GSK_ERROR_LDAP_INVALID_CREDENTIALS	Ein für eine LDAP-Abfrage verwendetes Kennwort ist nicht korrekt.
0x000000ce	206	GSK_ERROR_BAD_INDEX	Ein Index in der Liste von LDAP-Servern für Funktionsübernahme war nicht korrekt.
0x000000cd	207	GSK_ERROR_FIPS_NOT_SUPPORTED	Der Versuch, GSKit in den FIPS-Modus zu versetzen, ist fehlgeschlagen.

Tabelle 2. Allgemeine GSKit-Rückkehrcodes (Forts.)

Rückkehrcode (hexadezimal)	Rückkehrcode (dezimal)	Konstante	Erläuterung
0x0000012d	301	GSK_CLOSE_FAILED	Dieser Rückkehrcode gibt an, dass die Anforderung CLOSE zum Schließen der GSKit-Umgebung nicht ordnungsgemäß verarbeitet wurde. Dies liegt wahrscheinlich daran, dass versucht wurde, einen Befehl 'gsk_secure_socket*()' nach einem Aufruf 'gsk_close_environment()' auszuführen.
0x00000191	401	GSK_ERROR_BAD_DATE	Für das Systemdatum wurde ein ungültiger Wert festgelegt.
0x00000192	402	GSK_ERROR_NO_CIPHERS	Weder SSLV2 noch SSLV3 ist aktiviert.
0x00000193	403	GSK_ERROR_NO_CERTIFICATE	Das erforderliche Zertifikat vom Partner wurde nicht empfangen.
0x00000194	404	GSK_ERROR_BAD_CERTIFICATE	Das empfangene Zertifikat war nicht ordnungsgemäß formatiert.
0x00000195	405	GSK_ERROR_UNSUPPORTED_ CERTIFICATE_TYPE	Der empfangene Zertifikatstyp wurde nicht unterstützt.
0x00000196	406	GSK_ERROR_IO	Bei einer Operation zum Lesen oder Schreiben von Daten trat ein E/A-Fehler auf.
0x00000197	407	GSK_ERROR_BAD_KEYFILE_LABEL	Der angegebene Kennsatz in der Schlüsseldatei konnte nicht gefunden werden.
0x00000198	408	GSK_ERROR_BAD_KEYFILE_ PASSWORD	Das angegebene Schlüsseldateikennwort ist falsch. Die Schlüsseldatei konnte nicht verwendet werden. Die Schlüsseldatei kann möglicherweise auch beschädigt sein.
0x00000199	409	GSK_ERROR_BAD_KEY_LEN_ FOR_EXPORT	In einer eingeschränkten Verschlüsselungsumgebung wird die Schlüsselgröße nicht unterstützt, weil sie zu lang ist.
0x0000019a	410	GSK_ERROR_BAD_MESSAGE	Eine vom Partner empfangene SSL-Nachricht war nicht ordnungsgemäß formatiert.
0x0000019b	411	GSK_ERROR_BAD_MAC	Der Nachrichtenauthentifizierungscode (MAC) wurde nicht erfolgreich geprüft.
0x0000019c	412	GSK_ERROR_UNSUPPORTED	Nicht unterstütztes SSL-Protokoll oder nicht unterstützter Zertifikatstyp.
0x0000019d	413	GSK_ERROR_BAD_CERT_SIG	Das empfangene Zertifikat enthielt eine falsche Signatur.

Tabelle 2. Allgemeine GSKit-Rückkehrcodes (Forts.)

Rückkehrcode (hexadezimal)	Rückkehrcode (dezimal)	Konstante	Erläuterung
0x0000019e	414	GSK_ERROR_BAD_CERT	Das vom Partner empfangene Zertifikat war nicht ordnungsgemäß formatiert.
0x0000019f	415	GSK_ERROR_BAD_PEER	Das vom Partner empfangene SSL-Protokoll war ungültig.
0x000001a0	416	GSK_ERROR_PERMISSION_DENIED	Melden Sie diesen internen Fehler an den Service.
0x000001a1	417	GSK_ERROR_SELF_SIGNED	Das selbst signierte Zertifikat ist nicht gültig.
0x000001a2	418	GSK_ERROR_NO_READ_FUNCTION	Die Leseoperation ist fehlgeschlagen. Melden Sie diesen internen Fehler an den Service.
0x000001a3	419	GSK_ERROR_NO_WRITE_FUNCTION	Die Schreiboperation ist fehlgeschlagen. Melden Sie diesen internen Fehler an den Service.
0x000001a4	420	GSK_ERROR_SOCKET_CLOSED	Der Partner hat das Socket vor Abschluss des Protokolls geschlossen.
0x000001a5	421	GSK_ERROR_BAD_V2_CIPHER	Die angegebene V2-Verschlüsselung ist nicht gültig.
0x000001a6	422	GSK_ERROR_BAD_V3_CIPHER	Die angegebene V3-Verschlüsselung ist nicht gültig.
0x000001a7	423	GSK_ERROR_BAD_SEC_TYPE	Melden Sie diesen internen Fehler an den Service.
0x000001a8	424	GSK_ERROR_BAD_SEC_ TYPE_COMBINATION	Melden Sie diesen internen Fehler an den Service.
0x000001a9	425	GSK_ERROR_HANDLE_ CREATION_FAILED	Die Kennung konnte nicht erstellt werden. Melden Sie diesen internen Fehler an den Service.
0x000001aa	426	GSK_ERROR_INITIALIZATION_ FAILED	Initialisierung fehlgeschlagen. Melden Sie diesen internen Fehler an den Service.
0x000001ab	427	GSK_ERROR_LDAP_NOT_AVAILABLE	Beim Prüfen eines Zertifikats bestand kein Zugriff auf das angegebene LDAP-Verzeichnis.
0x000001ac	428	GSK_ERROR_NO_PRIVATE_KEY	Der angegebene Schlüssel enthielt keinen privaten Schlüssel.
0x000001ad	429	GSK_ERROR_PKCS11_ LIBRARY_NOTLOADED	Ein Versuch, die angegebene gemeinsam genutzte PKCS11-Bibliothek zu laden, ist fehlgeschlagen.
0x000001ae	430	GSK_ERROR_PKCS11_TOKEN_ LABELMISMATCH	Der PKCS #11-Treiber konnte das vom aufrufenden Programm angegebene Token nicht finden.
0x000001af	431	GSK_ERROR_PKCS11_TOKEN_ NOTPRESENT	Im Slot ist kein PKCS #11-Token vorhanden.

Tabelle 2. Allgemeine GSKit-Rückkehrcodes (Forts.)

Rückkehrcode (hexadezimal)	Rückkehr-code (dezimal)	Konstante	Erläuterung
0x000001b0	432	GSK_ERROR_PKCS11_TOKEN_ BADPASSWORD	Das Kennwort/die PIN für den Zugriff auf das PKCS #11-Token ist ungültig.
0x000001b1	433	GSK_ERROR_INVALID_V2_HEADER	Der empfangene SSL-Header war für SSLV2 nicht ordnungsgemäß formatiert.
0x000001b2	434	GSK_CSP_OPEN_ERROR	Auf den hardwarebasierten Verschlüsselungsserviceanbieter (CSP) konnte nicht zugegriffen werden. Entweder ist der angegebene CSP-Name nicht im System registriert oder der CSP-Name ist zwar registriert, aber der Zertifikatsspeicher konnte nicht geöffnet werden.
0x000001b3	435	GSK_CONFLICTING_ATTRIBUTE_ SETTING	Konflikt bei den Attributeinstellungen zwischen PKCS11, der CMS-Schlüsseldatenbank und der Microsoft-Verschlüsselungs-API.
0x000001b4	436	GSK_UNSUPPORTED_PLATFORM	Die angeforderte Funktion wird auf der von der Anwendung ausgeführten Plattform nicht unterstützt. Beispiel: Die Microsoft-Verschlüsselungs-API wird ausschließlich auf der Windows 2000-Plattform unterstützt.
0x000001b5	437	GSK_ERROR_INCORRECT_ SESSION_TYPE	Von der Rückruffunktion zum Zurücksetzen des Sitzungstyps wurde ein falscher Wert zurückgegeben. Für GSKit ist nur GSK_SERVER_SESSION oder GSK_SERVER_SESSION_ WITH_CL_AUTH zulässig.
0x000001f5	501	GSK_INVALID_BUFFER_SIZE	Die Puffergröße ist negativ oder null.
0x000001f6	502	GSK_WOULD_BLOCK	Verwendung mit nicht geblockter Eingabe/Ausgabe.
0x00000259	601	GSK_ERROR_NOT_SSLV3	Für 'reset_cipher' ist SSLV3 erforderlich, die Verbindung verwendet jedoch SSLV2.
0x0000025a	602	GSK_MISC_INVALID_ID	Für den Funktionsaufruf 'gsk_secure_soc_misc' wurde eine ungültige Kennung angegeben.

Tabelle 2. Allgemeine GSKit-Rückkehrcodes (Forts.)

Rückkehrcode (hexadezimal)	Rückkehrcode (dezimal)	Konstante	Erläuterung
0x000002bd	701	GSK_ATTRIBUTE_INVALID_ID	Der Funktionsaufruf wies eine ungültige Kennung auf. Dies kann möglicherweise auch daran liegen, dass eine Umgebungskennung angegeben wurde, obwohl eine Kennung für eine SSL-Verbindung erforderlich war.
0x000002be	702	GSK_ATTRIBUTE_INVALID_LENGTH	Das Attribut hat eine negative Länge. Dies ist ungültig.
0x000002bf	703	GSK_ATTRIBUTE_INVALID_ ENUMERATION	Der Aufzählungswert ist für den angegebenen Aufzählungstyp ungültig.
0x000002c0	704	GSK_ATTRIBUTE_INVALID_ SID_CACHE	Ungültige Parameterliste zum Ersetzen der Cache-Routinen der Sitzungs-ID (SID).
0x000002c1	705	GSK_ATTRIBUTE_INVALID_ NUMERIC_VALUE	Der angegebene Wert ist für das betreffende numerische Attribut ungültig.
0x000002c2	706	GSK_CONFLICTING_VALIDATION_ SETTING	Für die zusätzliche Zertifikatsprüfung wurden Parameter gesetzt, die miteinander in Konflikt stehen.
0x000002c3	707	GSK_AES_UNSUPPORTED	Die Verschlüsselungsspezifikation enthielt eine AES-Verschlüsselung (Advanced Encryption Standard), die im ausgeführten System nicht unterstützt wird.
0x000002c4	708	GSK_PEERID_LENGTH_ERROR	Die Länge der Peer-ID ist falsch. Die Länge muss kleiner-gleich 16 Byte sein.
0x000002c5	709	GSK_CIPHER_INVALID_WHEN_ FIPS_MODE_OFF	Die angegebene Verschlüsselung ist bei inaktiviertem FIPS-Modus nicht zulässig.
0x000002c6	710	GSK_CIPHER_INVALID_WHEN_ FIPS_MODE_ON	Im FIPS-Modus wurde keine für FIPS genehmigte Verschlüsselung ausgewählt.
0x00000641	1601	GSK_TRACE_STARTED	Der Trace wurde erfolgreich gestartet.
0x00000642	1602	GSK_TRACE_STOPPED	Der Trace wurde erfolgreich gestoppt.
0x00000643	1603	GSK_TRACE_NOT_STARTED	Es wurde bisher keine Tracedatei gestartet. Daher kann auch keine Tracedatei gestoppt werden.
0x00000644	1604	GSK_TRACE_ALREADY_STARTED	Die Tracedatei wurde bereits gestartet und kann daher nicht erneut gestartet werden.

Tabelle 2. Allgemeine GSKit-Rückkehrcodes (Forts.)

Rückkehrcode (hexadezimal)	Rückkehr-code (dezimal)	Konstante	Erläuterung
0x00000645	1605	GSK_TRACE_OPEN_FAILED	Die Tracedatei kann nicht geöffnet werden. Der erste Parameter von 'gsk_start_trace()' muss ein gültiger Dateiname mit einem vollständigen Pfad sein.

Rückkehrcodes für das Schlüsselmanagement

Tabelle 3. Rückkehrcodes für das Schlüsselmanagement

Rückkehrcode (hexadezimal)	Rückkehrcode (dezimal)	Konstante
0x00000000	0	GSKKM_ERR_OK
0x00000000	0	GSKKM_ERR_SUCCESS
0x00000001	1	GSKKM_ERR_UNKNOWN
0x00000002	2	GSKKM_ERR_ASN
0x00000003	3	GSKKM_ERR_ASN_INITIALIZATION
0x00000004	4	GSKKM_ERR_ASN_PARAMETER
0x00000005	5	GSKKM_ERR_DATABASE
0x00000006	6	GSKKM_ERR_DATABASE_OPEN
0x00000007	7	GSKKM_ERR_DATABASE_RE_OPEN
0x00000008	8	GSKKM_ERR_DATABASE_CREATE
0x00000009	9	GSKKM_ERR_DATABASE_ALREADY_EXISTS
0x0000000a	10	GSKKM_ERR_DATABASE_DELETE
0x0000000b	11	GSKKM_ERR_DATABASE_NOT_OPENED
0x0000000c	12	GSKKM_ERR_DATABASE_READ
0x0000000d	13	GSKKM_ERR_DATABASE_WRITE
0x0000000e	14	GSKKM_ERR_DATABASE_VALIDATION
0x0000000f	15	GSKKM_ERR_DATABASE_INVALID_VERSION
0x00000010	16	GSKKM_ERR_DATABASE_INVALID_PASSWORD
0x00000011	17	GSKKM_ERR_DATABASE_INVALID_FILE_TYPE
0x00000012	18	GSKKM_ERR_DATABASE_CORRUPTION
0x00000013	19	GSKKM_ERR_DATABASE_PASSWORD_ CORRUPTION
0x00000014	20	GSKKM_ERR_DATABASE_KEY_INTEGRITY
0x00000015	21	GSKKM_ERR_DATABASE_DUPLICATE_KEY
0x00000016	22	GSKKM_ERR_DATABASE_DUPLICATE_ KEY_RECORD_ID
0x00000017	23	GSKKM_ERR_DATABASE_DUPLICATE_ KEY_LABEL
0x00000018	24	GSKKM_ERR_DATABASE_DUPLICATE_ KEY_SIGNATURE
0x00000019	25	GSKKM_ERR_DATABASE_DUPLICATE_ KEY_UNSIGNED_CERTIFICATE
0x0000001a	26	GSKKM_ERR_DATABASE_DUPLICATE_KEY_ ISSUER_AND_SERIAL_NUMBER
0x0000001b	27	GSKKM_ERR_DATABASE_DUPLICATE_KEY_ SUBJECT_PUBLIC_KEY_INFO
0x0000001c	28	GSKKM_ERR_DATABASE_DUPLICATE_KEY_ UNSIGNED_CRL
0x0000001d	29	GSKKM_ERR_DATABASE_DUPLICATE_LABEL
0x0000001e	30	GSKKM_ERR_DATABASE_PASSWORD_ ENCRYPTION

Tabelle 3. Rückkehrcodes für das Schlüsselmanagement (Forts.)

Rückkehrcode (hexadezimal)	Rückkehrcode (dezimal)	Konstante
0x0000001f	31	GSKKM_ERR_DATABASE_LDAP
0x00000020	32	GSKKM_ERR_CRYPTO
0x00000021	33	GSKKM_ERR_CRYPTO_ENGINE
0x00000022	34	GSKKM_ERR_CRYPTO_ALGORITHM
0x00000023	35	GSKKM_ERR_CRYPTO_SIGN
0x00000024	36	GSKKM_ERR_CRYPTO_VERIFY
0x00000025	37	GSKKM_ERR_CRYPTO_DIGEST
0x00000026	38	GSKKM_ERR_CRYPTO_PARAMETER
0x00000027	39	GSKKM_ERR_CRYPTO_UNSUPPORTED_ ALGORITHM
0x00000028	40	GSKKM_ERR_CRYPTO_INPUT_GREATER_ THAN_MODULUS
0x00000029	41	GSKKM_ERR_CRYPTO_UNSUPPORTED_ MODULUS_SIZE
0x0000002a	42	GSKKM_ERR_VALIDATION
0x0000002b	43	GSKKM_ERR_VALIDATION_KEY
0x0000002c	44	GSKKM_ERR_VALIDATION_DUPLICATE_ EXTENSIONS
0x0000002d	45	GSKKM_ERR_VALIDATION_KEY_WRONG_ VERSION
0x0000002e	46	GSKKM_ERR_VALIDATION_KEY_ EXTENSIONS_REQUIRED
0x0000002f	47	GSKKM_ERR_VALIDATION_KEY_VALIDITY
0x00000030	48	GSKKM_ERR_VALIDATION_KEY_VALIDITY_ PERIOD
0x00000031	49	GSKKM_ERR_VALIDATION_KEY_VALIDITY_ PRIVATE_KEY_USAGE
0x00000032	50	GSKKM_ERR_VALIDATION_KEY_ISSUER_ NOT_FOUND
0x00000033	51	GSKKM_ERR_VALIDATION_KEY_MISSING_ REQUIRED_EXTENSIONS
0x00000034	52	GSKKM_ERR_VALIDATION_KEY_BASIC_ CONSTRAINTS
0x00000035	53	GSKKM_ERR_VALIDATION_KEY_SIGNATURE
0x00000036	54	GSKKM_ERR_VALIDATION_KEY_ROOT_KEY_ NOT_TRUSTED
0x00000037	55	GSKKM_ERR_VALIDATION_KEY_IS_REVOKED
0x00000038	56	GSKKM_ERR_VALIDATION_KEY_AUTHORITY_ KEY_IDENTIFIER
0x00000039	57	GSKKM_ERR_VALIDATION_KEY_PRIVATE_KEY_ USAGE_PERIOD
0x0000003a	58	GSKKM_ERR_VALIDATION_SUBJECT_ ALTERNATIVE_NAME

Tabelle 3. Rückkehrcodes für das Schlüsselmanagement (Forts.)

Rückkehrcode (hexadezimal)	Rückkehrcode (dezimal)	Konstante
0x0000003b	59	GSKKM_ERR_VALIDATION_ISSUER_ ALTERNATIVE_NAME
0x0000003c	60	GSKKM_ERR_VALIDATION_KEY_USAGE
0x0000003d	61	GSKKM_ERR_VALIDATION_KEY_ UNKNOWN_CRITICAL_EXTENSION
0x0000003e	62	GSKKM_ERR_VALIDATION_KEY_PAIR
0x0000003f	63	GSKKM_ERR_VALIDATION_CRL
0x00000040	64	GSKKM_ERR_MUTEX
0x00000041	65	GSKKM_ERR_PARAMETER
0x00000042	66	GSKKM_ERR_NULL_PARAMETER
0x00000043	67	GSKKM_ERR_NUMBER_SIZE
0x00000044	68	GSKKM_ERR_OLD_PASSWORD
0x00000045	69	GSKKM_ERR_NEW_PASSWORD
0x00000046	70	GSKKM_ERR_PASSWORD_EXPIRATION_TIME
0x00000047	71	GSKKM_ERR_THREAD
0x00000048	72	GSKKM_ERR_THREAD_CREATE
0x00000049	73	GSKKM_ERR_THREAD_WAIT_FOR_EXIT
0x0000004a	74	GSKKM_ERR_IO
0x0000004b	75	GSKKM_ERR_LOAD
0x0000004c	76	GSKKM_ERR_PKCS11
0x0000004d	77	GSKKM_ERR_NOT_INITIALIZED
0x0000004e	78	GSKKM_ERR_DB_TABLE_CORRUPTED
0x0000004f	79	GSKKM_ERR_MEMORY_ALLOCATE
0x00000050	80	GSKKM_ERR_UNSUPPORTED_OPTION
0x00000051	81	GSKKM_ERR_GET_TIME
0x00000052	82	GSKKM_ERR_CREATE_MUTEX
0x00000053	83	GSKKM_ERR_CMDCAT_OPEN
0x00000054	84	GSKKM_ERR_ERRCAT_OPEN
0x00000055	85	GSKKM_ERR_FILENAME_NULL
0x00000056	86	GSKKM_ERR_FILE_OPEN
0x00000057	87	GSKKM_ERR_FILE_OPEN_TO_READ
0x00000058	88	GSKKM_ERR_FILE_OPEN_TO_WRITE
0x00000059	89	GSKKM_ERR_FILE_OPEN_NOT_EXIST
0x0000005a	90	GSKKM_ERR_FILE_OPEN_NOT_ALLOWED
0x0000005b	91	GSKKM_ERR_FILE_WRITE
0x0000005c	92	GSKKM_ERR_FILE_REMOVE
0x0000005d	93	GSKKM_ERR_BASE64_INVALID_DATA
0x0000005e	94	GSKKM_ERR_BASE64_INVALID_MSGTYPE
0x0000005f	95	GSKKM_ERR_BASE64_ENCODING
0x00000060	96	GSKKM_ERR_BASE64_DECODING

Tabelle 3. Rückkehrcodes für das Schlüsselmanagement (Forts.)

Rückkehrcode (hexadezimal)	Rückkehrcode (dezimal)	Konstante
0x00000061	97	GSKKM_ERR_DN_TAG_NULL
0x00000062	98	GSKKM_ERR_DN_CN_NULL
0x00000063	99	GSKKM_ERR_DN_C_NULL
0x00000064	100	GSKKM_ERR_INVALID_DB_HANDLE
0x00000065	101	GSKKM_ERR_KEYDB_NOT_EXIST
0x00000066	102	GSKKM_ERR_KEYPAIRDB_NOT_EXIST
0x00000067	103	GSKKM_ERR_PWDFILE_NOT_EXIST
0x00000068	104	GSKKM_ERR_PASSWORD_CHANGE_MATCH
0x00000069	105	GSKKM_ERR_KEYDB_NULL
0x0000006a	106	GSKKM_ERR_REQKEYDB_NULL
0x0000006b	107	GSKKM_ERR_KEYDB_TRUSTCA_NULL
0x0000006c	108	GSKKM_ERR_REQKEY_FOR_CERT_NULL
0x0000006d	109	GSKKM_ERR_KEYDB_PRIVATE_KEY_NULL
0x0000006e	110	GSKKM_ERR_KEYDB_DEFAULT_KEY_NULL
0x0000006f	111	GSKKM_ERR_KEYREC_PRIVATE_KEY_NULL
0x00000070	112	GSKKM_ERR_KEYREC_CERTIFICATE_NULL
0x00000071	113	GSKKM_ERR_CRLS_NULL
0x00000072	114	GSKKM_ERR_INVALID_KEYDB_NAME
0x00000073	115	GSKKM_ERR_UNDEFINED_KEY_TYPE
0x00000074	116	GSKKM_ERR_INVALID_DN_INPUT
0x00000075	117	GSKKM_ERR_KEY_GET_BY_LABEL
0x00000076	118	GSKKM_ERR_LABEL_LIST_CORRUPT
0x00000077	119	GSKKM_ERR_INVALID_PKCS12_DATA
0x00000078	120	GSKKM_ERR_PKCS12_PWD_CORRUPTION
0x00000079	121	GSKKM_ERR_EXPORT_TYPE
0x0000007a	122	GSKKM_ERR_PBE_ALG_UNSUPPORT
0x0000007b	123	GSKKM_ERR_KYR2KDB
0x0000007c	124	GSKKM_ERR_KDB2KYR
0x0000007d	125	GSKKM_ERR_ISSUING_CERTIFICATE
0x0000007e	126	GSKKM_ERR_FIND_ISSUER_CHAIN
0x0000007f	127	GSKKM_ERR_WEBDB_DATA_BAD_FORMAT
0x00000080	128	GSKKM_ERR_WEBDB_NOTHING_TO_WRITE
0x00000081	129	GSKKM_ERR_EXPIRE_DAYS_TOO_LARGE
0x00000082	130	GSKKM_ERR_PWD_TOO_SHORT
0x00000083	131	GSKKM_ERR_PWD_NO_NUMBER
0x00000084	132	GSKKM_ERR_PWD_NO_CONTROL_KEY
0x00000085	133	GSKKM_ERR_SIGNATURE_ALGORITHM
0x00000086	134	GSKKM_ERR_INVALID_DATABASE_TYPE
0x00000087	135	GSKKM_ERR_SECONDARY_KEYDB_TO_OTHER

Tabelle 3. Rückkehrcodes für das Schlüsselmanagement (Forts.)

Rückkehrcode (hexadezimal)	**Rückkehrcode (dezimal)**	**Konstante**
0x00000088	136	GSKKM_ERR_NO_SECONDARY_KEYDB
0x00000089	137	GSKKM_ERR_CRYPTOGRAPHIC_TOKEN_ LABEL_NOT_EXIST
0x0000008a	138	GSKKM_ERR_CRYPTOGRAPHIC_TOKEN_ PASSWORD_REQUIRED
0x0000008b	139	GSKKM_ERR_CRYPTOGRAPHIC_TOKEN_ PASSWORD_NOT_REQUIRED
0x0000008c	140	GSKKM_ERR_CRYPTOGRAPHIC_TOKEN_ LIBRARY_NOT_LOADED
0x0000008d	141	GSKKM_ERR_CRYPTOGRAPHIC_TOKEN_ NOT_SUPPORT
0x0000008e	142	GSKKM_ERR_CRYPTOGRAPHIC_TOKEN_ FUNCTION_FAILED
0x0000008f	143	GSKKM_ERR_LDAP_USER_NOT_FOUND
0x00000090	144	GSKKM_ERR_LDAP_INVALID_PASSWORD
0x00000091	145	GSKKM_ERR_LDAP_QUERY_ENTRY_FAILED
0x00000092	146	GSKKM_ERR_INVALID_CERT_CHAIN
0x00000093	147	GSKKM_ERR_CERT_ROOT_NOT_TRUSTED
0x00000094	148	GSKKM_ERR_CERT_REVOKED
0x00000095	149	GSKKM_ERR_CRYPTOGRAPHIC_OBJECT_ FUNCTION_FAILED
0x00000096	150	GSKKM_ERR_NO_AVAILABLE_CRL_ DATASOURCE
0x00000097	151	GSKKM_ERR_NO_TOKEN_PRESENT
0x00000098	152	GSKKM_ERR_FIPS_NOT_SUPPORTED
0x00000099	153	GSKKM_ERR_FIPS_CONFLICT_SETTING
0x0000009a	154	GSKKM_ERR_PASSWORD_STRENGTH_FAILED

IBM Database Encryption Expert für die Verschlüsselung von ruhenden Daten (Data At Rest, DAR)

IBM Database Encryption Expert ist eine umfassende Softwarelösung für Datensicherheit, die bei Verwendung zusammen mit der nativen DB2-Sicherheit einen effektiven Schutz der Daten und der Datenbankanwendung vor einem breiten Spektrum an Sicherheitsrisiken bietet.

Database Encryption Expert hilft Unternehmen dabei sicherzustellen, dass private und vertrauliche Daten wirksam geschützt werden und mit den Verordnungen und gesetzlichen Bestimmungen konform gehen. Die Hauptvorteile von Database Encryption Expert sehen wie folgt aus:

- Bewährte, effektive Datensicherheit für das DB2-Datenbanksystem
- Schutz von Live-Dateien, Konfigurationsdateien, Protokolldateien und Backupdaten
- Transparenz für Anwendungs-, Datenbank- und Speicherumgebungen
- Einheitliches Richtlinien- und Schlüsselmanagement für den Schutz von Daten in Online- wie auch in Offline-Umgebungen
- Erfüllung von Leistungsanforderungen

Database Encryption Expert ermöglicht Ihnen das Verschlüsseln von Offline-Datenbankbackups sowie das Verschlüsseln von Online-Datenbankdateien (Live-Datenbankdateien). Hierbei handelt es sich um die Verschlüsselung von Daten auf dem Datenträger, die bisweilen auch als ruhende Daten (Data At Rest, DAS) bezeichnet werden, im Gegensatz zur Verschlüsselung von momentan aktiven oder übertragenen Daten (Data In Flight, DIF).

- Bei Backups werden die Daten während der Sicherung verschlüsselt, d. h. die Daten werden auf der Sicherungseinheit verschlüsselt. Sollte eine Recovery der Daten erforderlich werden, erkennt der Wiederherstellungsserver, dass die Daten verschlüsselt sind und führt eine Entschlüsselung der Daten durch.
- Bei Datenbankdateien werden die Datendateien des Betriebssystems verschlüsselt, die die Daten aus der DB2-Datenbank enthalten. Dadurch werden die Datendateien vor nicht berechtigten Benutzern geschützt, die versuchen, die unaufbereitete Datenbankdatei zu lesen.

Database Encryption Expert ist für Benutzer, Datenbanken, Anwendungen und Speicher transparent. Es sind keine Codeänderungen oder Änderungen an der vorhandenen Infrastruktur erforderlich. Database Encryption Expert kann Daten in jeder beliebigen Speicherumgebung schützen, während die Benutzer weiterhin wie zuvor auf die Daten zugreifen.

Database Encryption Expert kann Datenbankanwendungen schützen, weil das Produkt Änderungen an ausführbaren Dateien, Konfigurationsdateien, Bibliotheken etc. verhindern und dadurch Attacken auf die betreffende Anwendung verhindern kann.

Anmerkung: Database Encryption Expert kann nicht in DB2 pureScale-Umgebungen verwendet werden.

Architektur von Database Encryption Expert

Database Encryption Expert ist eine Gruppe aus Agenten- und Server-Softwarepaketen, die Sie mithilfe einer webbasierten Benutzerschnittstelle und von Befehlszeilendienstprogrammen verwalten können. Database Encryption Expert Administrator konfiguriert Sicherheitsrichtlinien, die regeln, wie Sicherheit und Verschlüsselung implementiert werden.

Je nachdem, wie diese Sicherheitsrichtlinien definiert sind, verschlüsselt der Database Encryption Expert-Backupagent die DB2-Backups und der Database Encryption Expert-Dateisystemagent die DB2-Datendateien.

Der Encryption Expert-Sicherheitsserver speichert die Sicherheitsrichtlinien, die Verschlüsselungsschlüssel und die Ereignisprotokolldateien. Sicherheitsrichtlinien enthalten Gruppen von Sicherheitsregeln, die eingehalten werden müssen. Von der Einhaltung dieser Regeln hängt ab, ob auf die Daten zugegriffen werden kann oder ob der Zugriff zurückgewiesen wird. Jede Sicherheitsregel wertet aus, wer wann und wie auf welche geschützten Daten zugreift, und anhand dieser Auswertung lässt der Sicherheitsserver den Zugriff auf die betreffenden Daten entweder zu oder nicht.

Abb. 4 veranschaulicht die Architektur von Database Encryption Expert.

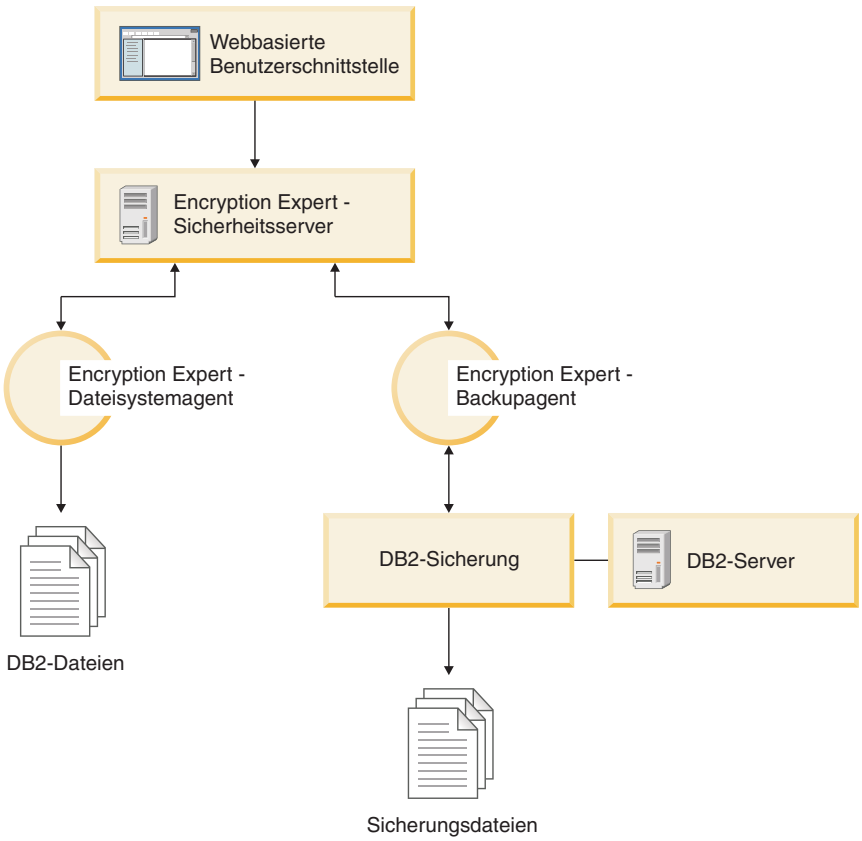


Abbildung 4. Architektur von Database Encryption Expert

Dateisystemagent

Der Prozess des Database Encryption Expert-Dateisystemagenten ist immer im Hintergrund aktiv. Der Agent fängt alle Versuche ab, auf geschützte Datendateien, Verzeichnisse oder ausführbare Dateien zuzugreifen. Der Database Encryption Expert-Dateisystemagent leitet den Zugriffsversuch an den Sicherheitsserver weiter, und in Abhängigkeit von der jeweils angewandten Richtlinie wird der Zugriff vom Sicherheitsserver gewährt oder zurückgewiesen.

Der Schutz von Database Encryption Expert ermöglicht jedoch nicht nur das Zulassen oder Zurückweisen von Zugriffen auf Dateien, sondern darüber hinaus auch das Verschlüsseln von Dateien. Hierbei wird lediglich der Dateiinhalt verschlüsselt, während die Dateimetadaten unverändert bleiben. Daher ist es nicht erforderlich, eine verschlüsselte Datei zu entschlüsseln, wenn Sie lediglich den Namen, den Typ, die Zeitmarken etc. der Datei anzeigen wollen. Dadurch können Datenmanagementanwendungen ihre Funktionen ausführen, ohne den Dateiinhalt zugänglich zu machen. So können beispielsweise Backup-Manager bestimmte Daten sichern, ohne deren Inhalt einsehen zu können.

Wenn ein nicht berechtigter Benutzer auf eine verschlüsselte Datei zugreift, ist ihr Inhalt ohne entsprechende Zugriffsgenehmigung des Sicherheitsservers und ohne die entsprechenden Verschlüsselungsschlüssel nutzlos. Benutzer hingegen, die über die korrekten Richtlinien und Berechtigungen verfügen, merken nicht einmal, dass eine Verschlüsselung und Entschlüsselung stattfinden.

Backupagent

Sämtliche Datenbankbackupfunktionen, die normalerweise vom DB2-Backup-API-System ausgeführt werden, werden vom Database Encryption Expert-Server unterstützt, einschließlich der nativen Datenbankkomprimierung. Abgesehen von einem zusätzlichen Befehlszeilenargument merken DB2-Backupbediener nicht, dass Database Encryption Expert eingreift. Database Encryption Expert führt Backups und Restores sowohl für statische ruhende Daten (Data At Rest, DAR) als auch für aktive Online-Daten durch.

Eine Basiskonfiguration von Backups und Restores wird unterstützt. In dieser Basiskonfiguration werden Daten mit einem Server und mehreren Agenten verschlüsselt und gesichert. Die Entschlüsselung und Wiederherstellung der Daten erfolgt auf einem Agenten, der für denselben Server konfiguriert ist, der ursprünglich zum Erstellen des Backups verwendet wurde.

Konfigurationen an einem einzigen und an mehreren Standorten werden ebenfalls für Backups und Restores unterstützt. In einem Szenario mit einem einzelnen Standort werden Konfigurationsdaten in einem einzigen Rechenzentrum auf mehreren Sicherheitsservern gespiegelt. In einem Szenario mit mehreren Standorten werden Backups auf verschiedenen Encryption Expert-Servern in verschiedenen Rechenzentren wiederhergestellt.

Prüfprotokollierung

Die Aktivität von Database Encryption Expert-Agenten wird eng überwacht und mithilfe einer zentralen Prüfprotokollierungseinrichtung aufgezeichnet. Alle überprüfbaren Ereignisse (einschließlich Backups, Restores und Operationen des Sicherheitsadministrators) können protokolliert werden. Dies umfasst auch Database Encryption Expert-Systemereignisse (wie beispielsweise das Initialisieren, Herunterfahren und Neustarten des Systems) sowie das Herstellen und Unterbrechen von Verbindungen zwischen unterschiedlichen Database Encryption Expert-Komponenten.

Database Encryption Expert-Dokumentation

Weitere Informationen zu Database Encryption Expert finden Sie auf der folgenden Webseite: http://publib.boulder.ibm.com/infocenter/mptoolic/v1r0/topic/com.ibm.db2tools.eet.doc.ug/eetwelcome.htm

Datenbankverschlüsselung mit AIX EFS

Wird DB2 Enterprise Server Edition unter dem Betriebssystem AIX ausgeführt, können Sie eine verschlüsselte Datenbank mit AIX EFS (Encrypted File System, verschlüsseltes Dateisystem) einrichten. Ausführliche Informationen zu EFS können Sie der AIX-Dokumentation entnehmen.

Anmerkung: Wenn Sie in einer Umgebung mit partitionierten Datenbanken arbeiten, sollten Sie EFS nur verwenden, wenn die Datenbank sich auf einer einzigen Datenbankpartition befindet.

Sie können die Betriebssystemdateien, die die Daten in Datenbanktabellen enthalten, verschlüsseln, indem Sie das verfügbare EFS mit dem Dateisystem JFS2 verwenden.

Richten Sie die Verschlüsselung wie folgt ein:

1. Aktivieren Sie EFS auf dem System.
2. Laden Sie die Schlüsselspeicher für das Benutzerkonto, unter dem die DB2-Datenbankdämonen ausgeführt werden.
3. Aktivieren Sie EFS für das Datenbankdateisystem.
4. Bestimmen Sie die zu verschlüsselnde Betriebssystemdatei.
5. Verschlüsseln Sie die Datei mit den Datenbanktabellen, die den EFS-Schutz benötigen.

Aktivieren von EFS für das System

Vor der EFS-Aktivierung muss die Dateigruppe `clic.rte` installiert werden. Das Installationsimage `clic.rte` finden Sie auf der CD mit dem Erweiterungspaket.

Führen Sie zum Aktivieren von EFS für das System den folgenden Befehl als Root aus:

```
% efsenable -a
```

Sie müssen den Befehl **efsenable** nur einmalig ausführen.

Laden der Schlüsselspeicher

Bei den folgenden Konfigurationsbeispielen handelt es sich bei dem DB2-Benutzerkonto, unter dem die Datenbankdämonen ausgeführt werden, um das Konto 'abst'. Der Benutzer 'abst' und jede Gruppe, der 'abst' angehört, müssen über einen Schlüsselspeicher verfügen.

1. Alle Schlüsselspeicher müssen dem Prozess 'abst' zugeordnet sein, bevor die DB2-Dämonen gestartet werden.

   Sie können sicherstellen, dass diese Schlüsselspeicher zugeordnet sind, indem Sie den Befehl **efskeymgr -V** wie im folgenden Beispiel ausgeben:

   ```
   # lsuser abst
   abst id=203 pgrp=abstgp groups=abstgp,staff ...

   # efskeymgr -V
   Liste der im aktuellen Prozess geladenen Schlüssel:
     Key #0:
                        Kind ..................... User key
                        Id (uid / gid) ......... 203
                        Type ..................... Private key
                        Algorithm ................ RSA_1024
                        Validity ................. Key is valid
                        Fingerprint ..............
   24c88df2:d91cb6a2:c3e11b6a:4c13f8b4:666fabd8

     Key #1:
                        Kind ..................... Group key
                        Id (uid / gid) ......... 1
                        Type ..................... Private key
                        Algorithm ................ RSA_1024
                        Validity ................. Key is valid
                        Fingerprint ..............
   03fead42:57e7646e:a1715626:cfa56c8e:8abed1c1

     Key #2:
                        Kind ..................... Group key
                        Id (uid / gid) ......... 212
                        Type ..................... Private key
                        Algorithm ................ RSA_1024
                        Validity ................. Key is valid
                        Fingerprint ..............
   339dfb19:bc850f4c:5551c975:7fe4961b:2dddf3bc
   ```

2. Versuchen Sie, die Schlüsselspeicher mit dem folgenden Befehl zu laden, wenn dem Prozess 'abst' keine Schlüsselspeicher als zugeordnet angezeigt werden: `% efskeymgr -o ksh`

   Dieser Befehl fordert zur Eingabe des Schlüsselspeicherkennworts auf, das ursprünglich mit dem Anmeldekennwort definiert ist.

3. Stellen Sie sicher, dass die Benutzer- und Gruppenschlüssel geladen sind, indem Sie den Befehl wiederholen: `% efskeymgr -V`

   Es müssen Benutzer- und Gruppenschlüssel aufgelistet werden. Fahren Sie mit Schritt 4 fort, wenn die Gruppenschlüsselspeicher weiterhin nicht aufgelistet werden.

4. Bei Gruppen, die auf bestimmte Arten erstellt wurden, sind möglicherweise keine Gruppenschlüsselspeicher vorhanden. Werden die Gruppenschlüsselspeicher des Benutzers nach Eingabe des Befehls **efskeymgr -V** nicht aufgelistet, müssen Sie die Gruppenschlüsselspeicher erstellen.

   Erstellen Sie den Gruppenschlüsselspeicher als Root oder mit der RBAC-Rolle *aix.efs_admin*:

   ```
   % efskeymgr -C gruppenname
   ```

5. Ordnen Sie den entsprechenden Benutzern den Zugriff auf den Gruppenschlüsselspeicher zu:

   ```
   % efskeymgr -k group /gruppenname -s user/benutzername
   ```

   Benutzer, die zu diesem Zeitpunkt bereits angemeldet sind, erhalten nicht sofort einen Zugriff auf den Gruppenschlüsselspeicher. Diese Benutzer sollten die Schlüsselspeicher mit dem Befehl **efskeymgr -o ksh** erneut laden oder sich erneut anmelden.

Aktivieren von EFS für das Datenbankdateisystem

EFS kann nur in JFS2-Dateisystemen ausgeführt werden und muss speziell aktiviert werden.

Wenn sich Ihre Datenbank in einem vorhandenen Dateisystem befindet, müssen Sie EFS aktivieren, indem Sie den Befehl `% chfs -a efs=yes dateisystem` ausführen. Dazu folgendes Beispiel:

```
% chfs –a efs=yes /test01
```

Wenn Sie ein neues Dateisystem erstellen, können Sie EFS mit der Option `-a efs=yes` über den Befehl **smit** oder den Befehl **crfs** aktivieren. Dazu folgendes Beispiel:

```
% crfs -v jfs2 -a efs=yes -m mountpunkt -d einheit -A yes
```

EFS ist jetzt im Dateisystem aktiviert, jedoch noch nicht gestartet. Starten Sie EFS nur für die Datenbanktabellen, die verschlüsselte Daten erfordern (nähere Informationen zum Befehl **efsmgr** sowie zur Vererbung finden Sie in der AIX-Dokumentation zu EFS).

Bestimmen der zu verschlüsselnden Datei

Die Datei, die eine bestimmte Datenbanktabelle enthält, die Sie mit EFS-Verschlüsselung schützen möchten, können Sie wie im Folgenden angegeben bestimmen. Im folgenden Beispiel handelt es sich dabei um die Tabelle EMPLOYEE.

1. Verwenden Sie eine Abfrage wie im folgenden Beispiel, um die Tabellenbereichs-ID TBSPACEID für die Tabelle zu ermitteln:

   ```
   SELECT TABNAME, TBSPACEID FROM syscat.tables WHERE tabname='EMPLOYEE'
   ```

   Diese Abfrage könnte beispielsweise folgende Ergebnisse liefern:

TABNAME	TBSPACEID
EMPLOYEE	2

2. Ermitteln Sie die Tabellenbereiche für die Beispieltabellenbereichs-ID 2 mit einer Abfrage wie der folgenden:

   ```
   LIST TABLESPACE CONTAINERS FOR 2
   ```

   Diese Abfrage könnte beispielsweise folgende Ergebnisse liefern:

Container-ID	Name	Typ
0	/test01/abst/NODE0000/BAR/T0000002/ C0000000.LRG	Datei

   Jetzt wissen Sie, dass dieser Tabellenbereich in der Betriebssystemdatei `/test01/abst/NODE0000/BAR/T0000002/C0000000.LRG` enthalten ist. Hierbei handelt es sich um die Datei, die Sie verschlüsseln müssen.

Verschlüsseln der Datei

Wie bei allen größeren Änderungen an Daten oder Datenbanken insgesamt müssen Sie die Datenbank zunächst sichern.

Verschlüsseln Sie die Datei wie folgt:

1. Listen Sie die Datei auf. Dazu folgendes Beispiel:

   ```
   # ls -U /test01/abst/NODE0000/BAR/T0000002/C0000000.LRG

   -rw-------- 1 abst abstgp 33554432 Jul 30 18:01
   /test01/abst/NODE0000/BAR/T0000002/C0000000.LRG
   ```

2. Verschlüsseln Sie die Datei mit dem Befehl **efsmgr**, wie im folgenden Beispielbefehl angegeben:

   ```
   # efsmgr -e /test01/abst/NODE0000/BAR/T0000002/C0000000.LRG
   ```

   Wenn Sie die Datei erneut auflisten, wird am Ende der Berechtigungsfolge durch ein zusätzliches „e" (für encrypted) angezeigt, dass die Datei verschlüsselt ist. Beispiel:

   ```
   # ls -U /test01/abst/NODE0000/BAR/T0000002/C0000000.LRG

   -rw-------e 1 abst abstgp 33554432 Jul 30 18:03
   /test01/abst/NODE0000/BAR/T0000002/C0000000.LRG
   ```

3. Starten Sie den DB2-Datenbankmanager und verwenden Sie ihn wie üblich. Alle zu der Tabelle EMPLOYEE hinzugefügten Daten und der verschlüsselte Tabellenbereich sind im Dateisystem über EFS verschlüsselt. Wenn die Daten abgerufen werden, werden sie entschlüsselt und über den DB2-Datenbankmanager wie üblich angezeigt.

Prüfen von DB2-Aktivitäten

Einführung in die DB2-Prüffunktion

Zur Verwaltung des Zugriffs auf schutzwürdige Daten können Sie eine Reihe von Authentifizierungs- und Zugriffssteuerungsmechanismen verwenden, um Regeln und Steuerelemente für zulässige Datenzugriffe einzurichten. Zum Schutz gegen unbekanntes bzw. nicht akzeptables Zugriffsverhalten und zur Erkennung eines solchen Verhaltens können Sie den Datenzugriff mit der DB2-Prüffunktion überwachen.

Die erfolgreiche Überwachung unerwünschter Datenzugriffe mit anschließender Analyse dieser Vorgänge kann zu verbesserter Datenzugriffssteuerung und damit zur Verhinderung böswilliger oder unbedachter Datenzugriffe beitragen. Durch Überwachung von Anwendungs- und Benutzerzugriffen, einschließlich der Systemverwaltungsaktionen, kann ein fortlaufendes Protokoll aller an Ihrer Datenbank ausgeführten Aktivitäten erstellt werden.

Die DB2-Prüffunktion generiert ein Prüfprotokoll für eine Reihe vordefinierter Datenbankereignisse und ermöglicht Ihnen die Verwaltung dieses Protokolls. Die von dieser Funktion generierten Datensätze werden in einer Prüfprotokolldatei aufgezeichnet. Die Analyse dieser Datensätze macht Nutzungsmuster deutlich, an denen Systemmissbrauch erkennbar wird. Sobald der Systemmissbrauch identifiziert ist, können entsprechende Gegenmaßnahmen eingeleitet werden.

Die Prüffunktion bietet die Möglichkeit einer Prüfung sowohl auf Instanzebene auch auf der Ebene einzelner Datenbanken und zeichnet alle Aktivitäten auf Instanz- und Datenbankebene unabhängig voneinander jeweils in separaten Protokol-

len auf. Der Systemadministrator (der die Berechtigung SYSADM besitzt) kann das Tool **db2audit** verwenden, um eine Prüfung auf Instanzebene zu konfigurieren und zu steuern, wann die entsprechenden Prüfdaten erfasst werden. Der Systemadministrator kann das Tool **db2audit** zum Archivieren von Prüfprotokollen sowohl der Instanz als auch der Datenbank sowie zum Extrahieren von Prüfdaten aus den archivierten Protokollen beider Typen verwenden.

Der Sicherheitsadministrator (der die Berechtigung SECADM für eine Datenbank besitzt) kann Prüfrichtlinien in Verbindung mit der SQL-Anweisung AUDIT verwenden, um die Prüfvorschriften für eine einzelne Datenbank zu konfigurieren und zu steuern. Der Sicherheitsadministrator kann die angegebenen Tasks mit den folgenden Prüfroutinen ausführen:

- Die gespeicherte Prozedur SYSPROC.AUDIT_ARCHIVE archiviert Prüfprotokolle.
- Die Tabellenfunktion SYSPROC.AUDIT_LIST_LOGS ermöglicht es Ihnen, für Ihre Zwecke interessante Protokolle zu ermitteln.
- Die gespeicherte Prozedur SYSPROC.AUDIT_DELIM_EXTRACT extrahiert Daten zu Analysezwecken in Dateien mit begrenzter Satzlänge.

Der Sicherheitsadministrator kann einem anderen Benutzer das Zugriffsrecht EXECUTE für diese Routinen erteilen und die in seinen Bereich fallenden Aufgaben auf diese Weise bei Bedarf delegieren.

Beim Arbeiten in einer Umgebung mit partitionierten Datenbanken treten viele der zu protokollierenden Ereignisse in der Datenbankpartition auf, mit der der Benutzer verbunden ist (Koordinatorpartition), oder in der Katalogpartition (sofern diese beiden Datenbankpartitionen nicht identisch sind). Dies bedeutet, dass Prüfsätze von mehreren Datenbankpartitionen generiert werden können. Ein Teil jedes Prüfsatzes enthält Informationen, die die Koordinatorpartition und die Ursprungsdatenbankpartition (die Partition, aus der der Prüfsatz generiert wurde), angeben.

Auf der Instanzebene muss die Prüffunktion explizit mit den Befehlen **db2audit start** und **db2audit stop** gestartet bzw. gestoppt werden. Wenn Sie die Prüfung auf Instanzebene starten, arbeitet die Prüffunktion mit vorhandenen Prüfkonfigurationsdaten. Da die Prüffunktion vom DB2-Datenbankserver unabhängig ist, bleibt sie auch dann aktiv, wenn die Instanz gestoppt wird. Beim Stoppen der Instanz kann sogar ein Prüfsatz im Prüfprotokoll generiert werden. Zum Starten einer Prüfung auf Datenbankebene müssen Sie zunächst eine Prüfrichtlinie erstellen und diese Prüfrichtlinie den zu prüfenden Objekten zuordnen, z. B. Berechtigungs-IDs, Datenbankberechtigungen, gesicherten Kontexten oder bestimmten Tabellen.

Kategorien von Prüfsätzen

Verschiedene Kategorien von Prüfsätzen können generiert werden. Beachten Sie in der folgenden Beschreibung der Kategorien möglicher Prüfereignisse, dass auf jeden Kategorienamen ein aus einem Wort bestehendes Schlüsselwort folgt, das den Typ der Kategorie identifiziert. Folgende Kategorien von Prüfereignissen stehen zur Verfügung:

- Prüfung (AUDIT). Generiert Prüfsätze, wenn Prüfeinstellungen geändert werden oder auf das Prüfprotokoll zugegriffen wird.
- Berechtigungsprüfung (CHECKING). Generiert Prüfsätze während der Berechtigungsprüfung, wenn versucht wird, auf DB2-Datenbankobjekte oder -funktionen zuzugreifen oder sie zu manipulieren.
- Objektpflege (OBJMAINT). Generiert Prüfsätze beim Erstellen oder Löschen von Datenobjekten sowie beim Ändern bestimmter Objekte.

- Sicherheitspflege (SECMAINT). Generiert Prüfsätze bei folgenden Aktivitäten:
  - Erteilen oder Entziehen von Objektzugriffsrechten oder Datenbankberechtigungen
  - Erteilen oder Entziehen von Sicherheitskennsätzen oder Freistellungen
  - Ändern der Gruppenberechtigung, Rollenberechtigung oder Überschreiben bzw. Einschränken von Attributen einer LBAC-Sicherheitsrichtlinie
  - Erteilen oder Entziehen des Zugriffsrechts SETSESSIONUSER
  - Ändern der Konfigurationsparameter SYSADM_GROUP, SYSCTRL_GROUP, SYSMAINT_GROUP oder SYSMON_GROUP
- Systemverwaltung (SYSADMIN). Generiert Prüfsätze, wenn Operationen ausgeführt werden, für die die Berechtigung SYSADM, SYSMAINT oder SYSCTRL erforderlich ist.
- Benutzergültigkeitsprüfung (VALIDATE). Generiert Prüfsätze bei der Vergabe von Benutzerberechtigungen und beim Abrufen von Systemsicherheitsdaten.
- Operationskontext (CONTEXT). Generiert Prüfsätze, die den Operationskontext darstellen, wenn eine Datenbankoperation ausgeführt wird. Diese Prüfkategorie erleichtert die Interpretation der Prüfprotokolldatei. Bei Verwendung zusammen mit dem Ereigniskorrelationsfeld des Protokolls kann eine Gruppe von Ereignissen auf eine einzige Datenbankoperation zurückgeführt werden. Beispiel: Eine Abfrageanweisung für dynamische Abfragen, eine Paketkennung für statische Abfragen oder ein Indikator des ausgeführten Operationstyps (z. B. CONNECT) kann den erforderlichen Kontext zum Analysieren der Prüfergebnisse liefern.

  Anmerkung: Die SQL- oder XQuery-Anweisung, die den Operationskontext bereitstellt, kann sehr lang sein. Sie wird jedoch im CONTEXT-Prüfsatz vollständig angegeben. Dadurch kann der CONTEXT-Prüfsatz sehr umfangreich werden.
- Ausführen (EXECUTE). Generiert Prüfsätze während der Ausführung von SQL-Anweisungen.

Für alle zuvor aufgeführten Kategorien können Sie auf fehlgeschlagene oder erfolgreich ausgeführte Operationen oder beide Arten prüfen.

Alle Operationen auf dem Datenbankserver können mehrere Prüfsätze generieren. Die tatsächliche Anzahl der im Prüfprotokoll generierten Prüfsätze hängt von der Anzahl der aufzuzeichnenden Ereigniskategorien ab, die in der Konfiguration der Prüffunktion angegeben werden. Sie ist außerdem davon abhängig, ob erfolgreiche oder fehlgeschlagene Operationen oder beide geprüft werden. Deshalb sollte gezielt ausgewählt werden, welche Ereignisse zu protokollieren sind.

Prüfrichtlinien

Der Sicherheitsadministrator kann mithilfe von Prüfrichtlinien die Prüffunktion so konfigurieren, dass nur Informationen zu den Daten und Objekten erfasst werden, die benötigt werden.

Der Sicherheitsadministrator kann Prüfrichtlinien erstellen, um zu steuern, welche Elemente in einer einzelnen Datenbank zu prüfen sind. Den folgenden Objekten kann eine Prüfrichtlinie zugeordnet werden:

- Gesamte Datenbank

  Alle prüfbaren Ereignisse, die in der Datenbank auftreten, werden der Prüfrichtlinie entsprechend protokolliert.
- Tabellen

  Alle Zugriffe durch die Datenbearbeitungssprache (DML) und XQUERY-Zugriffe auf die Tabelle (nicht typisiert), die MQT (Materialized Query Table) oder den

Kurznamen werden protokolliert. Nur Prüfereignisse der Kategorie EXECUTE werden mit oder ohne Daten generiert, wenn auf die Tabelle zugegriffen wird, selbst wenn die Richtlinie angibt, dass andere Kategorien protokolliert werden sollen.

- Gesicherte Kontexte

  Alle prüfbaren Ereignisse, die in einer gesicherten Verbindung auftreten, die durch den bestimmten gesicherten Kontext definiert wird, werden der Prüfrichtlinie entsprechend protokolliert.

- Berechtigungs-IDs von Benutzern, Gruppen oder Rollen

  Alle prüfbaren Ereignisse, die von dem angegebenen Benutzer eingeleitet werden, werden der Prüfrichtlinie entsprechend protokolliert.

  Alle prüfbaren Ereignisse, die von Benutzern eingeleitet werden, die zu einer Gruppe oder Rolle gehören, werden der Prüfrichtlinie entsprechend protokolliert. Dies schließt indirekte Rollenzugehörigkeiten zum Beispiel durch andere Rollen und Gruppen mit ein.

  Sie können ähnliche Daten mithilfe der Workload Management-Ereignismonitore, erfassen, indem Sie eine Auslastung (Workload) für eine Gruppe definieren und die Aktivitätsdetails erfassen. Sie sollten sich jedoch darüber im Klaren sein, dass die Zuordnung von Auslastungen neben der Berechtigungs-ID auch auf Attribute zurückgreift, die dazu führen können, dass nicht die gewünschte Granularität beim Prüfen erzielt wird oder, falls diese anderen Attribute geändert werden, dass Verbindungen möglicherweise anderen (eventuell nicht überwachten) Auslastungen zugeordnet werden. Die Lösung der Prüffunktion bietet hingegen die Garantie, dass ein Benutzer, eine Gruppe oder eine Rolle geprüft wird.

- Berechtigungen (SYSADM, SECADM, DBADM, SQLADM, WLMADM, ACCESSCTRL, DATAACCESS, SYSCTRL, SYSMAINT, SYSMON)

  Alle prüfbaren Ereignisse, die von einem Benutzer eingeleitet werden, der die angegebene Berechtigung hat, selbst wenn diese Berechtigung für das Ereignis nicht erforderlich ist, werden der Prüfrichtlinie entsprechend protokolliert.

Der Sicherheitsadministrator kann mehrere Prüfrichtlinien erstellen. Zum Beispiel kann es für ein Unternehmen sinnvoll sein, eine Richtlinie zur Prüfung schutzwürdiger Daten und eine Richtlinie zur Prüfung der Aktivitäten von Benutzern mit der Berechtigung DBADM zu haben. Wenn mehrere Prüfrichtlinien für eine Anweisung verwendet werden, werden alle Ereignisse, die für jede der Prüfrichtlinien geprüft werden müssen, geprüft (jedoch nur einmal). Wenn zum Beispiel die Prüfrichtlinie der Datenbank eine Prüfung erfolgreicher EXECUTE-Ereignisse für eine bestimmte Tabelle erfordert und die Prüfrichtlinie des Benutzers eine Prüfung fehlgeschlagener EXECUTE-Ereignisse für dieselbe Tabelle erfordert, werden sowohl erfolgreiche als auch fehlgeschlagene Zugriffsversuche für diese Tabelle protokolliert.

Für ein bestimmtes Objekt kann nur eine Prüfrichtlinie in Kraft sein. Es ist zum Beispiel nicht möglich, mehrere Prüfrichtlinien gleichzeitig derselben Tabelle zuzuordnen.

Eine Prüfrichtlinie kann nicht einer Sicht oder einer typisierten Tabelle zugeordnet werden. Sichten, die auf eine Tabelle zugreifen, der eine Prüfrichtlinie zugeordnet ist, werden entsprechend der Prüfrichtlinie der zugrunde liegenden Tabelle geprüft.

Eine Prüfrichtlinie, die für eine Tabelle gilt, gilt nicht automatisch auch für eine MQT (Materialized Query Table), die auf dieser Tabelle basiert. Wenn Sie eine Prüfrichtlinie einer Tabelle zuordnen, sollten Sie dieselbe Richtlinie auch jeder MQT zuordnen, die auf dieser Tabelle basiert.

Eine Prüfung, die während einer Transaktion ausgeführt wird, erfolgt auf der Basis der Prüfrichtlinien und ihrer Zuordnungen zu Beginn der Transaktion. Wenn zum Beispiel der Sicherheitsadministrator eine Prüfrichtlinie einem Benutzer zuordnet und sich dieser Benutzer zu diesem Zeitpunkt in einer Transaktion befindet, wirkt sich die Prüfrichtlinie nicht auf die verbleibenden Anweisungen aus, die in dieser Transaktion ausgeführt werden. Auch Änderungen an einer Prüfrichtlinie treten erst in Kraft, wenn sie festgeschrieben (COMMIT) werden. Wenn ein Sicherheitsadministrator eine Anweisung ALTER AUDIT POLICY ausführt, wird sie erst wirksam, wenn die Anweisung mit COMMIT festgeschrieben wird.

Der Sicherheitsadministrator verwendet die Anweisung CREATE AUDIT POLICY zum Erstellen einer Prüfrichtlinie und die Anweisung ALTER AUDIT POLICY zum Ändern einer Prüfrichtlinie. In diesen Anweisungen können folgende Spezifikationen angegeben werden:

- Die Statuswerte für zu prüfende Ereignisse: `NONE` (Kein), `SUCCESS` (Erfolg), `FAILURE` (Fehlschlag) oder `BOTH` (Beides).

  Nur prüfbare Ereignisse, die dem angegebenen Statuswert entsprechen, werden protokolliert.
- Das Serververhalten, wenn Fehler während der Prüfung auftreten.

Der Sicherheitsadministrator verwendet die Anweisung AUDIT, um eine Prüfrichtlinie der aktuellen Datenbank oder einem Datenbankobjekt auf dem aktuellen Server zuzuordnen. Bei jeder Verwendung dieses Objekts wird es der zugeordneten Prüfrichtlinie entsprechend geprüft.

Der Sicherheitsadministrator verwendet die Anweisung DROP zum Löschen einer Prüfrichtlinie. Eine Prüfrichtlinie kann nicht gelöscht werden, wenn sie einem Objekt zugeordnet ist. Mit der Anweisung AUDIT REMOVE kann jede verbliebene Zuordnung zu einem Objekt entfernt werden. Wenn einer Prüfrichtlinie Metadaten hinzugefügt werden sollen, kann der Sicherheitsadministrator dazu die Anweisung COMMENT verwenden.

Vor Einrichtung einer vollständigen Verbindung generierte Ereignisse

Für einige Ereignisse, die während eines Verbindungsaufbaus und einer Benutzerwechseloperation generiert werden, sind nur die Informationen der Prüfrichtlinie verfügbar, die der Datenbank zugeordnet ist. Diese Ereignisse sind in der folgenden Tabelle aufgeführt:

Tabelle 4. Verbindungsereignisse

Ereignis	Prüf-kategorie	Kommentar
CONNECT	CONTEXT	
CONNECT_RESET	CONTEXT	
AUTHENTICATION	VALIDATE	Dieses Ereignis erfasst die Authentifizierung sowohl beim Verbindungsaufbau als auch beim Benutzerwechsel in einer gesicherten Verbindung.
CHECKING_FUNC	CHECKING	Der versuchte Zugriff ist SWITCH_USER.

Diese Ereignisse werden nur auf der Basis der Prüfrichtlinie, die der Datenbank zugeordnet ist, und nicht durch Prüfrichtlinien, die anderen Objekten, wie Benutzern, deren Gruppen oder Berechtigungen, zugeordnet sind, protokolliert. Für die

Ereignisse CONNECT und AUTHENTICATION, die während des Verbindungsaufbaus auftreten, werden die Prüfeinstellungen auf der Instanzebene verwendet, bis die Datenbank aktiviert wird. Die Datenbank wird aktiviert, wenn die erste Verbindung zu ihr hergestellt wird oder wenn der Befehl ACTIVATE DATABASE ausgeführt wird.

Wirkung des Benutzerwechsels

Wenn ein Benutzer innerhalb einer gesicherten Verbindung gewechselt wird, bleiben keine Spuren des ursprünglichen Benutzers zurück. In diesem Fall wird eine Prüfrichtlinie, die dem ursprünglichen Benutzer zugeordnet ist, nicht mehr berücksichtigt, und die anzuwendenden Prüfrichtlinien werden auf der Basis des neuen Benutzers erneut ausgewertet. Eine der gesicherten Verbindung zugeordnete Prüfrichtlinie bleibt weiterhin in Kraft.

Wenn die Anweisung SET SESSION USER verwendet wird, wird nur die Sitzungsberechtigungs-ID gewechselt. Die Prüfrichtlinie der Berechtigungs-ID des ursprünglichen Benutzers (die Systemberechtigungs-ID) bleibt in Kraft und die Prüfrichtlinie des neuen Benutzers wird zusätzlich verwendet. Wenn mehrere Anweisungen SET SESSION USER innerhalb einer Sitzung ausgeführt werden, werden nur die Prüfrichtlinie, die dem ursprünglichen Benutzer (die Systemberechtigungs-ID) zugeordnet ist, und die Prüfrichtlinie, die dem aktuellen Benutzer (die Sitzungsberechtigungs-ID) zugeordnet ist, berücksichtigt.

Einschränkungen für die Datendefinitionssprache (DDL)

Die folgenden Anweisungen der Datendefinitionssprache (DDL-Anweisungen) werden als exklusive SQL-Anweisungen der Prüffunktion (exklusive AUDIT-Anweisungen) bezeichnet:

- AUDIT
- CREATE AUDIT POLICY, ALTER AUDIT POLICY und DROP AUDIT POLICY
- DROP ROLE und DROP TRUSTED CONTEXT, wenn die Rolle bzw. der gesicherte Kontext, die bzw. der gelöscht wird, einer Prüfrichtlinie zugeordnet ist

Exklusive AUDIT-Anweisungen unterliegen in ihrer Verwendung bestimmten Einschränkungen:

- Auf jede Anweisung muss eine Anweisung COMMIT oder ROLLBACK folgen.
- Diese Anweisungen können nicht innerhalb einer globalen Transaktion, zum Beispiel einer XA-Transaktion, ausgeführt werden.

Nur eine nicht festgeschriebene exklusive DDL-Anweisung der Prüffunktion ist für alle Partitionen gleichzeitig zulässig. Wenn eine nicht festgeschriebene exklusive DDL-Anweisung der Prüffunktion ausgeführt wird, warten nachfolgende exklusive DDL-Anweisungen der Prüffunktion, bis die aktuelle Anweisung festgeschrieben (COMMIT) oder rückgängig gemacht (ROLLBACK) wird.

Anmerkung: Änderungen werden in den Katalog geschrieben, werden jedoch erst wirksam, wenn sie festgeschrieben werden. Dies gilt auch für die Verbindung, die die Anweisung ausführt.

Beispiel für das Prüfen eines beliebigen Zugriffs auf eine bestimmte Tabelle

Betrachten Sie zum Beispiel ein Unternehmen, bei dem die Tabelle EMPLOYEE extrem sensible Informationen enthält, sodass das Unternehmen sämtliche SQL-Zugriffe auf die Daten in dieser Tabelle prüfen möchte. Zur Verfolgung aller Zugriffe

auf eine Tabelle kann die Kategorie EXECUTE verwendet werden. Sie prüft die SQL-Anweisung und optional den für diese Anweisung bei der Ausführung angegebenen Eingabedatenwert.

Die Verfolgung der Aktivitäten an der Tabelle erfolgt in zwei Schritten. Zunächst erstellt der Sicherheitsadministrator eine Prüfrichtlinie, die die Kategorie EXECUTE angibt. Anschließend ordnet der Sicherheitsadministrator diese Richtlinie der Tabelle zu:

```
CREATE AUDIT POLICY SENSITIVEDATAPOLICY
    CATEGORIES EXECUTE STATUS BOTH ERROR TYPE AUDIT
COMMIT

AUDIT TABLE EMPLOYEE USING POLICY SENSITIVEDATAPOLICY
COMMIT
```

Beispiel für das Prüfen beliebiger Aktionen durch SYSADM oder DBADM

Ein Unternehmen muss zur Erfüllung einer Sicherheitszertifizierung nachweisen, dass alle Aktivitäten in der Datenbank durch Personen, die die Berechtigung SYSADM (Systemverwaltung) oder DBADM (Datenbankverwaltung) besitzen, überwacht werden können.

Zur Erfassung aller Aktionen in der Datenbank sollten sowohl die Kategorie EXECUTE als auch die Kategorie SYSADMIN geprüft werden. Der Sicherheitsadministrator erstellt eine Prüfrichtlinie, die diese beiden Kategorie prüft. Der Sicherheitsadministrator kann diese Prüfrichtlinie mit der Anweisung AUDIT den Berechtigungen SYSADM und DBADM zuordnen. Für jeden Benutzer, der entweder über die Berechtigung SYSADM oder die Berechtigung DBADM verfügt, werden anschließend alle prüfbaren Ereignisse protokolliert. Das folgende Beispiel zeigt, wie eine solche Prüfrichtlinie erstellt und den Berechtigungen SYSADM und DBADM zugeordnet wird:

```
CREATE AUDIT POLICY ADMINSPOLICY CATEGORIES EXECUTE STATUS BOTH,
   SYSADMIN STATUS BOTH ERROR TYPE AUDIT
COMMIT
AUDIT SYSADM, DBADM USING POLICY ADMINSPOLICY
COMMIT
```

Beispiel für das Prüfen eines beliebigen Zugriffs durch eine bestimmte Rolle

In einem Unternehmen können die Webanwendungen des Unternehmens auf die Unternehmensdatenbank zugreifen. Die genauen Personen, die mit den Webanwendungen arbeiten, sind nicht bekannt. Bekannt sind nur die verwendeten Rollen, die daher zur Verwaltung der Datenbankberechtigungen verwendet werden. Das Unternehmen möchte die Aktionen aller Benutzer überwachen, die zu dieser Rolle gehören, um die Anforderungen zu untersuchen, die sie an die Datenbank übergeben. Dadurch soll sichergestellt werden, dass sie nur über die Webanwendungen auf die Datenbank zugreifen.

Die Kategorie EXECUTE enthält die erforderliche Prüfstufe, um die Aktivitäten der Benutzer in diesem Fall zu verfolgen. Der erste Schritt besteht darin, die entsprechende Prüfrichtlinie zu erstellen und sie den Rollen zuzuordnen, die von den Webanwendungen verwendet werden (in diesem Beispiel sind dies die Rollen TELLER und CLERK):

```
CREATE AUDIT POLICY WEBAPPPOLICY CATEGORIES EXECUTE WITH DATA
   STATUS BOTH ERROR TYPE AUDIT
COMMIT
AUDIT ROLE TELLER, ROLE CLERK USING POLICY WEBAPPPOLICY
COMMIT
```

Speicherung und Analyse von Prüfprotokollen

Die Archivierung des Prüfprotokolls versetzt das aktive Prüfprotokoll in ein Archivverzeichnis, während der Server gleichzeitig mit dem Schreiben in ein neues aktives Prüfprotokoll beginnt. Zu einem späteren Zeitpunkt können Sie Daten aus einem archivierten Protokoll in Dateien mit begrenzter Satzlänge extrahieren und anschließend Daten zu Analysezwecken aus diesen Dateien in DB2-Datenbanktabellen laden.

Die Konfiguration der Position für die Prüfprotokolle bietet die Möglichkeit, die Prüfprotokolle auf einem großen Hochgeschwindigkeitsdatenträger zu platzieren, wobei für jedes Member in einer Umgebung mit partitionierten Datenbanken optional separate Platten verwendet werden können. In einer Umgebung mit partitionierten Datenbanken kann der Pfad für das aktive Prüfprotokoll ein Verzeichnis sein, das für jedes Member eindeutig ist. Ein eindeutiges Verzeichnis für jedes Member hilft bei der Vermeidung von Dateizugriffskonflikten, da jedes Member auf eine andere Platte schreibt.

Der Standardpfad für die Prüfprotokolle unter Windows-Betriebssystemen ist *instanz*\security\auditdata. Unter Linux- und UNIX-Betriebssystemen ist dieser Pfad *instanz*/security/auditdata. Wenn Sie die Standardposition nicht verwenden wollen, können Sie andere Verzeichnisse auswählen. (Sie können neue Verzeichnisse auf Ihrem System zur Verwendung als alternative Positionen erstellen, falls die gewünschten Verzeichnisse noch nicht vorhanden sind.) Zum Festlegen des Pfads der Position für das aktive Prüfprotokoll und der Position für die archivierten Prüfprotokolle verwenden Sie den Befehl **db2audit configure** mit den Parametern **datapath** und **archivepath** wie im folgenden Beispiel:

```
db2audit configure datapath /auditlog archivepath /auditarchive
```

Die Speicherpositionen für Prüfprotokolle, die Sie mit dem Befehl **db2audit** festlegen, gelten für alle Datenbanken in der Instanz.

Anmerkung: Wenn mehrere Instanzen auf dem Server vorhanden sind, sollte jede Instanz separate Daten- und Archivpfade haben.

Pfad für aktive Prüfprotokolle (datapath) in einer Umgebung mit partitionierten Datenbanken

In einer Umgebung mit partitionierten Datenbanken muss in jeder Partition die gleiche Position für das aktive Prüfprotokoll (durch den Parameter **datapath** festgelegt) verwendet werden. Dies lässt sich mit zwei Methoden erreichen:

1. Verwenden Sie Datenbankpartitionsausdrücke, wenn Sie den Parameter **datapath** angeben. Durch die Verwendung von Datenbankpartitionsausdrücken kann die Partitionsnummer in den Pfad für die Prüfprotokolldateien eingefügt werden, sodass sich in jeder Datenbankpartition ein anderer Pfad angeben lässt.
2. Verwenden Sie ein gemeinsam genutztes Laufwerk, das für alle Member identisch ist.

Sie können Datenbankpartitionsausdrücke an einer beliebigen Stelle innerhalb des Werts verwenden, den Sie für den Parameter **datapath** angeben. Auf einem System

mit drei Membern, auf dem die Datenbankpartitionsnummer 10 lautet, kann zum Beispiel der folgende Befehl verwendet werden:

```
db2audit configure datapath '/pfadFuerKnoten $N'
```

verwendet die folgenden Pfade:

- `/pfadFuerKnoten10`
- `/pfadFuerKnoten20`
- `/pfadFuerKnoten30`

Anmerkung: Zur Angabe des Archivpfads (Parameter **archivepath**) für Protokolldateien können keine Datenbankpartitionsausdrücke verwendet werden.

Archivieren aktiver Prüfprotokolle

Der Systemadministrator kann das Tool **db2audit** zum Archivieren von Prüfprotokollen sowohl der Instanz als auch der Datenbank sowie zum Extrahieren von Prüfdaten aus den archivierten Protokollen beider Typen verwenden.

Der Sicherheitsadministrator oder ein Benutzer, dem vom Sicherheitsadministrator das Zugriffsrecht EXECUTE für Prüfroutinen erteilt wurde, kann das aktive Prüfprotokoll mithilfe der gespeicherten Prozedur SYSPROC.AUDIT_ARCHIVE archivieren. Zum Extrahieren von Daten aus dem Protokoll und zum Laden dieser Daten in Dateien mit begrenzter Satzlänge kann dabei die gespeicherte Prozedur SYSPROC.AUDIT_DELIM_EXTRACT verwendet werden.

Der Sicherheitsadministrator oder ein Benutzer, dem vom Sicherheitsadministrator das Zugriffsrecht EXECUTE für Prüfroutinen erteilt wurde, muss zum Archivieren und Extrahieren von Prüfprotokollen mithilfe der Prüfroutinen wie folgt vorgehen:

1. Er muss eine Anwendung terminieren, die regelmäßige Archivierungen des aktiven Prüfprotokolls mithilfe der gespeicherten Prozedur SYSPROC.AUDIT_ARCHIVE ausführt.
2. Er muss feststellen, welche archivierten Protokolldateien von Interesse sind. Dabei kann er die Tabellenfunktion SYSPROC.AUDIT_LIST_LOGS verwenden, um alle archivierten Prüfprotokolle aufzulisten.
3. Er muss den ermittelten Dateinamen als Parameter an die gespeicherte Prozedur SYSPROC.AUDIT_DELIM_EXTRACT übergeben, um Daten aus dem Protokoll zu extrahieren und in Dateien mit begrenzter Satzlänge zu laden.
4. Er muss die Prüfdaten zur Analyse in DB2-Datenbanktabellen laden.

Die archivierten Protokolldateien müssen nicht sofort zur Analyse in Tabellen geladen werden. Sie können für eine spätere Analyse gespeichert werden. Zum Beispiel ist es möglich, dass sie überhaupt nur untersucht werden, wenn eine Unternehmensprüfung stattfindet.

Falls während der Archivierung ein Problem auftritt, zum Beispiel, wenn der Plattenspeicherplatz im Archivpfad nicht ausreicht oder der Archivpfad nicht vorhanden ist, schlägt der Archivierungsprozess fehl. In diesem Fall wird eine vorläufige Protokolldatei mit der Dateierweiterung `.bk` im Datenpfad für das Prüfprotokoll generiert. Beispiel: `db2audit.instance.log.0.20070508172043640941.bk`. Wenn das Problem beseitigt ist (durch Zuordnen eines ausreichenden Plattenspeicherbereichs im Archivpfad bzw. durch Erstellen des Archivpfads) müssen Sie diese vorläufige Protokolldatei in den Archivpfad versetzen. Anschließend können Sie diese Datei genauso wie ein erfolgreich archiviertes Protokoll behandeln.

Archivieren aktiver Prüfprotokolle in einer Umgebung mit partitionierten Datenbanken

Wenn in einer Umgebung mit partitionierten Datenbanken der Archivierungsbefehl ausgeführt wird, während die Instanz aktiv ist, wird der Archivierungsprozess automatisch für jedes Member ausgeführt. Für alle Member wird im Dateinamen der archivierten Protokolldatei dieselbe Zeitmarke verwendet. Auf einem System mit drei Membern, auf dem die Datenbankpartitionsnummer 10 lautet, kann zum Beispiel der folgende Befehl verwendet werden:

```
db2audit archive to /auditarchive
```

Dieser Befehl erstellt die folgenden Dateien:

- `/auditarchive/db2audit.log.10.`*`zeitmarke`*
- `/auditarchive/db2audit.log.20.`*`zeitmarke`*
- `/auditarchive/db2audit.log.30.`*`zeitmarke`*

Wenn der Archivierungsbefehl abgesetzt wird, während die Instanz nicht aktiv ist, können Sie mit einer der folgenden Methoden steuern, für welches Member die Archivierung ausgeführt wird:

- Sie können die Option **node** im Befehl **db2audit** verwenden, um die Archivierung nur für das aktuelle Member auszuführen.
- Sie können den Befehl **db2_all** verwenden, um die Archivierung für alle Member auszuführen.

  Beispiel:

  ```
  db2_all db2audit archive node to /auditarchive
  ```

  Dieser Befehl definiert die Umgebungsvariable **DB2NODE**, um anzugeben, für welche Member der Befehl aufgerufen wird.

Alternativ hierzu können Sie für jedes Member einen separaten Archivierungsbefehl absetzen. Beispiel:

- Auf Member 10:

  ```
  db2audit archive node 10 to /auditarchive
  ```
- Auf Member 20:

  ```
  db2audit archive node 20 to /auditarchive
  ```
- Auf Member 30:

  ```
  db2audit archive node 30 to /auditarchive
  ```

Anmerkung: Wenn die Instanz nicht aktiv ist, dann sind die Zeitmarken in den Dateinamen der archivierten Prüfprotokolle nicht für alle Member gleich.

Anmerkung: Es wird empfohlen, für alle Member denselben Archivpfad gemeinsam zu nutzen, dies ist jedoch nicht zwingend erforderlich.

Anmerkung: Die gespeicherte Prozedur AUDIT_DELIM_EXTRACT und die Tabellenfunktion AUDIT_LIST_LOGS können nur auf die archivierten Protokolldateien zugreifen, die über das aktuelle Member (Koordinatormember) sichtbar sind.

Beispiel für das Archivieren eines Protokolls und das Extrahieren von Daten in eine Tabelle

Zur Sicherstellung, dass die Prüfdaten erfasst und zur zukünftigen Verwendung gespeichert werden, muss ein Unternehmen alle sechs Stunden ein neues Prüfpro-

tokoll erstellen und das aktuelle Prüfprotokoll auf einem Laufwerk mit WORM-Funktionalität archivieren. Das Unternehmen terminiert den folgenden Aufruf der gespeicherten Prozedur SYSPROC.AUDIT_ARCHIVE so, dass der Aufruf jeweils im Abstand von sechs Stunden vom Sicherheitsadministrator oder einem Benutzer ausgegeben wird, dem der Sicherheitsadministrator das Zugriffsrecht EXECUTE für die gespeicherte Prozedur AUDIT_ARCHIVE erteilt hat. Der Pfad zum archivierten Protokoll ist der Standardarchivpfad `/auditarchive`. Die Archivierung wird für alle Member ausgeführt:

```
CALL SYSPROC.AUDIT_ARCHIVE( '/auditarchive', -2 )
```

Im Rahmen seiner Sicherheitsprozeduren hat das Unternehmen eine Reihe verdächtiger Verhaltensweisen bzw. unzulässiger Aktivitäten erkannt und definiert, nach denen in den Prüfdaten zu suchen ist. Das Verfahren sieht vor, alle Daten aus einem oder mehreren Prüfprotokollen zu extrahieren, diese Daten in eine relationale Tabelle zu laden und anschließend mithilfe von SQL-Abfragen nach solchen Aktivitäten zu suchen. Das Unternehmen hat die geeigneten Kategorien für die Prüfung festgelegt und der Datenbank oder anderen Datenbankobjekten die erforderlichen Prüfrichtlinien zugeordnet.

Nun kann zum Beispiel die gespeicherte Prozedur SYSPROC.AUDIT_DELIM_EXTRACT aufgerufen werden, um die archivierten Prüfprotokolle für alle Kategorien aus allen Membern zu extrahieren, die mit einer Zeitmarke vom April 2006 erstellt wurden. Dabei wird der Standardbegrenzer verwendet:

```
CALL SYSPROC.AUDIT_DELIM_EXTRACT(
     '', '', '/auditarchive',  'db2audit.%.200604%', '' )
```

In einem weiteren Beispiel kann die gespeicherte Prozedur SYSPROC.AUDIT_DELIM_EXTRACT aufgerufen werden, um die archivierten Prüfsätze mit erfolgreichen Ereignissen der Kategorie EXECUTE und fehlgeschlagenen Ereignissen der Kategorie CHECKING aus einer Datei mit der gewünschten Zeitmarke zu extrahieren:

```
CALL SYSPROC.AUDIT_DELIM_EXTRACT( '', '', '/auditarchive',
    'db2audit.%.20060419034937', 'category
     execute status success, checking status failure );
```

Namen der Prüfprotokolldateien:

Die Prüfprotokolldateien haben Namen, die zu erkennen geben, ob sie Prüfprotokolle auf Instanzebene oder Datenbankebene sind und aus welcher Partition in einer partitionierten Datenbankumgebung sie stammen. Archivierte Prüfprotokolle haben die an ihren Dateinamen angehängte Zeitmarke für den Zeitpunkt, zu dem der Archivierungsbefehl ausgeführt wurde.

Dateinamen für das aktive Prüfprotokoll

In einer partitionierten Datenbankumgebung kann der Pfad für das aktive Prüfprotokoll ein Verzeichnis sein, das für jede Partition eindeutig ist, sodass jede Partition in eine eigene Datei schreibt. Um die Herkunft der Prüfsätze exakt verfolgen zu können, wird die Partitionsnummer in die Dateinamen der Prüfprotokolle eingefügt. Für Partition 20 lautet der Dateiname des Prüfprotokolls auf Instanzebene zum Beispiel `db2audit.instance.log.20`. Für eine Datenbank mit dem Namen 'testdb' heißt die Prüfprotokolldatei `db2audit.db.testdb.log.20`.

In einer nicht partitionierten Datenbankumgebung wird als Partitionsnummer der Wert 0 (null) angenommen. In diesem Fall lautet der Prüfprotokolldateiname auf Instanzebene `db2audit.instance.log.0`. Für eine Datenbank mit dem Namen 'test-

db' in dieser Instanz ist der Name des Prüfprotokolls `db2audit.db.testdb.log.0`.

Dateinamen für archivierte Prüfprotokolle

Wenn das aktive Prüfprotokoll archiviert wird, wird die aktuelle Zeitmarke im folgenden Format an den Dateinamen angehängt: *JJJJMMTTHHMMSS*. (Dabei gibt *JJJJ* das Jahr, *MM* den Monat, *TT* den Tag, *HH* die Stunde, *MM* die Minuten und *SS* die Sekunden an.)

Das Dateinamenformat für ein archiviertes Prüfprotokoll hängt von der Ebene des Prüfprotokolls ab:

Archiviertes Prüfprotokoll auf Instanzebene
Der Dateiname für das archivierte Prüfprotokoll auf Instanzebene ist: `db2audit.instance.log.`*`partition`*`.`*`JJJJMMTTHHMMSS`*.

Archiviertes Prüfprotokoll auf Datenbankebene
Der Dateiname für das archivierte Prüfprotokoll auf Datenbankebene ist: `db2audit.db`*`datenbank`*`.log.`*`partition.JJJJMMTTHHMMSS`*.

In einer nicht partitionierten Datenbankumgebung hat *partition* den Wert `0` (null).

Die Zeitmarke stellt den Zeitpunkt dar, zu dem der Archivierungsbefehl ausgeführt wurde. Daher ist dieser Zeitpunkt nicht immer mit dem Zeitpunkt des letzten Prüfsatzes im Protokoll identisch. Die archivierte Prüfprotokolldatei kann Prüfsätze mit Zeitmarken enthalten, die einen um einige Sekunden späteren Zeitpunkt angeben als die Zeitmarke im Namen der Protokolldatei. Dies kann folgende Gründe haben:

- Wenn der Archivierungsbefehl abgesetzt wird, wartet die Prüffunktion darauf, dass laufende Schreiboperationen für Prüfsätze abgeschlossen werden, bevor die archivierte Protokolldatei erstellt wird.
- In einer Umgebung mit mehreren Systemen ist möglicherweise die Systemzeit auf einem fernen System nicht mit der Systemzeit des Systems synchron, auf dem der Archivierungsbefehl abgesetzt wird.

Wenn in einer partitionierten Datenbankumgebung der Server aktiv ist, wenn die Archivierung ausgeführt wird, ist die Zeitmarke über alle Partitionen hinweg konsistent und stellt die Zeitmarke dar, die in der Partition generiert wurde, in der die Archivierung ausgeführt wurde.

Erstellen von Tabellen zur Aufnahme der DB2-Prüfdaten:

Bevor Sie mit Prüfdaten in Datenbanktabellen arbeiten können, müssen Sie die Tabellen erstellen, in denen die Daten gespeichert werden sollen. Sie sollten in Betracht ziehen, diese Tabelle in einem separaten Schema zu erstellen, um die Daten in den Tabellen von unbefugten Benutzern zu isolieren.

Vorbereitende Schritte

- Die zur Erstellung eines Schemas erforderlichen Berechtigungen und Zugriffsrechte finden Sie in den Informationen zur Anweisung CREATE SCHEMA.
- Die zur Erstellung einer Tabelle erforderlichen Berechtigungen und Zugriffsrechte finden Sie in den Informationen zur Anweisung CREATE TABLE.
- Legen Sie fest, welcher Tabellenbereich zum Speichern der Tabellen verwendet werden soll. (Dieser Abschnitt behandelt keine Erstellung von Tabellenbereichen.)

Anmerkung: Das Format der Tabellen, die Sie zur Aufnahme der Prüfdaten erstellen müssen, kann sich von Release zu Release ändern. Möglicherweise werden neue Spalten hinzugefügt oder die Größe einer vorhandenen Spalte wird geändert. Das Script `db2audit.ddl` erstellt Tabellen im korrekten Format, um die Prüfsätze aufzunehmen.

Informationen zu diesem Vorgang

In den folgenden Beispielen wird gezeigt, wie die Tabellen erstellt werden, die die Datensätze aus den Dateien mit begrenzter Satzlänge aufnehmen können. Falls erwünscht, können Sie ein separates Schema zum Speichern dieser Tabellen erstellen.

Wenn Sie nicht alle Daten verwenden wollen, die in den Dateien enthalten sind, können Sie je nach Bedarf Spalten aus den Tabellendefinitionen weglassen oder die Erstellung bestimmter Tabellen übergehen. Wenn Sie Spalten aus den Tabellendefinitionen weglassen, müssen Sie die Befehle modifizieren, die Sie zum Laden von Daten in diese Tabellen verwenden.

Vorgehensweise

1. Führen Sie den Befehl **db2** aus, um ein DB2-Befehlsfenster zu öffnen.
2. Optional: Erstellen Sie ein Schema zum Speichern der Tabellen. In diesem Beispiel erhält das Schema den Namen AUDIT:
   ```
   CREATE SCHEMA AUDIT
   ```
3. Optional: Wenn Sie das Schema AUDIT erstellt haben, wechseln Sie zu diesem Schema, bevor Sie Tabellen erstellen:
   ```
   SET CURRENT SCHEMA = 'AUDIT'
   ```
4. Führen Sie das Script `db2audit.ddl` aus, um die Tabellen zu erstellen, die die Prüfsätze aufnehmen sollen.

   Das Script `db2audit.ddl` befindet sich im Verzeichnis `sqllib/misc` (`sqllib\misc` unter Windows). Das Script setzt voraus, dass eine Verbindung zu der Datenbank besteht und dass ein 8K-Tabellenbereich verfügbar ist. Der Befehl zum Ausführen des Scripts sieht wie folgt aus: **db2 +o -tf sqllib/misc/db2audit.ddl**. Die folgenden Tabellen werden durch das Script erstellt: AUDIT, CHECKING, OBJMAINT, SECMAINT, SYSADMIN, VALIDATE, CONTEXT und EXECUTE.
5. Nach der Erstellung der Tabellen kann der Sicherheitsadministrator die gespeicherte Prozedur SYSPROC.AUDIT_DELIM_EXTRACT bzw. der Systemadministrator den Befehl **db2audit extract** verwenden, um die Prüfsätze aus den archivierten Prüfprotokolldateien in die Dateien mit begrenzter Satzlänge zu extrahieren. Sie können die Prüfdaten aus den Dateien mit begrenzter Satzlänge in die Datenbanktabellen laden, die Sie zuvor erstellt haben.

Laden von DB2-Prüfdaten in Tabellen:

Nachdem Sie die Prüfprotokolldatei archiviert und in Dateien mit begrenzter Satzlänge extrahiert haben und die Datenbanktabellen zur Aufnahme der Prüfdaten erstellt wurden, können Sie die Prüfdaten aus den Dateien mit begrenzter Satzlänge zur Analyse in die Datenbanktabellen laden.

Informationen zu diesem Vorgang

Die Prüfdaten werden mit dem Dienstprogramm LOAD in die Tabellen geladen. Führen Sie für jede Tabelle einen getrennten LOAD-Befehl aus. Wenn Sie eine oder

mehrere Spalten aus den Tabellendefinitionen weggelassen haben, müssen Sie die Version des **LOAD**-Befehls modifizieren, um die Daten erfolgreich zu laden. Wenn Sie ein anderes Begrenzungszeichen als das Standardzeichen beim Extrahieren der Prüfdaten angegeben haben, müssen Sie die von Ihnen verwendete Version des **LOAD**-Befehls ebenfalls modifizieren.

Vorgehensweise

1. Führen Sie den Befehl **db2** aus, um ein DB2-Befehlsfenster zu öffnen.
2. Zum Laden von Daten in die Tabelle AUDIT führen Sie den folgenden Befehl aus:

   ```
   LOAD FROM audit.del OF DEL MODIFIED BY DELPRIORITYCHAR LOBSINFILE
            INSERT INTO schema.AUDIT
   ```

   Anmerkung: Geben Sie den Modifikator **DELPRIORITYCHAR** an, um ein ordnungsgemäßes Parsing der Binärdaten sicherzustellen.

   Anmerkung: Geben Sie die Option **LOBSINFILE** des Befehls **LOAD** an (wegen der Einschränkung, dass integrierte Daten für große Objekte auf 32 K begrenzt werden müssen). In einen Fällen kann es auch erforderlich sein, die Option **LOBS FROM** zu verwenden.

   Anmerkung: Verwenden Sie zur Angabe des Dateinamens den vollständig qualifizierten Pfadnamen. Wenn das DB2-Datenbanksystem zum Beispiel auf dem Laufwerk C: eines Windows-Betriebssystems installiert ist, müsste `C:\Program Files\IBM\SQLLIB\instanz\security\audit.del` als vollständig qualifizierter Dateiname für die Datei `audit.del` angegeben werden.
3. Zum Laden von Daten in die Tabelle CHECKING führen Sie den folgenden Befehl aus:

   ```
   LOAD FROM checking.del OF DEL MODIFIED BY DELPRIORITYCHAR LOBSINFILE
            INSERT INTO schema.CHECKING
   ```
4. Zum Laden von Daten in die Tabelle OBJMAINT führen Sie den folgenden Befehl aus:

   ```
   LOAD FROM objmaint.del OF DEL MODIFIED BY DELPRIORITYCHAR LOBSINFILE
            INSERT INTO schema.OBJMAINT
   ```
5. Zum Laden von Daten in die Tabelle SECMAINT führen Sie den folgenden Befehl aus:

   ```
   LOAD FROM secmaint.del OF DEL MODIFIED BY DELPRIORITYCHAR LOBSINFILE
            INSERT INTO schema.SECMAINT
   ```
6. Zum Laden von Daten in die Tabelle SYSADMIN führen Sie den folgenden Befehl aus:

   ```
   LOAD FROM sysadmin.del OF DEL MODIFIED BY DELPRIORITYCHAR LOBSINFILE
            INSERT INTO schema.SYSADMIN
   ```
7. Zum Laden von Daten in die Tabelle VALIDATE führen Sie den folgenden Befehl aus:

   ```
   LOAD FROM validate.del OF DEL MODIFIED BY DELPRIORITYCHAR LOBSINFILE
            INSERT INTO schema.VALIDATE
   ```
8. Zum Laden von Daten in die Tabelle CONTEXT führen Sie den folgenden Befehl aus:

   ```
   LOAD FROM context.del OF DEL MODIFIED BY DELPRIORITYCHAR LOBSINFILE
            INSERT INTO schema.CONTEXT
   ```
9. Zum Laden von Daten in die Tabelle EXECUTE führen Sie den folgenden Befehl aus:

   ```
   LOAD FROM execute.del OF DEL MODIFIED BY DELPRIORITYCHAR LOBSINFILE
            INSERT INTO schema.EXECUTE
   ```

10. Nach Abschluss des Ladens der Daten in die Tabellen, löschen Sie die `.del`-Dateien aus dem Unterverzeichnis `security/auditdata` des Verzeichnisses `sqllib`.
11. Wenn Sie die Prüfdaten in die Tabellen geladen haben, können Sie die Daten aus diesen Tabellen zur Analyse auswählen.

Nächste Schritte

Wenn Sie die Tabellen bereits ein erstes Mal gefüllt haben und dies wiederholen wollen, verwenden Sie die Option **INSERT**, um die neuen Tabellendaten den vorhandenen Tabellendaten hinzuzufügen. Wenn Sie die Datensätze der vorigen Operation **db2audit extract** aus den Tabellen entfernen wollen, laden Sie die neuen Daten mit der Option **REPLACE** in die Tabellen.

Gespeicherte Prozeduren zum Archivieren und Extrahieren von Prüfprotokollen:

Der Sicherheitsadministrator kann mithilfe der gespeicherten Prozedur und Tabellenfunktion SYSPROC.AUDIT_ARCHIVE, der gespeicherten Prozedur SYSPROC.AUDIT_DELIM_EXTRACT und mit der Tabellenfunktion SYSPROC.AUDIT_LIST_LOGS Prüfprotokolle der Datenbank, mit der er momentan verbunden ist, archivieren und Daten in Dateien mit begrenzter Satzlänge extrahieren.

Der Sicherheitsadministrator kann die Verwendung dieser Routinen an einen anderen Benutzer delegieren, indem er diesem Benutzer die Berechtigung EXECUTE für diese Routinen erteilt. Das Zugriffsrecht EXECUTE für diese Routinen kann nur vom Sicherheitsadministrator erteilt werden. Das Zugriffsrecht EXECUTE mit WITH GRANT OPTION kann für diese Routinen nicht erteilt werden (SQLSTATE 42501).

Sie müssen mit einer Datenbank verbunden sein, um die Prüfprotokolle der Datenbank mit diesen gespeicherten Prozeduren und Tabellenfunktionen archivieren bzw. auflisten zu können.

Wenn Sie die archivierten Dateien in ein anderes Datenbanksystem kopieren und die gespeicherten Prozeduren und Tabellenfunktionen für den Zugriff auf sie verwenden wollen, müssen Sie sicherstellen, dass der Datenbankname identisch ist oder die Dateien umbenennen, sodass sie den gleichen Datenbanknamen enthalten.

Diese gespeicherten Prozeduren und Tabellenfunktionen dienen nicht zum Archivieren bzw. Auflisten des Prüfprotokolls der Instanzebene. Das Prüfprotokoll der Instanzebene muss vom Systemadministrator mit dem Befehl **db2audit** archiviert und extrahiert werden.

Sie können diese gespeicherten Prozeduren und Tabellenfunktionen zur Ausführung der folgenden Operationen verwenden:

Tabelle 5. Gespeicherte Prozeduren und Tabellenfunktionen für das Prüfsystem

Gespeicherte Prozedur bzw. Tabellenfunktion	**Operation**	**Kommentare**
AUDIT_ARCHIVE	Archiviert das aktuelle Prüfprotokoll.	Akzeptiert den Archivpfad als Eingabe. Wenn der Archivpfad nicht angegeben wird, entnimmt diese gespeicherte Prozedur den Archivpfad der Prüfkonfigurationsdatei. Die Archivierung wird für jedes Member ausgeführt. An den Namen der Prüfprotokolldatei wird eine synchronisierte Zeitmarke angehängt.
AUDIT_LIST_LOGS	Gibt eine Liste der archivierten Prüfprotokolle im angegebenen Pfad für die aktuelle Datenbank zurück.	

Tabelle 5. Gespeicherte Prozeduren und Tabellenfunktionen für das Prüfsystem (Forts.)

Gespeicherte Prozedur bzw. Tabellenfunktion	Operation	Kommentare
AUDIT_ DELIM_EXTRACT	Extrahiert Daten aus den archivierten Binärprotokollen und lädt sie in Dateien mit begrenzter Satzlänge.	Die extrahierten Prüfsätze werden ein begrenztes Format umgesetzt, das sich zum Laden in DB2-Datenbanktabellen eignet. Die Ausgabe wird in separaten Dateien, jeweils eine pro Kategorie, erstellt. Darüber hinaus wird die Datei **auditlobs** erstellt, in der alle großen Objekte (LOBs) gespeichert werden, die in den Prüfdaten enthalten sind. Die Dateien haben die folgenden Dateinamen: • **audit.del** • **checking.del** • **objmaint.del** • **secmaint.del** • **sysadmin.del** • **validate.del** • **context.del** • **execute.del** • **auditlobs** Wenn die Dateien bereits vorhanden sind, wird die Ausgabe an sie angehängt. Die Datei **auditlobs** wird erstellt, wenn Daten der Kategorien CONTEXT oder EXECUTE extrahiert werden. Es können nur Daten der archivierten Prüfprotokolle für die aktuelle Datenbank extrahiert werden. Nur Dateien, die für das Koordinatormember sichtbar sind, werden extrahiert. Nur der Instanzeigner kann archivierte Prüfprotokolle löschen.

Kategorie EXECUTE zum Prüfen von SQL-Anweisungen

Die Kategorie EXECUTE bietet die Möglichkeit, die von einem Benutzer ausgeführten SQL-Anweisungen präzise zu verfolgen. (Vor Version 9.5 musste die Kategorie CONTEXT zur Erfassung solcher Informationen verwendet werden.)

Im Rahmen einer umfassenden Sicherheitsrichtlinie kann es für ein Unternehmen erforderlich sein, die Möglichkeit zu haben, bis zu einer festgelegten Anzahl Jahre rückwirkend die Auswirkungen einer bestimmten Anforderung auf bestimmte Tabellen in der Datenbank zu analysieren.Zu diesem Zweck muss ein Unternehmen eine Richtlinie zur Archivierung der wöchentlichen Backups und der zugehörigen Protokolldateien einrichten, sodass die Datenbank für jeden gewünschten Zeitpunkt rekonstruiert werden kann. Außerdem müssen genügend Datenbankprüfinformationen zu jeder Anforderung erfasst werden, die in der Datenbank ausgeführt

wird, um die Wiederholung und Analyse jeder beliebigen Anforderung für die betreffende wiederhergestellte Datenbank für jeden gewünschten zukünftigen Zeitpunkt zu ermöglichen. Diese Voraussetzung kann sowohl statistische als auch dynamische SQL-Anweisungen mit einschließen.

Die Kategorie EXECUTE erfasst den SQL-Anweisungstext sowie die Kompilierungsumgebung und andere Werte, die erforderlich sind, um die Anweisung zu einem späteren Zeitpunkt zu wiederholen. Zum Beispiel kann eine Wiederholung der Anweisung zeigen, welche Zeilen genau durch eine Anweisung SELECT zurückgegeben wurden. Zur Wiederholung einer Anweisung müssen die Datenbanktabellen zunächst in den Status zurückversetzt werden, den sie hatten, als die Anweisung ursprünglich ausgeführt wurde.

Wenn Sie eine Prüfung mit der Kategorie EXECUTE ausführen, wird der Anweisungstext für statisches und dynamisches SQL aufgezeichnet, ebenso wie Eingabeparametermarken und Hostvariablen. Sie können die Kategorie EXECUTE zur Prüfung mit oder ohne Eingabewerte konfigurieren.

Anmerkung: Globale Variablen werden nicht geprüft.

Die Prüfung von EXECUTE-Ereignissen erfolgt am Ende des Ereignisses (bei SELECT-Anweisungen ist dies nach dem Schließen des Cursors). Der Status, mit dem das Ereignis abgeschlossen wurde, wird ebenfalls gespeichert. Da EXECUTE-Ereignisse bei Abschluss geprüft werden, werden Abfragen mit langer Laufzeit nicht sofort in das Prüfprotokoll eingetragen.

Anmerkung: Die Vorbereitung einer Anweisung wird nicht als Teil der Ausführung betrachtet. Die meisten Berechtigungsprüfungen werden bei der Vorbereitung durchgeführt (z. B. für das Zugriffsrecht SELECT). Dies bedeutet, dass Anweisungen, die während der Vorbereitung aufgrund von Berechtigungsfehlern fehlschlagen, keine EXECUTE-Ereignisse generieren.

Die Felder 'Statement Value Index' (Anweisungswertindex), 'Statement Value Type' (Anweisungswerttyp) und 'Statement Value Data' (Anweisungswertdaten) können für einen gegebenen EXECUTE-Prüfsatz mehrfach erfasst werden. Bei dem durch die Extraktion generierten Berichtsformat listet jeder Prüfsatz mehrere Werte auf. Beim Dateiformat mit begrenzter Satzlänge werden mehrere Zeilen verwendet. Die erste Zeile enthält den Ereignistyp STATEMENT und keine Werte. Die nachfolgenden Zeilen enthalten den Ereignistyp DATA, wobei für jeden Datenwert, der der SQL-Anweisung zugeordnet ist, eine Zeile verwendet wird. Über die Felder für den Ereigniskorrelator und die Anwendungs-ID können Sie STATEMENT-Zeilen und DATA-Zeilen korrelieren. Die Spalten 'Statement Text' (Anweisungstext), 'Statement Isolation Level' (Isolationsstufe der Anweisung) und 'Compilation Environment Description' (Beschreibung der Kompilierungsumgebung) sind in den DATA-Ereignissen nicht enthalten.

Der Anweisungstext und die Eingabedatenwerte, die geprüft werden, werden in die Codepage der Datenbank konvertiert, wenn sie auf Platte gespeichert werden. (Alle geprüften Felder werden in der Codepage der Datenbank gespeichert.) Es wird kein Fehler zurückgemeldet, wenn die Codepage der Eingabedaten nicht mit der Codepage der Datenbank kompatibel ist. Stattdessen werden die nicht konvertierten Daten protokolliert. Da jede Datenbank über ein eigenes Prüfprotokoll verfügt, stellen Datenbanken mit unterschiedlichen Codepages kein Problem dar.

Die Anweisungen ROLLBACK und COMMIT werden geprüft, wenn sie von der Anwendung ausgeführt werden und ebenso wenn sie implizit als Teil eines anderen Befehls, zum Beispiel BIND, ausgeführt werden.

Wenn ein EXECUTE-Ereignis aufgrund eines Zugriffs auf eine geprüfte Tabelle protokolliert wurde, werden alle Anweisungen geprüft, die sich darauf auswirken, welche anderen Anweisungen innerhalb einer UOW ausgeführt werden. Diese Anweisungen sind COMMIT, ROLLBACK, ROLLBACK TO SAVEPOINT und SAVEPOINT.

Feld 'Savepoint ID'

Mithilfe des Feldes 'Savepoint ID' (Sicherungspunkt-ID) können Sie verfolgen, welche Anweisungen von einer Anweisung ROLLBACK TO SAVEPOINT betroffen waren. Bei einer normalen DML-Anweisung (z. B. SELECT, INSERT usw.) wird die aktuelle Sicherungspunkt-ID protokolliert. Bei der Anweisung ROLLBACK TO SAVEPOINT wird hingegen die Sicherungspunkt-ID protokolliert, bis zu der die ROLLBACK-Operation ausgeführt wird. Das heißt, jede Anweisung mit einer Sicherungspunkt-ID, die größer oder gleich dieser ID ist, wird durch ROLLBACK rückgängig gemacht, wie im folgenden Beispiel veranschaulicht. Die Tabelle zeigt die Abfolge der ausgeführten Anweisungen. Alle Ereignisse mit einer Sicherungspunkt-ID größer oder gleich 2 werden rückgängig gemacht. Nur der Wert 3 (aus der ersten Anweisung INSERT) wird in die Tabelle T1 eingefügt.

Tabelle 6. Abfolge von Anweisungen zur Veranschaulichung der Wirkung einer Anweisung ROLLBACK TO SAVEPOINT

Anweisung	Sicherungspunkt-ID
INSERT INTO T1 VALUES (3)	1
SAVEPOINT A	2
INSERT INTO T1 VALUES (5)	2
SAVEPOINT B	3
INSERT INTO T1 VALUES (6)	3
ROLLBACK TO SAVEPOINT A	2
COMMIT	

Option WITH DATA

Nicht alle Eingabewerte werden geprüft, wenn Sie die Option WITH DATA angeben. LOB-, LONG- und XML-Parameter sowie Parameter strukturierter Typen werden als NULL erfasst.

Datums-, Zeit- und Zeitmarkenfelder werden im ISO-Format aufgezeichnet.

Wenn in einer Richtlinie, die Objekten zugeordnet ist, die von der Ausführung der SQL-Anweisung betroffen sind, die Option WITH DATA angegeben wird, und in einer anderen, ebensolchen Objekten zugeordneten Richtlinie die Option WITHOUT DATA angegeben wird, erhält die Option WITH DATA Vorrang, sodass für diese bestimmte Anweisung Daten erfasst werden. Wenn zum Beispiel die einem Benutzer zugeordnete Prüfrichtlinie die Option WITHOUT DATA angibt, die einer Tabelle zugeordnete Prüfrichtlinie jedoch die Option WITH DATA, werden bei einem Zugriff dieses Benutzers auf diese Tabelle die für die Anweisung verwendeten Eingabedaten protokolliert.

Sie haben keine Möglichkeit, die Zeilen zu ermitteln, die durch eine Anweisung zur positionierten Aktualisierung oder positionierten Löschung geändert wurden. Es wird nur die Ausführung der zugrunde liegenden Anweisung SELECT protokolliert, nicht jedoch die einzelne FETCH-Operation. Es ist nicht möglich, anhand eines EXECUTE-Prüfsatzes festzustellen, auf welcher Zeile sich der Cursor befand, als die Anweisung ausgeführt wurde. Wenn die Anweisung zu einem späteren Zeitpunkt wiederholt wird, ist es nur möglich, die Anweisung SELECT auszuführen, um festzustellen, welcher Bereich von Zeilen möglicherweise betroffen war.

Beispiel für die Wiederholung früherer Aktivitäten

Betrachten Sie das folgende Beispiel: Im Rahmen einer umfassenden Sicherheitsrichtlinie ist es für ein Unternehmen erforderlich, die Möglichkeit zu behalten, bis zu sieben Jahren rückwirkend die Auswirkungen einer bestimmten Anforderung für bestimmte Tabellen in der Datenbank zu analysieren. Zu diesem Zweck wird eine Richtlinie zur Archivierung der wöchentlichen Backups und der zugehörigen Protokolldateien eingerichtet, sodass sich die Datenbank für jeden ausgewählten Zeitpunkt rekonstruieren lässt. Es ist erforderlich, dass die Datenbankprüfung ausreichend Informationen zu jeder Anforderung erfasst, die an der Datenbank ausgeführt wird, sodass es möglich ist, jede beliebige Anforderung an der relevanten, für den Zeitpunkt wiederhergestellten Datenbank zu wiederholen. Diese Voraussetzung schließt statische und dynamische SQL-Anweisungen mit ein.

Dieses Beispiel zeigt die Prüfrichtlinie, die eingerichtet sein muss, wenn die SQL-Anweisung ausgeführt wird, und die Schritte, die zur Archivierung der Prüfprotokolle sowie zur späteren Extraktion und Analyse von Daten erforderlich sind.

1. Es muss eine Prüfrichtlinie mit der Kategorie EXECUTE erstellt und der Datenbank zugeordnet werden:

   ```
   CREATE AUDIT POLICY STATEMENTS CATEGORIES EXECUTE WITH DATA
      STATUS BOTH ERROR TYPE AUDIT
   COMMIT

   AUDIT DATABASE USING POLICY STATEMENTS
   COMMIT
   ```

2. Das Prüfprotokoll muss regelmäßig archiviert werden, um eine Archivkopie zu erstellen.

   Die folgende Anweisung sollte vom Sicherheitsadministrator oder einem Benutzer, dem das Zugriffsrecht EXECUTE für die gespeicherte Prozedur SYSPROC.AUDIT_ARCHIVE erteilt wurde, regelmäßig ausgeführt werden, zum Beispiel einmal pro Woche oder einmal pro Tag, je nachdem, wie viel Daten protokolliert werden. Diese archivierten Dateien können nach Bedarf über einen beliebig langen Zeitraum aufbewahrt werden. Die Prozedur AUDIT_ARCHIVE wird mit zwei Eingabeparametern aufgerufen. Dabei handelt es sich um den Pfad zum Archivverzeichnis und den Wert '-2', um anzugeben, dass die Archivierung für alle Member ausgeführt werden soll:

   ```
   CALL SYSPROC.AUDIT_ARCHIVE( '/auditarchive', -2 )
   ```

3. Der Sicherheitsadministrator bzw. ein Benutzer, dem das Zugriffsrecht EXECUTE für die Tabellenfunktion SYSPROC.AUDIT_LIST_LOGS erteilt wurde, verwendet SYSPROC.AUDIT_LIST_LOGS, um z. B. alle verfügbaren Prüfprotokolle von April 2006 zu untersuchen und festzustellen, welche Protokolle die benötigten Daten enthalten:

   ```
   SELECT FILE FROM TABLE(SYSPROC.AUDIT_LIST_LOGS('/auditarchive'))
      AS T WHERE FILE LIKE 'db2audit.dbname.log.0.200604%'
   FILE
   ------------------------------------
   ```

```
...
db2audit.dbname.log.0.20060418235612
db2audit.dbname.log.0.20060419234937
db2audit.dbname.log.0.20060420235128
```

4. Dieser Ausgabe entnimmt der Sicherheitsadministrator, dass sich die benötigten Dateien in nur einer Datei befinden sollten: `db2audit.dbname.log.20060419234937`. Die Zeitmarke zeigt, dass diese Datei am Ende des Tages an dem Tag, den die Prüfer sehen möchten, archiviert wurde.

   Der Sicherheitsadministrator bzw. ein Benutzer, dem das Zugriffsrecht EXECUTE für die gespeicherte Prozedur SYSPROC.AUDIT_DELIM_EXTRACT erteilt wurde, verwendet den Dateinamen als Eingabe für AUDIT_DELIM_EXTRACT, um die Prüfdaten in Dateien mit begrenzter Satzlänge zu extrahieren. Die Prüfdaten in diesen Dateien können in DB2-Datenbanktabellen geladen werden, wo sie analysiert werden können, um die bestimmte Anweisung zu ermitteln, für die sich die Prüfer interessieren. Obwohl die Prüfer nur an einer einzigen SQL-Anweisung interessiert sind, müssen möglicherweise mehrere Anweisungen aus dieser UOW untersucht werden, falls sie eine Auswirkung auf die fragliche Anweisung haben.

5. Zur Wiederholung der Anweisung muss der Sicherheitsadministrator die folgenden Aktionen ausführen:
   - Ermitteln der genauen Anweisung, die aus diesem Prüfsatz ausgeführt werden muss
   - Ermitteln des Benutzers, der die Anweisung aus diesem Prüfsatz ausgeführt hat
   - Erneutes Erstellen der genauen Berechtigungen (einschließlich LBAC-Schutz) des Benutzers zu dem Zeitpunkt, als er die Anweisung ausführte
   - Reproduzieren der Kompilierungsumgebung unter Verwendung der Spalte mit der Beschreibung der Kompilierungsumgebung im Prüfsatz in Kombination mit der Anweisung SET COMPILATION ENVIRONMENT
   - Wiederherstellen der Datenbank in exakt dem Status, den sie zum Zeitpunkt der Ausführung der Anweisung hatte

   Zur Vermeidung von Störungen des Produktionssystem sollten alle Wiederherstellungen der Datenbank und Wiederholungen von Anweisungen auf einem zweiten Datenbanksystem durchgeführt werden. Der Sicherheitsadministrator kann unter der Benutzer-ID des Benutzers, der die Anweisung ausgeführt hat, die Anweisung, wie sie im Anweisungstext vorliegt, mit den in den Anweisungsdatenelementen bereitgestellten Eingabevariablen erneut ausführen.

Wiederholung früherer Aktivitäten aktivieren:

Im Rahmen einer umfassenden Sicherheitsrichtlinie kann es für ein Unternehmen erforderlich sein, die Möglichkeit zu haben, bis zu einer festgelegten Anzahl Jahre rückwirkend die Auswirkungen einer bestimmten Anforderung auf bestimmte Tabellen in der Datenbank zu analysieren.

Vorbereitende Schritte

Ein Unternahmen muss eine Richtlinie zur Archivierung der wöchentlichen Backups und der zugehörigen Protokolldateien einrichten, sodass die Datenbank für jeden ausgewählten Zeitpunkt rekonstruiert werden kann.

Informationen zu diesem Vorgang

Damit jede beliebige Anforderung für die betreffende wiederhergestellte Datenbank an jedem beliebigen zukünftigen Zeitpunkt wiederholt und analysiert werden kann, müssen genügend Datenbankprüfdaten zu jeder für die Datenbank ausgeführten Anforderung erfasst werden. Diese Voraussetzung kann sowohl statistische als auch dynamische SQL-Anweisungen mit einschließen. Wenn die EXECUTE-Kategorie mit Daten (WITH DATA) protokolliert wird, enthält sie die erforderlichen Informationen zum Wiederholen früherer SQL-Anweisungen, sofern die Daten in der Datenbank bis zu dem Punkt wiederhergestellt werden, an dem die Anweisung gestartet wurde.

Einschränkungen

Die folgenden Berechtigungen und Zugriffsrechte sind erforderlich:

- Die Berechtigung SECADM ist zum Erstellen der Prüfrichtlinien erforderlich
- Das Zugriffsrecht EXECUTE ist für die Prüfroutinen und -prozeduren erforderlich

Vorgehensweise

Gehen Sie als Benutzer mit der Berechtigung SECADM wie folgt vor, um die Wiederholung früherer Aktivitäten zu aktivieren:

1. Eine Prüfrichtlinie für die Kategorie EXECUTE muss erstellt und auf die Datenbank angewendet werden.

   ```
   CREATE AUDIT POLICY STATEMENTS CATEGORIES EXECUTE WITH DATA
      STATUS BOTH ERROR TYPE AUDIT
   COMMIT
   AUDIT DATABASE USING POLICY STATEMENTS
   COMMIT
   ```

2. Das Prüfprotokoll muss regelmäßig archiviert werden, um eine Archivkopie zu erstellen. Führen Sie zum Archivieren des Prüfprotokolls den folgenden Befehl regelmäßig aus und geben Sie dabei den Pfad zum Archivierungsverzeichnis und den Wert '-2' an, wenn die Archivierung für alle Member ausgeführt werden soll:

   ```
   CALL SYSPROC.AUDIT_ARCHIVE( '/auditarchive', -2 )
   ```

3. Überprüfen Sie, dass die Prüfprotokolldateien erstellt wurden. Die archivierten Dateien werden so viele Jahre aufbewahrt, wie die Geschäftsrichtlinie des Unternehmens dies vorsieht. Führen Sie den folgenden Befehl aus, um die Prüfprotokolldateien zu überprüfen:

   ```
   SELECT FILE FROM SESSION.AUDIT_ARCHIVE_RESULTS
   ```

Ergebnisse

Ihre Umgebung ist jetzt so eingerichtet, dass die erforderlichen Daten und Informationen für die zukünftige Wiederholung protokollierter Datenbankaktivtäten archiviert werden.

Wiederholung früherer Datenbankaktivitäten:

Die Wiederholung früherer Datenbankaktivitäten ist möglich, wenn alle erforderlichen Daten, Protokolle und Informationen verfügbar sind. Dieses Referenzthema zeigt an einem Beispiel, wie ein Sicherheitsadministrator (Benutzer mit der Berechtigung SECADM) frühere Datenbankaktivitäten wiederholen kann.

Beschreibung

Es kann vorkommen, dass die Prüfer eines Unternehmens analysieren möchten, welche Aktivitäten ein bestimmter Benutzer in der Vergangenheit durchgeführt hat. Der Sicherheitsadministrator (SECADM-Benutzer) kann mithilfe von Backup-Images der Datenbank, Backup-Protokollen und Prüfprotokollen die betreffende Datenbank rekonstruieren und die Aktivität wiederholen, die die Prüfer analysieren möchten. Angenommen, es geht um die Aktivitäten eines bestimmten Benutzers am 19. April 2006. Das folgende Beispiel zeigt, mit welchem Verarbeitungsablauf ein SECADM-Benutzer die Prüfer bei der Durchführung der Analyse unterstützen kann.

Beispiel

1. Der SECADM-Benutzer sucht mit dem Parameter AUDIT_LIST_LOGS alle verfügbaren Prüfprotokolle für April 2006.

```
SELECT FILE FROM TABLE(SYSPROC.AUDIT_LIST_LOGS('/auditarchive'))
   AS T WHERE FILE LIKE 'db2audit.db.sample.log.0.200604%'
FILENAME
----------------------------------------
...
db2audit.db.sample.log.0.20060418235612
db2audit.db.sample.log.0.20060419234937
db2audit.db.sample.log.0.20060420235128
```

2. Dieser Ausgabe entnimmt der SECADM-Benutzer, dass sich die benötigten Protokolle in der Datei db2audit.db.sample.log.20060419234937 befinden sollten. Das Protokoll wurde bei Geschäftsschluss am 19. April 2006 erstellt.
3. Diese Datei wird als Eingabe für die gespeicherte Prozedur SYSPROC.AUDIT_DELIM_EXTRACT verwendet. Folgende Argumente werden an die Prozedur übergeben:
   - Begrenzungszeichen (Standardwert),
   - Ausgabepfad,
   - Pfad zu den archivierten Prüfprotokollen,
   - Dateinamenfilter zum Festlegen der zu extrahierenden Dateien,
   - Status für jede zu extrahierende Kategorie (in diesem Fall nur die Kategorie EXECUTE)

```
CALL SYSPROC.AUDIT_DELIM_EXTRACT( '', '', '/auditarchive',
   'db2audit.db.sample.log.0.20060419234937',
   'category execute' )
```

4. Die Prüfdaten befinden sich nun in Dateien mit begrenzter Satzlänge. Der SECADM-Benutzer lädt die Prüfdaten aus der Kategorie EXECUTE in die Tabelle AUDITDATA.EXECUTE. Diese Tabelle kann durch Ausführen der folgenden Befehle erstellt werden:

```
db2 CONNECT TO sample
db2 SET CURRENT SCHEMA AUDITDATA
db2 -tvf sqllib/misc/db2audit.ddl
```

5. Laden Sie nun die Daten aus execute.del in die Tabelle AUDITDATA.EXECUTE. Führen Sie dazu den folgenden Befehl aus:

```
db2 LOAD FROM FILE execute.del OF DEL MODIFIED BY LOBSINFILE INSERT INTO AUDITDATA.EXECUTE
```

6. Damit verfügt der SECADM-Benutzer über alle Prüfdaten aus den Prüftabellen in AUDITDATA schema. Diese Daten können nun analysiert werden, um die bestimmte Anweisung zu ermitteln, für die sich die Prüfer interessieren.

Anmerkung: Obwohl die Prüfer nur an einer einzigen SQL-Anweisung interessiert sind, müssen möglicherweise mehrere Anweisungen aus dieser UOW untersucht werden, falls sie eine Auswirkung auf die fragliche Anweisung haben.

7. Zur Wiederholung der Anweisung müssen die folgenden Aktionen ausgeführt werden:
   - Ermitteln der genauen Anweisung aus dem Prüfsatz
   - Ermitteln des Benutzers, der die Anweisung ausgeführt hat, aus dem Prüfsatz
   - Erneutes Erstellen der genauen Berechtigungen (einschließlich LBAC-Schutz) des Benutzers zu dem Zeitpunkt, als er die Anweisung ausführte
   - Reproduzieren der Kompilierungsumgebung unter Verwendung der Spalte mit der Beschreibung der Kompilierungsumgebung im Prüfsatz in Kombination mit der Anweisung SET COMPILATION ENVIRONMENT
   - Wiederherstellen der Datenbank in genau dem Status, den sie zum Zeitpunkt der Ausführung der Anweisung hatte

   Anmerkung: Zur Vermeidung von Störungen des Produktionssystems sollten alle Wiederherstellungen der Datenbank und Wiederholungen von Anweisungen auf einem zweiten Datenbanksystem durchgeführt werden.
8. Der SECADM-Benutzer müsste eine aktualisierende Wiederherstellung bis zum Ausführungsbeginn der Anweisung durchführen. Die lokale Startzeit der Anweisung (local_start_time) ist im EXECUTE-Prüfsatz enthalten. Verwenden Sie als Beispiel den folgenden EXECUTE-Prüfsatz:

```
timestamp=2006-04-10-13.20.51.029203;
  category=EXECUTE;
  audit event=STATEMENT;
  event correlator=1;
  event status=0;
  database=SAMPLE;
  userid=smith;
  authid=SMITH;
  session authid=SMITH;
  application id=*LOCAL.prodrig.060410172044;
  application name=myapp;
  package schema=NULLID;
  package name=SQLC2F0A;
  package section=201;
  uow id=2;
  activity id=3;
  statement invocation id=0;
  statement nesting level=0;
  statement text=SELECT * FROM DEPARTMENT WHERE DEPTNO = ? AND DEPTNAME = ?;
  statement isolation level=CS;
  compilation environment=
    isolation level=CS
    query optimization=5
    min_dec_div_3=NO
    degree=1
    sqlrules=DB2
    refresh age=+00000000000000.000000
    schema=SMITH
    maintained table type=SYSTEM
    resolution timestamp=2006-04-10-13.20.51.000000
    federated asynchrony=0;
  value index=0;
  value type=CHAR;
  value data=C01;
  value index=1;
  value type=VARCHAR;
  value index=INFORMATION CENTER;  local_start_time=2006-04-10-13.20.51.021507;
```

Die Anweisung ROLLFORWARD sieht wie folgt aus:

```
ROLLFORWARD DATABASE sample TO 2006-04-10-13.20.51.021507 USING LOCAL TIME AND COMPLETE
```

9. Außerdem muss die Kompilierungsumgebung festgelegt werden. Die Variable für die Kompilierungsumgebung kann mit der Anweisung SET COMPILATION ENVIRONMENT festgelegt werden. Der SECADM-Benutzer kann unter der Benutzer-ID des Benutzers, der die Anweisung ausgeführt hat, die Anweisung, wie sie im Anweisungstext vorliegt, mit den in den Anweisungsdatenelementen bereitgestellten Eingabevariablen erneut ausführen. Das folgende Beispielprogramm in der Programmiersprache C mit eingebettetem SQL legt die Kompilierungsumgebung (COMPILATION ENVIRONMENT) fest und wiederholt die Anweisung SELECT, die die Prüfer analysieren möchten:

```
EXEC SQL INCLUDE SQLCA;

EXEC SQL  BEGIN DECLARE SECTION;
     SQL TYPE IS BLOB(1M) hv_blob;
EXEC SQL  END DECLARE SECTION;

EXEC SQL DECLARE c1 CURSOR FOR SELECT COMPENVDESC FROM AUDITDATA.EXECUTE
TIMESAMP= '2006-04-10-13.20.51.029203';
EXEC SQL DECLARE c2 CURSOR FOR SELECT * FROM DEPARTMENT WHERE DEPTNO = 'C01'
AND DEPTNAME = 'INFORMATION CENTER';
EXEC SQL OPEN c1;

EXEC SQL FETCH c1 INTO :hv_blob;

EXEC SQL SET COMPILATION ENVIRONMENT :hv_blob;

EXEC SQL OPEN c2;

....

EXEC SQL CLOSE c1;
EXEC SQL CLOSE c2;
```

Verwalten der Prüffunktion

Funktionsweise der Prüffunktion

Dieser Abschnitt enthält Hintergrundinformationen, die Ihnen beim Verständnis helfen sollen, wie der zeitliche Ablauf beim Schreiben von Prüfsätzen in das Protokoll die Datenbankleistung beeinflussen kann, wie Fehler behandelt werden können, die in der Prüffunktion auftreten, und wie Prüfsätze in den verschiedenen Situationen generiert werden.

Steuern des Zeitpunkts für das Schreiben von Prüfsätzen in das aktive Protokoll

Das Schreiben der Prüfsätze in das aktive Protokoll kann synchron oder asynchron zum Auftreten der Ereignisse erfolgen, die die Generierung dieser Prüfsätze auslösen. Der Wert des Konfigurationsparameters **audit_buf_sz** des Datenbankmanagers legt fest, wann das Schreiben der Prüfsätze stattfindet.

Wenn der Parameter **audit_buf_sz** den Wert null (0) hat, erfolgt das Schreiben synchron. Das den Prüfsatz auslösende Ereignis bleibt im Wartestatus bis der Prüfsatz auf die Platte geschrieben ist. Die Wartezeit beim Schreiben der einzelnen Prüfsätze wirkt sich nachteilig auf die Verarbeitungsleistung der DB2-Datenbank aus.

Wenn der Wert des Parameters **audit_buf_sz** größer als null ist, wird der Schreibvorgang asynchron ausgeführt. Wenn der Parameterwert für **audit_buf_sz** größer als null ist, entspricht er der Anzahl an 4-KB-Seiten, die zum Erstellen eines inter-

nen Puffers verwendet werden. In dem internen Puffer werden Prüfsätze zwischengespeichert, bis sie gruppenweise auf die Platte geschrieben werden. Die Anweisung, die den aus einem Prüfereignis resultierenden Prüfsatz generiert, wartet nicht, bis der Prüfsatz auf die Platte geschrieben ist, sondern kann die Verarbeitung ohne Verzögerung fortsetzen.

Beim asynchronen Schreiben verbleiben die Prüfsätze gegebenenfalls für einige Zeit in einem nur teilweise gefüllten Puffer. Damit dies nicht über zu lange Zeiträume der Fall ist, erzwingt der Datenbankmanager das regelmäßige Schreiben der Prüfsätze auf die Platte. Ein berechtigter Benutzer der Prüffunktion kann den Prüfpuffer auch mit einer expliziten Anforderung (FLUSH) leeren. Die Puffer werden zudem bei einer Archivierungsoperation automatisch geleert.

Je nachdem, ob der Schreibvorgang synchron oder asynchron erfolgt, ergeben sich Unterschiede beim Auftreten von Fehlern. Im asynchronen Modus gehen möglicherweise einige Datensätze verloren, da die Prüfsätze gepuffert werden, bevor sie auf die Platte geschrieben werden. Im synchronen Modus geht, wenn überhaupt, ein Datensatz verloren, da durch den Fehler höchstens ein Prüfsatz nicht geschrieben werden kann.

Verwalten von Fehlern der Prüffunktion

Die Einstellung des Prüffunktionsparameters ERRORTYPE legt fest, wie auftretende Fehler zwischen dem DB2-Datenbanksystem und der Prüffunktion verwaltet werden. Wenn die Prüffunktion aktiv und der Prüffunktionsparameter ERRORTYPE auf `AUDIT` gesetzt ist, wird die Prüffunktion genau so behandelt wie jede andere Komponente der DB2-Datenbank. Ein Prüfsatz muss geschrieben werden (im synchronen Modus auf die Platte, im asynchronen Modus in den Prüfpuffer), damit ein Prüfereignis, das sich auf eine Anweisung bezieht, als erfolgreich eingestuft wird. Bei jedem in diesem Modus festgestellten Fehler wird für die Anweisung, die einen Prüfsatz generiert hat, ein negativer SQLCODE-Wert an die Anwendung zurückgegeben.

Ist der Parameter ERRORTYPE auf `NORMAL` gesetzt, werden alle durch **`db2audit`** verursachten Fehler ignoriert und der SQLCODE-Wert der Operation wird zurückgegeben.

In verschiedenen Situationen generierte Prüfsätze

In Abhängigkeit von der API bzw. Abfrageanweisung und den Prüfeinstellungen können für ein bestimmtes Ereignis kein Prüfsatz, ein Prüfsatz oder mehrere Prüfsätze generiert werden. Beispiel: Eine SQL-Anweisung UPDATE mit einer Unterabfrage SELECT kann zur Generierung eines Prüfsatzes mit den Ergebnissen der Berechtigungsprüfung für das Zugriffsrecht UPDATE einer Tabelle und zur Generierung eines weiteren Prüfsatzes mit den Ergebnissen der Berechtigungsprüfung für das Zugriffsrecht SELECT einer Tabelle führen.

Für dynamische DML-Anweisungen (DML = Data Manipulation Language) werden Prüfsätze für alle Berechtigungsprüfungen zum Zeitpunkt der Anweisungsvorbereitung (Prepare) generiert. Die erneute Verwendung dieser Anweisungen durch denselben Benutzer wird nicht erneut überprüft, weil zu diesem Zeitpunkt keine Berechtigungsprüfung stattfindet. Wenn jedoch eine der Katalogtabellen mit Berechtigungsinformationen geändert wurde, werden in der nächsten UOW die Berechtigungen für die im Cache gespeicherten dynamischen SQL- oder XQuery-Anweisungen erneut geprüft und mindestens ein neuer Prüfsatz erstellt.

Für ein Paket, das ausschließlich statische DML-Anweisungen enthält, ist die Berechtigungsprüfung, mit der geprüft wird, ob ein Benutzer die erforderliche Berechtigung zum Ausführen des Pakets hat, das einzige prüfbare Ereignis, das einen Prüfsatz generieren könnte. Die Berechtigungsprüfung und die gegebenenfalls erforderliche Prüfsatzerstellung für die statischen SQL- oder XQuery-Anweisungen des Pakets werden beim Vorkompilieren oder Binden des Pakets durchgeführt. Die Ausführung der statischen SQL- bzw. XQuery-Anweisungen in dem Paket kann mit der Kategorie EXECUTE (Ausführen) geprüft werden. Wird für ein Paket entweder explizit vom Benutzer oder implizit vom System ein Rebind durchgeführt, werden Prüfsätze für die von den statischen SQL- oder XQuery-Anweisungen benötigten Berechtigungsprüfungen generiert.

Für Anweisungen, bei denen die Berechtigungsprüfung zum Zeitpunkt der Ausführung erfolgt (z. B. Data Definition Language (DDL) oder GRANT- und REVOKE-Anweisungen), werden bei jeder Verwendung dieser Anweisungen Prüfsätze generiert.

Anmerkung: Bei der Ausführung von DDL ist die aufgezeichnete Abschnittsnummer für alle Ereignisse (ausgenommen Kontextereignisse) im Prüfsatz null (0), und zwar unabhängig von der tatsächlichen Abschnittsnummer der Anweisung.

Tipps und Techniken für die Prüffunktion

Es empfiehlt sich, bei der Verwaltung der Prüfdaten für eine regelmäßige Archivierung des Prüfprotokolls zu sorgen, beim Erstellen einer Prüfrichtlinie den Fehlertyp AUDIT zu verwenden und die übrigen Tipps in diesem Abschnitt umzusetzen.

Archivieren des Prüfprotokolls

Sie sollten das Prüfprotokoll regelmäßig archivieren. Die Archivierung des Prüfprotokolls versetzt das aktuelle Prüfprotokoll in ein Archivverzeichnis, während der Server gleichzeitig mit dem Schreiben in ein neues, aktives Prüfprotokoll beginnt. Der Name jeder archivierten Protokolldatei enthält eine Zeitmarke, an der Sie interessante Protokolldateien bei einer späteren Analyse identifizieren können.

Für die langfristige Speicherung ist möglicherweise eine Komprimierung von Gruppen archivierter Dateien sinnvoll.

Für archivierte Prüfprotokolle, die für Sie von keinem Interesse mehr sind, kann der Instanzeigner einfach die Dateien aus dem Betriebssystem löschen.

Fehlerbehandlung

Wenn Sie eine Prüfrichtlinie erstellen, sollten Sie den Fehlertyp AUDIT verwenden, sofern Sie nicht nur eine Testprüfrichtlinie erstellen. Wenn zum Beispiel der Fehlertyp auf AUDIT gesetzt ist und ein Fehler auftritt, zum Beispiel durch nicht ausreichenden Plattenspeicherplatz, wird eine Fehlernachricht zurückgegeben. Die Fehlerbedingung muss behoben werden, bevor alle weiteren, von der Prüffunktion überwachten Aktionen fortgesetzt werden können. Wenn der Fehlertyp jedoch auf NORMAL gesetzt war, würde die Protokollierung einfach fehlschlagen, ohne einen Fehler an den Benutzer zurückzumelden. Der Betrieb wird so fortgesetzt, als wäre der Fehler nicht aufgetreten.

Falls während der Archivierung ein Problem auftritt, zum Beispiel, wenn der Plattenspeicherplatz im Archivpfad nicht ausreicht oder der Archivpfad nicht vorhanden ist, schlägt der Archivierungsprozess fehl. In diesem Fall wird eine vorläufige Protokolldatei mit der Dateierweiterung `.bk` im Datenpfad für das Prüfprotokoll

generiert. Beispiel: db2audit.instance.log.0.20070508172043640941.bk. Wenn das Problem beseitigt ist (durch Zuordnen eines ausreichenden Plattenspeicherbereichs im Archivpfad bzw. durch Erstellen des Archivpfads) müssen Sie diese vorläufige Protokolldatei in den Archivpfad versetzen. Anschließend können Sie diese Datei genauso wie ein erfolgreich archiviertes Protokoll behandeln.

Einschränkungen für DDL-Anweisungen

Einige Anweisungen der Datendefinitionssprache (DDL-Anweisungen), die als exklusive SQL-Anweisungen der Prüffunktion (exklusive AUDIT-Anweisungen) bezeichnet werden, werden erst bei der nächsten UOW (Unit of Work, Arbeitseinheit) wirksam. Daher gilt die Empfehlung, die Anweisung COMMIT unmittelbar nach jeder dieser Anweisungen zu verwenden.

Bei den exklusiven SQL-Anweisungen der Prüffunktion handelt es sich um folgende Anweisungen:

- AUDIT
- CREATE AUDIT POLICY, ALTER AUDIT POLICY und DROP AUDIT POLICY
- DROP ROLE und DROP TRUSTED CONTEXT, wenn die Rolle bzw. der gesicherte Kontext, die bzw. der gelöscht wird, einer Prüfrichtlinie zugeordnet ist

Mögliche Änderung des Tabellenformats zum Speichern archivierter Daten

Mit der gespeicherten Prozedur SYSPROC.AUDIT_DEL_EXTRACT kann der Sicherheitsadministrator, bzw. der Systemadministrator mit dem Befehl **db2audit extract**, Prüfsätze aus den archivierten Prüfprotokolldateien in Dateien mit begrenzter Satzlänge extrahieren. Sie können die Prüfdaten aus den Dateien mit begrenzter Satzlänge zur Analyse in DB2-Datenbanktabellen laden. Das Format der Tabellen, die Sie zur Aufnahme der Prüfdaten erstellen müssen, kann sich von Release zu Release ändern.

Wichtig: Das Script db2audit.ddl erstellt Tabellen im korrekten Format, um die Prüfsätze aufzunehmen. Sie sollten die Ausführung des Scripts db2audit.ddl für jedes Release vorsehen, da möglicherweise neue Spalten hinzugefügt oder die Größe einer vorhandenen Spalte geändert werden.

Verwenden von CHECKING-Ereignissen

Beim Arbeiten mit CHECKING-Ereignissen wird meist das Feld für den Objekttyp im Prüfsatz überprüft, um festzustellen, ob die Benutzer-ID, die auf das Objekt zuzugreifen versucht, über das erforderliche Zugriffsrecht bzw. die erforderliche Berechtigung verfügt. Beispiel: Wenn ein Benutzer versucht, mit ALTER eine Tabellenspalte hinzuzufügen, weist der Prüfsatz des CHECKING-Ereignisses darauf hin, dass ein Zugriff mit „ALTER" auf ein Objekt des Typs „TABLE" (Tabelle) versucht wurde (nicht Spalte, weil in diesem Fall die Tabellenzugriffsrechte geprüft werden).

Muss beim Überprüfen jedoch festgestellt werden, ob die erforderliche Datenbankberechtigung vorhanden ist, mit der eine Benutzer-ID eine CREATE- oder BIND-Operation an einem Objekt ausführen oder ein Objekt löschen (DROP) kann, dann erfolgt zwar eine Prüfung in der Datenbank, aber im Feld für den Objekttyp wird das zu erstellende, zu bindende oder zu löschende Objekt angegeben (und nicht die Datenbank).

Bei der Indexerstellung für eine Tabelle ist das Zugriffsrecht zum Erstellen eines Index erforderlich, deshalb wird im Prüfsatz des CHECKING-Ereignisses der Zugriffstyp „index" und nicht „create" angegeben.

Für das Binden eines Pakets erstellte Prüfsätze

Beim Binden eines bereits vorhandenen Pakets wird ein OBJMAINT-Prüfsatz für die DROP-Operation des Pakets erstellt und danach ein weiterer OBJMAINT-Prüfsatz für die CREATE-Operation der neuen Paketkopie.

Verwenden von CONTEXT-Ereignisdaten nach ROLLBACK

Die Datendefinitionssprache (Data Definition Language, DDL) kann OBJMAINT- oder SECMAINT-Ereignisse generieren, die als erfolgreich protokolliert werden. Es kann jedoch vorkommen, dass nach der Protokollierung des Ereignisses durch einen nachfolgenden Fehler ein Rollback ausgelöst wird. Dadurch würde das Objekt als nicht erstellt bzw. die GRANT- oder REVOKE-Operation als unvollständig ausgewiesen. In diesem Fall erweist sich die Verwendung von CONTEXT-Ereignissen als hilfreich. Die Prüfsätze von CONTEXT-Ereignissen, insbesondere die abschließende Anweisung des Ereignisses, geben an, wie die versuchte Operation beendet wurde.

Ladebegrenzer

Beim Extrahieren von Prüfsätzen in begrenztem Format, das sich zum Laden in eine DB2-Datenbanktabelle eignet, ist darauf zu achten, dass im Textfeld der Anweisung der richtige Begrenzer verwendet wird. Dies kann beim Extrahieren der Datei mit begrenzter Satzlänge zum Beispiel mit folgendem Befehl erreicht werden:

```
db2audit extract delasc delimiter ladebegrenzer
```

Dabei kann *ladebegrenzer* ein Einzelzeichen (z. B. ") oder ein Vierbytezeichen sein, das eine Hexadezimalzahl darstellt (z. B. „0xff"). Folgende Beispiele zeigen gültige Befehle:

```
db2audit extract delasc
db2audit extract delasc delimiter !
db2audit extract delasc delimiter 0xff
```

Wenn beim Extrahieren nicht der Standardladebegrenzer (") verwendet wurde, sollte die Option **MODIFIED BY** im Befehl **LOAD** verwendet werden. Das folgende Beispiel zeigt einen abgekürzten Befehl **LOAD** mit dem Begrenzer „0xff":

```
db2 load from context.del of del modified by chardel0xff replace into ...
```

Dadurch wird der Standardzeichenfolgebegrenzer " (Anführungszeichen) für LOAD außer Kraft gesetzt.

Sicherheitsmodell für den Befehl 'db2cluster'

Der Befehl **db2cluster** ist die Hauptschnittstelle für die Untersuchung der DB2-Cluster-Services. Er interagiert in dieser Eigenschaft sowohl mit dem Cluster-Manager als auch mit dem Cluster für gemeinsam genutzte Dateisysteme, die für IBM DB2 pureScale Feature bereitgestellt werden. Die Optionen des Befehls **db2cluster**, die einem Benutzer zur Verfügung stehen, hängen von der Berechtigung des Benutzers ab.

In Hinblick auf das Sicherheitsmodell für den Befehl **db2cluster** gibt es drei Benutzergruppen, die je nach dem Typ von Tasks, die sie wahrscheinlich ausführen, unterschieden werden:

- Jeder Benutzer mit einer Benutzer-ID auf dem System

  Benutzer dieser Gruppe können den Befehl **db2cluster** verwenden, um Informationen zur DB2 pureScale-Instanz aufzulisten. Sie dürfen jedoch keine Änderungen vornehmen.
- Gruppe SYSADM, SYSCTL oder SYSMAINT

  Benutzer dieser Gruppe können den Befehl **db2cluster** verwenden, um die Instanz betriebsbereit zu halten sowie um einige Verwaltungstasks für den Cluster-Manager auszuführen. Per Definition ist ein Benutzer dieser Gruppe entweder die Benutzer-ID der Instanz, ein Mitglied der Primärgruppe des Instanzeigners oder ein Mitglied einer anderen Gruppe des Instanzeigners als der Primärgruppe. Für DB2 wird empfohlen, alltägliche Aktivitäten mithilfe einer Benutzer-ID mit Mitgliedschaft in einer nicht primären Gruppe des Instanzeigners durchzuführen.
- Administrator der DB2-Cluster-Services

  Benutzer in dieser Gruppe haben keinen Bedarf an Zugriff auf Daten in der Datenbank. Diese Rolle ist eine Verwaltungsrolle für folgende Zwecke:
  - Installation und Konfiguration des DB2-Cluster-Services-Teils von DB2
  - Verwaltung von im Cluster konfigurierten Instanzen in der Clusterdomäne und Verwaltung des Clusters für gemeinsam genutzte Dateisysteme

  Der Administrator von DB2-Cluster-Services ist ein Endbenutzer mit Zugriff auf eine Benutzer-ID des Benutzers 'root' für das Betriebssystem (z. B. ein Betriebssystemadministrator). DB2-Cluster-Services kann für alle Clusterumgebungen gelten, ganz gleich, ob Sie DB2 pureScale Feature oder eine Umgebung mit partitionierten Datenbanken mit integrierter hoher Verfügbarkeit verwenden. Daher bieten Rollen wie DBADM, SECADM, SQLADM, WLMADM, EXPLAIN, ACCESSCTRL und DATAACCESS, die auf Datenbanken anwendbar sind, nicht die angemessene Berechtigungsstufe für die Clusterverwaltung. Der Administrator von DB2-Cluster-Services kann dieselbe Person sein wie eine Person mit einer Benutzer-ID in der Gruppe SYSADM, SYSCTL oder SYSMAINT.

  Anmerkung: Die Tatsache, dass ein Benutzer über die Berechtigung SYSADM verfügt, bedeutet allein noch nicht, dass der Benutzer über Administratorberechtigungen für das Betriebssystem verfügt.

Cluster-Manager-Tasks für 'db2cluster'

- Jeder Benutzer mit einer Benutzer-ID auf dem System kann Informationen zum aktuellen Status der Clusterdomäne mit den Optionen **-list** und **-verify** abrufen.
- Benutzer der Gruppe SYSADM, SYSMAINT oder SYSCTL können die bevorzugte primäre Cluster-Caching-Funktion mit den Optionen **-list** und **-set** abfragen und ändern. Darüber hinaus können diese Benutzer die Option **-clear -alert** verwenden, um Alerts für beliebige Hosts, Member oder Cluster-Caching-Funktionen in der aktuellen Instanz (definiert durch die Registrierdatenbankvariable DB2INSTANCE) zu löschen. Benutzer in dieser Gruppe können außerdem Clusterressourcen erstellen und löschen und das Cluster-Manager-Ressourcenmodell reparieren. Es wird jedoch dringend empfohlen, diese Tasks nur unter Anleitung eines DB2-Kundendienstmitarbeiters durchzuführen.
- Der Administrator der DB2-Cluster-Services kann Verwaltungstasks ausführen, die die DB2-Cluster-Services als Ganzes über alle Clusterinstanzen hinweg auf allen Hosts in der Clusterdomäne betreffen. Dieser Benutzer kann mithilfe der

Option **-set** verschiedene Konfigurationstasks durchführen, wie das Festlegen der Tiebreakereinheit und der Zeitspanne zum Erkennen von Hostfehlern. Außerdem kann der Administrator von DB2-Cluster-Services mithilfe der Option **-enter** Wartungstasks, wie das Versetzen von Hosts in den Wartungsmodus, durchführen oder mithilfe der Option **-commit** Änderungen oder Aktualisierungen im Cluster-Manager festschreiben. Zudem kann er erweiterte Wartungsoperationen auf der Peerdomäne des Cluster-Managers durchführen, wie das Erstellen, Löschen, Starten oder Stoppen der Domäne und das Hinzufügen oder Entfernen von Hosts. Es wird jedoch dringend empfohlen, diese Tasks nur unter Anleitung eines DB2-Kundendienstmitarbeiters durchzuführen.

Tasks in Bezug auf das gemeinsam genutzte Dateisystem für 'db2cluster'

- Jeder Benutzer mit einer Benutzer-ID auf dem System kann Informationen zum aktuellen Status der Clusterdomäne mit den Optionen **-list** und **-verify** abrufen. Diese Benutzer können außerdem eine ganze Reihe von Dateisystemoperationen mit den Optionen des Befehls **db2cluster** ausführen. Die genauen Operationen, die sie ausführen können, werden durch reguläre Dateisystemberechtigungen einschränkt. Solange die Benutzer-ID, die den Befehl ausführt, Lese- und Schreibzugriff für die verwendete Einheit besitzt, kann der Benutzer Dateisysteme erstellen und Platten hinzufügen. Wenn ein Dateisystem einmal erstellt oder durch Mount angehängt wurde, ist der Zugriff auf dieses Dateisystem auf die Benutzer-ID, die es erstellt hat, sowie auf den Administrator der DB2-Cluster-Services beschränkt. Das heißt, dass nur diese Benutzer ein Dateisystem entfernen, löschen oder neu ausgleichen können. Wie bei einem normalen Dateisystem auch kann entweder die Benutzer-ID, die das System erstellt hat, oder der DB2-Cluster-Services-Administrator Verzeichnisse erstellen, auf die andere Benutzer Zugang haben.
- Der Administrator der DB2-Cluster-Services kann Verwaltungstasks ausführen, die die DB2-Cluster-Services als Ganzes über alle Clusterinstanzen hinweg auf allen Hosts in der Clusterdomäne betreffen. Dieser Benutzer kann Änderungsoptionen für die Tiebreakereinheit mit der Option **-set** ausführen. Außerdem kann der Administrator von DB2-Cluster-Services mithilfe der Option **-enter** Wartungstasks, wie das Versetzen von Hosts in den Wartungsmodus, durchführen oder mithilfe der Option **-commit** Änderungen oder Aktualisierungen im gemeinsam genutzten Dateisystem festschreiben. Zudem kann er erweiterte Wartungsoperationen im gemeinsam genutzten Dateisystemcluster durchführen, wie das Erstellen, Löschen, Starten oder Stoppen der Domäne und das Hinzufügen oder Entfernen von Hosts. Es wird jedoch dringend empfohlen, diese Tasks nur unter Anleitung eines DB2-Kundendienstmitarbeiters durchzuführen.

Kapitel 2. Rollen

Rollen vereinfachen die Verwaltung und das Management von Zugriffsrechten, indem sie eine äquivalente Funktionalität zu Gruppen bieten, jedoch ohne deren Einschränkungen.

Eine Rolle ist ein Datenbankobjekt, das ein oder mehrere Zugriffsrechte zusammenfasst und Benutzern, Gruppen, PUBLIC oder anderen Rollen durch eine Anweisung GRANT oder einem gesicherten Kontext durch eine Anweisung CREATE TRUSTED CONTEXT oder ALTER TRUSTED CONTEXT erteilt werden kann. Eine Rolle kann für das Verbindungsattribut SESSION_USER ROLE in einer Auslastungsdefinition (Workloaddefinition) angegeben werden.

Rollen bieten verschiedene Vorteile, die das Management von Zugriffsrechten in einem Datenbanksystem vereinfachen:

- Sicherheitsadministratoren können den Zugriff auf ihre Datenbanken auf eine Weise steuern, die die Struktur ihrer Unternehmen widerspiegelt. (Sie können Rollen in der Datenbank erstellen, die die Aufgabenbereiche in ihren Unternehmen direkt abbilden.)
- Benutzern wird die Zugehörigkeit zu den Rollen erteilt, die ihren Zuständigkeiten entsprechen. Wenn sich Zuständigkeiten ändern, lässt sich die Rollenzugehörigkeit problemlos neu erteilen und entziehen.
- Die Zuordnung von Zugriffsrechten wird vereinfacht. Anstatt die gleiche Gruppe von Zugriffsrechten jedem einzelnen Benutzer mit einem bestimmten Aufgabenbereich zu erteilen, kann der Administrator diese Gruppe von Zugriffsrechten einer Rolle erteilen, die diesen Aufgabenbereich darstellt, und anschließend diese Rolle jedem Benutzer mit diesem Aufgabenbereich erteilen.
- Die Zugriffsrechte einer Rolle können aktualisiert werden, und alle Benutzer, denen diese Rolle erteilt wurde, empfangen die Aktualisierung. Der Administrator braucht die Zugriffsrechte nicht für jeden Benutzer einzeln zu aktualisieren.
- Die Zugriffsrechte und Berechtigungen, die Rollen erteilt sind, werden bei jeder Erstellung von Sichten, Triggern, MQTs (Materialized Query Tables), statischem SQL und SQL-Routinen verwendet, während die Zugriffsrechte und Berechtigungen, die Gruppen (direkt oder indirekt) erteilt sind, nicht verwendet werden.

  Dies liegt daran, dass das DB2-Datenbanksystem nicht feststellen kann, wenn sich die Zugehörigkeit zu einer Gruppe ändert, da die Gruppe von der Software eines anderen Anbieters (z. B. dem Betriebssystem oder einem LDAP-Verzeichnis) verwaltet wird. Da Rollen innerhalb der Datenbank verwaltet werden, kann das DB2-Datenbanksystem feststellen, wenn sich eine Berechtigung ändert, und sich entsprechend verhalten. Rollen, die Gruppen erteilt sind, werden aus demselben Grund nicht berücksichtigt, aus dem Gruppen nicht berücksichtigt werden.
- Alle Rollen, die einem Benutzer zugeordnet sind, werden aktiviert, wenn dieser Benutzer eine Verbindung herstellt, sodass alle Zugriffsrechte und Berechtigungen, die Rollen erteilt sind, bei der Herstellung einer Verbindung durch einen Benutzer ausgewertet werden. Rollen können nicht explizit aktiviert oder inaktiviert werden.
- Der Sicherheitsadministrator kann das Management einer Rolle an andere Personen delegieren.

Einer Rolle können alle DB2-Zugriffsrechte und -Berechtigungen, die innerhalb einer Datenbank erteilt werden können, erteilt werden. Zum Beispiel können einer Rolle beliebige der folgenden Berechtigungen und Zugriffsrechte erteilt werden:

- Datenbankberechtigungen DBADM, SECADM, DATAACCESS, ACCESSCTRL, SQLADM, WLMADM, LOAD und IMPLICIT_SCHEMA
- Datenbankberechtigungen CONNECT, CREATETAB, CREATE_NOT_FENCED, BINDADD, CREATE_EXTERNAL_ROUTINE und QUIESCE_CONNECT
- Alle Zugriffsrechte für Datenbankobjekte (einschließlich CONTROL)

Die Rollen eines Benutzers werden automatisch aktiviert und bei der Berechtigungsverarbeitung berücksichtigt, wenn der Benutzer eine Verbindung zur Datenbank herstellt. Sie brauchen eine Rolle nicht durch die Anweisung SET ROLE zu aktivieren. Wenn Sie zum Beispiel eine Sicht, eine MQT, einen Trigger, ein Paket oder eine SQL-Routine erstellen, werden die Zugriffsrechte, über die Sie durch Rollen verfügen, angewendet. Die Zugriffsrechte, über die Sie durch Rollen verfügen, die Gruppen erteilt sind, deren Mitglied Sie sind, werden hingegen nicht angewendet.

Eine Rolle hat keinen Eigner. Der Sicherheitsadministrator kann mithilfe der Klausel WITH ADMIN OPTION der Anweisung GRANT das Management der Rolle an einen anderen Benutzer delegieren, sodass der andere Benutzer die Rollenzugehörigkeit steuern kann.

Einschränkungen

Bei der Verwendung von Rollen sind einige Einschränkungen zu beachten:

- Eine Rolle kann nicht Eigner von Datenbankobjekten sein.
- Berechtigungen und Rollen, die Gruppen erteilt sind, werden beim Erstellen der folgenden Datenbankobjekte nicht geprüft:
  - Pakete mit statischem SQL
  - Sichten
  - MQTs (Materialized Query Tables)
  - Trigger
  - SQL-Routinen

  Beim Erstellen dieser Objekte werden nur die Rollen geprüft, die dem Benutzer, der das Objekt erstellt, oder PUBLIC direkt oder indirekt (z. B. durch eine Rollenhierarchie) erteilt sind.

Erstellen von Rollen und Erteilen der Zugehörigkeit

Der Sicherheitsadministrator hat die Berechtigung, eine Rolle zu erstellen, zu löschen, zu erteilen und zu entziehen sowie einer Rolle einen Kommentar hinzuzufügen. Der Sicherheitsadministrator verwendet die Anweisung GRANT (Rolle), um einer Berechtigungs-ID die Zugehörigkeit zu einer Rolle zu erteilen, und die Anweisung REVOKE (Rolle), um einer Berechtigungs-ID die Zugehörigkeit zu einer Rolle zu entziehen.

Der Sicherheitsadministrator kann das Management der Zugehörigkeit zu einer Rolle an eine Berechtigungs-ID delegieren, indem er der Berechtigungs-ID die Zugehörigkeit zu dieser Rolle mit der Klausel WITH ADMIN OPTION in der Anweisung GRANT erteilt. Die Klausel WITH ADMIN OPTION der Anweisung GRANT (Rolle) gibt einem anderen Benutzer folgende Möglichkeiten:

- Er kann Rollen anderen Benutzern erteilen.

- Er kann Rollen anderen Benutzern entziehen.
- Er kann eine Rolle mit Kommentaren versehen.

Die Klausel WITH ADMIN OPTION schränkt den berechtigten Benutzer jedoch wie folgt ein:

- Er kann die Rolle nicht löschen.
- Er kann einer Berechtigungs-ID die Berechtigung mit der Klausel WITH ADMIN OPTION für die Rolle nicht entziehen.
- Er kann einem anderen Benutzer die Rolle nicht mit der Klausel WITH ADMIN OPTION erteilen (sofern er nicht die Berechtigung SECADM besitzt).

Wenn der Sicherheitsadministrator eine Rolle erstellt hat, kann der Datenbankadministrator der Rolle mithilfe der Anweisung GRANT Berechtigungen und Zugriffsrechte zuordnen. Einer Rolle können alle DB2-Zugriffsrechte und -Berechtigungen, die innerhalb einer Datenbank erteilt werden können, erteilt werden. Berechtigungen auf der Instanzebene, wie zum Beispiel die Berechtigung SYSADM (Systemadministrator), können einer Rolle nicht zugeordnet werden.

Der Sicherheitsadministrator bzw. jeder Benutzer, dem der Sicherheitsadministrator die Zugehörigkeit zu einer Rolle mit der Klausel WITH ADMIN OPTION erteilt hat, kann die Anweisung GRANT (Rolle) verwenden, um die Zugehörigkeit zu dieser Rolle anderen Benutzern, Gruppen, der speziellen Gruppe PUBLIC oder Rollen zu erteilen. Einem Benutzer kann die Zugehörigkeit zu einer Rolle mit der Klausel WITH ADMIN OPTION entweder direkt oder indirekt über PUBLIC, eine Gruppe oder eine Rolle erteilt werden.

Alle Rollen, die einem Benutzer zugeordnet sind, werden aktiviert, wenn dieser Benutzer eine Sitzung einrichtet. Alle Zugriffsrechte und Berechtigungen, die den Rollen eines Benutzers zugeordnet sind, werden bei der Berechtigungsprüfung durch das DB2-Datenbanksystem berücksichtigt. Einige Datenbanksysteme verwenden die Anweisung SET ROLE, um eine bestimmte Rolle zu aktivieren. Das DB2-Datenbanksystem unterstützt die Anweisung SET ROLE aus Gründen der Kompatibilität mit anderen Produkten, die mit der Anweisung SET ROLE arbeiten. In einem DB2-Datenbanksystem überprüft die Anweisung SET ROLE, ob dem Sitzungsbenutzer die Rolle erteilt ist, und gibt einen Fehler zurück, wenn dies nicht der Fall ist.

Zum Entziehen der Rollenzugehörigkeit eines Benutzers verwendet der Sicherheitsadministrator bzw. ein Benutzer, der das Zugriffsrecht WITH ADMIN OPTION für die Rolle besitzt, die Anweisung REVOKE (Rolle).

Beispiel

Eine Rolle besitzt eine Reihe von Zugriffsrechten, und ein Benutzer, dem diese Rolle erteilt wird, übernimmt diese Zugriffsrechte. Diese Übernahme von Zugriffsrechten vermeidet die Verwaltung einzelner Zugriffsrechte, wenn Zugriffsrechte eines Benutzers auf einen anderen Benutzer übertragen werden. Die einzigen Operationen, die bei Verwendung von Rollen dazu erforderlich sind, bestehen darin, die Zugehörigkeit zu der Rolle dem einen Benutzer zu entziehen und sie dem anderen Benutzer zu erteilen.

Betrachten Sie ein Beispiel: Die Mitarbeiter BOB und ALICE, die in der Abteilung DEV ('Entwicklung') arbeiten, haben das Zugriffsrecht SELECT für die Tabellen SERVER, CLIENT und TOOLS. Eines Tages entscheidet das Management, sie in eine andere Abteilung, zum Beispiel QA, zu versetzen. Der Datenbankmanager

muss ihnen das Zugriffsrecht SELECT für die Tabellen SERVER, CLIENT und TOOLS entziehen. Die Abteilung DEV stellt nachfolgend einen neuen Mitarbeiter TOM ein, sodass der Datenbankadministrator dem Benutzer TOM das Zugriffsrecht SELECT für die Tabellen SERVER, CLIENT und TOOLS erteilen muss.

Bei Verwendung von Rollen sind in diesem Fall die folgenden Schritte erforderlich:

1. Der Sicherheitsadministrator erstellt die Rolle DEVELOPER ('Entwickler'):
   ```
   CREATE ROLE DEVELOPER
   ```
2. Der Datenbankadministrator (mit der Berechtigung DBADM) erteilt der Rolle DEVELOPER das Zugriffsrecht SELECT für die Tabellen SERVER, CLIENT und TOOLS:
   ```
   GRANT  SELECT ON TABLE SERVER TO ROLE DEVELOPER
   GRANT  SELECT ON TABLE CLIENT TO ROLE DEVELOPER
   GRANT  SELECT ON TABLE TOOLS TO ROLE DEVELOPER
   ```
3. Der Sicherheitsadministrator erteilt den Benutzern BOB und ALICE in der Abteilung DEV die Rolle DEVELOPER:
   ```
   GRANT ROLE DEVELOPER TO USER BOB, USER ALICE
   ```
4. Wenn BOB und ALICE die Abteilung DEV verlassen, entzieht der Sicherheitsadministrator den Benutzern BOB und ALICE die Rolle DEVELOPER:
   ```
   REVOKE ROLE DEVELOPER FROM USER BOB, USER ALICE
   ```
5. Wenn TOM in der Abteilung DEV angestellt wird, erteilt der Sicherheitsadministrator dem Benutzer TOM die Rolle DEVELOPER:
   ```
   GRANT ROLE DEVELOPER TO USER TOM
   ```

Rollenhierarchien

Eine Rollenhierarchie entsteht, wenn einer Rolle die Zugehörigkeit zu einer anderen Rolle erteilt wird.

Eine Rolle *enthält* eine andere Rolle, wenn die andere Rolle der ersten Rolle erteilt wird. Die andere Rolle übernimmt alle Zugriffsrechte der ersten Rolle. Wenn zum Beispiel die Rolle DOCTOR der Rolle SURGEON (Chirurg) erteilt wird, dann lässt sich sagen, dass die Rolle SURGEON die Rolle DOCTOR enthält. Die Rolle SURGEON übernimmt alle Zugriffsrechte der Rolle DOCTOR.

Zyklen in Rollenhierarchien sind nicht zulässig. Ein *Zyklus* entsteht, wenn erteilte Rollen einen abgeschlossenen Kreis bilden, indem zum Beispiel eine Rolle einer anderen Rolle erteilt wird und diese andere Rolle wiederum der ersten Rolle erteilt wird. Ein Beispiel für einen Zyklus wäre, wenn die Rolle DOCTOR der Rolle SURGEON erteilt würde und anschließend die Rolle SURGEON wieder der Rolle DOCTOR erteilt würde. Wenn Sie einen Zyklus in einer Rollenhierarchie erstellen, wird ein Fehler zurückgegeben (SQLSTATE-Wert 428GF).

Beispiel für den Aufbau einer Rollenhierarchie

Das folgende Beispiel zeigt, wie eine Rollenhierarchie aufgebaut wird, um die medizinischen Ebenen in einem Krankenhaus darzustellen.

Betrachten Sie die folgenden Rollen: DOCTOR (Arzt), SPECIALIST (Facharzt) und SURGEON (Chirurg). Eine Rollenhierarchie wird aufgebaut, indem eine Rolle einer anderen Rolle erteilt wird, jedoch ohne Zyklen zu bilden. Die Rolle DOCTOR wird der Rolle SPECIALIST, die Rolle SPECIALIST der Rolle SURGEON erteilt.

Wenn die Rolle SURGEON der Rolle DOCTOR erteilt würde, entstünde ein Zyklus, der nicht zulässig wäre.

Der Sicherheitsadministrator führt die folgenden SQL-Anweisungen aus, um die Rollenhierarchie aufzubauen:

```
CREATE ROLE DOCTOR
CREATE ROLE SPECIALIST
CREATE ROLE SURGEON

GRANT ROLE DOCTOR TO ROLE SPECIALIST

GRANT ROLE SPECIALIST TO ROLE SURGEON
```

Wirkung des Widerrufens von Zugriffsrechten für Rollen

Wenn Zugriffsrechte entzogen werden, kann dies in manchen Fällen dazu führen, dass abhängige Datenbankobjekte, wie zum Beispiel Sichten, Pakete oder Trigger, ungültig oder unbrauchbar werden.

Die folgenden Beispiele zeigen, was mit einem Datenbankobjekt geschieht, wenn einige Zugriffsrechte einer Berechtigungs-ID entzogen werden und die Zugriffsrechte durch eine Rolle oder andere Methoden erteilt wurden.

Beispiel für das Entziehen von Zugriffsrechten von Rollen

1. Der Sicherheitsadministrator erstellt die Rolle DEVELOPER und erteilt dem Benutzer BOB die Zugehörigkeit zu dieser Rolle:

   ```
   CREATE ROLE DEVELOPER
   GRANT ROLE DEVELOPER TO USER BOB
   ```

2. Die Benutzerin ALICE erstellt die Tabelle WORKITEM:

   ```
   CREATE TABLE WORKITEM (x int)
   ```

3. Der Datenbankadministrator erteilt PUBLIC und der Rolle DEVELOPER die Zugriffsrechte SELECT und INSERT für die Tabelle WORKITEM:

   ```
   GRANT SELECT ON TABLE ALICE.WORKITEM TO PUBLIC
   GRANT INSERT ON TABLE ALICE.WORKITEM TO PUBLIC
   GRANT SELECT ON TABLE ALICE.WORKITEM TO ROLE DEVELOPER
   GRANT INSERT ON TABLE ALICE.WORKITEM TO ROLE DEVELOPER
   ```

4. Der Benutzer BOB erstellt die Sicht PROJECT, die auf der Tabelle WORKITEM basiert, und das Paket PKG1, das von der Tabelle WORKITEM abhängig ist:

   ```
   CREATE VIEW PROJECT AS SELECT * FROM ALICE.WORKITEM
   PREP emb001.sqc BINDFILE PACKAGE USING PKG1 VERSION 1
   ```

5. Wenn der Datenbankadministrator der Gruppe PUBLIC das Zugriffsrecht SELECT für die Tabelle ALICE.WORKITEM entzieht, bleiben die Sicht BOB.PROJECT brauchbar und das Paket PKG1 gültig, weil der Benutzer BOB, der die Sicht und das Paket definiert hat, weiterhin die erforderlichen Zugriffsrechte durch seine Zugehörigkeit zur Rolle DEVELOPER besitzt:

   ```
   REVOKE SELECT ON TABLE ALICE.WORKITEM FROM PUBLIC
   ```

6. Wenn der Datenbankadministrator der Rolle DEVELOPER das Zugriffsrecht SELECT für die Tabelle ALICE.WORKITEM entzieht, werden die Sicht BOB.PROJECT unbrauchbar und das Paket PKG1 ungültig, weil der Benutzer BOB, der die Sicht und das Paket definiert hat, die erforderlichen Zugriffsrechte jetzt nicht mehr durch andere Methoden besitzt:

   ```
   REVOKE SELECT ON TABLE ALICE.WORKITEM FROM ROLE DEVELOPER
   ```

Beispiel für das Entziehen der Berechtigung DBADM

In diesem Beispiel erhält die Rolle DEVELOPER die Berechtigung DBADM und wird dem Benutzer BOB erteilt.

1. Der Sicherheitsadministrator erstellt die Rolle DEVELOPER:

   ```
   CREATE ROLE DEVELOPER
   ```

2. Der Systemadministrator erteilt der Rolle DEVELOPER die Berechtigung DBADM:

   ```
   GRANT DBADM ON DATABASE TO ROLE DEVELOPER
   ```

3. Der Sicherheitsadministrator erteilt dem Benutzer BOB die Zugehörigkeit zu dieser Rolle:

   ```
   GRANT ROLE DEVELOPER TO USER BOB
   ```

4. Die Benutzerin ALICE erstellt die Tabelle WORKITEM:

   ```
   CREATE TABLE WORKITEM (x int)
   ```

5. Der Benutzer BOB erstellt die Sicht PROJECT, die auf der Tabelle WORKITEM basiert, das Paket PKG1, das von der Tabelle WORKITEM abhängig ist, und den Trigger TRG1, der ebenfalls von der Tabelle WORKITEM abhängig ist:

   ```
   CREATE VIEW PROJECT AS SELECT * FROM ALICE.WORKITEM
   PREP emb001.sqc BINDFILE PACKAGE USING PKG1 VERSION 1
   CREATE TRIGGER TRG1 AFTER DELETE ON ALICE.WORKITEM
                FOR EACH STATEMENT MODE DB2SQL
          INSERT INTO ALICE.WORKITEM VALUES (1)
   ```

6. Der Sicherheitsadministrator entzieht dem Benutzer BOB die Rolle DEVELOPER:

   ```
   REVOKE ROLE DEVELOPER FROM USER BOB
   ```

   Durch das Entziehen der Rolle DEVELOPER verliert der Benutzer BOB die Berechtigung DBADM, weil ihm die Rolle, die diese Berechtigung besitzt, entzogen wird. Die Sicht, das Paket und der Trigger werden davon wie folgt betroffen:

   - Die Sicht BOB.PROJECT ist weiterhin gültig.
   - Das Paket PKG1 wird ungültig.
   - Der Trigger BOB.TRG1 ist weiterhin gültig.

   Die Sicht BOB.PROJECT und der Trigger BOB.TRG1 bleiben verwendbar, während das Paket PKG1 nicht mehr verwendet werden kann. Sicht- und Triggerobjekte, die durch eine Berechtigungs-ID mit der Berechtigung DBADM erstellt werden, werden nicht betroffen, wenn die Berechtigung DBADM entzogen wird.

Delegieren der Rollenverwaltung mit der Klausel WITH ADMIN OPTION

Mithilfe der Klausel WITH ADMIN OPTION der SQL-Anweisung GRANT (Rolle) kann der Sicherheitsadministrator das Management und die Steuerung der Zugehörigkeit zu einer Rolle an eine andere Person delegieren.

Die Klausel WITH ADMIN OPTION gibt einem anderen Benutzer die Berechtigung, die Zugehörigkeit zur jeweiligen Rolle anderen Benutzern zu erteilen, anderen Benutzern, die diese Rolle haben, diese Rolle zu entziehen und diese Rolle mit Kommentaren zu versehen. Jedoch kann er die Rolle nicht löschen.

Die Klausel WITH ADMIN OPTION gibt einem anderen Benutzer nicht die Berechtigung, die Rolle wiederum einem anderen Benutzer ebenfalls mit der Klausel WITH ADMIN OPTION zu erteilen. Darüber hinaus gibt sie nicht die Berechti-

gung, einer anderen Berechtigungs-ID, die diese Rolle mit dem Zugriffsrecht WITH ADMIN OPTION besitzt, das Zugriffsrecht WITH ADMIN OPTION für diese Rolle zu entziehen.

Beispiel für die Verwendung der Klausel WITH ADMIN OPTION

1. Ein Sicherheitsadministrator erstellt die Rolle DEVELOPER und erteilt die neue Rolle dem Benutzer BOB mithilfe der Anweisung GRANT mit der Klausel WITH ADMIN OPTION:

   ```
   CREATE ROLE DEVELOPER
   GRANT ROLE DEVELOPER TO USER BOB WITH ADMIN OPTION
   ```

2. Der Benutzer BOB kann jetzt die Zugehörigkeit zu dieser Rolle anderen Benutzern, zum Beispiel ALICE, erteilen und entziehen:

   ```
   GRANT ROLE DEVELOPER TO USER ALICE
   REVOKE ROLE DEVELOPER FROM USER ALICE
   ```

3. Der Benutzer BOB kann die Rolle jedoch nicht löschen oder sie einem anderen Benutzer mit der Klausel WITH ADMIN OPTION erteilen. (Diese beiden Operationen können nur vom Sicherheitsadministrator ausgeführt werden.) Die folgenden Befehle schlagen fehl, wenn sie vom Benutzer BOB ausgeführt werden:

   ```
   DROP ROLE DEVELOPER - FEHLER!
                       - Nur ein Sicherheitsadministrator darf die Rolle löschen.
   GRANT ROLE DEVELOPER TO USER ALICE WITH ADMIN OPTION - FEHLER!
                       - Nur ein Sicherheitsadministrator kann die Rolle mit der
                         Klausel WITH ADMIN OPTION erteilen.
   ```

4. Der Benutzer BOB kann die Verwaltungszugriffsrechte für die Rolle (die durch WITH ADMIN OPTION erteilt wurden) Benutzern mit der Rolle DEVELOPER nicht entziehen, weil er nicht die Berechtigung SECADM (Sicherheitsadministrator) besitzt. Wenn BOB den folgenden Befehl eingibt, schlägt dieser fehl:

   ```
   REVOKE ADMIN OPTION FOR ROLE DEVELOPER FROM USER SANJAY - FEHLER!
   ```

5. Ein Sicherheitsadministrator kann die Rollenverwaltungszugriffsrechte für die Rolle DEVELOPER (die mit der Klausel WITH ADMIN OPTION übertragen wurden) dem Benutzer BOB entziehen, sodass ihm weiterhin die Rolle DEVELOPER erteilt bleibt:

   ```
   REVOKE ADMIN OPTION FOR ROLE DEVELOPER FROM USER BOB
   ```

   Wenn ein Sicherheitsadministrator dem Benutzer BOB andererseits die Rolle DEVELOPER entzieht, verliert BOB alle Zugriffsrechte, die er durch die Zugehörigkeit zur Rolle DEVELOPER besaß, und die Berechtigung für die Rolle, die er durch die Klausel WITH ADMIN OPTION erhalten hatte:

   ```
   REVOKE ROLE DEVELOPER FROM USER BOB
   ```

Rollen im Vergleich zu Gruppen

Zugriffsrechte und Berechtigungen, die Gruppen erteilt sind, werden bei der Erstellung von Sichten, MQTs (Materialized Query Tables), SQL-Routinen, Triggern und Paketen mit statischem SQL nicht berücksichtigt. Diese Einschränkung kann durch die Verwendung von Rollen anstelle von Gruppen überwunden werden.

Durch Rollen können Benutzer Datenbankobjekte erstellen und dabei ihre durch Rollen erhaltenen Zugriffsrechte verwenden, die durch das DB2-Datenbanksystem gesteuert werden. Gruppen und Benutzer werden außerhalb des DB2-Datenbanksystems, zum Beispiel durch das Betriebssystem oder einen LDAP-Server, gesteuert.

Beispiel für das Ersetzen von Gruppen durch Rollen

Das folgende Beispiel zeigt, wie Gruppen durch die Verwendung von Rollen ersetzt werden können.

Nehmen Sie an, es seien drei Gruppen vorhanden: DEVELOPER_G, TESTER_G und SALES_G. Die Benutzer BOB, ALICE und TOM sind Mitglieder dieser Gruppen, wie in der folgenden Tabelle dargestellt:

Tabelle 7. Beispiel für Gruppen und Benutzer

Gruppe	Mitglieder der Gruppe
DEVELOPER_G	BOB
TESTER_G	ALICE, TOM
SALES_G	ALICE, BOB

1. Der Sicherheitsadministrator erstellt die Rollen DEVELOPER, TESTER und SALES, die an Stelle der Gruppen verwendet werden sollen.
   ```
   CREATE ROLE DEVELOPER
   CREATE ROLE TESTER
   CREATE ROLE SALES
   ```
2. Der Sicherheitsadministrator erteilt Benutzern die Zugehörigkeit zu diesen Rollen (die Festlegung der Mitgliedschaft von Benutzern in Gruppen lag in der Zuständigkeit des Systemadministrators):
   ```
   GRANT  ROLE DEVELOPER TO USER BOB
   GRANT  ROLE TESTER TO USER ALICE, USER TOM
   GRANT  ROLE SALES TO USER BOB, USER ALICE
   ```
3. Der Datenbankadministrator kann den Rollen ähnliche Zugriffsrechte oder Berechtigungen erteilen, wie sie auch die Gruppen hatten. Beispiel:
   ```
   GRANT zugriffsrecht ON objekt TO ROLE DEVELOPER
   ```

   Der Datenbankadministrator kann anschließend den Gruppen diese Zugriffsrechte entziehen und den Systemadministrator bitten, die Gruppen aus dem System zu entfernen.

Beispiel für das Erstellen eines Triggers mit den durch eine Rolle erteilten Zugriffsrechten

Dieses Beispiel zeigt, dass der Benutzer BOB einen Trigger (TRG1) erfolgreich erstellen kann, wenn er das erforderliche Zugriffsrecht durch die Rolle DEVELOPER besitzt.

1. Zunächst erstellt die Benutzerin ALICE die Tabelle WORKITEM:
   ```
   CREATE TABLE WORKITEM (x int)
   ```
2. Anschließend wird das Zugriffsrecht ALTER zum Ändern der Tabelle der Benutzerin ALICE vom Datenbankadministrator der Rolle DEVELOPER erteilt.
   ```
   GRANT ALTER ON ALICE.WORKITEM TO ROLE DEVELOPER
   ```
3. Der Benutzer BOB kann den Trigger TRG1 erfolgreich erstellen, weil er die Zugehörigkeit zur Rolle DEVELOPER besitzt.
   ```
   CREATE TRIGGER TRG1 AFTER DELETE ON ALICE.WORKITEM
              FOR EACH STATEMENT MODE DB2SQL INSERT INTO ALICE.WORKITEM VALUES (1)
   ```

Verwenden von Rollen nach der Migration von IBM Informix Dynamic Server

Wenn Sie eine Migration von IBM Informix Dynamic Server auf das DB2-Datenbanksystem ausgeführt haben und Rollen verwenden, müssen Sie einige Gesichtspunkte beachten.

Die SQL-Anweisung GRANT ROLE von Informix Dynamic Server (IDS) ermöglicht die Verwendung der Klausel WITH GRANT OPTION. Die Anweisung GRANT ROLE des DB2-Datenbanksystems stellt die Klausel WITH ADMIN OPTION (dies entspricht dem SQL-Standard) zur Verfügung, die dieselbe Funktionalität besitzt. Während einer Migration von IDS auf das DB2-Datenbanksystem generiert das Tool **dbschema** Anweisungen CREATE ROLE und GRANT ROLE. Anschließend ersetzt das Tool **dbschema** alle Vorkommen der Klausel WITH GRANT OPTION durch WITH ADMIN OPTION.

In einem IDS-Datenbanksystem aktiviert die Anweisung SET ROLE eine bestimmte Rolle. Das DB2-Datenbanksystem unterstützt die Anweisung SET ROLE, jedoch nur, um Kompatibilität mit anderen Produkten herzustellen, die diese SQL-Anweisung verwenden. Die Anweisung SET ROLE überprüft, ob der Sitzungsbenutzer zu der Rolle gehört und gibt einen Fehler zurück, wenn er dies nicht tut.

Beispielausgabe für das Tool 'dbschema'

Nehmen Sie an, dass eine IDS-Datenbank die Rollen DEVELOPER, TESTER und SALES enthält. Den Benutzern BOB, ALICE und TOM wurden jeweils verschiedene Rollen erteilt. Die Rolle DEVELOPER wurde dem Benutzer BOB erteilt, die Rolle TESTER der Benutzerin ALICE und die Rollen TESTER und SALES dem Benutzer TOM. Zur Migration auf das DB2-Datenbanksystem verwenden Sie das Tool **dbschema**, um die Anweisungen CREATE ROLE und GRANT ROLE für die Datenbank zu generieren:

```
CREATE ROLE DEVELOPER
CREATE ROLE TESTER
CREATE ROLE SALES

GRANT  DEVELOPER TO BOB
GRANT  TESTER TO ALICE, TOM
GRANT  SALES TO TOM
```

Sie müssen die Datenbank im DB2-Datenbanksystem erstellen und anschließend die oben gezeigten Anweisungen in dieser Datenbank ausführen, um die Rollen und die Zuordnung der Rollen erneut zu erstellen.

Kapitel 3. Verwenden gesicherter Kontexte und gesicherter Verbindungen

Sie können eine explizite gesicherte Verbindung herstellen, indem Sie in einer Anwendung eine Anforderung absetzen, wenn eine Verbindung zu einer DB2-Datenbank hergestellt wird. Der Sicherheitsadministrator muss zuvor mithilfe der Anweisung CREATE TRUSTED CONTEXT einen gesicherten Kontext definiert haben, dessen Attribute mit denen der Verbindung übereinstimmen, die Sie herstellen (siehe Schritt 1 unten).

Vorbereitende Schritte

Die Anwendungsprogrammierschnittstelle (API), die Sie bei der Verbindungsherstellung zum Anfordern einer expliziten gesicherten Verbindung verwenden, hängt von dem Typ der Anwendung ab, den Sie verwenden (siehe Tabelle in Schritt 2).

Nachdem Sie eine explizite gesicherte Verbindung hergestellt haben, kann die Anwendung die Benutzer-ID der Verbindung mithilfe der entsprechenden API für den Typ von Anwendung (siehe Tabelle in Schritt 3) in eine andere Benutzer-ID ändern.

Vorgehensweise

1. Der Sicherheitsadministrator definiert einen gesicherten Kontext auf dem Server mit der Anweisung CREATE TRUSTED CONTEXT. Beispiel:

```
CREATE TRUSTED CONTEXT MYTCX
       BASED UPON CONNECTION USING SYSTEM AUTHID NEWTON
       ATTRIBUTES (ADDRESS   '192.0.2.1')
       WITH USE FOR PUBLIC WITHOUT AUTHENTICATION
       ENABLE
```

2. Zum Herstellen einer gesicherten Verbindung müssen Sie in Ihrer Anwendung eine der folgenden APIs verwenden:

Option	**Bezeichnung**
Anwendung	**API**
CLI/ODBC	SQLConnect, SQLSetConnectAttr
XA CLI/ODBC	Xa_open
JAVA	getDB2TrustedPooledConnection, getDB2TrustedXAConnection

3. Für den Wechsel zu einem anderen Benutzer mit oder ohne Authentifizierung verwenden Sie in Ihrer Anwendung eine der folgenden APIs:

Option	**Bezeichnung**
Anwendung	**API**
CLI/ODBC	SQLSetConnectAttr
XA CLI/ODBC	SQLSetConnectAttr
JAVA	getDB2Connection, reuseDB2Connection
.NET	DB2Connection.ConnectionString Schlüsselwörter: TrustedContextSystemUserID und TrustedContextSystemPassword

Der Wechsel kann mit oder ohne Authentifizierung der neuen Benutzer-ID erfolgen. Dies hängt von der Definition des Objekts für den gesicherten Kontext ab, das der expliziten gesicherten Verbindung zugeordnet wird. Nehmen Sie zum Beispiel an, dass der Sicherheitsadministrator das folgende Objekt für einen gesicherten Kontext erstellt:

```
CREATE TRUSTED CONTEXT CTX1
 BASED UPON CONNECTION USING SYSTEM AUTHID BENUTZER1
 ATTRIBUTES (ADDRESS   '192.0.2.1')
  WITH USE FOR BENUTZER2 WITH AUTHENTICATION,
               BENUTZER3 WITHOUT AUTHENTICATION
  ENABLE
```

Nehmen Sie weiterhin an, dass eine explizite gesicherte Verbindung hergestellt wird. In diesem Fall ist ein Wechsel der Benutzer-ID für die gesicherte Verbindung zu BENUTZER3 ohne die Bereitstellung von Authentifizierungsdaten zulässig, weil BENUTZER3 als ein Benutzer des gesicherten Kontextes CTX1 definiert wurde, für den keine Authentifizierung erforderlich ist. Eine Anforderung zum Wechseln der Benutzer-ID für die gesicherte Verbindung zu BENUTZER2 schlägt dagegen fehl, wenn keine Authentifizierungsdaten angegeben werden, weil BENUTZER2 als ein Benutzer des gesicherten Kontextes CTX1 definiert ist, für den Authentifizierungsdaten angegeben werden müssen.

Beispiel für das Herstellen einer expliziten gesicherten Verbindung und den Wechsel des Benutzers

Im folgenden Beispiel muss ein Server der Mittelschicht Anforderungen für einen Endbenutzer absetzen, verfügt jedoch über keinen Zugriff auf die Berechtigungsnachweise des Endbenutzers, um eine Datenbankverbindung für diesen Endbenutzer herzustellen.

Sie können ein Objekt für einen gesicherten Kontext auf dem Datenbankserver erstellen, mit dem der Server der Mittelschicht eine explizite gesicherte Verbindung zur Datenbank herstellen kann. Nach der Herstellung einer expliziten gesicherten Verbindung kann der Server der Mittelschicht die aktuelle Benutzer-ID der Verbindung wechseln und eine neue Benutzer-ID angeben, ohne dass die neue Benutzer-ID auf dem Datenbankserver authentifiziert werden muss. Das folgende CLI-Codefragment zeigt, wie eine gesicherte Verbindung mit dem gesicherten Kontext MYTCX hergestellt werden kann, der in Schritt 1 definiert wurde, und wie der Benutzer der gesicherten Verbindung ohne Authentifizierung gewechselt werden kann.

```
int main(int argc, char *argv[])
{
 SQLHANDLE henv;         /* Umgebungskennung */
 SQLHANDLE hdbc1;       /* Verbindungskennung */
 char origUserid[10] = "newton";
  char password[10] = "test";
   char switchUserid[10] = "zurbie";
 char dbName[10] = "testdb";

// Kennungen zuordnen
SQLAllocHandle( SQL_HANDLE_ENV, &henv );
SQLAllocHandle( SQL_HANDLE_DBC, &hdbc1 );

// Attribut für gesicherte Verbindung festlegen
SQLSetConnectAttr( hdbc1, SQL_ATTR_USE_TRUSTED_CONTEXT,
SQL_TRUE, SQL_IS_INTEGER );

// Gesicherte Verbindung einrichten
SQLConnect( hdbc1, dbName, SQL_NTS, origUserid, SQL_NTS,
password, SQL_NTS );
```

```
// Einige Arbeitsoperationen unter Benutzer-ID "newton" ausführen
. . . . . .

 // Commit durchführen
SQLEndTran(SQL_HANDLE_DBC, hdbc1, SQL_COMMIT);


// Benutzer-ID der gesicherten wechseln
SQLSetConnectAttr( hdbc1,
SQL_ATTR_TRUSTED_CONTEXT_USERID, switchUserid,
SQL_IS_POINTER
);

//Neue Arbeitsoperationen unter Benutzer-ID "zurbie" ausführen
. . . . . .

// Commit durchführen
SQLEndTranSQL_HANDLE_DBC, hdbc1, SQL_COMMIT);

// Datenbankverbindung trennen
SQLDisconnect( hdbc1 );

 return 0;

}  /* Ende von 'main' */
```

Nächste Schritte

Tatsächlicher Zeitpunkt des Wechsels der Benutzer-ID

Nachdem der Befehl zum Wechseln des Benutzers auf der gesicherten Verbindung abgesetzt wurde, wird die Benutzerwechselanforderung erst ausgeführt, wenn die nächste Anweisung an den Server gesendet wird. Dies wird im folgenden Beispiel veranschaulicht, in dem der Befehl **list applications** die ursprüngliche Benutzer-ID anzeigt, bis die nächste Anweisung abgesetzt wird.

1. Stellen Sie eine explizite gesicherte Verbindung mit BENUTZERID1 her.
2. Setzen Sie den Befehl zum Wechseln des Benutzers ab, wie zum Beispiel **getDB2Connection** für BENUTZERID2.
3. Führen Sie den Befehl `db2 list applications` aus. Die Ausgabe des Befehls zeigt weiterhin an, dass BENUTZERID1 verbunden ist.
4. Setzen Sie eine Anweisung über die gesicherte Verbindung, wie zum Beispiel `executeQuery("values current sqlid")` ab, sodass die Benutzerwechselanforderung auf dem Server ausgeführt wird.
5. Führen Sie den Befehl `db2 list applications` erneut aus. Die Ausgabe des Befehls zeigt jetzt an, dass BENUTZERID2 verbunden ist.

Gesicherte Kontexte und Verbindungen

Ein gesicherter Kontext ist ein Datenbankobjekt, das eine Vertrauensbeziehung für eine Verbindung zwischen der Datenbank und einer externen Entität wie beispielsweise einem Anwendungsserver definiert.

Die Vertrauensbeziehung basiert auf der folgenden Gruppe von Attributen:

- Systemberechtigungs-ID: Ist dem Benutzer zugeordnet, der eine Datenbankverbindung herstellt.
- IP-Adresse (oder Domänenname): Ist dem Host zugeordnet, über den eine Datenbankverbindung hergestellt wird.

- Datenstromverschlüsselung: Stellt die Verschlüsselungseinstellung (sofern vorhanden) für die Datenübertragung zwischen dem Datenbankserver und dem Datenbankclient dar.

Wenn ein Benutzer eine Datenbankverbindung herstellt, überprüft das DB2-Datenbanksystem, ob die Verbindung mit der Definition eines Objekts für einen gesicherten Kontext in der Datenbank übereinstimmt. Wenn eine Übereinstimmung vorhanden ist, dann wird die Datenbankverbindung als gesichert eingestuft.

Eine gesicherte Verbindung ermöglicht dem Initiator dieser Verbindung das Anfordern weiterer Funktionen, die außerhalb der gesicherten Verbindung möglicherweise nicht verfügbar sind. Diese zusätzlichen Funktionen können abhängig davon, ob es sich um eine explizite oder implizite gesicherte Verbindung handelt, variieren.

Der Initiator einer expliziten gesicherten Verbindung verfügt über folgende Möglichkeiten:

- Wechseln der aktuellen Benutzer-ID der Verbindung und Angabe einer anderen Benutzer-ID mit oder ohne Authentifizierung.
- Anfordern zusätzlicher Zugriffsrechte über die Funktion für die Rollenvererbung in gesicherten Kontexten.

Eine implizite gesicherte Verbindung stellt eine gesicherte Verbindung dar, die nicht explizit angefordert wurde. Die implizite gesicherte Verbindung resultiert aus einer normalen Verbindungsanforderung und nicht aus der Anforderung einer expliziten gesicherten Verbindung. Zum Anfordern einer impliziten Verbindung sind keine Änderungen am Anwendungscode erforderlich. Darüber hinaus hat die Entscheidung zum Anfordern einer impliziten gesicherten Verbindung oder eines anderen Verbindungstyps keine Auswirkungen auf den Verbindungsrückkehrcode (wenn Sie eine explizite gesicherte Verbindung anfordern, gibt der Verbindungsrückkehrcode an, ob die Anforderung erfolgreich ausgeführt werden konnte oder nicht). Der Initiator einer impliziten gesicherten Verbindung kann zusätzliche Zugriffsrechte nur über die Funktion für die Rollenvererbung in gesicherten Kontexten anfordern. Ein Wechsel der Benutzer-ID ist hingegen nicht möglich.

Verbesserte Sicherheit durch Verwendung gesicherter Kontexte

Das auf drei Schichten basierende Anwendungsmodell erweitert das standardmäßige zweischichtige Client- und Servermodell durch eine Mittelschicht, die zwischen der Clientanwendung und dem Datenbankserver eingefügt wird. In den letzten Jahren fand dieses Modell mit dem verstärkten Einsatz webbasierter Technologien und der J2EE-Plattform (J2EE = Java 2 Enterprise Edition) zunehmend Verbreitung. Als Beispiel für ein Softwareprodukt, das das dreischichtige Anwendungsmodell unterstützt, kann IBM WebSphere Application Server (WAS) aufgeführt werden.

In einem dreischichtigen Anwendungsmodell ist die Mittelschicht für die Authentifizierung der Benutzer verantwortlich, die die Clientanwendungen ausführen, und darüber hinaus auch für die Verwaltung der Interaktion mit dem Datenbankserver. Traditionell erfolgt die gesamte Interaktion mit dem Datenbankserver über eine Datenbankverbindung, die von der Mittelschicht mithilfe einer Kombination aus Benutzer-ID und einem Berechtigungsnachweis hergestellt wird, mit der die Mittelschicht gegenüber dem Datenbankserver identifiziert wird. Dies bedeutet, dass der Datenbankserver für alle Berechtigungsprüfungs- und Protokollierungsoperationen, die für den Datenbankzugriff erforderlich sind, die Datenbankzugriffsrechte verwendet, die der Benutzer-ID der Mittelschicht zugeordnet sind. Dies gilt auch für die Zugriffsoperationen der Mittelschicht für einen Benutzer.

Obwohl das dreischichtige Anwendungsmodell zahlreiche Vorteile aufweist, verursacht die Ausführung der gesamten Interaktion mit dem Datenbankserver (z. B. eine Benutzeranforderung) unter der Berechtigungs-ID der Mittelschicht jedoch verschiedene Sicherheitsrisiken, die wie folgt zusammengefasst werden können:

- Verlust der Benutzeridentität

  Für bestimmte Unternehmen ist es aus Gründen der Zugriffskontrolle wichtig, die Identität des Benutzers zu kennen, der auf die Datenbank zugreift.

- Reduzierte Benutzerverantwortlichkeit

  Die Möglichkeit zur Feststellung der Verantwortlichkeit mithilfe entsprechender Protokollierungsoperationen ist ein Grundprinzip der Datenbanksicherheit. Dadurch, dass die Benutzeridentität nicht bekannt ist, wird die Unterscheidung von Transaktionen der Mittelschicht für eigene Zwecke und Transaktionen der Mittelschicht für einen Benutzer erheblich erschwert.

- Gewährung nicht erforderlicher Zugriffsrechte für die Berechtigungs-ID der Mittelschicht

  Die Berechtigungs-ID der Mittelschicht muss über alle Zugriffsrechte verfügen, die zur Ausführung sämtlicher Anforderungen aller Benutzer erforderlich sind. Hierdurch entsteht das Sicherheitsproblem, dass Benutzern, die keinen Zugriff auf bestimmte Informationen benötigen, dieser Zugriff dennoch gewährt wird.

- Beeinträchtigte Sicherheit

  Zusätzlich zum Problem der Zugriffsrechte, das im vorherigen Listenpunkt erläutert wurde, ist es bei diesem Ansatz erforderlich, dass die von der Mittelschicht für die Herstellung von Verbindungen verwendete Berechtigungs-ID über Zugriffsrechte für alle Ressourcen verfügen muss, auf die mit Benutzeranforderungen zugegriffen werden kann. Wenn diese Berechtigungs-ID der Mittelschicht in ihrer Sicherheit beeinträchtigt wird, dann sind alle diese Ressourcen einem erhöhten Sicherheitsrisiko ausgesetzt.

- "Überlauf" zwischen Benutzern derselben Verbindung

  Änderungen eines vorherigen Benutzers können sich auf den aktuellen Benutzer auswirken.

Es besteht die Notwendigkeit zur Einrichtung von Verfahren, durch die die Identität und die Datenbankzugriffsrechte des aktuellen Benutzers für Datenbankanforderungen verwendet werden können, die von der Mittelschicht für diesen Benutzer ausgeführt werden. Der einfachste Ansatz zur Erreichung dieser Zielsetzung besteht darin, dass die Mittelschicht mithilfe der Benutzer-ID und des zugehörigen Kennworts eine neue Verbindung herstellt und die Benutzeranforderungen dann über diese Verbindung leitet. Obwohl dieser Ansatz einfach ist, birgt er verschiedene Nachteile, die im Folgenden aufgeführt sind:

- Nichtanwendbarkeit für bestimmte Mittelschichten. Zahlreiche Server der Mittelschicht verfügen nicht über die Berechtigungsnachweise für die Benutzerauthentifizierung, die für die Herstellung einer Verbindung erforderlich sind.
- Leistungseinbußen. Mit der Herstellung einer neuen physischen Verbindung und der erneuten Authentifizierung des Benutzers auf dem Datenbankserver sind gewisse Leistungseinbußen verbunden.
- Verwaltungsaufwand. In bestimmten Fällen, in denen keine zentrale Sicherheitsfunktion definiert und eingesetzt wird oder in denen die einmalige Anmeldung (Single Sign-on) nicht verwendet wird, entsteht durch das Vorhandensein von zwei Benutzerdefinitionen (in der Mittelschicht und auf dem Server) ein erhöhter Verwaltungsaufwand. Hierdurch ist es erforderlich, das Kennwort an unterschiedlichen Stellen zu ändern.

Dieses Problem kann mit der Funktion für gesicherte Kontexte behoben werden. Der Sicherheitsadministrator kann in der Datenbank ein Objekt für einen gesicher-

ten Kontext erstellen, mit dem eine Vertrauensbeziehung zwischen der Datenbank und der Mittelschicht definiert wird. Die Mittelschicht kann dann eine explizite gesicherte Verbindung zur Datenbank herstellen, wodurch sie die Möglichkeit erhält, die aktuelle Benutzer-ID für die Verbindung mit oder ohne Authentifizierung zu wechseln und eine andere Benutzer-ID zu verwenden. Zusätzlich zur Behebung des Problems mit der Identitätsprüfung für den Endbenutzer bieten gesicherte Kontexte einen weiteren Vorteil. Mit gesicherten Kontexten können Sie steuern, wann einem Datenbankbenutzer ein Zugriffsrecht gewährt wird. Wenn diese Möglichkeit nicht besteht, dann kann dies das gesamte Sicherheitskonzept des Systems beeinträchtigen. Zugriffsrechte können z. B. zu anderen Zwecken als den ursprünglich vorgesehenen verwendet werden. Der Sicherheitsadministrator kann einer Rolle ein oder auch mehrere Zugriffsrechte zuordnen und diese Rolle dann einem Objekt für einen gesicherten Kontext zuordnen. Nur gesicherte Datenbankverbindungen (explizit oder implizit), die mit der Definition dieses gesicherten Kontextes übereinstimmen, können die Zugriffsrechte, die dieser Rolle zugeordnet sind, nutzen.

Verbessern der Leistung

Bei Verwendung gesicherter Verbindungen können Sie die Leistung aufgrund der folgenden Vorteile maximieren:

- Beim Wechsel der aktuellen Benutzer-ID der Verbindung wird keine neue Verbindung hergestellt.
- Wenn in der Definition des gesicherten Kontextes die Authentifizierung der Benutzer-ID, zu der gewechselt werden soll, nicht als erforderlich definiert ist, dann entsteht der mit der Authentifizierung eines neuen Benutzers auf dem Datenbankserver verbundene Systemaufwand nicht.

Beispiel zum Erstellen eines gesicherten Kontextes

Der Sicherheitsadministrator erstellt das folgende Objekt eines gesicherten Kontextes:

```
CREATE TRUSTED CONTEXT CTX1
  BASED UPON CONNECTION USING SYSTEM AUTHID BENUTZER2
  ATTRIBUTES (ADDRESS   '192.0.2.1')
  DEFAULT ROLE managerRole
   ENABLE
```

Wenn Benutzer *benutzer1* eine gesicherte Verbindung über die IP-Adresse 192.0.2.1 anfordert, gibt das DB2-Datenbanksystem eine Warnung (SQLSTATE 01679, SQLCODE +20360) zurück, um anzuzeigen, dass keine gesicherte Verbindung hergestellt werden konnte und dass dem Benutzer *benutzer1* nun eine nicht gesicherte Verbindung zugeordnet wurde. Wenn hingegen der Benutzer *benutzer2* eine gesicherte Verbindung über die IP-Adresse 192.0.2.1 anfordert, wird diese Anforderung ausgeführt, weil die Verbindungsattribute vom gesicherten Kontext CTX1 erfüllt werden. Nachdem der Benutzer *benutzer2* nun eine gesicherte Verbindung hergestellt hat, kann er nun alle Zugriffsrechte und Berechtigungen anfordern, die der Rolle managerRole des gesicherten Kontextes zugeordnet sind. Diese Zugriffsrechte und Berechtigungen stehen dem Benutzer *benutzer2* außerhalb des Geltungsbereichs dieser gesicherten Verbindung möglicherweise nicht zur Verfügung.

Übernahme der Rollenzugehörigkeit über einen gesicherten Kontext

Der aktuelle Benutzer einer gesicherten Verbindung kann mithilfe der automatischen Rollenübernahme über den gesicherten Kontext zusätzliche Zugriffsrechte anfordern, wenn diese Möglichkeit vom Sicherheitsadministrator in der Definition des relevanten gesicherten Kontextes so festgelegt wurde.

Eine Rolle kann standardmäßig von allen Benutzern einer gesicherten Verbindung übernommen werden. Der Sicherheitsadministrator kann die Definition des gesicherten Kontextes auch verwenden, um eine Rolle anzugeben, die nur von ganz bestimmten Benutzern übernommen werden soll.

Die aktiven Rollen, über die eine Sitzungsberechtigungs-ID verfügen kann, während sie über eine gesicherte Verbindung verfügt, lauten wie folgt:

- Die Rollen, die der Sitzungsberechtigungs-ID normalerweise zugeordnet sind, sowie
- entweder die Standardrolle des gesicherten Kontextes oder die Rolle eines speziellen Benutzers des gesicherten Kontextes, sofern diese definiert sind.

Anmerkung:

- Wenn Sie die Benutzerauthentifizierung mithilfe eines angepassten Sicherheits-Plug-ins konfigurieren, das so konzipiert ist, dass die Systemberechtigungs-ID und die Sitzungsberechtigungs-ID, die bei erfolgreicher Verbindungsherstellung von diesem Plug-in erzeugt werden, nicht identisch sind, kann die Rolle eines gesicherten Kontextes nicht über diese Verbindung übernommen werden. Dies gilt auch dann, wenn es sich hierbei um eine gesicherte Verbindung handelt.
- Zugriffsrechte gesicherter Kontexte, die über eine Rolle angefordert wurden, gelten ausschließlich für dynamische DML-Operationen. Sie gelten hingegen nicht für die folgenden Komponenten:
  - DDL-Operationen
  - Operationen mit nicht dynamischem SQL (Operationen mit statischen SQL-Anweisungen wie z. B. BIND und REBIND sowie implizite Rebinds, inkrementelle Bindungen usw.)

Anfordern benutzerspezifischer Zugriffsrechte im gesicherten Kontext

Der Sicherheitsadministrator kann die Definition des gesicherten Kontextes verwenden, um dem gesicherten Kontext bestimmte Rollen zuzuordnen, sodass Folgendes gilt:

- Alle Benutzer der gesicherten Verbindung können standardmäßig eine angegebene Rolle übernehmen.
- Bestimmte Benutzer der gesicherten Verbindung können eine angegebene Rolle übernehmen.

Wenn beim Benutzer einer gesicherten Verbindung zu einer neuen Berechtigungs-ID gewechselt wird und für diese neue Berechtigungs-ID eine benutzerspezifische Rolle des gesicherten Kontextes vorhanden ist, dann setzt die benutzerspezifische Rolle die Standardrolle des gesicherten Kontextes außer Kraft, sofern eine solche Rolle vorhanden ist. Diese Vorgehensweise wird im folgenden Beispiel dargestellt.

Beispiel zum Erstellen eines gesicherten Kontextes, in dem eine Standardrolle und eine benutzerspezifische Rolle zugeordnet wird

Der Sicherheitsadministrator erstellt das folgende Objekt eines gesicherten Kontextes:

```
CREATE TRUSTED CONTEXT CTX1
  BASED UPON CONNECTION USING SYSTEM AUTHID BENUTZER1
 ATTRIBUTES (ADDRESS   '192.0.2.1')
  WITH USE FOR BENUTZER2 WITH AUTHENTICATION,
               BENUTZER3 WITHOUT AUTHENTICATION
  DEFAULT ROLE AUDITOR
  ENABLE
```

Wenn BENUTZER1 eine gesicherte Verbindung herstellt, werden die Zugriffsrechte, die der Rolle AUDITOR gewährt wurden, von dieser Berechtigungs-ID übernommen. Diese Zugriffsrechte werden auch von BENUTZER3 übernommen, wenn die aktuelle Berechtigungs-ID für die gesicherte Verbindung gewechselt und dann die Benutzer-ID von BENUTZER3 verwendet wird. (Wenn die Benutzer-ID der Verbindung zu einem bestimmten Zeitpunkt gewechselt und dann die Benutzter-ID von BENUTZER2 verwendet wird, dann übernimmt BENUTZER2 auch die Standardrolle des gesicherten Kontextes, AUDITOR.) Der Sicherheitsadministrator kann angeben, dass BENUTZER3 eine andere Rolle als die Standardrolle des gesicherten Kontextes übernehmen soll. Hierzu kann diesem Benutzer wie folgt eine bestimmte Rolle zugeordnet werden:

```
CREATE TRUSTED CONTEXT CTX1
 BASED UPON CONNECTION USING SYSTEM AUTHID BENUTZER1
 ATTRIBUTES (ADDRESS   '192.0.2.1')
  WITH USE FOR BENUTZER2 WITH AUTHENTICATION,
               BENUTZER3 WITHOUT AUTHENTICATION ROLE OTHER_ROLE
  DEFAULT ROLE AUDITOR
  ENABLE
```

Wenn die aktuelle Benutzer-ID der gesicherten Verbindung gewechselt und dann die Benutzer-ID von BENUTZER3 verwendet wird, übernimmt dieser Benutzer nicht mehr die Standardrolle des gesicherten Kontextes. Stattdessen wird die spezielle Rolle OTHER_ROLE übernommen, die dem Benutzer vom Sicherheitsadministrator zugeordnet wurde.

Regeln zum Wechseln der Benutzer-ID bei einer expliziten gesicherten Verbindung

Bei einer expliziten gesicherten Verbindung können Sie die Benutzer-ID der Verbindung wechseln. Es sind bestimmte Regeln zu beachten.

1. Wenn die Anforderung zum Wechseln nicht über eine explizite gesicherte Verbindung abgesetzt wurde und die Anforderung zum Wechseln zur Verarbeitung an den Server gesendet wird, dann wird die Verbindung geschlossen und das System gibt eine Fehlernachricht (SQLSTATE 08001, SQLCODE -30082 mit Ursachencode 41) zurück.
2. Wenn die Anforderung zum Wechseln nicht an einer Transaktionsgrenze abgesetzt wird, wird die Transaktion mit einem Rollback zurückgesetzt und die Anforderung zum Wechseln wird zur Verarbeitung an den Server gesendet. Die Verbindung wird in den Status 'Nicht verbunden' versetzt und das System gibt eine Fehlernachricht (SQLSTATE 58009, SQLCODE -30020) zurück.
3. Wenn die Anforderung zum Wechseln innerhalb einer gespeicherten Prozedur abgesetzt wird, gibt das System eine Fehlernachricht (SQLCODE -30090, Ursa-

chencode 29) zurück, in der darauf hingewiesen wird, dass diese Operation in der vorliegenden Umgebung nicht zulässig ist. Der Verbindungsstatus wird beibehalten und die Verbindung wird nicht in den Status 'Nicht verbunden' versetzt. Nachfolgende Anforderungen können verarbeitet werden.

4. Wenn die Anforderung zum Wechseln über eine Instanzverbindung (und nicht über eine Datenbankverbindung) geleitet wird, dann wird diese Verbindung geschlossen und das System gibt eine Fehlernachricht (SQLCODE -30005) zurück.
5. Wenn die Anforderung zum Wechseln mit einer Berechtigungs-ID ausgeführt wird, die auf der gesicherten Verbindung nicht zulässig ist, wird der Fehler (SQLSTATE 42517, SQLCODE -20361) zurückgegeben und die Verbindung wird in den Status 'Nicht verbunden' versetzt.
6. Wenn die Anforderung zum Wechseln mit einer Berechtigungs-ID ausgeführt wird, die auf der gesicherten Verbindung mit der Einstellung WITH AUTHENTICATION zulässig ist, das erforderliche Authentifizierungstoken jedoch nicht bereitgestellt wird, wird ein Fehler (SQLSTATE 42517, SQLCODE -20361) zurückgegeben und die Verbindung wird in den Status 'Nicht verbunden' versetzt.
7. Wenn das Objekt für den gesicherten Kontext, das der gesicherten Verbindung zugeordnet ist, inaktiviert wurde und eine Anforderung zum Wechseln für diese gesicherte Verbindung abgesetzt wird, dann gibt das System einen Fehler (SQLSTATE 42517, SQLCODE -20361) zurück und die Verbindung wird in den Status 'Nicht verbunden' versetzt.

   In diesem Fall werden nur Benutzerwechselanforderungen akzeptiert, in denen die Benutzer-ID, über die die gesicherte Verbindung hergestellt wurde, oder die Benutzer-ID NULL angegeben ist. Wenn zu der Benutzer-ID gewechselt wird, mit der die gesicherte Verbindung hergestellt wurde, dann übernimmt diese Benutzer-ID keine der für den gesicherten Kontext definierten Rollen (d. h. weder die Standardrolle des gesicherten Kontextes noch seine benutzerspezifische Rolle).
8. Wenn das Attribut für die Systemberechtigungs-ID des Objekts für den gesicherten Kontext, das der gesicherten Verbindung zugeordnet ist, geändert wurde und eine Anforderung zum Wechseln für diese gesicherte Verbindung abgesetzt wird, dann gibt das System einen Fehler (SQLSTATE 42517, SQLCODE -20361) zurück und die Verbindung wird in den Status 'Nicht verbunden' versetzt.

   In diesem Fall werden nur Benutzerwechselanforderungen akzeptiert, in denen die Benutzer-ID, über die die gesicherte Verbindung hergestellt wurde, oder die Benutzer-ID NULL angegeben ist. Wenn zu der Benutzer-ID gewechselt wird, mit der die gesicherte Verbindung hergestellt wurde, dann übernimmt diese Benutzer-ID keine der für den gesicherten Kontext definierten Rollen (d. h. weder die Standardrolle des gesicherten Kontextes noch seine benutzerspezifische Rolle).
9. Wenn das Objekt für den gesicherten Kontext, das der gesicherten Verbindung zugeordnet ist, gelöscht wird und eine Anforderung zum Wechseln für diese gesicherte Verbindung abgesetzt wird, dann gibt das System einen Fehler (SQLSTATE 42517, SQLCODE -20361) zurück und die Verbindung wird in den Status 'Nicht verbunden' versetzt.

   In diesem Fall werden nur Benutzerwechselanforderungen akzeptiert, in denen die Benutzer-ID, über die die gesicherte Verbindung hergestellt wurde, oder die Benutzer-ID NULL angegeben ist. Wenn zu der Benutzer-ID gewechselt wird, mit der die gesicherte Verbindung hergestellt wurde, dann über-

nimmt diese Benutzer-ID keine der für den gesicherten Kontext definierten Rollen (d. h. weder die Standardrolle des gesicherten Kontextes noch seine benutzerspezifische Rolle).

10. Wenn die Anforderung zum Wechseln mit einer Benutzer-ID ausgeführt wird, die auf der gesicherten Verbindung zulässig ist, diese Benutzer-ID jedoch nicht über das Zugriffsrecht CONNECT für die Datenbank verfügt, wird die Verbindung in den Status 'Nicht verbunden' versetzt und das System gibt eine Fehlernachricht (SQLSTATE 08004, SQLCODE -1060) zurück.
11. Wenn die Systemberechtigungs-ID des gesicherten Kontextes in der Klausel WITH USE FOR aufgeführt ist, dann berücksichtigt das DB2-Datenbanksystem die Authentifizierungseinstellung für die Systemberechtigungs-ID bei Benutzerwechselanforderungen, mit denen zurück zu dieser Systemberechtigungs-ID gewechselt werden soll. Wenn die Systemberechtigungs-ID des gesicherten Kontextes in der Klausel WITH USE FOR nicht aufgeführt wird, dann ist die Benutzerwechselanforderung zum Zurückwechseln zur Systemberechtigungs-ID immer zulässig. Dies gilt auch dann, wenn keine Authentifizierung durchgeführt wird.

Anmerkung: Wenn die Verbindung in den Status 'Nicht verbunden' versetzt wird, dann werden nur die folgenden Anforderungen vom System akzeptiert und führen nicht zur Ausgabe des Fehlers "Der Anwendungsstatus ist fehlerhaft. Die Verbindung zur Datenbank ging verloren." (SQLCODE -900):

- Benutzerwechselanforderung
- COMMIT- oder ROLLBACK-Anweisung
- DISCONNECT-, CONNECT RESET- oder CONNECT-Anforderung

Anmerkung: Wenn die Benutzer-ID der gesicherten Verbindung gewechselt wird, dann gehen alle Traces für die Verbindungsumgebung, die zu der alten Benutzer-ID gehören, verloren. Dies bedeutet, dass der Wechsel von Benutzer-IDs zu einer Umgebung führt, die einer neuen Verbindungsumgebung entspricht. Wenn die alte Benutzer-ID der Verbindung beispielsweise über temporäre Tabellen oder geöffnete WITH HOLD-Cursor verfügte, gehen diese Objekte vollständig verloren, wenn die Benutzer-ID für diese Verbindung gewechselt und eine neue Benutzer-ID definiert wird.

Fehlerbestimmung für gesicherte Kontexte

Eine explizite gesicherte Verbindung ist eine Verbindung, die mithilfe einer bestimmten, expliziten Anforderung für eine gesicherte Verbindung erfolgreich hergestellt werden kann. Wenn Sie eine explizite gesicherte Verbindung anfordern und nicht über die hierfür erforderlichen Berechtigungen verfügen, wird Ihnen eine reguläre Verbindung zugeteilt und das System gibt eine Warnung (+20360) aus.

Um festzustellen, warum ein Benutzer keine gesicherte Verbindung herstellen konnte, muss der Sicherheitsadministrator die Definition des gesicherten Kontextes in den Systemkatalogen und die Verbindungsattribute prüfen. Dabei müssen insbesondere die IP-Adresse, über die die Verbindung hergestellt werden soll, die Verschlüsselungsstufe des Datenstroms oder Netzes sowie die Systemberechtigungs-ID berücksichtigt werden, die zur Herstellung der Verbindung verwendet wird. Die Option **-application** des Dienstprogramms **db2pd** gibt diese sowie die folgenden zusätzlichen Informationen zurück:

- Sicherungstyp der Verbindung: Dieser Typ gibt an, ob die Verbindung gesichert ist oder nicht. Bei gesicherten Verbindungen enthält diese Angabe auch die Information, ob es sich um eine explizite oder implizite gesicherte Verbindung handelt.
- Name des gesicherten Kontextes: Der Name des gesicherten Kontextes, der der gesicherten Verbindung zugeordnet ist.
- Übernommene Rolle: Die Rolle, die durch die gesicherte Verbindung übernommen wurde.

Die folgenden Ursachen für Fehler beim Abrufen einer expliziten gesicherten Verbindung treten am häufigsten auf:

- Die Clientanwendung verwendet nicht TCP/IP zur Kommunikation mit dem DB2-Server. TCP/IP ist das einzige unterstützte Protokoll für eine Clientanwendung, über das mit dem DB2-Server kommuniziert werden kann und das zur Herstellung einer gesicherten Verbindung (explizit oder implizit) genutzt werden kann.
- Der Authentifizierungstyp des Datenbankservers ist auf den Wert CLIENT eingestellt.
- Der Datenbankserver hat kein aktiviertes Objekt für einen gesicherten Kontext. Die Definition eines Objekts für einen gesicherten Kontext muss explizit ENABLE (Aktivieren) angeben, damit der entsprechende gesicherte Kontext beim Abgleichen der Attribute einer eingehenden Verbindung in Betracht gezogen wird.
- Die Objekte für gesicherte Kontexte auf dem Datenbankserver stimmen nicht mit den entsprechenden Attributen für die gesicherte Verbindung überein, die dargestellt werden. Es kann zum Beispiel eine der folgenden Situationen vorliegen:
  - Die Systemberechtigungs-ID der Verbindung entspricht keiner Systemberechtigungs-ID eines Objekts für einen gesicherten Objekt.
  - Die IP-Adresse, von der die Verbindung ausgeht, stimmt mit keiner IP-Adresse in dem Objekt für den gesicherten Kontext überein, das für die Verbindung in Betracht kommt.
  - Die von der Verbindung verwendete Datenstromverschlüsselung stimmt nicht mit dem Wert des Attributs ENCRYPTION in dem Objekt für den gesicherten Kontext überein, das für die Verbindung in Betracht gezogen wird.

  Mithilfe des Tools **db2pd** können Sie die IP-Adresse, von der aus die Verbindung hergestellt wurde, die Verschlüsselungsstufe des Datenstroms oder des Netzes, das von der Verbindung verwendet wird, und die Systemberechtigungs-ID, unter der die Verbindung hergestellt wird, ermitteln. Sie können die Katalogsichten SYSCAT.CONTEXTS und SYSCAT.CONTEXTATTRIBUTES abfragen, um die Definition eines bestimmten Objekts für einen gesicherten Kontext, wie zum Beispiel die Systemberechtigungs-ID, die Gruppe der zulässigen IP-Adressen und den Wert des Attributs ENCRYPTION des Objekts, zu ermitteln.

Die folgenden Ursachen für einen Benutzerwechselfehler treten am häufigsten auf:

- Die Benutzer-ID, zu der gewechselt werden soll, besitzt keine CONNECT-Zugriffsrechte für die Datenbank. In diesem Fall wird der Fehler SQL1060N zurückgegeben.
- Die Benutzer-ID, zu der gewechselt werden soll, oder die Gruppe PUBLIC ist in der Klausel WITH USE FOR des Objekts des gesicherten Kontexts, das der explizit gesicherten Verbindung zugeordnet ist, nicht definiert.
- Das Wechseln des Benutzers ist mit Authentifizierung zulässig, jedoch gibt der Benutzer keine Berechtigungsnachweise bzw. falsche Berechtigungsnachweise an.

- Eine Anforderung zum Benutzerwechsel wird nicht an einer Transaktionsgrenze abgesetzt.
- Der gesicherte Kontext, der einer gesicherten Verbindung zugeordnet ist, wurde inaktiviert, gelöscht oder geändert. In diesem Fall ist nur ein Wechsel zu der Benutzer-ID zulässig, unter der die gesicherte Verbindung hergestellt wurde.

Kapitel 4. Zeilen- und Spaltenzugriffssteuerung (RCAC) - Übersicht

In DB2 Version 10.1 wird als zusätzliche Ebene der Datensicherheit die Zeilen- und Spaltenzugriffssteuerung (RCAC) eingeführt. Diese Zugriffssteuerung wird manchmal auch als differenzierte Zugriffssteuerung oder FGAC bezeichnet. RCAC steuert den Zugriff auf eine Tabelle auf Zeilen- oder Spaltenebene oder beides. RCAC kann als Ergänzung zum Modell für Tabellenzugriffsrechte verwendet werden.

Zur Einhaltung von gesetzlichen Richtlinien sollten Sie verschiedene Prozeduren und Methoden implementieren, die sicherstellen, dass die Informationen entsprechend geschützt sind. Die Mitarbeiter in Ihrem Unternehmen haben nur auf die Daten Zugriff, die für die Ausübung der jeweiligen Jobs-Tasks erforderlich sind. Beispiel: Die gesetzlichen Richtlinien in Ihrer Region sehen vor, dass ein Arzt nur die Krankenakten seiner eigenen Patienten einsehen darf, nicht aber die Akten anderer Patienten. Dieselben Verordnungen geben außerdem vor, dass ein Leistungsanbieter im Gesundheitswesen ohne ausdrückliche Zustimmung eines Patienten keinen Zugriff auf persönliche Patientendaten, wie die Telefonnummer des Patienten, haben darf.

Mithilfe der Zeilen- und Spaltenzugriffssteuerung können Sie sicherstellen, dass Ihre Benutzer nur auf die Daten Zugriff haben, die für ihre Arbeit relevant sind. Beispielsweise kann ein Krankenhaussystem, für das DB2 for Linux, UNIX and Windows und RCAC implementiert ist, Patienteninformationen so filtern, dass nur Daten ausgegeben werden, die ein bestimmter Arzt benötigt. Andere Patienten sind für diesen Arzt demnach gar nicht vorhanden. In ähnlicher Weise sind für Mitarbeiter des Patientenservice beim Abfragen von Daten aus der Patiententabelle desselben Krankenhauses zwar Name und Telefonnummer eines Patienten sichtbar, die Spalte mit der entsprechenden Krankengeschichte ist jedoch ausgeblendet. Wenn Daten ausgeblendet sind, wird anstelle der tatsächlichen Krankenstelle der Wert NULL oder ein alternativer Wert angezeigt.

Die Zeilen- und Spaltenzugriffssteuerung (RCAC) bietet die folgenden Vorteile:

- Alle Datenbankbenutzer unterliegen an sich den Regeln der Zeilen- und Spaltenzugriffssteuerung.

  Selbst Instanzen auf höherer Ebene, wie Benutzer mit der Berechtigung DATAACCESS, sind von diesen Regeln nicht ausgenommen. Allein Benutzer mit der Sicherheitsadministratorberechtigung (SECADM) können die Zeilen- und Spaltenzugriffssteuerung in einer Datenbank verwalten. RCAC kann daher verwendet werden, um Benutzern mit der Berechtigung DATAACCESS den freien Zugriff auf alle Daten in einer Datenbank zu verwehren.
- Tabellendaten sind geschützt, ganz gleich, wie auf die Tabelle über SQL zugegriffen wird.

  Anwendungen, unkonventionelle Abfragetools und Berichtserstellungstools unterliegen alle den RCAC-Regeln. Die Umsetzung dieser Regeln ist dabei datenorientiert.
- Es müssen keine Änderungen an Anwendungen vorgenommen werden, um diese zusätzliche Ebene der Datensicherheit nutzen zu können.

  Das heißt, die Zeilen- und Spaltenzugriffssteuerung wird auf eine Art und Weise eingerichtet und definiert, die für vorhandene Anwendungen nicht offensichtlich ist. RCAC stellt jedoch in konzeptioneller Hinsicht einen veränderten Blick-

winkel bereit, indem der Fokus nicht mehr darauf liegt, was gefragt wird, sondern wer was fragt. Die Ergebnisse für dieselbe Abfrage sind je nach Kontext, in dem die Abfrage gestellt wurde, unterschiedlich, und es werden keine Warnungen oder Fehler zurückgegeben. Aber genau dieses Verhalten der Lösung ist beabsichtigt. Es bedeutet, dass Anwendungsentwickler und Datenbankadministratoren sich bewusst sein müssen, dass bei Abfragen bezüglich der Daten in einer Tabelle nicht alle Daten mit eingezogen werden, es sei denn, es liegen die entsprechenden Berechtigungen dafür vor.

Zeilen- und Spaltenzugriffssteuerung (RCAC) - Regeln

Die Zeilen- und Spaltenzugriffssteuerung (RCAC) regelt die die Zugriffssteuerung auf Tabellenebene konkret für die Daten selbst. Für Zeilen und Spalten erstellte SQL-Regeln bilden dabei die Basis für die Implementierung dieser Funktionalität.

Die Zeilen- und Zugriffssteuerung ist ein Zugriffssteuerungsmodell, in dem ein Sicherheitsadministrator Datenschutz- und Sicherheitsrichtlinien verwaltet. Im Rahmen von RCAC haben alle Benutzer Zugriff auf dieselbe Tabelle, anstatt alternative Ansichten dieser Tabelle aufzurufen. RCAC beschränkt jedoch den Zugriff auf die Tabelle basierend auf den einzelnen Benutzerberechtigungen oder auf Regeln, die durch eine der Tabelle zugeordneten Richtlinie festgelegt sind. Es gibt zwei Gruppen von Regeln: eine bezieht sich auf Zeilen, die andere auf Spalten.

- Zeilenberechtigung
  - Eine Zeilenberechtigung ist ein Datenbankobjekt, das eine Zeilenzugriffssteuerungsregel für eine bestimmte Tabelle angibt.
  - Eine Zeilenzugriffssteuerungsregel ist eine SQL-Suchbedingung, die beschreibt, auf welche Zeilen ein Benutzer Zugriff hat.
- Spaltenmaske
  - Eine Spaltenmaske ist ein Datenbankobjekt, das eine Spaltenzugriffssteuerungsregel für eine bestimmte Spalte in einer Tabelle angibt.
  - Eine Spaltenzugriffssteuerungsregel ist ein CASE-Ausdruck in SQL, der beschreibt, welche Spaltenwerte ein Benutzer sehen darf und unter welchen Bedingungen.

SQL-Anweisungen zur Verwaltung von RCAC-Regeln

Mit den folgenden SQL-Anweisungen können Sie RCAC-Regeln erstellen, ändern oder löschen.

Integrierte Funktionen zum Verwalten von RCAC-Berechtigungen und -Masken

Mit den folgenden integrierten Skalarfunktionen können Sie verschiedene Bedingungen in Ihren Berechtigungen und Masken ausdrücken. Beispielsweise muss ein Benutzer zu mindestens einer Rolle oder Gruppe gehören, um auf eine bestimmte Zeile zugreifen zu können.

Szenario: ExampleHMO als Benutzer der Zeilen- und Spaltenzugriffssteuerung

Im diesem Szenario präsentiert sich ExampleHMO, ein nationales Unternehmen mit einer umfassenden und aktiven Liste von Patienten, als Benutzer der Zeilen- und Spaltenzugriffssteuerung. ExampleHMO verwendet diese Art der Zugriffssteuerung, um sicherzustellen, dass die Datenbankrichtlinien des Unternehmens den gesetzlichen Bestimmungen für Datenschutz und Sicherheit sowie den internen Unternehmenszielen entsprechen.

Unternehmen wie ExampleHMO, die mit Angaben zum Gesundheitszustand von Patienten und deren persönlichen Daten arbeiten, müssen gesetzliche Bestimmungen zu Datenschutz und Sicherheit (in den USA beispielsweise der Health Insurance Portability and Accountability Act (HIPAA)) einhalten. Diese Bestimmungen sorgen dafür, dass sämtliche vertrauliche Patienteninformationen nur von Instanzen angezeigt, gemeinsam genutzt und geändert werden können, die über die entsprechenden Berechtigungen dafür verfügen. Jegliche Nichteinhaltung solcher Bestimmungen kann eine hohe Geldstrafe sowie ein zivil- und strafrechtliches Verfahren zur Folge habe.

ExampleHMO muss sicherstellen, dass die in den Unternehmensdatenbanksystemen gespeicherten Daten ausreichend geschützt sind und nur dazu berechtigte Benutzer auf diese Daten zugreifen können. Im Rahmen typischer Datenschutzverordnungen können nur Benutzer mit den entsprechenden Berechtigungen auf bestimmte Patienteninformationen zugreifen und diese ändern.

Szenario: ExampleHMO als Benutzer der Zeilen- und Spaltenzugriffssteuerung - Sicherheitsrichtlinien

ExampleHMO implementiert eine Sicherheitsstrategie, bei der der Datenzugriff auf DB2-Datenbanken basierend auf bestimmten Sicherheitsrichtlinien gewährt wird.

Die Sicherheitsrichtlinien entsprechen den gesetzlichen Sicherheits- und Datenschutzbestimmungen. In der ersten Spalte sind die Richtlinien und die Herausforderungen aufgelistet, denen sich das Unternehmen gegenübersieht, die zweite Spalte enthält die Sicherheitsfunktion der Zeilen- und Spaltenzugriffssteuerung von DB2 für die entsprechende Herausforderung.

Tabelle 8.

Herausforderung	**Sicherheitsfunktion der Zeilen- und Spaltenzugriffssteuerung für diese Herausforderung**
Spaltenzugriff auf privilegierte Benutzer beschränken. Beispielsweise darf Jane als Arzneimittelforscherin eines Partnerunternehmens keine vertraulichen Patienteninformationen, wie Versicherungsnummern oder Informationen zum Gesundheitszustand der Patienten, anzeigen.	Mithilfe von Spaltenmasken können vertrauliche Daten gefiltert bzw. ausgeblendet (maskiert) werden.

Tabelle 8. (Forts.)

Herausforderung	Sicherheitsfunktion der Zeilen- und Spaltenzugriffssteuerung für diese Herausforderung
Zeilenzugriff auf privilegierte Benutzer beschränken. Dr. Lee darf nur die Patienteninformationen für seine eigenen Patienten einsehen, nicht aber für alle Patienten im ExampleHMO-System.	Es können Zeilenberechtigungen implementiert werden, mit denen gesteuert werden kann, welcher Benutzer eine bestimmte Zeile anzeigen darf.
Datenzugriff auf der Basis eines berechtigten Interesses einschränken.	Mit Zeilenberechtigungen können Daten in Tabellen auf Benutzerebene eingeschränkt werden.
Andere Datenbankobjekte, wie UDFs, Trigger und Ansichten, für mit RCAC gesicherte Daten einschränken.	Die Zeilen- und Spaltenzugriffssteuerung schützt Daten auf Datenebene. Es ist diese datenorientierte Natur der Zugriffssteuerung, die Sicherheitsrichtlinien sogar für Datenbankobjekte wie UDFs, Trigger und Ansichten durchsetzt.

Szenario: ExampleHMO als Benutzer der Zeilen- und Spaltenzugriffssteuerung - Datenbankbenutzer und -rollen

In diesem Szenario erstellen, sichern und verwenden verschiedene Personen ExampleHMO-Daten. Diese Personen verfügen über verschiedene Benutzerrechte und Datenbankberechtigungen.

Das Unternehmen ExampleHMO hat seine Sicherheitsstrategie so implementiert, dass die Daten nach Art des Zugriffs über die Datenbank klassifiziert werden können. Der interne und externe Datenzugriff basiert auf der Trennung der Aufgabenbereiche der Benutzer, die auf die Daten zugreifen, und deren Datenzugriffsberechtigungen. ExampleHMO hat die folgenden Datenbankrollen zur Trennung dieser Aufgaben erstellt:

PCP
Für behandelnde Ärzte.

DRUG_RESEARCH
Für Forschungsmitarbeiter.

ACCOUNTING
Für Buchhalter.

MEMBERSHIP
Für Mitglieder, die Patienten für das Opt-in oder Opt-out hinzufügen.

PATIENT
Für Patienten.

Die folgenden Personen erstellen, sichern und verwenden ExampleHMO-Daten:

Alex
Sicherheitsadministrator bei ExampleHMO. Er verfügt über die Berechtigung SECADM.

Peter
Datenbankadministrator bei ExampleHMO. Er verfügt über die Berechtigung DBADM.

Paul
Datenbankentwickler bei ExampleHMO. Er verfügt über die Berechtigung zum Erstellen von Triggern und benutzerdefinierten Funktionen.

Dr. Lee
Mediziner bei ExampleHMO. Er gehört der Rolle PCP an.

Jane
Arzneimittelforscherin bei Innovative Pharmaceutical Company, einem Partnerunternehmen von ExampleHMO. Sie gehört der Rolle DRUG_RESEARCH an.

John
Buchhalter bei ExampleHMO. Er gehört der Rolle ACCOUNTING an.

Tom
Mitarbeiter in der Mitgliedsschafabteilung bei ExampleHMO. Er gehört der Rolle MEMBERSHIP an.

Bob
Patient von ExampleHMO. Er gehört der Rolle PATIENT an.

Wenn Sie die SQL-Beispielanweisungen und -befehle in diesem Szenario anwenden möchten, müssen Sie diese Benutzer-IDs mit ihren aufgelisteten Berechtigungen erstellen.

Für die folgenden SQL-Beispielanweisungen wird angenommen, dass die Benutzer im System erstellt wurden. Mit den SQL-Anweisungen werden die einzelnen Rollen erstellt und den Benutzern die Berechtigungen **SELECT** und **INSERT** für die verschiedenen Tabellen in der ExampleHMO-Datenbank erteilt:

```
--Erstellen von Rollen und Erteilen von Berechtigungen

CREATE ROLE PCP;

CREATE ROLE DRUG_RESEARCH;

CREATE ROLE ACCOUNTING;

CREATE ROLE MEMBERSHIP;

CREATE ROLE PATIENT;

GRANT ROLE PCP TO USER LEE;
GRANT ROLE DRUG_RESEARCH TO USER JANE;
GRANT ROLE ACCOUNTING  TO USER JOHN;
GRANT ROLE MEMBERSHIP TO USER TOM;
GRANT ROLE PATIENT TO USER BOB;
```

Szenario: ExampleHMO als Benutzer der Zeilen- und Spaltenzugriffssteuerung - Datenbanktabellen

Bei diesem Szenario liegt der Fokus auf zwei Tabellen in der ExampleHMO-Datenbank: auf der Tabelle PATIENT und der Tabelle PATIENTCHOICE.

In der Tabelle PATIENT sind allgemeine Patienteninformationen und Angaben zum Gesundheitszustand der Patienten gespeichert. In diesem Szenario werden die folgenden Spalten in der Tabelle PATIENT betrachtet:

SSN
Die Versicherungsnummer des Patienten. Die Versicherungsnummer eines Patienten gilt als persönliche Information.

NAME
Der Name des Patienten. Der Name eines Patienten gilt als persönliche Information.

ADDRESS
Die Adresse des Patienten. Die Adresse eines Patienten gilt als persönliche Information.

USERID
Die Datenbank-ID des Patienten.

PHARMACY
Informationen zum Gesundheitszustand des Patienten.

ACCT_BALANCE
Die Abrechnungsinformationen des Patienten.

PCP_ID
Die Hausarztdatenbank-ID des Patienten

In der Tabelle PATIENTCHOICE werden die Opt-in- und Opt-out-Informationen der einzelnen Patienten gespeichert, die anzeigen, ob ein Patient möchte, dass die Informationen zu seinem Gesundheitszustand für Forschungszwecke für unternehmensfremde Personen in dieser Tabelle offengelegt werden. In diesem Szenario werden die folgenden Spalten in der Tabelle PATIENTCHOICE betrachtet:

SSN
Die Versicherungsnummer des Patienten wird zum Abgleich der Patienten mit ihrer Auswahl verwendet.

CHOICE
Der Name einer Auswahloption, die ein Patient wählen kann.

VALUE
Die getroffene Entscheidung des Patienten bezüglich der Auswahloption.

Die Zeile 123-45-6789, drug_research, opt-in gibt beispielsweise an, dass der Patient mit SSN 123-45-6789 bereit ist, seine persönlichen Informationen für medizinische Forschungszwecke offenzulegen.

Mit den folgenden SQL-Beispielanweisungen werden die Tabellen PATIENT, PATIENTCHOICE und ACCT_HISTORY erstellt. Es werden Berechtigungen für die Tabelle gewährt und Daten eingefügt:

```
--Tabelle PATIENT mit Informationen zu dem jeweiligen Patienten
CREATE TABLE PATIENT (
 SSN CHAR(11),
 USERID VARCHAR(18),
 NAME VARCHAR(128),
 ADDRESS VARCHAR(128),
 PHARMACY VARCHAR(250),
 ACCT_BALANCE DECIMAL(12,2) WITH DEFAULT,
 PCP_ID VARCHAR(18)
 );


--Tabelle PATIENTCHOICE, in der gespeichert wird, welche Informationen
--ein Patient bezüglich seiner Krankheitsgeschichte offenlegen möchte
CREATE TABLE PATIENTCHOICE (
 SSN CHAR(11),
 CHOICE VARCHAR(128),
 VALUE VARCHAR(128)
);
```

```
--Tabelle ACC_HISTORY, mit der Kontostände überwacht werden
CREATE TABLE ACCT_HISTORY(
 SSN   CHAR(11),
 BEFORE_BALANCE DECIMAL(12,2),
 AFTER_BALANCE DECIMAL(12,2),
 WHEN  DATE,
 BY_WHO VARCHAR(20)
);

--Erteilen von Berechtigungen

GRANT SELECT, UPDATE ON TABLE PATIENT TO ROLE PCP;

GRANT SELECT ON TABLE PATIENT TO ROLE DRUG_RESEARCH;

GRANT SELECT, UPDATE ON TABLE PATIENT TO ROLE ACCOUNTING;
GRANT SELECT ON TABLE ACCT_HISTORY TO ROLE ACCOUNTING;

GRANT SELECT, UPDATE, INSERT ON TABLE PATIENT TO ROLE MEMBERSHIP;
GRANT INSERT ON TABLE PATIENTCHOICE TO ROLE MEMBERSHIP;

GRANT SELECT ON TABLE PATIENT TO ROLE PATIENT;

GRANT SELECT, ALTER ON TABLE PATIENT TO USER ALEX;

GRANT ALTER, SELECT ON TABLE PATIENT TO USER PAUL;
GRANT INSERT ON TABLE ACCT_HISTORY TO USER PAUL;

--Einfügen von Patientendaten
INSERT INTO PATIENT VALUES('123-55-1234', 'MAX', 'Max', 'First Strt',
'hypertension', 89.70,'LEE');
INSERT INTO PATIENTCHOICE VALUES('123-55-1234', 'drug-research', 'opt-out');


INSERT INTO PATIENT VALUES('123-58-9812', 'MIKE', 'Mike', 'Long Strt', null, 8.30,'JAMES');
INSERT INTO PATIENTCHOICE VALUES('123-58-9812', 'drug-research', 'opt-out');


INSERT INTO PATIENT VALUES('123-11-9856', 'SAM', 'Sam', 'Big Strt', null, 0.00,'LEE');
INSERT INTO PATIENTCHOICE VALUES('123-11-9856', 'drug-research', 'opt-in');


INSERT INTO PATIENT VALUES('123-19-1454', 'DUG', 'Dug', 'Good Strt', null, 0.00,'JAMES');
INSERT INTO PATIENTCHOICE VALUES('123-19-1454', 'drug-research', 'opt-in');
```

Szenario: ExampleHMO als Benutzer der Zeilen- und Spaltenzugriffssteuerung - Verwaltung von Sicherheitsfunktionen

Die Verwaltung von Sicherheitsfunktionen sowie die Rolle des Sicherheitsadministrators (SECADM) spielen eine wichtige Rolle beim Schutz von Patienten- und Unternehmensdaten bei ExampleHMO. Bei ExampleHMO wurde auf höchster Ebene entschieden, dass verschiedene Personen über Berechtigungen für die Datenbankadministration und für die Verwaltung von Sicherheitsfunktionen verfügen.

Das Management von ExampleHMO beschließt daher, eine Rolle für die Zugriffsverwaltung auf die Unternehmensdaten zu erstellen. Außerdem wird beschlossen, dass Benutzer mit der Berechtigung DATAACCESS standardmäßig keinen Zugriff auf persönliche Daten und Informationen über den Gesundheitszustand von Patienten erhalten sollen.

Das Management wählt Alex als den alleinigen Sicherheitsadministrator für ExampleHMO aus. Ab nun kontrolliert Alex alle Berechtigungen für den Datenzugriff. Mit dieser Berechtigung kann Alex Sicherheitsregeln wie Zeilenberechtigungen und Spaltenmasken definieren und angeben, ob Funktionen und Trigger sicher sind oder nicht. Diese Regeln steuern, welche Benutzer Zugriff auf bestimmte Daten haben.

Anschließend erstellt Peter, der Datenbankadministrator, die erforderlichen Tabellen und richtet die benötigten Rollen ein. Datei nimmt er eine Aufgabenaufteilung vor. Die Aufgaben für die Datenbankverwaltung und die Verwaltung von Sicherheitsfunktionen werden so aufgeteilt, dass Alex der Sicherheitsadministrator wird.

Peter stellt eine Verbindung zur Datenbank her und erteilt Alex die Berechtigung SECADM. Dies kann er tun, da er zurzeit über die Berechtigungen DBADM, DATAACCESS und SECADM verfügt.

```
-- Um Aufgaben des Sicherheitsadministrator von denen des Systemadministrators zu trennen,
-- weist Peter als SECADMN dem Benutzer Alex die Berechtigung SECADM zu.

GRANT SECADM ON DATABASE TO USER ALEX;
```

Nachdem Alex die Berechtigung SECADM zugewiesen wurde, stellt er eine Verbindung zur Datenbank her und entzieht Peter die Administratorberechtigung. Die Aufgaben wurden somit aufgeteilt und nur Alex darf nun bei ExampleHMO internen und externen Benutzern Zugriff auf Daten erteilen. Die folgende SQL-Anweisung zeigt, wie Alex Peter die Berechtigung SECADM entzieht:

```
--Entziehen der Berechtigung SECADMIN von Peter

REVOKE SECADM ON DATABASE FROM USER PETER;
```

Szenario: ExampleHMO als Benutzer der Zeilen- und Spaltenzugriffssteuerung - Zeilenberechtigungen

Der Sicherheitsadministrator Alex beginnt damit, den Datenzugriff auf die ExampleHMO-Datenbank zu beschränken, indem er als Teil der Zeilen- und Spaltenzugriffssteuerung Zeilenberechtigungen verwendet. Mit einer Zeilenberechtigung werden die an Benutzer zurückgegebenen Daten nach Zeilen gefiltert.

Patienten dürfen ihre eigenen Daten einsehen. Ein behandelnder Arzt darf zwar die Daten seiner eigenen Patienten anzeigen, nicht jedoch die Daten von Patienten, die bei einem anderen Arzt in Behandlung sind. Benutzer mit den Rollen MEMBERSHIP, ACCOUNTING oder DRUG_RESEARCH haben Zugriff auf alle Patien-

teninformationen. Der Sicherheitsadministrator Alex wird gebeten, diese Berechtigungen so einzuschränken, dass nur Personen mit einem berechtigten Interesse auf bestimmte Zeilen zugreifen können.

Zeilenberechtigungen beschränken bzw. filtern Zeilen basierend auf dem an der Datenbank angemeldeten Benutzer. Im Beispielunternehmen ExampleHMO erzeugen die Zeilenberechtigungen eine horizontale Dateneinschränkung für die Tabelle PATIENT.

Alex implementiert die folgenden Zeilenberechtigungen so, dass die Benutzer in den verschiedenen Rollen jeweils nur das Ergebnis sehen können, das ihnen die zugewiesene Berechtigung erlaubt:

```
CREATE PERMISSION ROW_ACCESS ON PATIENT
------------------------------------------------------
-- Buchhaltungsinformationen:
-- Benutzer der Rolle PATIENT dürfen auf ihre eigene Zeile zugreifen
-- Benutzer der Rolle ROLE PCP dürfen auf die Zeilen ihrer eigenen Patienten zugreifen
-- Benutzer der Rollen MEMBERSHIP, ACCOUNTING und DRUG_RESEARCH dürfen
-- auf alle Zeilen zugreifen
------------------------------------------------------------
FOR ROWS WHERE(VERIFY_ROLE_FOR_USER(SESSION_USER,'PATIENT') = 1
AND
PATIENT.USERID = SESSION_USER) OR
(VERIFY_ROLE_FOR_USER(SESSION_USER,'PCP') = 1
AND
PATIENT.PCP_ID = SESSION_USER) OR
 (VERIFY_ROLE_FOR_USER(SESSION_USER,'MEMBERSHIP') = 1 OR
 VERIFY_ROLE_FOR_USER(SESSION_USER,'ACCOUNTING') = 1 OR
 VERIFY_ROLE_FOR_USER(SESSION_USER, 'DRUG_RESEARCH') = 1)
ENFORCED FOR ALL ACCESS
ENABLE;
```

Alex beobachtet jedoch, dass selbst nach der Erstellung einer Zeilenberechtigung die anderen Mitarbeiter immer noch alle Daten aufrufen können. Eine Zeilenberechtigung wird erst dann angewendet, wenn sie für die Tabelle aktiviert wird, für die sie definiert wurde. Daher muss Alex nun die Berechtigung aktivieren:

```
--Aktivieren der Zeilenzugriffssteuerung zum Implementieren von Zeilenberechtigungen

ALTER TABLE PATIENT ACTIVATE ROW ACCESS CONTROL;
```

Szenario: ExampleHMO als Benutzer der Zeilen- und Spaltenzugriffssteuerung - Spaltenmasken

Der Sicherheitsadministrator Alex beschränkt den Datenzugriff auf die ExampleHMO-Datenbank weiter, indem er als Teil der Zeilen- und Spaltenzugriffssteuerung Spaltenmasken verwendet. Spaltenmasken blenden Daten für Benutzer nach Spalten aus, es sei den, die Benutzer sind berechtigt, die entsprechenden Daten anzuzeigen.

Nur Benutzer in der Buchhaltung dürfen Zugang zu den Zahlungsinformationen von Patienten haben. Die jeweiligen Kontostände dürfen für keine anderen Datenbankbenutzer sichtbar sein. Alex wird gebeten, den Zugriff so zu beschränken, dass nur Benutzer der Rolle ACCOUNTING auf diese Informationen Zugriff haben.

Alex implementiert die folgende Spaltenmaske so, dass die Benutzer in den verschiedenen Rollen jeweils nur das Ergebnis sehen können, das ihnen die zugewiesene Berechtigung erlaubt:

```
--Erstellen einer Spaltenmaske für die Spalte ACCT_BALANCE in der Tabelle PATIENT

CREATE MASK ACCT_BALANCE_MASK ON PATIENT FOR
-----------------------------------------------------------
-- Buchhaltungsinformationen:
-- Benutzer der Rolle ACCOUNTING dürfen auf alle Informationen
-- in der Spalte ACCT_BALANCE zugreifen.
-- Benutzern anderer Rollen wird beim Zugriff auf diese Spalte nur ein
-- Nullwert angezeigt.
-----------------------------------------------------------
COLUMN ACCT_BALANCE RETURN
 CASE WHEN VERIFY_ROLE_FOR_USER(SESSION_USER,'ACCOUNTING') = 1
   THEN ACCT_BALANCE
  ELSE 0.00
 END
ENABLE;
```

Alex beobachtet jedoch, dass selbst nach der Erstellung einer Spaltenmaske die anderen Mitarbeiter immer noch alle Daten aufrufen können. Eine Spaltenmaske wird erst dann angewendet, wenn sie für die Tabelle aktiviert wird, für die sie definiert wurde. Daher muss Alex nun die Maske aktivieren:

```
--Aktivieren der Spaltenzugriffssteuerung zum Implementieren von Spaltenmasken

ALTER TABLE PATIENT ACTIVATE COLUMN ACCESS CONTROL;
```

Alex wird vom Management gebeten, die Versicherungsnummer der Patienten auszublenden. Nur Patienten, behandelnde Ärzte, Buchhalter oder Personen der Rolle MEMBERSHIP können die Spalte SSN anzeigen.

Um auch die Details in der Spalte PHARMACY eines Patienten zu schützen, dürfen diese Informationen nur von Arzneimittelforschern oder Ärzten eingesehen werden. Arzneimittelforscher können diese Daten nur sehen, wenn der Patient sich bereiterklärt hat, die Informationen offenzulegen.

Alex implementiert die folgenden Spaltenmasken so, dass die Benutzer in den verschiedenen Rollen jeweils nur das Ergebnis sehen können, das ihnen die zugewiesene Berechtigung erlaubt:

```
CREATE MASK SSN_MASK ON PATIENT FOR
-----------------------------------------------------
-- Persönliche Kontaktinformationen:
-- Benutzer der Rollen PATIENT, PCP, MEMBERSHIP und ACCOUNTING dürfen auf alle
-- Informationen in den Spalten SSN, USERID, NAME
-- und ADDRESS zugreifen. Benutzern anderer Rollen wird beim Zugriff auf diese Spalten
-- nur ein ausgeblendeter (maskierter) Wert angezeigt.
-----------------------------------------------------
COLUMN SSN RETURN
 CASE WHEN
  VERIFY_ROLE_FOR_USER(SESSION_USER,'PATIENT') = 1 OR
  VERIFY_ROLE_FOR_USER(SESSION_USER,'PCP') = 1 OR
  VERIFY_ROLE_FOR_USER(SESSION_USER,'MEMBERSHIP') = 1 OR
  VERIFY_ROLE_FOR_USER(SESSION_USER,'ACCOUNTING') = 1
 THEN SSN
  ELSE CHAR('XXX-XX-' || SUBSTR(SSN,8,4)) END
ENABLE;

CREATE MASK PHARMACY_MASK ON PATIENT FOR
-----------------------------------------------------------
-- Medizische Informationen:
-- Benutzer der Rolle PCP dürfen auf alle Informationen in der
-- Spalte PHARMACY zugreifen.
-- Im Rahmen ihrer Arzneimittelforschung dürfen Benutzer der Rolle DRUG_RESEARCH
-- bedingt auf die medizinischen Informationen für einen Patienten zugreifen.
-- Dies setzt jedoch die Zustimmung des Patienten voraus.
```

```
-- In allen anderen Fällen werden als Spaltenwerte Nullwerte
-- zurückgegeben.
-------------------------------------------------
COLUMN PHARMACY RETURN
  CASE WHEN
  VERIFY_ROLE_FOR_USER(SESSION_USER,'PCP') = 1 OR
  (VERIFY_ROLE_FOR_USER(SESSION_USER,'DRUG_RESEARCH')=1
  AND
   EXISTS (SELECT 1 FROM PATIENTCHOICE C
 WHERE PATIENT.SSN = C.SSN AND C.CHOICE = 'drug-research' AND C.VALUE = 'opt-in'))
  THEN PHARMACY
  ELSE NULL
END ENABLE;
```

Alex stellt fest, dass nach dem Erstellen dieser zwei Spaltenmasken die Daten nur für die Benutzer sichtbar sind, für die sie relevant sind. Für die Tabelle PATIENT war die Spaltenzugriffssteuerung bereits aktiviert.

Szenario: ExampleHMO als Benutzer der Zeilen- und Spaltenzugriffssteuerung - Einfügen von Daten

Wenn ein neuer Patient zur Behandlung in das Krankenhaus aufgenommen wird, muss in der ExampleHMO-Datenbank eine neue Krankenakte für den Patienten angelegt werden.

Bob ist ein neuer Patient, und seine Patientendaten müssen der ExampleHMO-Datenbank hinzugefügt werden. Dafür muss ein Benutzer mit der erforderlichen Sicherheitsberechtigung einen neuen Datensatz für Bob erstellen. Tom aus der Abteilung für Mitgliedschaften bei ExampleHMO, der der Rolle MEMBERSHIP zugeordnet ist, registriert Bob als neues Mitglied. Nachdem er eine Verbindung zur ExampleHMO-Datenban hergestellt hat, führt Tom die folgenden SQL-Anweisungen aus, um Bob in die ExampleHMO-Datenbank aufzunehmen:

```
INSERT INTO PATIENT VALUES('123-45-6789', 'BOB', 'Bob', '123 Some St.', 'hypertension',
9.00,'LEE');
INSERT INTO PATIENTCHOICE VALUES('123-45-6789', 'drug-research', 'opt-in');
```

Tom bestätigt, dass Bob der Datenbank hinzugefügt wurde, indem er eine entsprechende Abfrage an die Tabelle PATIENT in der ExampleHMO-Datenbank sendet:

```
Select * FROM PATIENT WHERE NAME = 'Bob';

SSN         USERID      NAME       ADDRESS      PHARMACY     ACCT_BALANCE   PCP_ID
----------- ----------  ---------- ------------ -----------  -------------- -----------
123-45-6789 BOB         Bob        123 Some St. XXXXXXXXXX  0.00           LEE
```

Szenario: ExampleHMO als Benutzer der Zeilen- und Spaltenzugriffssteuerung - Datenaktualisierungen

Während des Krankenhausaufenthalts wird die Behandlung für Bob geändert. Dies erfordert, dass auch die Patienteninformationen in der ExampleHMO-Datenbank entsprechend aktualisiert werden.

Bobs behandelnder Arzt Dr. Lee ordnet eine andere Behandlungsmethode an und verordnet ihm daher andere Medikamente. Bobs Patienteninformationen müssen dahingehend in den ExampleHMO-Systemen geändert werden. Die Regeln für die Zeilenberechtigung in der ExampleHMO-Datenbank geben an, dass jemand, der die Daten in einer Zeile nicht sehen kann, diese auch nicht aktualisieren kann. Da Bobs PCPID die ID von Dr. Lee enthält und die Zeilenberechtigung festgelegt ist, kann Dr. Lee Bobs Informationen mit der folgenden SQL-Anweisungen sowohl anzeigen als auch aktualisieren:

```
UPDATE PATIENT SET PHARMACY = 'codeine' WHERE NAME = 'Bob';
```

Dr. Lee überprüft die Aktualisierung:

```
Select * FROM PATIENT WHERE NAME = 'Bob';

SSN         USERID      NAME       ADDRESS      PHARMACY     ACCT_BALANCE   PCP_ID
----------- ----------  ---------- ------------ -----------  -------------- -----------
123-45-6789 BOB         Bob        123 Some St. codeine      0.00           LEE
```

Dug ist ein Patient, der bei Dr. James, einem Kollegen von Dr. Lee, in Behandlung ist. Dr. Lee versucht, die gleiche Aktualisierung für die Informationen von Dug durchzuführen:

```
UPDATE PATIENT SET PHARMACY = 'codeine' WHERE NAME = 'Dug';
SQL0100W  Es wurde keine Zeile für FETCH, UPDATE oder DELETE gefunden; oder
das Ergebnis der Abfrage ist eine leere Tabelle. SQLSTATE=02000
```

Da die PCPID von Dug nicht die ID von Dr. Lee enthält und die Zeilenberechtigung definiert ist, kann Dr. Lee die Informationen von Dug weder anzeigen noch aktualisieren.

Szenario: ExampleHMO als Benutzer der Zeilen- und Spaltenzugriffssteuerung - Datenabfragen

Im Rahmen der Zeilen- und Spaltenzugriffssteuerung können Personen mit verschiedenen Rollen von denselben Datenbankabfragen unterschiedliche Ergebnisse erhalten. Beispielsweise kann Peter, der Datenbankadministrator mit der Berechtigung DATAACCESS, keine Daten in der Tabelle PATIENT sehen.

Peter, Bob, Dr. Lee, Tom, Jane und John stellen jeweils eine Verbindung zur Datenbank her und geben die folgende SQL-Abfrage ein:

```
SELECT SSN, USERID, NAME, ADDRESS, PHARMACY, ACCT_BALANCE, PCP_ID FROM PATIENT;
```

Je nachdem, wer die Abfrage ausführt, wird ein anderes Ergebnis zurückgegeben. Die von Alex erstellten Regeln für die Zeilen- und Spaltenzugriffssteuerung werden auf diese Abfragen angewendet.

Hier ist das Ergebnis, das Peter sieht:

```
SSN         USERID      NAME        ADDRESS     PHARMACY    ACC_BALANCE PCP_ID

----------- ----------- ----------- ----------- ----------- ----------- -----------

  0 Satz/Sätze ausgewählt.
```

Obwohl sich Daten in der Tabelle befinden und Peter der Datenbankadministrator ist, verfügt er nicht über eine ausreichende Berechtigung, um alle Daten anzusehen.

Hier ist das Ergebnis, das Bob sieht:

```
SSN         USERID      NAME        ADDRESS     PHARMACY    ACC_BALANCE PCP_ID

----------- ----------- ----------- ----------- ----------- ----------- -----------
123-45-6789 BOB         Bob         123 Some St.XXXXXXXXXXX 0.00        LEE

1 Satz/Sätze ausgewählt.
```

Als Patient kann Bob nur seine eigenen Daten sehen. Er gehört der Rolle PATIENT an. Die Daten in den Spalten PHARMACY und ACC_BALANCE sind für ihn ausgeblendet.

Hier ist das Ergebnis, das Dr. Lee sieht:

```
SSN         USERID      NAME        ADDRESS     PHARMACY            ACC_BALANCE PCP_ID

----------- ----------- ----------- ----------- ------------------- ----------- -----------
123-55-1234 MAX         Max         First Strt  hypertension        0.00        LEE
123-11-9856 SAM         Sam         Big Strt    High blood pressure 0.00        LEE
123-45-6789 BOB         Bob         123 Some St.codeine             0.00        LEE

3 Satz/Sätze ausgewählt.
```

Dr. Lee kann nur die Daten der Patienten sehen, die bei ihm in Behandlung sind. Er gehört der Rolle PCP an. Die Daten in der Spalte ACC_BALANCE sind für ihn ausgeblendet.

Hier ist das Ergebnis, das Tom sieht:

```
SSN         USERID      NAME        ADDRESS     PHARMACY    ACC_BALANCE PCP_ID

----------- ----------- ----------- ----------- ----------- ----------- -----------
123-55-1234 MAX         Max         First Strt  XXXXXXXXXX 0.00        LEE
123-58-9812 MIKE        Mike        Long Strt   XXXXXXXXXX 0.00        JAMES
123-11-9856 SAM         Sam         Big Strt    XXXXXXXXXX 0.00        LEE
123-19-1454 DUG         Dug         Good Strt   XXXXXXXXXX 0.00        JAMES
123-45-6789 BOB         Bob         123 Some St.XXXXXXXXXX 0.00        LEE

     5 Satz/Sätze ausgewählt.
```

Tom kann alle Mitglieder sehen. Er gehört der Rolle MEMBERSHIP an. Die Daten in den Spalten PHARMACY und ACC_BALANCE sind für ihn jedoch ausgeblendet.

Hier ist das Ergebnis, das Jane sieht:

```
SSN         USERID      NAME        ADDRESS     PHARMACY            ACC_BALANCE PCP_ID

----------- ----------- ----------- ----------- ------------------- ----------- -----------
XXX-XX-1234 MAX         Max         First Strt  XXXXXXXXXX         0.00        LEE
XXX-XX-9812 MIKE        Mike        Long Strt   XXXXXXXXXX         0.00        JAMES
XXX-XX-9856 SAM         Sam         Big Strt    High blood pressure 0.00        LEE
XXX-XX-1454 DUG         Dug         Good Strt   Influenza           0.00        JAMES
XXX-XX-6789 BOB         Bob         123 Some St.codeine             0.00        LEE

     5 Satz/Sätze ausgewählt.
```

Jane kann alle Mitglieder sehen. Sie gehört der Rolle DRUG_RESEARCH an. Die Daten in den Spalten SSN und ACC_BALANCE sind für sie jedoch ausgeblendet. Die Daten in der Spalte PHARMACY sind nur dann verfügbar, wenn die Patienten zugestimmt haben (Opt-in), ihre Daten an Arzneiforschungsunternehmen weiterzugeben.

Hier ist das Ergebnis, das John sieht:

```
SSN         USERID      NAME        ADDRESS     PHARMACY    ACC_BALANCE PCP_ID

----------- ----------- ----------- ----------- ----------- ----------- -----------
123-55-1234 MAX         Max         First Strt  XXXXXXXXXX 89.70       LEE
123-58-9812 MIKE        Mike        Long Strt   XXXXXXXXXX 8.30        JAMES
123-11-9856 SAM         Sam         Big Strt    XXXXXXXXXX 0.00        LEE
123-19-1454 DUG         Dug         Good Strt   XXXXXXXXXX 0.00        JAMES
123-45-6789 BOB         Bob         123 Some St.XXXXXXXXXX 9.00        LEE

     5 Satz/Sätze ausgewählt.
```

John kann alle Mitglieder sehen. Er gehört der Rolle ACCOUNTING an. Die Daten in der Spalte PHARMACY sind für ihn ausgeblendet.

Szenario: ExampleHMO als Benutzer der Zeilen- und Spaltenzugriffssteuerung - Ansichtserstellung

Ansichten können für Tabellen erstellt werden, für die die Zeilen- und Spaltenzugriffssteuerung definiert ist. Der Sicherheitsadministrator Alex wird gebeten, eine Ansicht der Tabelle PATIENT zu erstellen, die Arzneimittelforscher nutzen können.

Forscher, die eine Partnerschaft mit ExampleHMO haben, verfügen über einen begrenzten Zugriff auf Patientendaten, wenn diese Patienten zugestimmt haben, diesen Zugriff zu erlauben. Alex und das IT-Team sollen eine Ansicht erstellen, in der nur für Forschungszwecke relevante Informationen bezüglich Patienten aufgelistet sind. Der Bericht soll die Versicherungsnummer und den Namen des Patienten sowie die ausgewählte Option zur Offenlegung von Daten des Patienten enthalten.

Die erstellte Ansicht enthält die Basisinformationen zum Patienten sowie die Option zur Offenlegung des Gesundheitszustandes. Sie stellt sicher, dass die Patienteninformationen geschützt sind und nur mit der Genehmigung des jeweiligen Patienten abgerufen werden.

Alex und das IT-Team implementieren die folgende Ansicht:

```
CREATE VIEW PATIENT_INFO_VIEW AS
SELECT P.SSN, P.NAME FROM PATIENT P, PATIENTCHOICE C
WHERE P.SSN = C.SSN AND
   C.CHOICE = 'drug-research' AND
   C.VALUE = 'opt-in';
```

Nachdem Alex und sein Team die Ansicht erstellt haben, können Benutzer diese abfragen. Sie können dabei Daten gemäß den Regeln für die Zeilen- und Spaltenzugriffssteuerung anzeigen, die basierend auf den Basistabellen definiert wurden, für die die Ansicht erstellt wurde.

Alex sieht für die folgende Abfrage der Ansicht das folgende Ergebnis:

```
SELECT SSN, NAME FROM PATIENT_INFO_VIEW;

SSN         NAME
----------- ----------

  0 Satz/Sätze ausgewählt.
```

Dr. Lee sieht für die folgende Abfrage der Ansicht das folgende Ergebnis:

```
SELECT SSN, NAME FROM PATIENT_INFO_VIEW;

SSN         NAME
----------- ----------
123-11-9856 Sam
123-45-6789 Bob

  2 Satz/Sätze ausgewählt.
```

Bob sieht für die folgende Abfrage der Ansicht das folgende Ergebnis:

```
SELECT SSN, NAME FROM PATIENT_INFO_VIEW;

SSN         NAME
```

```
---------- ----------
123-45-6789 Bob

  1 Satz/Sätze ausgewählt.
```

Szenario: ExampleHMO als Benutzer der Zeilen- und Spaltenzugriffssteuerung - Sichere Funktionen

Funktionen müssen als sicher gelten, bevor sie in den Definitionen der Zeilen- und Zugriffssteuerung aufgerufen werden können. Der Sicherheitsadministrator Alex überlegt, wie Paul, ein Datenbankentwickler bei ExampleHMO, für seine neue Buchhaltungsanwendung eine sichere Funktion erstellen kann.

Nach dem Inkrafttreten der Datenschutz- und Sicherheitsrichtlinien bei ExampleHMO wird Alex davon unterrichtet, dass die Buchhaltungsabteilung eine leistungsstarke Buchhaltungsanwendung entwickelt hat. ExampleHMOAccounting ist eine benutzerdefinierte SQL-Skalarfunktion (UDF), die in der Spaltenmaske ACCT_BALANCE_MASK für die Tabelle (und Zeile) PATIENT.ACCT_BALANCE verwendet wird.

Nur sichere UDFs können innerhalb einer Spaltenmaske aufgerufen werden. Alex erörtert die UDF zunächst mit Paul, der die UDF geschrieben hat, um sicherzustellen, dass die Vorgänge innerhalb der UDF sicher sind.

Nachdem sich Alex überzeugt hat, dass die Funktion sicher ist, gewährt er Paul eine Sicherheitsberechtigung, mit der dieser die UDF so bearbeiten kann, dass die UDF sicher ist:

```
GRANT CREATE_SECURE_OBJECT ON DATABASE TO USER PAUL;
```

Um eine gesicherte UDF zu erstellen oder eine UDF so zu ändern, dass sie sicher ist, muss einem Entwickler die Berechtigung CREATE_SECURE_OBJECT erteilt werden.

Paul erstellt die Funktion:

```
CREATE FUNCTION EXAMPLEHMOACCOUNTINGUDF(X DECIMAL(12,2))
       RETURNS DECIMAL(12,2)
       LANGUAGE SQL
       CONTAINS SQL
       DETERMINISTIC
       NO EXTERNAL ACTION
       RETURN X*(1.0 + RAND(X));
```

Paul ändert die Funktion so, dass sie sicher ist:

```
ALTER FUNCTION EXAMPLEHMOACCOUNTINGUDF SECURED;
```

Alex löscht die Maske ACC_BALANCE_MASK und erstellt sie dann neu, sodass die neue UDF verwendet wird:

```
--Löschen der Maske zur Neuerstellung

DROP MASK ACCT_BALANCE_MASK;



CREATE MASK EXAMPLEHMO.ACCT_BALANCE_MASK ONPATIENT FOR
------------------------------------------------------------
-- Buchhaltungsinformationen:
-- Benutzer der Rolle ACCOUNTING dürfen die gesicherte UDF aufrufen.
-- ExampleHMOAccountingUDFL übergibt ACCT_BALANCE als
-- das Eingabeargument.
```

```
-- Benutzern anderer Rollen wird beim Zugriff auf diese Spalte nur ein
-- Nullwert angezeigt.
-------------------------------------------------------------
COLUMN ACCT_BALANCE RETURN
CASE WHEN VERIFY_ROLE_FOR_USER(SESSION_USER,'ACCOUNTING') = 1
THEN EXAMPLEHMOACCOUNTINGUDF(ACCT_BALANCE)
ELSE 0.00
END ENABLE;
```

Dr. Lee, dem die Rolle PCP zugeordnet ist, muss eine benutzerdefinierte Funktion für die Arzneimittelanalyse aufrufen. DrugUDF gibt Informationen zu den Medikamenten von Patienten zurück. Bisher konnte Dr. Lee eine SELECT-Anweisung absetzen, mit der die Funktion DrugUDF aufgerufen und das Ergebnis schnell zurückgegeben wurde. Seitdem die Tabelle PATIENT jedoch durch die Zeilen- und Spaltenzugriffssteuerung geschützt ist, dauert es für dieselbe Abfrage wesentlich länger, um ein Ergebnis zu erhalten.

Dr. Lee wendet sich bezüglich dieser Leistungseinbuße an das IT-Team von ExampleHMO und den Sicherheitsadministrator Alex. Alex erklärt Dr. Lee, dass, wenn die UDF nicht sicher ist, die Abfrage nicht optimiert werden kann und es daher länger dauert, einen Ergebnissatz zurückzugeben.

Alex schaut sich die UDF mit Dr. Lee und Paul, der die UDF geschrieben hat, an, um sicherzustellen, dass die Vorgänge innerhalb der UDF sicher sind. Alex bittet Paul, der immer noch über die Berechtigung CREATE_SECURE_OBJECT verfügt, die ihm Alex erteilt hat, die UDF so zu ändern, dass sie sicher ist:

```
--Funktion für die Arzneimittelabteilung von ExampleHMO

CREATE FUNCTION DRUGUDF(PHARMACY VARCHAR(5000))
      RETURNS VARCHAR(5000)
      NO EXTERNAL ACTION
      BEGIN ATOMIC
       IF PHARMACY IS NULL THEN
          RETURN NULL;
        ELSE
          RETURN 'Normal';
        END IF;
      END;

--Sichern der UDF

ALTER FUNCTION DRUGUDF SECURED;

--Erteilen von Ausführungsberechtigungen für Dr. Lee

GRANT EXECUTE ON FUNCTION DRUGUDF TO USER LEE;
```

Dr. Lee kann die Abfrage absetzen und die Abfrage kann wie erwartet optimiert werden:

```
--Absetzen der Abfrage, nachdem die Funktion gesichert ist

SELECT PHARMACY FROM PATIENT
      WHERE DRUGUDF(PHARMACY) = 'Normal' AND SSN = '123-45-6789';

PHARMACY
--------
codeine

  1 Satz/Sätze ausgewählt.
```

Szenario: ExampleHMO als Benutzer der Zeilen- und Spaltenzugriffssteuerung - Sichere Trigger

Trigger, die für eine Tabelle mit aktivierter Zeilen- oder Spaltenzugriffssteuerung definiert sind, müssen sicher sein. Der Sicherheitsadministrator Alex überlegt, wie Paul, der Datenbankentwickler bei ExampleHMO, für seine neue Buchhaltungsanwendung einen sicheren Trigger erstellen kann.

Alex spricht mit Mitarbeitern der Buchhaltungsabteilung und erfährt, dass für die Tabelle PATIENT ein Trigger des Typs AFTER UPDATE benötigt wird. Mit dem Trigger wird der Verlauf der Spalte ACCT_BALANCE überwacht.

Alex erklärt Paul, der über die nötigen Berechtigungen zum Erstellen des Triggers verfügt, dass jeder Trigger, der für eine durch Zeilen- oder Spaltenzugriff geschützte Tabelle definiert ist, als sicher gekennzeichnet sein muss. Paul und Alex überprüfen das Verhalten des neuen Triggers und stufen ihn als sicher ein.

Mit ExampleHMO_ACCT_BALANCE_TRIGGER wird die Spalte ACCT_BALANCE in der Tabelle PATIENT überwacht. Jedes Mal, wenn diese Spalte aktualisiert wird, wird der Trigger ausgelöst und fügt die aktuellen Kontostandinformationen in die Tabelle ACCT_HISTORY ein.

Paul erstellt den Trigger:

```
CREATE TRIGGER HOSPITAL.NETHMO_ACCT_BALANCE_TRIGGER
  AFTER UPDATE OF ACCT_BALANCE ON PATIENT
  REFERENCING OLD AS O NEW AS N
  FOR EACH ROW MODE DB2SQL SECURED
  BEGIN ATOMIC
  INSERT INTO ACCT_HISTORY
  (SSN, BEFORE_BALANCE, AFTER_BALANCE, WHEN, BY_WHO)
  VALUES(O.SSN, O.ACCT_BALANCE, N.ACCT_BALANCE,
  CURRENT TIMESTAMP, SESSION_USER);
END;
```

John aus der Buchhaltungsabteilung muss den Kontostand für den Patienten Bob aktualisieren, dessen Sozialversicherungsnummer SSN '123-45-6789' lautet.

John sieht sich vor dem Ausführen der Aktualisierung die Daten für Bob an:

```
SELECT ACCT_BALANCE FROM PATIENT WHERE SSN = '123-45-6789';

ACCT_BALANCE
--------------
9.00

  1 Satz/Sätze ausgewählt.

SELECT * FROM ACCT_HISTORY WHERE SSN = '123-45-6789';

SSN         BEFORE_BALANCE AFTER_BALANCE  WHEN       BY_WHO
----------- -------------- -------------- ---------- --------------------

  0 Satz/Sätze ausgewählt.
```

Dann führt John die Aktualisierung aus:

```
UPDATE PATIENT SET ACCT_BALANCE = ACCT_BALANCE * 0.9 WHERE SSN = '123-45-6789';
```

Da für die Tabelle PATIENT ein Trigger definiert ist, löst die Aktualisierung den Trigger aus. Da der Trigger als SECURED definiert ist, wird die Aktualisierung erfolgreich abgeschlossen. John sieht sich nach dem Ausführen der Aktualisierung die Daten für Bob an:

```
SELECT ACCT_BALANCE FROM PATIENT WHERE SSN = '123-45-6789';

ACCT_BALANCE
--------------
8.10

  1 Satz/Sätze ausgewählt.

SELECT * FROM ACCT_HISTORY WHERE SSN = '123-45-6789';

SSN         BEFORE_BALANCE AFTER_BALANCE  WHEN       BY_WHO
----------- -------------- -------------- ---------- --------------------
123-45-6789 9.00           8.10           2010-10-10 JOHN
  1 Satz/Sätze ausgewählt.
```

Szenario: ExampleHMO als Benutzer der Zeilen- und Spaltenzugriffssteuerung - Entziehen von Berechtigungen

Alex entscheidet als Sicherheitsadministrator darüber, wer sichere Objekte erstellen kann. Wenn ein Entwickler mit der Erstellung sicherer Objekte fertig ist, entzieht Alex ihm die entsprechende Berechtigung für die Datenbank.

Der Datenbankentwickler Paul hat die Entwicklung verschiedener Objekte abgeschlossen. Alex entzieht ihm daraufhin sofort die Berechtigung zum Erstellen von sicheren Objekten:

```
REVOKE CREATE_SECURE_OBJECT ON DATABASE FROM USER PAUL;
```

Wenn Paul zu einem späteren Zeitpunkt weitere sichere Objekte erstellen muss, muss er sich an Alex wenden, damit ihm die entsprechende Berechtigung erneut erteilt wird.

Szenario: ExampleBANK als Benutzer der Zeilen- und Spaltenzugriffssteuerung

In diesem Szenario präsentiert sich ExampleBANK, ein Kreditinstitut mit einem breitgefächerten Kundenstamm, als Benutzer der Zeilen- und Spaltenzugriffssteuerung. ExampleBANK verwendet diese Art der Zugriffssteuerung, um sicherzustellen, dass die Datenbankrichtlinien des Unternehmens den internen Bestimmungen für Datenschutz und Sicherheit sowie den Unternehmenszielen entsprechen.

Unternehmen wie ExampleBANK, die Investitionen, Spareinlagen und persönlichen Daten von Kunden verwalten, stellen diese Informationen innerhalb des Unternehmens nur den Personen bereit, die ein berechtigtes Interessen daran haben. Diese Form des Datenschutzes sorgt dafür, dass sämtliche vertraulichen Finanzdaten bzw. persönlichen Informationen nur von Mitarbeitern angezeigt, gemeinsam genutzt und geändert werden können, die über die entsprechenden Berechtigungen dafür verfügen.

Szenario: ExampleBANK als Benutzer der Zeilen- und Spaltenzugriffssteuerung - Sicherheitsrichtlinien

ExampleBANK implementiert eine Sicherheitsstrategie, bei der der Datenzugriff auf DB2-Datenbanken basierend auf bestimmten Sicherheitsrichtlinien gewährt wird.

Die Sicherheitsrichtlinien entsprechen den Sicherheits- und Datenschutzbestimmungen von ExampleBANK. In der ersten Spalte sind die Richtlinien und die Herausforderungen aufgelistet, denen sich ExampleBANK gegenübersieht, die zweite Spalte enthält die Sicherheitsfunktion der Zeilen- und Spaltenzugriffssteuerung (RCAC) von DB2 für die entsprechende Herausforderung.

Tabelle 9.

Herausforderung	**Sicherheitsfunktion der Zeilen- und Spaltenzugriffssteuerung für diese Herausforderung**
Zeilenzugriff auf berechtigte Benutzer beschränken. Bankangestellte dürfen nur die Kundendaten ihrer eigenen Filiale einsehen, nicht jedoch die Daten aller Kunden von ExampleBANK, die im unternehmensweiten System gespeichert sind.	Es können Zeilenberechtigungen implementiert werden, mit denen gesteuert werden kann, welcher Benutzer eine bestimmte Zeile anzeigen darf.
Die Kontonummer ist nur dann für Kundendienstmitarbeiter zugänglich, wenn die Mitarbeiter die Anwendung zur Aktualisierung des Kontos verwenden. Diese Anwendung wird durch die gespeicherte Prozedur ACCOUNTS.ACCTUPDATE angegeben.	Mithilfe von Spaltenmasken können vertrauliche Daten für Kundendienstmitarbeiter gefiltert bzw. ausgeblendet werden, wenn die Mitarbeiter die Daten außerhalb der Anwendung ACCOUNTS.ACCTUPDATE abfragen.

Szenario: ExampleBANK als Benutzer der Zeilen- und Spaltenzugriffssteuerung - Datenbankbenutzer und -rollen

In diesem Beispiel verwenden verschiedene Personen ExampleBANK-Daten. Diese Personen verfügen über unterschiedliche Benutzerrechte.

Das Unternehmen ExampleBANK hat seine Sicherheitsstrategie so implementiert, dass die Daten nach Art des Zugriffs über die Datenbank klassifiziert werden können. Der interne Datenzugriff basiert auf der Trennung der Aufgabenbereiche der Benutzer, die auf die Daten zugreifen, und deren Datenzugriffsberechtigungen. ExampleBANK hat die folgenden Datenbankrollen zur Trennung dieser Aufgaben erstellt:

TELLER
Für Bankangestellte der einzelnen Filialen.

TELEMARKERT
Für Telefonmarketing- und Vertriebsmitarbeiter.

CSR
Für Kundendienstmitarbeiter.

Die folgenden Personen verwenden ExampleBANK-Daten:

ZURBIE
Eine Kundendienstmitarbeiterin von ExampleBANK. Sie gehört der Rolle CSR an.

NEWTON
Ein Bankangestellter in einer ExampleBANK-Filiale. Er gehört der Rolle TELLER an.

PLATO
Ein Telefonmarketing- und Vertriebsmitarbeiter bei ExampleBANK. Er gehört der Rolle TELEMARKETER an.

Wenn Sie die SQL-Beispielanweisungen und -befehle in diesem Szenario anwenden möchten, müssen Sie diese Benutzer-IDs mit ihren aufgelisteten Berechtigungen erstellen.

Für die folgenden SQL-Beispielanweisungen wird angenommen, dass die Benutzer im System erstellt wurden. Mit den SQL-Anweisungen werden die einzelnen Rollen erstellt und den Benutzern die Berechtigung **SELECT** für die verschiedenen Tabellen in der ExampleBANK-Datenbank erteilt:

```
--Erstellen von Rollen und Erteilen von Berechtigungen

CREATE ROLE TELLER;

CREATE ROLE CSR;

CREATE ROLE TELEMARKERTER;

GRANT ROLE TELLER TO USER NEWTON;
GRANT ROLE CSR TO USER ZURBIE;
GRANT ROLE TELEMARKETER TO USER PLATO;
```

Szenario: ExampleBANK als Benutzer der Zeilen- und Spaltenzugriffssteuerung - Datenbanktabellen

Bei diesem Szenario liegt der Fokus auf zwei Tabellen in der ExampleBANK-Datenbank: auf der Tabelle CUSTOMER und der Tabelle INTERNAL_INFO.

In der Tabelle INTERNAL_INFO sind Informationen zu Mitarbeitern von ExampleBANK gespeichert. In diesem Szenario werden die folgenden Spalten in der Tabelle INTERNAL_INFO betrachtet:

HOME_BRANCH
Die ID der Filiale, in der ein Mitarbeiter arbeitet.

EMP_ID
Die Mitarbeiter-ID.

In der Tabelle CUSTOMER sind verschiedene Kundeninformationen gespeichert:

ACCOUNT
Die Kontonummer des Kunden.

NAME
Der Name des Kunden.

INCOME
Das Einkommen des Kunden.

BRANCH
Die ID der Kundenfiliale.

Mit den folgenden SQL-Anweisungen werden die Tabellen CUSTOMER und INTERNAL_INFO erstellt. Es werden Berechtigungen für die Tabelle gewährt und Daten eingefügt:

```
--Tabelle CUSTOMER mit Informationen zu den einzelnen Kunden
CREATE TABLE RCACTSPM.CUSTOMER (
   ACCOUNT VARCHAR(19),
   NAME VARCHAR(20),
   INCOME INTEGER,
   BRANCH CHAR(1)
);

--Tabelle INTERNAL_INFO mit Mitarbeiterinformationen
```

```
CREATE TABLE RCACTSPM.INTERNAL_INFO (
   HOME_BRANCH CHAR(1),
   EMP_ID VARCHAR(10));

--Erteilen von Berechtigungen

GRANT SELECT ON RCACTSPM.CUSTOMER TO USER NEWTON, USER ZURBIE, USER PLATO;

--Einfügen von Daten

INSERT INTO RCACTSPM.CUSTOMER VALUES ('1111-2222-3333-4444', 'Alice', 22000, 'A');
INSERT INTO RCACTSPM.CUSTOMER VALUES ('2222-3333-4444-5555', 'Bob', 71000, 'A');
INSERT INTO RCACTSPM.CUSTOMER VALUES ('3333-4444-5555-6666', 'Carl', 123000, 'B');
INSERT INTO RCACTSPM.CUSTOMER VALUES ('4444-5555-6666-7777', 'David', 172000, 'C');

INSERT INTO RCACTSPM.INTERNAL_INFO VALUES ('A', 'NEWTON');
INSERT INTO RCACTSPM.INTERNAL_INFO VALUES ('B', 'ZURBIE');
INSERT INTO RCACTSPM.INTERNAL_INFO VALUES ('C', 'PLATO');
```

Szenario: ExampleBANK als Benutzer der Zeilen- und Spaltenzugriffssteuerung - Zeilenberechtigungen

Der Sicherheitsadministrator von ExampleBANK beginnt damit, den Datenzugriff zu beschränken, indem er als Teil der Zeilen- und Spaltenzugriffssteuerung Zeilenberechtigungen verwendet. Mit einer Zeilenberechtigung werden die an Benutzer zurückgegebenen Daten nach Zeilen gefiltert.

Bankangestellte dürfen nur Kundendaten für ihre eigene Filiale einsehen. Telemarketing- und Kundendienstmitarbeiter können alle ExampleBANK-Kunden im System anzeigen. Telemarketingmitarbeiter können jedoch dabei nicht die vollständige Kontonummer sehen.

Zeilenberechtigungen beschränken bzw. filtern Zeilen basierend auf dem an der Datenbank angemeldeten Benutzer. Im Beispielunternehmen ExampleBANK erzeugen die Zeilenberechtigungen eine horizontale Dateneinschränkung für die Tabelle CUSTOMER.

Der Sicherheitsadministrator implementiert die folgenden Zeilenberechtigungen so, dass die Benutzer in den verschiedenen Rollen jeweils nur das Ergebnis sehen können, das ihnen die zugewiesene Berechtigung erlaubt:

```
CREATE PERMISSION TELLER_ROW_ACCESS ON RCACTSPM.CUSTOMER
-------------------------------------------------------
-- Informationen zu den Bankangestellten:
-- Benutzer der Rolle TELLER dürfen nur auf Kundendaten in ihrer
-- eigenen Filiale zugreifen.
-------------------------------------------------------------
FOR ROWS WHERE VERIFY_ROLE_FOR_USER(USER, 'TELLER') = 1
AND
BRANCH = (SELECT HOME_BRANCH FROM RCACTSPM.INTERNAL_INFO WHERE EMP_ID = USER)
ENFORCED FOR ALL ACCESS
ENABLE;

CREATE PERMISSION CSR_ROW_ACCESS ON RCACTSPM.CUSTOMER
-----------------------------------------------------
-- Informationen zu Kundendienst- und Telemarketingmitarbeitern:
-- Benutzer der Rollen TELEMARKETER und CSR dürfen auf alle Zeilen mit Kundendaten
-- im ExampleBANK-System zugreifen.
-------------------------------------------------------------
FOR ROWS WHERE VERIFY_ROLE_FOR_USER (USER, 'CSR') = 1
```

```
OR
VERIFY_ROLE_FOR_USER (USER, 'TELEMARKETER') = 1
ENFORCED FOR ALL ACCESS
ENABLE;
```

Der Sicherheitsadministrator beobachtet jedoch, dass selbst nach der Erstellung einer Zeilenberechtigung die anderen Mitarbeiter immer noch alle Daten aufrufen können. Eine Zeilenberechtigung wird erst dann angewendet, wenn sie für die Tabelle aktiviert wird, für die sie definiert wurde. Daher muss der Sicherheitsadministrator nun die Berechtigung aktivieren:

```
--Aktivieren der Zeilenzugriffssteuerung zum Implementieren von Zeilenberechtigungen

ALTER TABLE RCACTSPM.CUSTOMER ACTIVATE ROW ACCESS CONTROL;
```

Szenario: ExampleBANK als Benutzer der Zeilen- und Spaltenzugriffssteuerung - Spaltenmasken

Der Sicherheitsadministrator von ExampleBANK beschränkt den Datenzugriff noch weiter, indem er als Teil der Zeilen- und Spaltenzugriffssteuerung Spaltenmasken verwendet. Spaltenmasken blenden Daten für Benutzer oder Anwendungen nach Spalten aus, es sei den, die Benutzer oder Anwendungen sind berechtigt, die entsprechenden Daten anzuzeigen.

Kundendienstmitarbeiter können zwar alle Kunden im ExampleBANK-System sehen, die jeweilige vollständige Kontonummer wird jedoch nur bei Verwendung einer bestimmten Anwendung angezeigt.

Der Sicherheitsadministrator implementiert die folgende Spaltenmaske so, dass ein Kundendienstmitarbeiter immer nur das Ergebnis sehen kann, das ihm die zugewiesene Berechtigung erlaubt:

```
CREATE MASK ACCOUNT_COL_MASK ON RCACTSPM.CUSTOMER FOR
-------------------------------------------------------------
-- Informationen zur Kontonummer:
-- Benutzer der Rolle CSR dürfen nur dann auf
-- Kontonummerinformationen zugreifen, wenn sie die Anwendung
-- zur Kontoaktualisierung verwenden. Diese Anwendung wird
-- durch die gespeicherte Prozedur ACCOUNTS.ACCTUPDATE angegeben.
-- Wenn ein Benutzer der Rolle CSR die Daten außerhalb dieser Anwendung abfragt, werden
-- die Kontoinformationen ausgeblendet (maskiert), und die ersten 12 Ziffern werden
-- durch "x" ersetzt.
-------------------------------------------------------------
COLUMN ACCOUNT RETURN
  CASE WHEN (VERIFY_ROLE_FOR_USER (USER, 'CSR') = 1 AND
             ROUTINE_SPECIFIC_NAME = 'ACCTUPDATE' AND
             ROUTINE_SCHEMA = 'ACCOUNTS' AND
             ROUTINE_TYPE = 'P')
       THEN ACCOUNT
       ELSE 'xxxx-xxxx-xxxx-' || SUBSTR(ACCOUNT,16,4)
  END ENABLE;
```

Der Sicherheitsadministrator beobachtet jedoch, dass selbst nach der Erstellung einer Spaltenmaske die Daten immer noch von allen Mitarbeitern abgerufen werden können. Eine Spaltenmaske wird erst dann angewendet, wenn sie für die Tabelle aktiviert wird, für die sie definiert wurde. Daher muss der Sicherheitsadministrator nun die Maske aktivieren:

```
--Aktivieren der Spaltenzugriffssteuerung zum Implementieren von Spaltenmasken

ALTER TABLE RCACTSPM.CUSTOMER ACTIVATE COLUMN ACCESS CONTROL;
```

Szenario: ExampleBANK als Benutzer der Zeilen- und Spaltenzugriffssteuerung - Datenabfragen

Im Rahmen der Zeilen- und Spaltenzugriffssteuerung können Personen mit verschiedenen Rollen von denselben Datenbankabfragen unterschiedliche Ergebnisse erhalten. Beispielsweise kann der Bankangestellte Newton keine Daten von Kunden außerhalb der entsprechenden Kundenfiliale sehen.

Newton, Zurbie und Plato stellen jeweils eine Verbindung zur Datenbank her und setzen die folgende SQL-Abfrage ab:

```
SELECT * FROM RCACTSPM.CUSTOMER;
```

Je nachdem, wer die Abfrage ausführt, wird ein anderes Ergebnis zurückgegeben. Die vom Sicherheitsadministrator erstellten Regeln für die Zeilen- und Spaltenzugriffssteuerung werden auf diese Abfragen angewendet.

Hier ist das Ergebnis, das Newton sieht:

```
ACCOUNT             NAME                 INCOME      BRANCH
------------------- -------------------- ----------- ------
1111-2222-3333-4444 Alice                     22000 A
2222-3333-4444-5555 Bob                       71000 A

  2 Satz/Sätze ausgewählt.
```

Newton als Bankangestellter in der Filiale A kann nur ExampleBANK-Kunden sehen, die dieser Filiale angehören.

Hier ist das Ergebnis, das Zurbie sieht:

```
ACCOUNT             NAME                 INCOME      BRANCH
------------------- -------------------- ----------- ------
xxxx-xxxx-xxxx-4444 Alice                     22000 A
xxxx-xxxx-xxxx-5555 Bob                       71000 A
xxxx-xxxx-xxxx-6666 Carl                     123000 B
xxxx-xxxx-xxxx-7777 David                    172000 C

       4 Satz/Sätze ausgewählt.
```

Als Kundendienstmitarbeiterin kann Zurbie alle ExampleBANK-Kunden im System sehen. Wenn sie jedoch die vollständige Kontonummer anzeigen möchte, muss sie die Anwendung ACCOUNTS.ACCTUPDATE verwenden. Da ihre Anfrage jedoch außerhalb von ACCOUNTS.ACCTUPDATE abgesetzt wurde, ist ein Teil der jeweiligen Kontonummern ausgeblendet.

Hier ist das Ergebnis, das Plato sieht:

```
ACCOUNT             NAME                 INCOME      BRANCH
------------------- -------------------- ----------- ------
1111-2222-3333-4444 Alice                     22000 A
2222-3333-4444-5555 Bob                       71000 A
3333-4444-5555-6666 Carl                     123000 B
4444-5555-6666-7777 David                    172000 C

       4 Satz/Sätze ausgewählt.
```

Als Telemarketingmitarbeiter kann Plato alle ExampleBANK-Kunden im System sehen.

Kapitel 5. Kennsatzbasierte Zugriffssteuerung (LBAC)

LBAC (Label-Based Access Control, kennsatzbasierte Zugriffssteuerung) bietet wesentlich bessere Kontrollmöglichkeiten für die Steuerung des Zugriffs auf Ihre Daten. Mit LBAC können Sie exakt festlegen, wer Schreibzugriff und wer Lesezugriff auf einzelne Zeilen und einzelne Spalten erhält.

Funktionsweise von LBAC

Die LBAC-Funktionalität ist umfassend konfigurierbar und lässt sich auf Ihre individuelle Sicherheitsumgebung abstimmen. Alle LBAC-Konfigurationsaktivitäten werden durch einen *Sicherheitsadministrator* ausgeführt. d. h. durch den Benutzer, dem die Berechtigung SECADM erteilt wurde.

Ein Sicherheitsadministrator konfiguriert das LBAC-System, indem er Sicherheitskennsatzkomponenten erstellt. Eine *Sicherheitskennsatzkomponente* ist ein Datenbankobjekt, das eine Bedingung darstellt, mit dem Sie feststellen wollen, ob ein Benutzer auf bestimmte Daten zugreifen darf. Die Bedingung kann z. B. sein, dass der Benutzer einer bestimmten Abteilung angehört oder dass er an einem bestimmten Projekt mitarbeitet. Eine *Sicherheitsrichtlinie* (Security Policy) beschreibt die Bedingungen, die zur Entscheidung herangezogen werden, wer auf welche Daten Zugriff besitzt. Eine Sicherheitsrichtlinie enthält mindestens eine Sicherheitskennsatzkomponente. Eine Tabelle kann nur durch eine Sicherheitsrichtlinie geschützt werden, jedoch können verschiedene Tabellen durch verschiedene Sicherheitsrichtlinien geschützt werden.

Nach der Erstellung einer Sicherheitsrichtlinie erstellt ein Sicherheitsadministrator Objekte, die als *Sicherheitskennsätze* (Security Labels) bezeichnet werden und die Teil der jeweiligen Richtlinie sind. Sicherheitskennsätze enthalten Sicherheitskennsatzkomponenten. Die exakten Bestandteile eines Sicherheitskennsatzes werden durch die Sicherheitsrichtlinie bestimmt und lassen sich so konfigurieren, dass sie die Bedingungen darstellen, auf die sich in Ihrem Unternehmen die Entscheidung gründet, wer auf welche bestimmten Datenelemente Zugriff haben soll. Wenn Sie zum Beispiel festlegen, dass die Entscheidung, welche Daten für eine Person zugänglich sein sollen, von ihrer Position im Unternehmen sowie von den Projekten, an denen sie beteiligt ist, abhängen soll, können Sie Ihre Sicherheitskennsätze so konfigurieren, dass jeder Kennsatz diese Informationen beinhaltet. LBAC bietet die Flexibilität, alle Arten von Kriterien zu konfigurieren, welche die gesamte Bandbreite von sehr komplexen bis hin zu sehr einfachen Systemen, bei denen jeder Kennsatz nur entweder einen "hohen" oder einen "niedrigen" Vertraulichkeitsgrad darstellt, abdecken.

Nach der Erstellung kann ein Sicherheitskennsatz einzelnen Spalten und Zeilen in einer Tabelle zugeordnet werden, um die dort gespeicherten Daten zu schützen. Daten, die durch einen Sicherheitskennsatz geschützt sind, werden als *geschützte Daten* bezeichnet. Ein Sicherheitsadministrator berechtigt Benutzer zum Zugriff auf geschützte Daten, indem er ihnen Sicherheitskennsätze erteilt. Wenn ein Benutzer versucht, auf geschützte Daten zuzugreifen, wird sein Sicherheitskennsatz mit dem Sicherheitskennsatz verglichen, der die Daten schützt. Der schützende Kennsatz blockiert einige Sicherheitskennsätze und andere nicht.

Ein Benutzer, eine Rolle oder eine Gruppe kann Sicherheitskennsätze für mehrere Sicherheitsrichtlinien besitzen. Für eine bestimmte Sicherheitsrichtlinie kann ein Be-

nutzer, eine Rolle oder eine Gruppe jedoch höchstens einen Kennsatz für den Lesezugriff und einen Kennsatz für den Schreibzugriff besitzen.

Ein Sicherheitsadministrator kann Benutzern außerdem Freistellungen erteilen. Eine *Freistellung* (engl. exemption) erlaubt einem Benutzer den Zugriff auf geschützte Daten, der ihm ansonsten durch seine Sicherheitskennsätze verwehrt wäre. Sicherheitskennsätze und Freistellungen werden zusammengenommen als *LBAC-Berechtigungsnachweise* (engl. LBAC credentials) eines Benutzers bezeichnet.

Wenn Sie versuchen, auf eine geschützte Spalte zuzugreifen, auf die der Zugriff durch Ihre LBAC-Berechtigungsnachweise nicht zugelassen wird, schlägt der Zugriffsversuch fehl und Sie empfangen eine Fehlernachricht.

Wenn Sie versuchen, geschützte Zeilen zu lesen, auf die ihre LBAC-Berechtigungsnachweise keinen Lesezugriff zulassen, reagiert DB2 so, als ob diese Zeilen nicht vorhanden wären. Diese Zeilen können durch keine von Ihnen ausgeführte SQL-Anweisung (d. h. Anweisungen SELECT, UPDATE oder DELETE) ausgewählt werden. Auch Spaltenfunktion ignorieren Zeilen, auf die Ihre LBAC-Berechtigungsnachweise keinen Lesezugriff zulassen. Die Funktion COUNT(*) gibt zum Beispiel nur die Anzahl der Zeilen zurück, auf die Sie Lesezugriff haben.

Sichten und LBAC

Sie können eine Sicht für eine geschützte Tabelle in der gleichen Weise definieren wie für eine nicht geschützte Tabelle. Beim Zugriff auf eine solche Sicht wird der LBAC-Schutz für die zugrunde liegende Tabelle wirksam. Dazu werden die LBAC-Berechtigungsnachweise der jeweiligen Sitzungsberechtigungs-ID verwendet. Zwei Benutzern, die auf die gleiche Sicht zugreifen, werden abhängig von ihren LBAC-Berechtigungsnachweisen möglicherweise verschiedene Zeilen angezeigt.

Referenzielle Integritätsbedingungen und LBAC

Die folgenden Regeln erläutern, wie LBAC-Regeln im Hinblick auf vorhandene referenzielle Integritätsbedingungen umgesetzt werden:

- **Regel 1:** Die LBAC-Regeln für den Lesezugriff werden auf intern generierte Suchen in untergeordneten Tabellen NICHT angewendet. Dadurch soll verhindert werden, dass auf untergeordnete Tabellen nicht mehr zugegriffen werden kann (verwaiste Tabelle).
- **Regel 2:** Die Regeln für den Lesezugriff werden auf intern generierte Suchen in übergeordneten Tabellen NICHT angewendet.
- **Regeln 3:** Die LBAC-Regeln für den Schreibzugriff werden angewendet, wenn eine kaskadierende Operation an untergeordneten Tabellen ausgeführt wird. Wenn ein Benutzer zum Beispiel ein übergeordnetes Element löscht, jedoch keines der untergeordneten Elemente wegen Verstoßes gegen die LBAC-Schreibzugriffsregel löschen kann, sollte die Löschung rückgängig gemacht und ein Fehler zurückgegeben werden.

Speichersystemaufwand bei der Verwendung von LBAC

Wenn Sie LBAC zum Schutz einer Tabelle auf Zeilenebene verwenden, fällt zusätzlicher Speicheraufwand in Form von Kosten für die Spalte der Zeilensicherheitskennsätze an. Diese Kosten sind vom Typ des ausgewählten Sicherheitskennsatzes abhängig. Wenn Sie zum Beispiel eine Sicherheitsrichtlinie mit zwei Komponenten zum Schutz einer Tabelle erstellen, belegt ein Sicherheitskennsatz aus dieser Sicherheitsrichtlinie 16 Byte (8 Byte für jede Komponente). Da die Spalte für die Zeilensi-

cherheitskennsätze als VARCHAR-Spalte, die keine Nullwerte enthalten darf (NOT NULLABLE) behandelt wird, beträgt der Gesamtaufwand in diesem Fall 20 Byte pro Zeile. Im Allgemeinen beträgt der Gesamtaufwand pro Zeile (*N**8 + 4) Byte. Datei ist *N* die Anzahl der Komponenten in der Sicherheitsrichtlinie, die die Tabelle schützt.

Wenn Sie LBAC zum Schutz einer Tabelle auf Spaltenebene verwenden, gehört der Spaltensicherheitskennsatz zu den Metadaten (d. h. er wird zusammen mit den Metadaten der Spalte in der Katalogtabelle SYSCOLUMNS gespeichert). Diese Metadaten bestehen einfach aus der ID des Sicherheitskennsatzes, der die Spalte schützt. In diesem Fall fällt für die Benutzertabelle kein zusätzlicher Speicheraufwand an.

Von LBAC nicht ausgeführte Aktionen

- LBAC wird nie einen Zugriff auf Daten zulassen, der durch eine eignerdefinierte Zugriffssteuerung verboten ist.

  Beispiel: Wenn Sie keine Berechtigung zum Lesen einer Tabelle haben, wird Ihnen das Lesen von Daten aus dieser Tabelle, selbst für Zeilen und Spalten, auf die Ihnen LBAC ansonsten Zugriff gewähren würde, nicht erlaubt.
- Ihre LBAC-Berechtigungsnachweise schränken nur Ihren Zugriff auf geschützte Daten ein. Sie wirken sich nicht auf Ihren Zugriff auf ungeschützte Daten aus.
- LBAC-Berechtigungsnachweise werden nicht überprüft, wenn Sie eine Tabelle oder eine Datenbank löschen, selbst wenn die Tabelle bzw. die Datenbank geschützte Daten enthält.
- LBAC-Berechtigungsnachweise werden nicht überprüft, wenn Sie ein Backup Ihrer Daten erstellen. Wenn Sie ein Backup für eine Tabelle ausführen können, unterliegen die durch das Backup gesicherten Zeilen keinerlei Einschränkung durch den LBAC-Schutz der Daten. Darüber hinaus ist zu beachten, dass die Daten auf dem Backupdatenträger nicht durch LBAC geschützt sind. Nur Daten in der Datenbank werden geschützt.
- LBAC kann nicht zum Schutz der folgenden Typen von Tabellen verwendet werden:
  - Eine Zwischenspeichertabelle
  - Eine Tabelle, von der eine Zwischenspeichertabelle abhängig ist
  - Eine typisierte Tabelle
- Der LBAC-Zugriffsschutz kann nicht auf einen Kurznamen angewendet werden.

LBAC-Lernprogramm

Ein Lernprogramm zu den LBAC-Grundlagen mit dem Titel DB2 Label-Based Access Control, a practical guide ist online unter der folgenden Adresse verfügbar: http://www.ibm.com/developerworks/data

LBAC-Sicherheitsrichtlinien

Der Sicherheitsadministrator definiert in einer Sicherheitsrichtlinie die Bedingungen, die bestimmen, wer Schreibzugriff und wer Lesezugriff auf einzelne Zeilen und einzelne Spalten von Tabellen erhält.

Eine Sicherheitsrichtlinie enthält folgende Informationen:

- Welche Sicherheitskennsatzkomponenten in den Sicherheitskennsätzen verwendet werden, die Teil der Richtlinie sind.

- Welche Regeln beim Vergleichen dieser Sicherheitskennsatzkomponenten verwendet werden.
- Welche von bestimmten optionalen Verhaltensweisen beim Zugriff auf Daten angewendet werden, die durch die Richtlinie geschützt werden.
- Welche zusätzlichen Sicherheitskennsätze und Freistellungen zu prüfen sind, wenn der Zugriff auf Daten, die durch die Sicherheitsrichtlinie geschützt werden, umgesetzt wird. Zum Beispiel wird die Option, ob die Sicherheitskennsätze zu prüfen sind, die Rollen und Gruppen erteilt wurden, durch die Sicherheitsrichtlinie gesteuert.

Jeder geschützten Tabelle darf eine und nur eine Sicherheitsrichtlinie zugeordnet werden. Zeilen und Spalten in einer solchen Tabelle können nur mit Sicherheitskennsätzen geschützt werden, die Teil dieser Sicherheitsrichtlinie sind. Jeglicher Zugriff auf geschützte Daten unterliegt den Regeln dieser Richtlinie. Sie können in einer Datenbank mehrere Sicherheitsrichtlinien haben, jedoch kann eine bestimmte Tabelle jeweils nur durch eine einzige Sicherheitsrichtlinie geschützt werden.

Erstellen einer Sicherheitsrichtlinie

Zur Erstellung einer Sicherheitsrichtlinie müssen Sie über die Berechtigung eines Sicherheitsadministrators verfügen. Eine Sicherheitsrichtlinie wird mithilfe der SQL-Anweisung CREATE SECURITY POLICY erstellt. Die in einer Sicherheitsrichtlinie aufgeführten Sicherheitskennsatzkomponenten müssen erstellt werden, bevor die Anweisung CREATE SECURITY POLICY ausgeführt wird. Die Reihenfolge, in der die Komponenten bei der Erstellung einer Sicherheitsrichtlinie aufgelistet werden, definiert keinerlei Art von Vorrangstellung oder irgendeine andere Beziehung zwischen den Komponenten. Es ist jedoch wichtig, diese Reihenfolge zu kennen, wenn Sicherheitskennsätze mit integrierten Funktionen wie SECLABEL erstellt werden.

Sie können von der Sicherheitsrichtlinie, die Sie erstellt haben, Sicherheitskennsätze erstellen, um Ihre Daten zu schützen.

Ändern einer Sicherheitsrichtlinie

Ein Sicherheitsadministrator kann eine Sicherheitsrichtlinie mithilfe der Anweisung ALTER SECURITY POLICY ändern.

Löschen einer Sicherheitsrichtlinie

Zum Löschen einer Sicherheitsrichtlinie müssen Sie über die Berechtigung eines Sicherheitsadministrators verfügen. Eine Sicherheitsrichtlinie wird mithilfe der SQL-Anweisung DROP gelöscht.

Sie können eine Sicherheitsrichtlinie nicht löschen, wenn Sie einer Tabelle zugeordnet (bzw. hinzugefügt) ist.

Komponenten von LBAC-Sicherheitskennsätzen - Übersicht

Eine *Sicherheitskennsatzkomponente* ist ein Datenbankobjekt, das Teil der LBAC-Funktionalität (Label-Based Access Control, kennsatzbasierte Zugriffssteuerung) ist. Mithilfe von Sicherheitskennsatzkomponenten können Sie die Sicherheitsstruktur Ihres Unternehmens modellieren.

Ein Sicherheitskennsatzkomponente kann ein beliebiges Kriterium darstellen, dass Sie möglicherweise zur Entscheidung heranziehen möchten, ob ein Benutzer Zugriff auf eine bestimmte Dateninformation haben soll. Typische Beispiele für solche Kriterien sind u. a.:

- Wie vertrauenswürdig der Benutzer ist.
- In welcher Abteilung der Benutzer tätig ist.
- Ob der Benutzer an einem bestimmten Projekt beteiligt ist.

Beispiel: Wenn Sie wollen, dass die Abteilung, der ein Benutzer angehört, Auswirkung auf die Daten hat, auf die der Benutzer zugreifen kann, können Sie eine Komponente mit dem Namen 'dept' erstellen und Elemente für diese Komponente definieren, die die verschiedenen Abteilungen in Ihrem Unternehmen benennen. Anschließend können Sie die Komponente 'dept' in Ihre Sicherheitsrichtlinie aufnehmen.

Ein *Element* einer Sicherheitskennsatzkomponente ist eine bestimmte "Einstellung", die für diese Komponente zulässig ist.

Beispiel: Eine Sicherheitskennsatzkomponente, die eine Vertraulichkeitsstufe darstellt, könnte zum Beispiel die folgenden vier Elemente haben: 'Top Secret' (Streng geheim), 'Secret' (Geheim), 'Classified' (Vertraulich) und 'Unclassified' (Nicht eingestuft).

Erstellen einer Sicherheitskennsatzkomponente

Zur Erstellung einer Sicherheitskennsatzkomponente müssen Sie über die Berechtigung eines Sicherheitsadministrators verfügen. Sicherheitskennsatzkomponenten werden mithilfe der SQL-Anweisung CREATE SECURITY LABEL COMPONENT erstellt.

Bei der Erstellung einer Sicherheitskennsatzkomponente müssen Sie folgende Informationen angeben:

- Einen Namen für die Komponente
- Den Typ der Komponente (ARRAY, TREE oder SET)
- Eine vollständige Liste aller zulässigen Elemente
- Für die Typen ARRAY und TREE: Eine Beschreibung, wie sich jedes einzelne Element in die Struktur der Komponente einfügt

Nachdem Sie Ihre Sicherheitskennsatzkomponenten erstellt haben, können Sie eine Sicherheitsrichtlinie auf der Basis dieser Komponenten erstellen. Sie können von dieser Sicherheitsrichtlinie Sicherheitskennsätze erstellen, um Ihre Daten zu schützen.

Typen von Komponenten

Es gibt drei Typen von Sicherheitskennsatzkomponenten:

- TREE: Jedes Element stellt einen Knoten in einer Baumstruktur dar.
- ARRAY: Jedes Element stellt einen Punkt auf einer linearen Skala dar.
- SET: Jedes Element stellt ein Element einer Menge dar.

Die Typen werden zur Modellierung der verschiedenen Möglichkeiten von Beziehungen zwischen den Elementen verwendet. Wenn Sie zum Beispiel eine Komponente erstellen, um eine oder mehrere Abteilungen in einem Unternehmen zu beschreiben, bietet sich wahrscheinlich der Komponententyp TREE an, da die meisten

Unternehmensstrukturen die Form einer Baumstruktur besitzen. Wenn Sie eine Komponente erstellen, die die Vertraulichkeitsstufe darstellt, den eine Person besitzt, würden Sie wahrscheinlich den Komponententyp ARRAY verwenden, da in der Regel für je zwei Vertraulichkeitsgrade stets einer höher als der andere ist.

Detaillierte Informationen zu den einzelnen Typen, einschließlich ausführlicher Beschreibungen der Beziehungen, die zwischen den Elementen bestehen können, finden Sie in den jeweiligen Beschreibungen zu den Typen.

Ändern der Komponenten von Sicherheitskennsätzen

Der Sicherheitsadministrator kann eine Sicherheitskennsatzkomponente mithilfe der Anweisung ALTER SECURITY LABEL COMPONENT ändern.

Löschen einer Sicherheitskennsatzkomponente

Zum Löschen einer Sicherheitskennsatzkomponente müssen Sie über die Berechtigung eines Sicherheitsadministrators verfügen. Eine Sicherheitskennsatzkomponente wird mithilfe der SQL-Anweisung DROP gelöscht.

Typ der LBAC-Sicherheitskennsatzkomponente: SET

SET ist ein Typ von Sicherheitskennsatzkomponente, der in einer LBAC-Sicherheitsrichtlinie (LBAC - kennsatzbasierte Zugriffssteuerung) verwendet werden kann.

Komponenten des Typs SET sind ungeordnete Listen von Elementen. Die einzige Vergleichsoperation, die für Elemente dieses Komponententyps durchgeführt werden kann, ist die Prüfung, ob ein bestimmtes Element in der Liste enthalten ist oder nicht.

Typ der LBAC-Sicherheitskennsatzkomponente: ARRAY

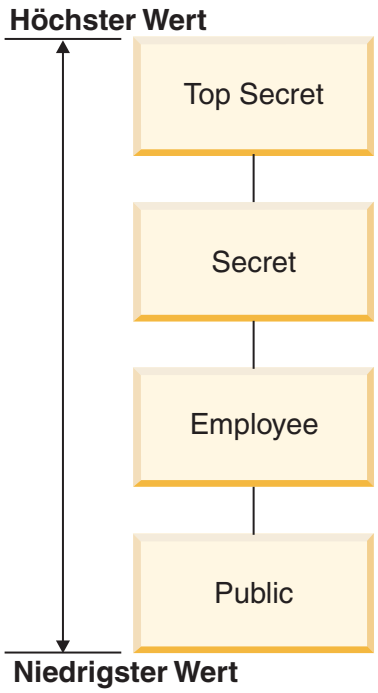
ARRAY ist ein Typ von Sicherheitskennsatzkomponente.

Bei dem Komponententyp ARRAY definiert die Reihenfolge, in der die Elemente beim Erstellen der Komponente aufgelistet werden, eine Skala, in der das erste aufgeführte Element den höchsten Wert und das zuletzt aufgeführte Element den niedrigsten Wert besitzt.

Beispiel: Eine Komponente 'mycomp' wird zum Beispiel auf folgende Weise definiert:

```
CREATE SECURITY LABEL COMPONENT mycomp
  ARRAY [ 'Top Secret', 'Secret', 'Employee', 'Public' ]
```

In diesem Fall werden die Elemente so behandelt, als wären sie in einer Struktur wie der folgenden angeordnet:



In einer Komponente des Typs ARRAY können Elemente die folgenden Arten von Beziehungen zueinander haben:

Höher als
Element A ist höher als Element B, wenn Element A in der Klausel ARRAY früher aufgeführt ist als Element B.

Niedriger als
Element A ist niedriger als Element B, wenn Element A in der Klausel ARRAY später aufgeführt ist als Element B.

Typ der LBAC-Sicherheitskennsatzkomponente: TREE

TREE ist ein Typ von Sicherheitskennsatzkomponente, der in einer LBAC-Sicherheitsrichtlinie (LBAC - kennsatzbasierte Zugriffssteuerung) verwendet werden kann.

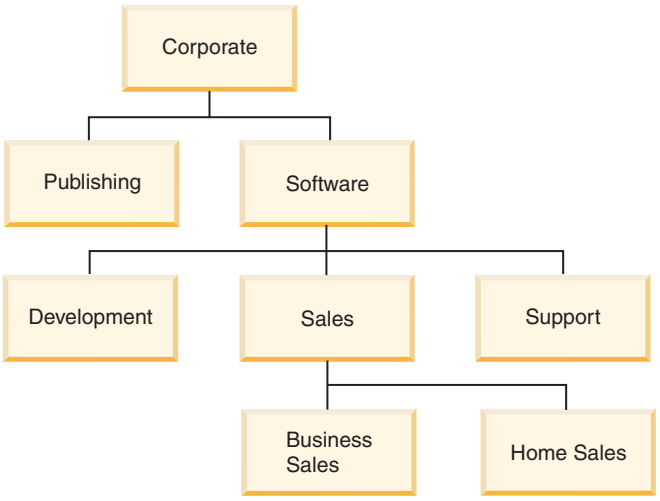
Bei dem Komponententyp TREE werden die Elemente so behandelt, als wären sie in einer Baumstruktur angeordnet. Wenn Sie ein Element angeben, das Teil einer Komponente des Typs TREE ist, müssen Sie auch angeben, unter welchem anderen Element es sich befindet. Einzige Ausnahme ist das erste Element, das als Stammelement (ROOT) der Baumstruktur angegeben werden muss. Auf diese Weise können die Elemente in einer Baumstruktur verwaltet werden.

Beispiel: Eine Komponente 'mycomp' wird zum Beispiel auf folgende Weise definiert:

```
CREATE SECURITY LABEL COMPONENT mycomp
TREE (
    'Corporate'      ROOT,
    'Publishing'     UNDER 'Corporate',
    'Software'       UNDER 'Corporate',
    'Development'    UNDER 'Software',
    'Sales'          UNDER 'Software',
```

```
    'Support'        UNDER 'Software'
    'Business Sales' UNDER 'Sales'
    'Home Sales'     UNDER 'Sales'
)
```

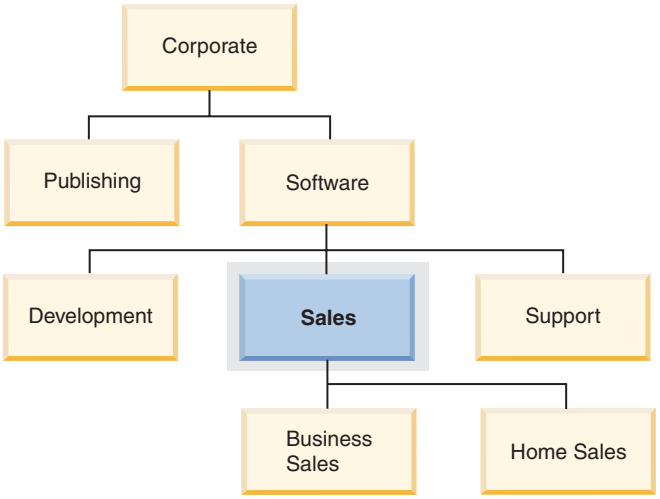
In diesem Fall werden die Elemente so behandelt, als wären sie in einer Baumstruktur wie der folgenden angeordnet:



In einer Komponente des Typs TREE können Elemente die folgenden Typen von Beziehungen zueinander haben:

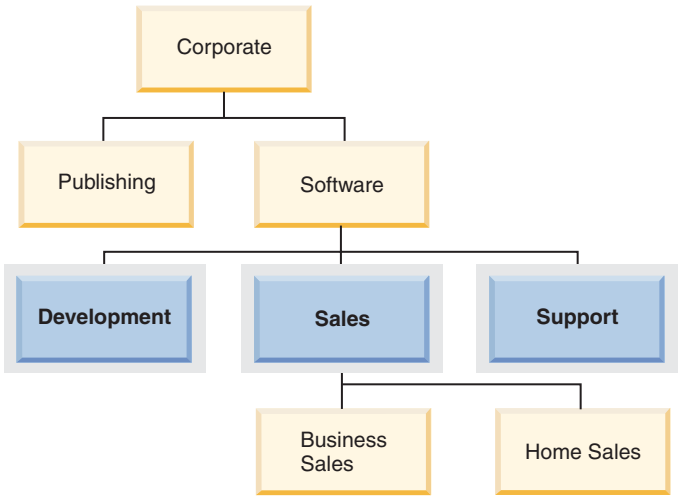
Direkt übergeordnetes Element (Elter)
Element A ist ein direkt übergeordnetes Element von Element B, wenn sich Element B *direkt unter* Element A befindet.

Beispiel: Das folgende Diagramm zeigt das direkt übergeordnete Element des Elements 'Business Sales':



Direkt untergeordnetes Element (Kind)
Element A ist ein direkt untergeordnetes Element von Element B, wenn sich Element A direkt unter Element B befindet.

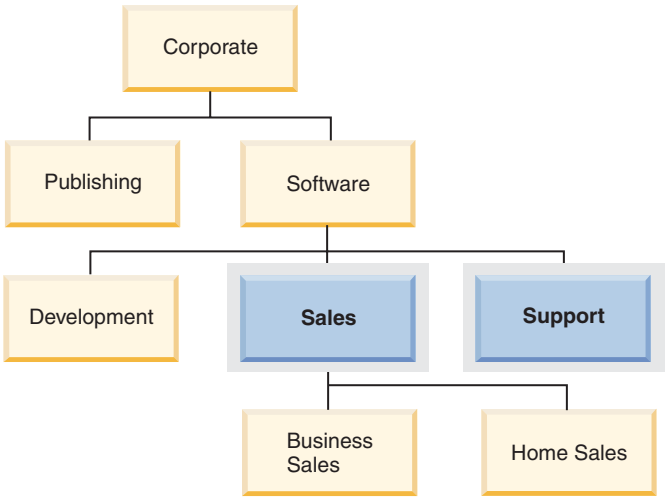
Beispiel: Das folgende Diagramm zeigt die direkt untergeordneten Elemente des Elements 'Software':



Gleichgeordnetes Element

Zwei Elemente sind einander gleichgeordnet, wenn sie dasselbe direkt übergeordnete Element besitzen.

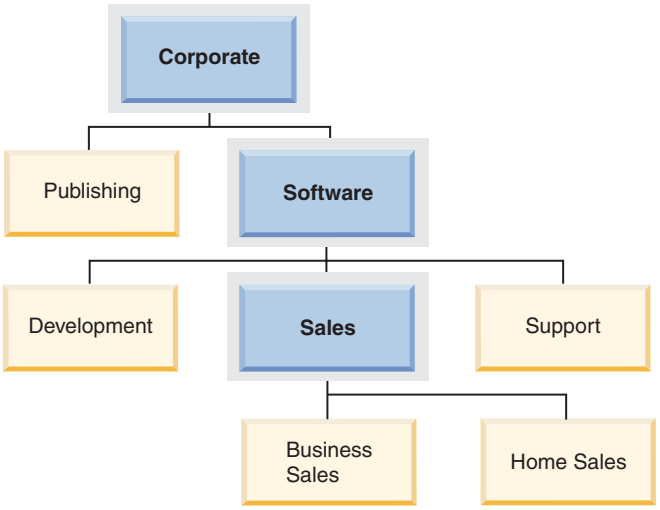
Beispiel: Das folgende Diagramm zeigt die gleichgeordneten Elemente des Elements 'Development':



Übergeordnetes Element (Vorfahre)

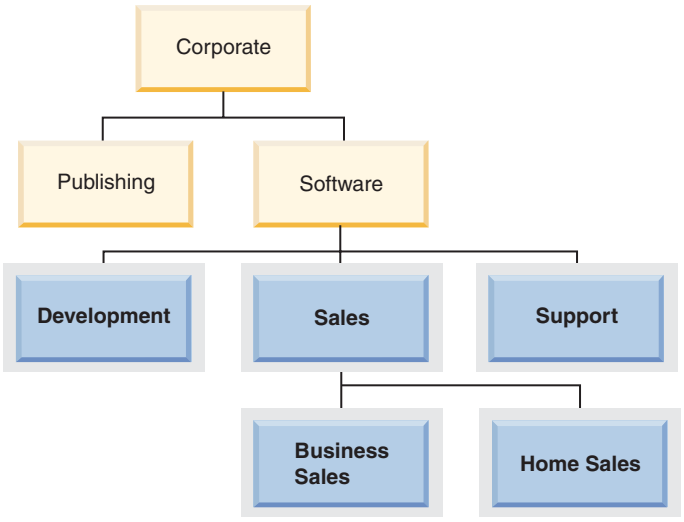
Element A ist ein übergeordnetes Element (Vorfahre) von Element B, wenn es das direkt übergeordnete Element von B oder ein entfernter übergeordnetes Element von B ist. Das Stammelement ist ein Vorfahre aller anderen Elemente in der Baumstruktur.

Beispiel: Das folgende Diagramm zeigt die Vorfahren des Elements 'Home Sales':



Untergeordnetes Element
Element A ist ein untergeordnetes Element von B, wenn es ein direkt oder entfernter untergeordnetes Element von B ist.

Beispiel: Das folgende Diagramm zeigt die untergeordneten Elemente des Elements 'Software':



LBAC-Sicherheitskennsätze

In LBAC (Label-Based Access Control, kennsatzbasierte Zugriffssteuerung) ist ein *Sicherheitskennsatz* ein Datenbankobjekt, das eine bestimmte Gruppe von Sicherheitskriterien beschreibt. Sicherheitskennsätze werden auf Daten angewendet, um die Daten zu schützen. Sie werden Benutzern erteilt, um Ihnen den Zugriff auf geschützte Daten zu ermöglichen.

Wenn ein Benutzer versucht, auf geschützte Daten zuzugreifen, wird sein Sicherheitskennsatz mit dem Sicherheitskennsatz verglichen, der die Daten schützt. Der schützende Sicherheitskennsatz blockiert einige Sicherheitskennsätze und andere nicht. Wenn der Sicherheitskennsatz eines Benutzers blockiert wird, erhält der Benutzer keinen Zugriff auf die Daten.

Jeder Sicherheitskennsatz ist Teil genau einer Sicherheitsrichtlinie und enthält einen Wert für jede Komponente in dieser Sicherheitsrichtlinie. Im Kontext einer Sicherheitskennsatzkomponente ist ein *Wert* eine Liste von null oder mehr Elementen, die durch diese Komponente zugelassen werden. Werte für Komponenten des Typs ARRAY können null oder ein Element enthalten, Werte für andere Typen können null oder mehrere Elemente enthalten. Ein Wert, der keine Elemente enthält, wird als *leerer Wert* bezeichnet.

Beispiel: Wenn eine Komponente des Typs TREE die drei Elemente 'Human Resources' (Personal), 'Sales' (Vertrieb) und 'Shipping' (Versand) enthält, stellen die folgenden Möglichkeiten einige der gültigen Werte für diese Komponente dar:

- Human Resources (oder ein beliebiges der Elemente allein)
- Human Resources, Shipping (oder eine beliebige andere Kombination der Elemente, solange kein Element mehr als einmal vorkommt)
- *Ein leerer Wert*

Ob ein bestimmter Sicherheitskennsatz einen anderen blockiert, wird durch die Werte der einzelnen Komponenten in den Kennsätzen sowie durch den LBAC-Regelsatz bestimmt, der in der Sicherheitsrichtlinie der Tabelle angegeben ist. Eine detaillierte Beschreibung dieses Vergleichs finden Sie in dem Abschnitt, der den Vergleich von LBAC-Sicherheitskennsätzen behandelt.

Wenn Sicherheitskennsätze in eine Textzeichenfolge konvertiert werden, verwenden sie das Format, das in dem Abschnitt beschrieben wird, der das Format für Sicherheitskennsatzwerte behandelt.

Erstellen von Sicherheitskennsätzen

Zur Erstellung eines Sicherheitskennsatzes müssen Sie über die Berechtigung eines Sicherheitsadministrators verfügen. Ein Sicherheitskennsatz wird mithilfe der SQL-Anweisung CREATE SECURITY LABEL erstellt. Bei der Erstellung eines Sicherheitskennsatzes müssen Sie folgende Informationen angeben:

- Einen Namen für den Kennsatz
- Die Sicherheitsrichtlinie, zu der der Kennsatz gehört
- Werte für eine oder mehrere der Komponenten, die in der Sicherheitsrichtlinie enthalten sind

Alle Komponenten, für die kein Wert angegeben wird, werden so interpretiert, als ob sie einen leeren Wert enthielten. Ein Sicherheitskennsatz muss mindestens einen nicht leeren Wert besitzen.

Ändern von Sicherheitskennsätzen

Sicherheitskennsätze können nicht geändert werden. Die einzige Möglichkeit zur Änderung eines Sicherheitskennsatzes besteht darin, diesen zu löschen und erneut zu erstellen. Die *Komponenten* eines Sicherheitskennsatzes können hingegen von einem Sicherheitsadministrator (mit der Anweisung ALTER SECURITY LABEL COMPONENT) geändert werden.

Löschen von Sicherheitskennsätzen

Zum Löschen eines Sicherheitskennsatzes müssen Sie über die Berechtigung eines Sicherheitsadministrators verfügen. Ein Sicherheitskennsatz wird mithilfe der SQL-Anweisung DROP gelöscht. Ein Sicherheitskennsatz, der in der Datenbank zum

Schutz von Daten verwendet wird oder zurzeit mindestens einem Benutzer erteilt ist, kann nicht gelöscht werden.

Erteilen von Sicherheitskennsätzen

Sie müssen über die Berechtigung eines Sicherheitsadministrators verfügen, um einem Benutzer, einer Gruppe oder einer Rolle einen Sicherheitskennsatz erteilen zu können. Ein Sicherheitskennsatz wird mithilfe der SQL-Anweisung GRANT SECURITY LABEL erteilt. Beim Erteilten eines Sicherheitskennsatzes können Sie ihn für den Lesezugriff, für den Schreibzugriff oder für den kombinierten Lese- und Schreibzugriff erteilen. Ein Benutzer, eine Gruppe bzw. eine Rolle kann nur einen Sicherheitskennsatz aus derselben Sicherheitsrichtlinie für denselben Typ von Zugriff besitzen.

Entziehen von Sicherheitskennsätzen

Sie müssen über die Berechtigung eines Sicherheitsadministrators verfügen, um einem Benutzer, einer Gruppe oder einer Rolle einen Sicherheitskennsatz entziehen zu können. Ein Sicherheitskennsatz wird mithilfe der SQL-Anweisung REVOKE SECURITY LABEL entzogen.

Mit Sicherheitskennsätzen kompatible Datentypen

Sicherheitskennsätze besitzen den Datentyp SYSPROC.DB2SECURITYLABEL. Zwischen den Datentypen SYSPROC.DB2SECURITYLABEL und VARCHAR(128) FOR BIT DATA wird eine Datenkonvertierung unterstützt.

Ermitteln der Benutzern erteilten Sicherheitskennsätze

Mithilfe der folgenden Abfrage können Sie die Sicherheitskennsätze ermitteln, die Benutzern erteilt wurden:

```
SELECT A.grantee, B.secpolicyname, c.seclabelname
FROM syscat.securitylabelaccess A, syscat.securitypolicies B, syscat.securitylabels C
WHERE A.seclabelid = C.seclabelid and B.secpolicyid = C.secpolicyid
```

Format für Werte von Sicherheitskennsätzen

Manchmal werden die Werte in einem Sicherheitskennsatz in Form einer Zeichenfolge dargestellt, zum Beispiel wenn die integrierte Funktion SECLABEL verwendet wird.

Wenn die Werte in einem Sicherheitskennsatz als Zeichenfolge dargestellt werden, weisen sie folgendes Format auf:

- Die Werte der Komponenten werden von links nach rechts in der gleichen Reihenfolge aufgelistet, in der die Komponenten in der Anweisung CREATE SECURITY POLICY für die Sicherheitsrichtlinie aufgelistet werden.
- Ein Element wird durch den Namen dieses Elements dargestellt.
- Elemente für verschiedene Komponenten werden durch einen Doppelpunkt (:) getrennt.
- Wenn mehr als ein Element für die gleiche Komponente angegeben wird, werden die Elemente in Klammern (()) angegeben und durch Kommata (,) getrennt.
- Leere Werte werden durch ein Paar leerer Klammern (()) dargestellt.

Beispiel: Ein Sicherheitskennsatz ist Teil einer Sicherheitsrichtlinie, welche die folgenden drei Komponenten in dieser Reihenfolge enthält: 'Level', 'Department' und 'Projects'. Der Sicherheitskennsatz hat die folgenden Werte:

Tabelle 10. Beispielwerte für einen Sicherheitskennsatz

Komponente	Werte
Level	Secret
Department	*Leerer Wert*
Projects	• Epsilon 37 • Megaphone • Cloverleaf

Die Werte dieses Sicherheitskennsatzes werden als Zeichenfolge wie folgt dargestellt:

```
'Secret:():(Epsilon 37,Megaphone,Cloverleaf)'
```

Vergleichen von LBAC-Sicherheitskennsätzen

Wenn Sie versuchen, auf Daten zuzugreifen, die dem LBAC-Schutz (Label-Based Access Control, kennsatzbasierte Zugriffssteuerung) unterliegen, werden Ihre LBAC-Berechtigungsnachweise mit einem oder mehreren Sicherheitskennsätzen verglichen, um zu prüfen, ob der Zugriff blockiert ist. Ihre LBAC-Berechtigungsnachweise bestehen aus allen Sicherheitskennsätzen und allen Freistellungen, die Sie besitzen.

Nur zwei Typen von Vergleichen können ausgeführt werden. Ihre LBAC-Berechtigungsnachweise können mit einem einzelnen Sicherheitskennsatz für Lesezugriff verglichen werden. Oder sie können mit einem einzelnen Sicherheitskennsatz für Schreibzugriff verglichen werden. Aktualisieren und Löschen werden wie ein Lesevorgang gefolgt von einem Schreibvorgang behandelt. Wenn eine Operation mehrere Vergleiche erfordert, wird jeder Vergleich separat ausgeführt.

Welcher Ihrer Sicherheitskennsätze wird verwendet?

Auch wenn Sie vielleicht mehrere Sicherheitskennsätze besitzen, wird nur einer mit dem schützenden Sicherheitskennsatz verglichen. Verwendet wird der Kennsatz, der folgende Bedingungen erfüllt:

- Er ist Teil der Sicherheitsrichtlinie, die die Tabelle schützt, auf die zugegriffen werden soll.
- Er wurde für den beabsichtigten Typ von Zugriff (Lese- oder Schreibzugriff) erteilt.

Wenn Sie keinen Sicherheitskennsatz besitzen, der diese Bedingungen erfüllt, wird ein Standardsicherheitskennsatz angenommen, der leere Werte für alle Komponenten enthält.

Ausführung des Vergleichs

Sicherheitskennsätze werden komponentenweise verglichen. Wenn ein Sicherheitskennsatz keinen Wert für eine der Komponenten enthält, wird ein leerer Wert angenommen. Bei der Prüfung der einzelnen Komponenten wird anhand der relevanten Regeln des LBAC-Regelsatzes entschieden, ob die Elemente in Ihrem Wert für die jeweilige Komponente durch Elemente im Wert für die gleiche Komponente im

schützenden Kennsatz blockiert werden sollen. Wenn irgendeiner Ihrer Werte blockiert wird, werden Ihre LBAC-Berechtigungsnachweise durch den schützenden Sicherheitskennsatz blockiert.

Der LBAC-Regelsatz, der für den Vergleich herangezogen wird, wird in der Sicherheitsrichtlinie festgelegt. Informationen zu den Regeln sowie zu ihrer Anwendung finden Sie in der Beschreibung des jeweiligen Regelsatzes.

Wirkung von Freistellungen auf den Vergleich

Wenn Sie eine Freistellung für die Regel besitzen, die zum Vergleichen zweier Werte verwendet wird, wird dieser Vergleich nicht ausgeführt und der schützende Wert wird so interpretiert, dass er den Wert in Ihrem Sicherheitskennsatz nicht blockiert.

Beispiel: Es wird der LBAC-Regelsatz DB2LBACRULES verwendet, und die Sicherheitsrichtlinie hat zwei Komponenten. Eine Komponente hat den Typ ARRAY, die andere den Typ TREE. Dem Benutzer wurde eine Freistellung für die Regel DB2LBACREADTREE erteilt. Diese Regel wird auf den Lesezugriff angewendet, wenn Komponentenwerte des Typs TREE verglichen werden. Wenn der Benutzer versucht, geschützte Daten zu lesen, blockiert jeder beliebige Wert, den der Benutzer für die TREE-Komponente besitzt, selbst wenn es sich um einen leeren Wert handelt, den Zugriff nicht, weil die Regel nicht angewendet wird. Ob der Benutzer die Daten lesen kann, hängt ganz von den Werten der ARRAY-Komponente der Kennsätze ab.

LBAC-Regelsätze - Übersicht

Ein LBAC-Regelsatz ist ein vordefinierter Satz von Regeln, die beim Vergleichen von Sicherheitskennsätzen verwendet werden. Wenn die Werte zweier Sicherheitskennsätze verglichen werden, wird mindestens eine der Regeln des Regelsatzes angewendet, um zu bestimmen, ob ein Wert einen anderen blockiert.

Jeder LBAC-Regelsatz wird durch einen eindeutigen Namen identifiziert. Beim Erstellen einer Sicherheitsrichtlinie müssen Sie den LBAC-Regelsatz angeben, der mit dieser Richtlinie verwendet werden soll. Für alle Vergleiche von Sicherheitskennsätzen, die Teil dieser Richtlinie sind, wird dann der angegebene LBAC-Regelsatz verwendet.

Jede einzelne Regel in einem Regelsatz wird ebenfalls durch einen eindeutigen Namen gekennzeichnet. Der Name einer Regel wird verwendet, wenn eine Freistellung von der betreffenden Regel erteilt wird.

Die Anzahl von Regeln in einem Satz und der Zeitpunkt, zu dem eine Regel verwendet wird, können je nach Regelsatz unterschiedlich sein.

Gegenwärtig wird nur ein LBAC-Regelsatz unterstützt. Der Name dieses Regelsatzes lautet DB2LBACRULES.

LBAC-Regelsatz: DB2LBACRULES

Der LBAC-Regelsatz DB2LBACRULES stellt einen traditionellen Satz von Regeln zum Vergleich der Werte von Sicherheitskennsatzkomponenten bereit. Er schützt gegen Write-up- und Write-down-Vorgänge.

Was bedeuten Write-up und Write-down?

Write-up und Write-down bezieht sich nur auf Komponenten des Typs ARRAY und nur auf den Schreibzugriff. Ein Write-up tritt auf, wenn der Wert, der Daten schützt, in die Sie schreiben wollen, höher ist als Ihr Wert. Wenn der Wert, der die Daten schützt, niedriger als Ihr Wert ist, wird dies als Write-down bezeichnet. Standardmäßig werden weder Write-up- noch Write-down-Vorgänge zugelassen. Das bedeutet, dass Sie nur in Daten schreiben können, die durch den gleichen Wert geschützt werden, den auch Sie besitzen.

Wenn zwei Werte für dieselbe Komponente verglichen werden, hängen die Regeln, die dazu verwendet werden, vom Typ der Komponente (ARRAY, SET oder TREE) sowie vom Typ des versuchten Zugriffs (Schreib- oder Lesezugriff) ab. In der folgenden Tabelle werden die Regeln aufgelistet und beschrieben, wann sie jeweils verwendet werden und wie sie bestimmen, ob der Zugriff blockiert wird.

Tabelle 11. Zusammenfassung der DB2LBACRULES-Regeln

Regelname	Typ von Komponente, für deren Wertevergleich die Regel gilt	Typ von Zugriffsversuch, für den die Regel gilt	Bedingung, unter der der Zugriff blockiert wird
DB2LBACREADARRAY	ARRAY	Lesen	Der Wert des Benutzers ist niedriger als der schützende Wert.
DB2LBACREADSET	SET	Lesen	Es gibt mindestens einen schützenden Wert, den der Benutzer nicht besitzt.
DB2LBACREADTREE	TREE	Lesen	Keiner der Werte des Benutzers ist gleich einem der schützenden Werte oder einem der übergeordneten Werte (Vorfahren) der schützenden Werte.
DB2LBACWRITEARRAY	ARRAY	Schreiben	Der Wert des Benutzers ist höher oder niedriger als der schützende Wert.[1]
DB2LBACWRITESET	SET	Schreiben	Es gibt mindestens einen schützenden Wert, den der Benutzer nicht besitzt.
DB2LBACWRITETREE	TREE	Schreiben	Keiner der Werte des Benutzers ist gleich einem der schützenden Werte oder einem der übergeordneten Werte (Vorfahren) der schützenden Werte.

Anmerkung:

1. Die Regel DB2LBACWRITEARRAY kann als Kombination aus zwei verschiedenen Regeln betrachtet werden. Die eine verhindert das Schreiben in Daten, die sich auf einer höheren als Ihre Stufe befinden (Write-up), während die andere das Schreiben in Daten verhindert, die sich auf einer niedrigeren als Ihre Stufe befinden (Write-down). Wenn Sie eine Freistellung von dieser Regel erteilen, können Sie den Benutzer von der einen oder der anderen oder auch von beiden Regeln freistellen.

Behandlung leerer Werte durch die Regeln

Alle Regeln behandeln leere Werte in gleicher Weise. Ein leerer Wert blockiert keine anderen Werte und wird selbst von jedem nicht leeren Wert blockiert.

Beispiele für DB2LBACREADSET und DB2LBACWRITESET

Diese Beispiele gelten für einen Benutzer, der versucht, geschützte Daten zu lesen oder in geschützte Daten zu schreiben. Für sie wird angenommen, dass die Werte für eine Komponente des Typs SET definiert sind, die die folgenden Elemente besitzt: eins, zwei, drei, vier.

Tabelle 12. Beispiele für die Anwendung der Regeln DB2LBACREADSET und DB2LBACWRITESET

Wert des Benutzers	Schützender Wert	Zugriff blockiert?
'eins'	'eins'	Nicht blockiert. Die Werte sind identisch.
'(eins,zwei,drei)'	'eins'	Nicht blockiert. Der Wert des Benutzers enthält das Element 'eins'.
'(eins,zwei)'	'(eins,zwei,vier)'	Blockiert. Das Element 'vier' ist im schützenden Wert, jedoch nicht im Wert des Benutzers enthalten.
'()'	'eins'	Blockiert. Ein leerer Wert wird von jedem beliebigen nicht leeren Wert blockiert.
'eins'	'()'	Nicht blockiert. Kein Wert wird durch einen leeren Wert blockiert.
'()'	'()'	Nicht blockiert. Kein Wert wird durch einen leeren Wert blockiert.

DB2LBACREADTREE und DB2LBACWRITETREE

Die folgenden Beispiele gelten sowohl für den Lese- als auch für den Schreibzugriff. Für sie wird angenommen, dass die Werte für eine Komponente des Typs TREE gelten, die auf folgende Weise definiert ist:

```
CREATE SECURITY LABEL COMPONENT mycomp
TREE (
    'Corporate'      ROOT,
    'Publishing'     UNDER 'Corporate',
    'Software'       UNDER 'Corporate',
    'Development'    UNDER 'Software',
    'Sales'          UNDER 'Software',
    'Support'        UNDER 'Software'
    'Business Sales' UNDER 'Sales'
    'Home Sales'     UNDER 'Sales'
)
```

Dies bedeutet, dass die Elemente wie folgt angeordnet sind:

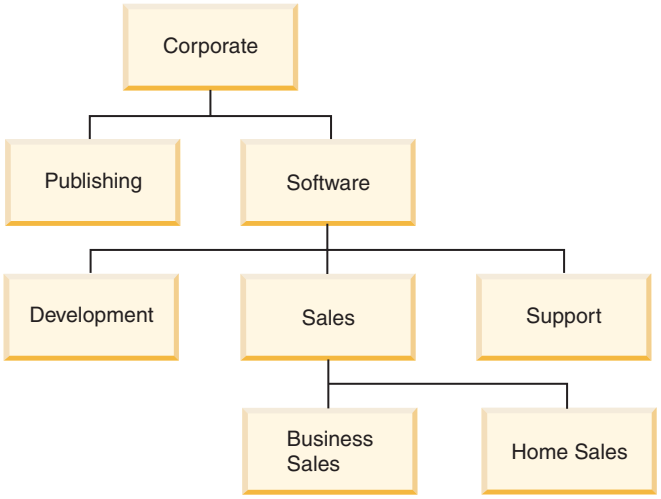


Tabelle 13. Beispiel für die Anwendung der Regeln DB2LBACREADTREE und DB2LBACWRITETREE

Wert des Benutzers	Schützender Wert	Zugriff blockiert?
'(Support,Sales)'	'Development'	Blockiert. Das Element 'Development' ist nicht in den Werten des Benutzers enthalten und weder 'Support' noch 'Sales' ist ein übergeordnetes Element (Vorfahre) von 'Development'.
'(Development,Software)'	'(Business Sales,Publishing)'	Nicht blockiert. Das Element 'Software' ist ein übergeordnetes Element (Vorfahre) des Elements 'Business Sales'.
'(Publishing,Sales)'	'(Publishing,Support)'	Nicht blockiert. Das Element 'Publishing' ist in beiden Gruppen von Werten enthalten.
'Corporate'	'Development'	Nicht blockiert. Das Stammelement ist ein übergeordnetes Element (Vorfahre) aller anderen Werte.
'()'	'Sales'	Blockiert. Ein leerer Wert wird von jedem beliebigen nicht leeren Wert blockiert.
'Home Sales'	'()'	Nicht blockiert. Kein Wert wird durch einen leeren Wert blockiert.
'()'	'()'	Nicht blockiert. Kein Wert wird durch einen leeren Wert blockiert.

Beispiele für DB2LBACREADARRAY

Diese Beispiele gelten nur für den Lesezugriff. Für sie wird angenommen, dass die Werte für eine Komponente des Typs ARRAY definiert sind, die die folgenden Elemente in der angegebenen Anordnung enthält:

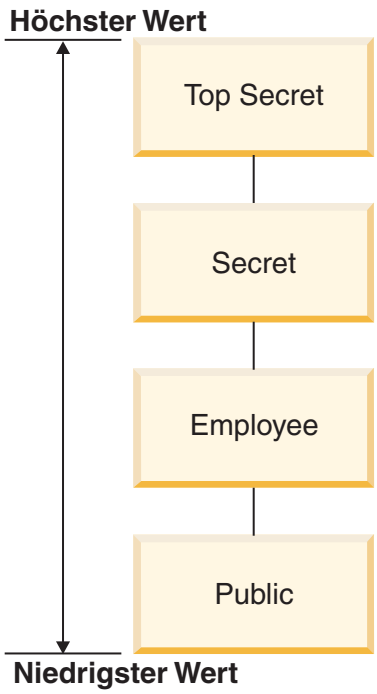


Tabelle 14. Beispiele für die Anwendung der Regel DB2LBACREADARRAY

Wert des Benutzers	Schützender Wert	Lesezugriff blockiert?
'Secret'	'Employee'	Nicht blockiert. Das Element 'Secret' ist höher als das Element 'Employee'.
'Secret'	'Secret'	Nicht blockiert. Die Werte sind identisch.
'Secret'	'Top Secret'	Blockiert. Das Element 'Top Secret' ist höher als das Element 'Secret'.
'()'	'Public'	Blockiert. Ein leerer Wert wird von jedem beliebigen nicht leeren Wert blockiert.
'Public'	'()'	Nicht blockiert. Kein Wert wird durch einen leeren Wert blockiert.
'()'	'()'	Nicht blockiert. Kein Wert wird durch einen leeren Wert blockiert.

Beispiele für DB2LBACWRITEARRAY

Diese Beispiele gelten nur für den Schreibzugriff. Für sie wird angenommen, dass die Werte für eine Komponente des Typs ARRAY definiert sind, die die folgenden Elemente in der angegebenen Anordnung enthält:

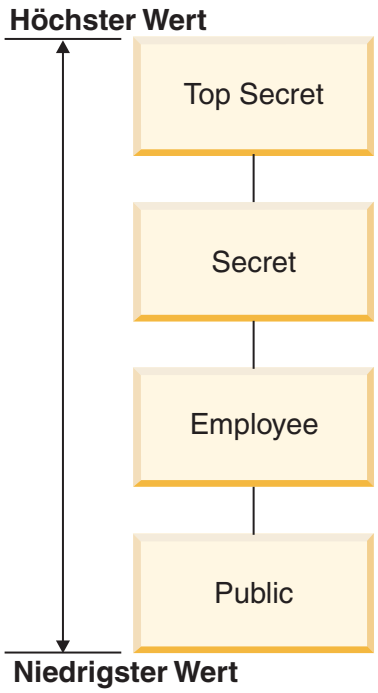


Tabelle 15. Beispiele für die Anwendung der Regel DB2LBACWRITEARRAY

Wert des Benutzers	Schützender Wert	Schreibzugriff blockiert?
'Secret'	'Employee'	Blockiert. Das Element 'Employee' ist niedriger als das Element 'Secret'.
'Secret'	'Secret'	Nicht blockiert. Die Werte sind identisch.
'Secret'	'Top Secret'	Blockiert. Das Element 'Top Secret' ist höher als das Element 'Secret'.
'()'	'Public'	Blockiert. Ein leerer Wert wird von jedem beliebigen nicht leeren Wert blockiert.
'Public'	'()'	Nicht blockiert. Kein Wert wird durch einen leeren Wert blockiert.
'()'	'()'	Nicht blockiert. Kein Wert wird durch einen leeren Wert blockiert.

Freistellungen von LBAC-Regeln

Wenn Sie eine LBAC-Regelfreistellung für eine bestimmte Regel einer bestimmten Sicherheitsrichtlinie haben, wird diese Regel nicht angewendet, wenn Sie versuchen, auf die Daten zuzugreifen, die durch diese Sicherheitsrichtlinie geschützt werden.

Eine Freistellung hat keine Wirkung, wenn Sicherheitskennsätze einer anderen Sicherheitsrichtlinie als die, für die sie erteilt wurde, verglichen werden.

Beispiel:

Es sind zwei Tabellen vorhanden: T1 und T2. T1 wird durch die Sicherheitsrichtlinie P1, T2 durch die Sicherheitsrichtlinie P2 geschützt. Beide Sicherheitsrichtlinien enthalten eine Komponente. Die Komponenten sind jeweils vom Typ ARRAY. T1

und T2 enthalten jeweils nur eine Zeile von Daten. Der Sicherheitskennsatz, den Sie für den Schreibzugriff unter der Sicherheitsrichtlinie P1 besitzen, erlaubt Ihnen keinen Zugriff auf die Zeile in T1. Der Sicherheitskennsatz, den Sie für den Lesezugriff unter der Sicherheitsrichtlinie P2 besitzen, erlaubt Ihnen keinen Zugriff auf die Zeile in T2.

In dieser Situation wird Ihnen eine Freistellung für DB2LBACREADARRAY unter P1 erteilt. Jetzt können Sie die Zeile in T1 lesen, die Zeile in T2 jedoch nicht, weil T2 von einer anderen Sicherheitsrichtlinie geschützt wird und Sie keine Freistellung für die Regel DB2LBACREADARRAY in dieser Richtlinie haben.

Sie können mehrere Freistellungen besitzen. Wenn Sie eine Freistellung für jede Regel besitzen, die von einer Sicherheitsrichtlinie verwendet wird, haben Sie vollständigen Zugriff auf alle Daten, die durch diese Sicherheitsrichtlinie geschützt werden.

Erteilen von Freistellungen für LBAC-Regeln

Zum Erteilen einer Freistellung für eine LBAC-Regel müssen Sie ein Sicherheitsadministrator sein. Eine Freistellung für eine LBAC-Regel wird mithilfe der SQL-Anweisung GRANT EXEMPTION ON RULE erteilt.

Beim Erteilen einer Freistellung für eine LBAC-Regel geben Sie die folgenden Informationen an:

- Die Regel bzw. die Regeln, für die die Freistellung gilt
- Die Sicherheitsrichtlinie, für die die Freistellung gilt
- Den Benutzer, die Gruppe oder die Rolle, für die Sie die Freistellung erteilen

Wichtig: Freistellungen von LBAC-Regeln gewähren einen sehr mächtigen Zugriff. Erteilen Sie Freistellungen nur nach reiflicher Überlegung.

Entziehen von Freistellungen für LBAC-Regeln

Zum Entziehen einer Freistellung für eine LBAC-Regel müssen Sie ein Sicherheitsadministrator sein. Eine Freistellung für eine LBAC-Regel wird mithilfe der SQL-Anweisung REVOKE EXEMPTION ON RULE entzogen.

Ermitteln der Benutzern erteilten Regelfreistellungen

Mithilfe der folgenden Abfrage können Sie die Regelfreistellungen ermitteln, die Benutzern erteilt wurden:

```
SELECT A.grantee, A.accessrulename, B.secpolicyname
FROM syscat.securitypolicyexemptions A, syscat.securitypolicies B
WHERE A.secpolicyid = B.secpolicyid
```

Integrierte Funktionen zum Verwalten von LBAC-Sicherheitskennsätzen

Die integrierten Funktionen SECLABEL, SECLABEL_BY_NAME und SECLABEL_TO_CHAR dienen zur Verwaltung von Sicherheitskennsätzen der kennsatzbasierten Zugriffssteuerung (LBAC, Label-Based Access Control).

Jede von ihnen wird im Folgenden kurz beschrieben. Eine detaillierte Beschreibung finden Sie im Handbuch *SQL Reference*.

SECLABEL

Diese integrierte Funktion dient zum Aufbau eines Sicherheitskennsatzes. Dazu werden eine Sicherheitsrichtlinie und Werte für die einzelnen Komponenten in diesem Kennsatz angegeben. Der zurückgegebene Wert besitzt den Datentyp DB2SECURITYLABEL und ist ein Sicherheitskennsatz, der zur angegebenen Sicherheitsrichtlinie gehört und die angegebenen Werte für die Komponenten besitzt. Es ist nicht erforderlich, dass ein Sicherheitskennsatz mit den angegebenen Werten bereits vorhanden ist.

Beispiel: Tabelle T1 hat zwei Spalten. Die erste hat den Datentyp DB2SECURITYLABEL und die zweite den Datentyp INTEGER. T1 wird durch die Sicherheitsrichtlinie P1 geschützt, die drei Sicherheitskennsatzkomponenten enthält: 'level', 'departments' und 'groups'. Wenn 'UNCLASSIFIED' ein Element der Komponente 'level', 'ALPHA' und 'SIGMA' beides Elemente der Komponente 'departments' und 'G2' ein Element der Komponente 'groups' sind, kann zum Beispiel ein Sicherheitskennsatz wie der folgende eingefügt werden:

```
INSERT INTO T1 VALUES
   ( SECLABEL( 'P1', 'UNCLASSIFIED:(ALPHA,SIGMA):G2' ), 22 )
```

SECLABEL_BY_NAME

Diese integrierte Funktion akzeptiert den Namen einer Sicherheitsrichtlinie und den Namen eines Sicherheitskennsatzes, der Teil dieser Sicherheitsrichtlinie ist. Dann gibt Sie den angegebenen Sicherheitskennsatz als Wert des Typs DB2SECURITYLABEL zurück. Diese Funktion muss verwendet werden, wenn ein vorhandener Sicherheitskennsatz in eine Spalte eingefügt wird, die den Datentyp DB2SECURITYLABEL besitzt.

Beispiel: Tabelle T1 hat zwei Spalten. Die erste hat den Datentyp DB2SECURITYLABEL und die zweite den Datentyp INTEGER. Der Sicherheitskennsatz mit dem Namen L1 ist Teil der Sicherheitsrichtlinie P1. Die folgende SQL-Anweisung kann in diesem Fall zum Einfügen des Sicherheitskennsatzes verwendet werden:

```
INSERT INTO T1 VALUES ( SECLABEL_BY_NAME( 'P1', 'L1' ), 22 )
```

Die folgende SQL-Anweisung funktioniert hingegen nicht:

```
INSERT INTO T1 VALUES ( P1.L1, 22 )     // Syntaxfehler!
```

SECLABEL_TO_CHAR

Diese integrierte Funktion liefert eine Zeichenfolgedarstellung der Werte, die einen Sicherheitskennsatz bilden.

Beispiel: Spalte C1 in Tabelle T1 hat den Datentyp DB2SECURITYLABEL. T1 wird durch die Sicherheitsrichtlinie P1 geschützt, die drei Sicherheitskennsatzkomponenten enthält: 'level', 'departments' und 'groups'. Die Tabelle T1 enthält eine Zeile, wobei der Wert in Spalte C1 die folgenden Elemente für die einzelnen Komponenten enthält:

Komponente	Elemente
'level'	SECRET
'departments'	DELTA und SIGMA
'groups'	G3

Ein Benutzer, der über LBAC-Berechtigungsnachweise verfügt, die einen Lesezugriff auf die Zeilen zulassen, kann zum Beispiel die folgende SQL-Anweisung ausführen:

```
SELECT SECLABEL_TO_CHAR( 'P1', C1 ) AS C1 FROM T1
```

Die Ausgabe sähe wie folgt aus:

```
C1

'SECRET:(DELTA,SIGMA):G3'
```

Schützen von Daten mit LBAC

Die LBAC-Funktionalität (Label-Based Access Control, kennsatzbasierte Zugriffssteuerung) kann zum Schutz von Datenzeilen, Datenspalten oder beidem verwendet werden. Daten in einer Tabelle können nur durch Sicherheitskennsätze geschützt werden, die Teil der Sicherheitsrichtlinie sind, die die Tabelle schützt. Der Datenschutz, einschließlich des Hinzufügens einer Sicherheitsrichtlinie, kann beim Erstellen oder später beim Ändern einer Tabelle eingefügt werden.

Sie können innerhalb einer Anweisung CREATE TABLE oder ALTER TABLE einer Tabelle eine Sicherheitsrichtlinie hinzufügen und Daten in dieser Tabelle schützen.

Als allgemeine Regel gilt, dass es nicht zulässig ist, Daten so zu schützen, dass die eigenen aktuellen LBAC-Berechtigungsnachweise keinen Schreibzugriff auf diese Daten erlauben.

Hinzufügen einer Sicherheitsrichtlinie zu einer Tabelle

Sie können einer zu erstellenden Tabelle eine Sicherheitsrichtlinie hinzufügen, indem Sie die Klausel SECURITY POLICY in der Anweisung CREATE TABLE verwenden. Sie können einer vorhandenen Tabelle eine Sicherheitsrichtlinie hinzufügen, indem Sie die Klausel ADD SECURITY POLICY der Anweisung ALTER TABLE verwenden. Zum Hinzufügen einer Sicherheitsrichtlinie zu einer Tabelle sind weder eine Berechtigung SECADM noch LBAC-Berechtigungsnachweise erforderlich.

Sicherheitsrichtlinien können keinen Typen von Tabellen hinzugefügt werden, die nicht durch LBAC geschützt werden können. In der Übersicht zu LBAC finden Sie eine Liste von Tabellentypen, die nicht durch LBAC geschützt werden können.

Jeder Tabelle kann nur eine Sicherheitsrichtlinie hinzugefügt werden.

Schützen von Zeilen

Sie können geschützte Zeilen in einer neuen Tabelle zulassen, indem Sie eine Spalte mit dem Datentyp DB2SECURITYLABEL beim Erstellen der Tabelle mit einfügen. Die Anweisung CREATE TABLE muss der Tabelle außerdem eine Sicherheitsrichtlinie hinzufügen. Zur Erstellung einer solchen Tabelle sind weder eine Berechtigung SECADM noch LBAC-Berechtigungsnachweise erforderlich.

Sie können geschützte Zeilen in einer vorhandenen Tabelle zulassen, indem Sie eine Spalte mit dem Datentyp DB2SECURITYLABEL hinzufügen. Um eine solche Spalte hinzufügen zu können, muss die Tabelle entweder bereits durch eine Sicherheitsrichtlinie geschützt sein oder die Anweisung ALTER TABLE, die die Spalte hinzufügt, muss der Tabelle auch eine Sicherheitsrichtlinie hinzufügen. Wenn die Spalte hinzugefügt ist, wird der Sicherheitskennsatz, den Sie für den Schreibzugriff

besitzen, zum Schutz aller vorhandenen Zeilen verwendet. Wenn Sie keinen Sicherheitskennsatz für den Schreibzugriff besitzen, der Teil der Sicherheitsrichtlinie ist, die die Tabelle schützt, können Sie keine Spalte mit dem Datentyp DB2SECURITYLABEL hinzufügen.

Wenn eine Spalte des Typs DB2SECURITYLABEL in eine Tabelle eingefügt ist, schützen Sie jede neue Zeile von Daten, indem Sie einen Sicherheitskennsatz in dieser Spalte speichern. Detaillierte Informationen dazu, wie dies funktioniert, enthalten die Themenabschnitte zum Einfügen und Aktualisieren von LBAC-geschützten Daten. Sie müssen über LBAC-Berechtigungsnachweise verfügen, um Zeilen in eine Tabelle einzufügen, die eine Spalte des Typs DB2SECURITYLABEL besitzt.

Eine Spalte, die den Datentyp DB2SECURITYLABEL besitzt, kann weder gelöscht noch in einen anderen Datentyp geändert werden.

Schützen von Spalten

Sie können bei der Erstellung der Tabelle eine Spalte schützen, indem Sie die Spaltenoption SECURED WITH der Anweisung CREATE TABLE verwenden. Sie können einer vorhandenen Spalte einen Schutz hinzufügen, indem Sie die Option SECURED WITH in einer Anweisung ALTER TABLE verwenden.

Zum Schützen einer Spalte durch einen bestimmten Sicherheitskennsatz müssen Sie über LBAC-Berechtigungsnachweise verfügen, die Ihnen einen Schreibzugriff auf die durch diesen Sicherheitskennsatz geschützten Daten erlauben. Die Berechtigung SECADM ist nicht erforderlich.

Spalten können nur durch Sicherheitskennsätze geschützt werden, die Teil der Sicherheitsrichtlinie sind, die die Tabelle schützt. Spalten in einer Tabelle, die keine Sicherheitsrichtlinie besitzt, können nicht geschützt werden. Es ist zulässig, eine Tabelle durch eine Sicherheitsrichtlinie und gleichzeitig eine oder mehrere Spalten in derselben Anweisung zu schützen.

Sie können eine beliebige Anzahl von Spalten in einer Tabelle schützen, jedoch kann eine Spalte nur durch einen einzigen Sicherheitskennsatz geschützt werden.

Lesen von LBAC-geschützten Daten

Wenn Sie versuchen, Daten zu lesen, die einem LBAC-Schutz (Label-Based Access Control, kennsatzbasierte Zugriffssteuerung) unterliegen, werden Ihre LBAC-Berechtigungsnachweise für den Lesezugriff mit dem Sicherheitskennsatz verglichen, der die Daten schützt. Wenn der schützende Kennsatz Ihre Berechtigungsnachweise nicht blockiert, wird Ihr Lesezugriff auf die Daten zugelassen.

Bei einer geschützten Spalte ist der schützende Sicherheitskennsatz im Schema der Tabelle definiert. Der schützende Sicherheitskennsatz für diese Spalte ist für jede Zeile in der Tabelle der gleiche. Bei einer geschützten Zeile wird der schützende Sicherheitskennsatz der Zeile in einer Spalte des Typs DB2SECURITYLABEL gespeichert. Der Wert dieser Spalte kann für jede Zeile in der Tabelle unterschiedlich sein.

Detaillierte Informationen dazu, wie Ihre LBAC-Berechtigungsnachweise mit einem Sicherheitskennsatz verglichen werden, finden Sie in dem Abschnitt über den Vergleich von LBAC-Sicherheitskennsätzen.

Lesen geschützter Spalten

Wenn Sie versuchen, Daten aus einer geschützten Spalte zu lesen, werden Ihre LBAC-Berechtigungsnachweise mit dem Sicherheitskennsatz verglichen, der die Spalte schützt. Abhängig vom Ergebnis dieses Vergleichs wird der Zugriff entweder blockiert oder zugelassen. Wenn der Zugriff blockiert wird, wird ein Fehler zurückgegeben und die Ausführung der Anweisung schlägt fehl. Anderenfalls wird die Ausführung der Anweisung normal fortgesetzt.

Ein Versuch, eine Spalte zu lesen, auf die Ihre LBAC-Berechtigungsnachweise keinen Lesezugriff erlauben, bewirkt, dass die gesamte Anweisung fehlschlägt.

Beispiel:

Die Tabelle T1 enthält zwei geschützte Spalten. Die Spalte C1 wird durch den Sicherheitskennsatz L1 geschützt. Die Spalte C2 wird durch den Sicherheitskennsatz L2 geschützt.

Nehmen Sie an, die Benutzerin Jyoti verfügt über LBAC-Berechtigungsnachweise für einen Lesezugriff, der durch den Sicherheitskennsatz L1 zugelassen wird, durch den Sicherheitskennsatz L2 jedoch nicht. Wenn Jyoti die folgende SQL-Anweisung absetzt, schlägt die Ausführung der Anweisung fehl:

```
SELECT * FROM T1
```

Die Ausführung der Anweisung schlägt fehl, weil Spalte C2 durch das Platzhalterzeichen (*) in die SELECT-Klausel mit eingeschlossen ist.

Wenn Jyoti die folgende SQL-Anweisung absetzt, ist die Ausführung erfolgreich:

```
SELECT C1 FROM T1
```

Die einzige geschützte Spalte in der SELECT-Klausel ist C1, und die LBAC-Berechtigungsnachweise von Jyoti lassen den Lesezugriff auf diese Spalte zu.

Lesen geschützter Zeilen

Wenn Sie keine LBAC-Berechtigungsnachweise besitzen, die Ihnen einen Lesezugriff auf eine Zeile erlauben, wird Ihr Lesezugriff so verarbeitet, als ob diese Zeile für Sie nicht vorhanden wäre.

Wenn Sie geschützte Zeilen lesen, werden nur solche Zeilen zurückgegeben, auf die Ihre LBAC-Berechtigungsnachweise einen Lesezugriff zulassen. Dies gilt auch dann, wenn die Spalte des Typs DB2SECURITYLABEL nicht in die SELECT-Klausel eingeschlossen ist.

Abhängig von den LBAC-Berechtigungsnachweisen werden verschiedenen Benutzern möglicherweise verschiedene Zeilen aus einer Tabelle angezeigt, die geschützte Zeilen enthält. Zum Beispiel empfangen zwei Benutzer, die die Anweisung `SELECT COUNT(*) FROM T1` ausführen, vielleicht unterschiedliche Ergebnisse, wenn die Tabelle T1 geschützte Spalten enthält und die Benutzer über unterschiedliche LBAC-Berechtigungsnachweise verfügen.

Ihre LBAC-Berechtigungsnachweise gelten nicht nur für SELECT-Anweisungen, sondern auch für andere SQL-Anweisungen wie UPDATE und DELETE. Wenn Sie keine LBAC-Berechtigungsnachweise besitzen, die Ihnen einen Lesezugriff auf eine Zeile erlauben, können Sie diese Zeile nicht bearbeiten.

Beispiel:

Die Tabelle 1 enthält die folgenden Zeilen und Spalten. Die Spalte ROWSECURITYLABEL hat den Datentyp DB2SECURITYLABEL.

Tabelle 16. Beispielwerte in Tabelle T1

LASTNAME	DEPTNO	ROWSECURITYLABEL
Rjaibi	55	L2
Miller	77	L1
Fielding	11	L3
Bird	55	L2

Nehmen Sie an, dass der Benutzer Dan über LBAC-Berechtigungsnachweise verfügt, die ihm einen Lesezugriff auf Daten erlauben, die durch den Sicherheitskennsatz L1 geschützt werden, jedoch keinen Lesezugriff auf Daten, die durch die Sicherheitskennsätze L2 oder L3 geschützt werden.

Dan setzt die folgende SQL-Anweisung ab:

```
SELECT * FROM T1
```

Die SELECT-Anweisung liefert nur die Zeile für Miller. Es werden keine Nachrichten oder Warnungen zurückgegeben.

Dan erhält die folgende Sicht auf die Tabelle T1:

Tabelle 17. Beispielwerte in der Sicht von Tabelle T1

LASTNAME	DEPTNO	ROWSECURITYLABEL
Miller	77	L1

Die Zeilen für Rjaibi, Fielding und Bird werden nicht zurückgegeben, weil der Lesezugriff durch die zugehörigen Sicherheitskennsätze blockiert wird. Dan kann diese Zeile nicht löschen oder aktualisieren. Diese Zeile werden darüber hinaus auch aus allen Spaltenfunktionen ausgeklammert. Für Dan verhält es sich so, als ob diese Zeilen nicht vorhanden wären.

Dan setzt die folgende SQL-Anweisung ab:

```
SELECT COUNT(*) FROM T1
```

Die Anweisung liefert den Wert 1, da nur die Zeile für Miller durch den Benutzer Dan gelesen werden kann.

Lesen geschützter Zeilen, die geschützte Spalten enthalten

Der Spaltenzugriff wird vor dem Zeilenzugriff überprüft. Wenn Ihre LBAC-Berechtigungsnachweise für den Lesezugriff durch den Sicherheitskennsatz blockiert werden, der eine der Spalten schützt, die Sie auswählen, schlägt die Ausführung der gesamten Anweisung fehl. Ist dies nicht der Fall, wird die Ausführung der Anweisung fortgesetzt, wobei nur die Zeilen zurückgegeben werden, die durch Sicherheitskennsätze geschützt werden, die Ihren LBAC-Berechtigungsnachweisen einen Lesezugriff auf die Zeilen gestatten.

Beispiel

Die Spalte LASTNAME der Tabelle T1 wird durch den Sicherheitskennsatz L1 geschützt. Die Spalte DEPTNO wird durch den Sicherheitskennsatz L2 geschützt. Die Spalte ROWSECURITYLABEL hat den Datentyp DB2SECURITYLABEL. Die Tabelle T1 sieht einschließlich Daten wie folgt aus:

Tabelle 18. Beispielwerte in Tabelle T1

LASTNAME *geschützt durch L1*	**DEPTNO** *geschützt durch L2*	**ROWSECURITYLABEL**
Rjaibi	55	L2
Miller	77	L1
Fielding	11	L3

Nehmen Sie an, dass die Benutzerin Sakari über LBAC-Berechtigungsnachweise verfügt, die ihr einen Lesezugriff auf Daten erlauben, die durch den Sicherheitskennsatz L1 geschützt werden, jedoch keinen Lesezugriff auf Daten, die durch die Sicherheitskennsätze L2 oder L3 geschützt werden.

Sakari setzt die folgende SQL-Anweisung ab:

```
SELECT * FROM T1
```

Die Ausführung der Anweisung schlägt fehl, weil in der SELECT-Klausel das Platzhalterzeichen (*) verwendet wird, das auch die Spalte DEPTNO mit einschließt. Die Spalte DEPTNO wird durch den Sicherheitskennsatz L2 geschützt, auf den die LBAC-Berechtigungsnachweise von Sakari keinen Lesezugriff zulassen.

Sakari setzt als Nächstes die folgende SQL-Anweisung ab:

```
SELECT LASTNAME, ROWSECURITYLABEL FROM T1
```

Die SELECT-Klausel schließt keine Spalten mit ein, die von Sakari nicht gelesen werden können, sodass die Ausführung der Anweisung fortgesetzt wird. Es wird jedoch nur eine Zeile zurückgegeben, da jede der anderen Zeilen durch den Sicherheitskennsatz L2 bzw. L3 geschützt wird.

Tabelle 19. Beispielausgabe der Abfrage auf Tabelle T1

LASTNAME	**ROWSECURITYLABEL**
Miller	L1

Einfügen von LBAC-geschützten Daten

Wenn Sie versuchen, Daten in eine geschützte Spalte einzufügen oder eine neue Zeile in eine Tabelle mit geschützten Zeilen, entscheiden Ihre LBAC-Berechtigungsnachweise, wie die Anweisung INSERT verarbeitet wird.

Einfügen in geschützte Spalten

Wenn Sie versuchen, Daten in eine geschützte Spalte einzufügen, werden Ihre LBAC-Berechtigungsnachweise für den Schreibzugriff mit dem Sicherheitskennsatz verglichen, der diese Spalte schützt. Abhängig vom Ergebnis dieses Vergleichs wird der Zugriff entweder blockiert oder zugelassen.

Detaillierte Informationen dazu, wie zwei Sicherheitskennsätze verglichen werden, finden Sie in dem Abschnitt über den Vergleich von LBAC-Sicherheitskennsätzen.

Wenn der Zugriff zugelassen wird, wird die Ausführung der Anweisung normal fortgesetzt. Wenn der Zugriff blockiert wird, schlägt das Einfügen fehl und ein Fehler wird zurückgegeben.

Wenn Sie eine Zeile einfügen, jedoch keinen Wert für eine geschützte Spalte angeben, wird der Standardwert, sofern verfügbar, eingefügt. Dies geschieht auch dann, wenn Ihre LBAC-Berechtigungsnachweise keinen Schreibzugriff auf diese Spalte zulassen. Ein Standardwert ist in folgenden Fällen verfügbar:

- Die Spalte wurde mit der Option WITH DEFAULT deklariert.
- Die Spalte ist eine generierte Spalte.
- Die Spalte besitzt einen Standardwert, der durch einen Vortrigger eingefügt wird.
- Die Spalte hat den Datentyp DB2SECURITYLABEL, wobei in diesem Fall der Standardwert der Sicherheitskennsatz ist, den Sie für den Schreibzugriff besitzen.

Einfügen in geschützte Zeilen

Wenn Sie eine neue Zeile in eine Tabelle mit geschützten Zeilen einfügen, brauchen Sie keinen Wert für die Spalte des Typs DB2SECURITYLABEL anzugeben. Wenn Sie keinen Wert für diese Spalte angeben, wird die Spalte automatisch mit dem Sicherheitskennsatz gefüllt, der Ihnen für den Schreibzugriff erteilt wurde. Wenn Ihnen kein Sicherheitskennsatz für den Schreibzugriff erteilt wurde, wird ein Fehler zurückgegeben und die Einfügung schlägt fehl.

Durch die Verwendung solcher integrierter Funktionen wie SECLABEL können Sie explizit einen Sicherheitskennsatz angeben, der in eine Spalte des Typs DB2SECURITYLABEL einzufügen ist. Der angegebene Sicherheitskennsatz wird jedoch nur verwendet, wenn Ihre LBAC-Berechtigungsnachweise einen Schreibzugriff auf die Daten zulassen würden, die durch den Sicherheitskennsatz geschützt werden, den Sie einzufügen versuchen.

Wenn Sie einen Sicherheitskennsatz angeben, der Ihnen keinen Schreibzugriff gewähren würde, hängt die weitere Verarbeitung von der Sicherheitsrichtlinie ab, welche die Tabelle schützt. Wenn die Sicherheitsrichtlinie die Option RESTRICT NOT AUTHORIZED WRITE SECURITY LABEL enthält, schlägt die Einfügung fehl und es wird ein Fehler zurückgegeben. Wenn die Sicherheitsrichtlinie die Option RESTRICT NOT AUTHORIZED WRITE SECURITY LABEL nicht enthält oder stattdessen die Option OVERRIDE NOT AUTHORIZED WRITE SECURITY LABEL enthält, wird der Sicherheitskennsatz, den Sie angeben, ignoriert und der Sicherheitskennsatz für den Schreibzugriff verwendet, sofern Sie einen haben. Wenn Sie keinen Sicherheitskennsatz für den Schreibzugriff haben, wird ein Fehler zurückgegeben.

Beispiele

Die Tabelle T1 wird durch eine Sicherheitsrichtlinie mit dem Namen P1 geschützt, die ohne die Option RESTRICT NOT AUTHORIZED WRITE SECURITY LABEL erstellt wurde. Die Tabelle T1 enthält zwei Spalten, jedoch keine Zeilen. Die Spalten haben die Bezeichnungen LASTNAME und LABEL. Die Spalte LABEL hat den Datentyp DB2SECURITYLABEL.

Der Benutzer Joe besitzt den Sicherheitskennsatz L2 für den Schreibzugriff. Nehmen Sie an, dass ihm der Sicherheitskennsatz L2 den Schreibzugriff auf Daten erlaubt, die durch den Sicherheitskennsatz L2 geschützt werden, jedoch keinen Schreibzugriff auf Daten, die durch die Sicherheitskennsätze L1 oder L3 geschützt werden.

Joe setzt die folgende SQL-Anweisung ab:

```
INSERT INTO T1 (LASTNAME, DEPTNO) VALUES ('Rjaibi', 11)
```

Weil kein Sicherheitskennsatz in der INSERT-Anweisung angegeben wurde, wird Joes Sicherheitskennsatz für den Schreibzugriff in die Spalte LABEL der Zeile eingefügt.

Die Tabelle T1 sieht nun folgendermaßen aus:

Tabelle 20. Werte in der Beispieltabelle T1 nach der ersten INSERT-Anweisung

LASTNAME	LABEL
Rjaibi	L2

Joe setzt die folgende SQL-Anweisung ab, in der er den Sicherheitskennsatz explizit angibt, der in die Spalte LABEL einzufügen ist:

```
INSERT INTO T1 VALUES ('Miller', SECLABEL_BY_NAME('P1', 'L1') )
```

Die Funktion SECLABEL_BY_NAME in der Anweisung gibt einen Sicherheitskennsatz zurück, der Teil der Sicherheitsrichtlinie P1 ist und den Namen L1 hat. Joe ist jedoch nicht berechtigt, in Daten zu schreiben, die durch den Sicherheitskennsatz L1 geschützt werden. Daher darf er den Wert L1 nicht in die Spalte LABEL einfügen.

Da die Sicherheitsrichtlinie, welche die Tabelle T1 schützt, ohne die Option RESTRICT NOT AUTHORIZED WRITE SECURITY LABEL erstellt wurde, wird stattdessen der Sicherheitskennsatz eingefügt, den Joe für den Schreibzugriff besitzt. Es wird weder ein Fehler noch eine Nachricht zurückgegeben.

Die Tabelle sieht nun folgendermaßen aus:

Tabelle 21. Werte in der Beispieltabelle T1 nach der zweiten INSERT-Anweisung

LASTNAME	LABEL
Rjaibi	L2
Miller	L2

Wenn die Sicherheitsrichtlinie, welche die Tabelle schützt, mit der Option RESTRICT NOT AUTHORIZED WRITE SECURITY LABEL erstellt worden wäre, wäre die Einfügung unter Rückgabe eines Fehlers fehlgeschlagen.

Als Nächstes wird Joe eine Freistellung für eine der LBAC-Regeln erteilt. Nehmen Sie an, dass seine neuen LBAC-Berechtigungsnachweise ihm nun einen Schreibzugriff auf Daten gestatten, die durch die Sicherheitskennsätze L1 und L2 geschützt werden. Der Sicherheitskennsatz, der Joe für den Schreibzugriff erteilt ist, bleibt unverändert L2.

Joe setzt die folgende SQL-Anweisung ab:

```
INSERT INTO T1 VALUES ('Bird', SECLABEL_BY_NAME('P1', 'L1') )
```

Aufgrund der neuen LBAC-Berechtigungsnachweise hat Joe nun Schreibzugriff auf Daten, die durch den Sicherheitskennsatz L1 geschützt werden. Die Einfügung des Werts L1 wird daher zugelassen. Die Tabelle sieht nun folgendermaßen aus:

Tabelle 22. Werte in der Beispieltabelle T1 nach der dritten INSERT-Anweisung

LASTNAME	LABEL
Rjaibi	L2
Miller	L2
Bird	L1

Aktualisieren von LBAC-geschützten Daten

Ihre LBAC-Berechtigungsnachweise müssen Ihnen Schreibzugriff auf Daten erlauben, damit Sie sie aktualisieren können. Für die Aktualisierung einer geschützten Zeile müssen Ihnen Ihre LBAC-Berechtigungsnachweise außerdem den Lesezugriff auf die Zeile erlauben.

Aktualisieren geschützter Spalten

Wenn Sie versuchen, Daten in einer geschützten Spalte zu aktualisieren, werden Ihre LBAC-Berechtigungsnachweise mit dem Sicherheitskennsatz verglichen, der die Spalte schützt. Der Vergleich, der vorgenommen wird, bezieht sich auf den Schreibzugriff. Wenn der Schreibzugriff blockiert wird, wird ein Fehler zurückgegeben und die Ausführung der Anweisung schlägt fehl. Anderenfalls wird die Aktualisierung fortgesetzt.

Detaillierte Informationen dazu, wie Ihre LBAC-Berechtigungsnachweise mit einem Sicherheitskennsatz verglichen werden, finden Sie in dem Abschnitt über den Vergleich von LBAC-Sicherheitskennsätzen.

Beispiel:

Nehmen Sie an, in einer Tabelle T1 werden die Spalte DEPTNO durch einen Sicherheitskennsatz L2 und die Spalte PAYSCALE durch einen Sicherheitskennsatz L3 geschützt. Die Tabelle T1 sieht einschließlich ihrer Daten wie folgt aus:

Tabelle 23. Tabelle T1

EMPNO	LASTNAME	DEPTNO *geschützt durch* L2	PAYSCALE *geschützt durch* L3
1	Rjaibi	11	4
2	Miller	11	7
3	Bird	11	9

Der Benutzer Lhakpa besitzt keine LBAC-Berechtigungsnachweise. Er setzt die folgende SQL-Anweisung ab:

```
UPDATE T1 SET EMPNO = 4
   WHERE LASTNAME = "Bird"
```

Diese Anweisung wird ohne Fehler ausgeführt, da sie keine geschützten Spalten aktualisiert. Die Tabelle T1 sieht nun folgendermaßen aus:

Tabelle 24. Tabelle T1 nach der Aktualisierung

EMPNO	**LASTNAME**	**DEPTNO** *geschützt durch* **L2**	**PAYSCALE** *geschützt durch* **L3**
1	Rjaibi	11	4
2	Miller	11	7
4	Bird	11	9

Als Nächstes setzt Lhakpa die folgende SQL-Anweisung ab:

```
UPDATE T1 SET DEPTNO = 55
   WHERE LASTNAME = "Miller"
```

Die Ausführung dieser Anweisung schlägt unter Rückgabe eines Fehlers fehl, weil die Spalte DEPTNO geschützt wird und der Benutzer Lhakpa keine LBAC-Berechtigungsnachweise besitzt.

Nehmen Sie an, dem Benutzer Lhakpa werden LBAC-Berechtigungsnachweise erteilt, die ihm die in der folgenden Tabelle zusammengefassten Zugriffsmöglichkeiten gewähren. Wie diese Berechtigungsnachweise im Einzelnen aussehen und welche Elemente sich in den Sicherheitskennsätzen befinden, spielt für dieses Beispiel keine wichtige Rolle.

Schützender Sicherheitskennsatz der Daten	**Lesezugriff?**	**Schreibzugriff?**
L2	Nein	Ja
L3	Nein	Nein

Lhakpa setzt die folgende SQL-Anweisung erneut ab:

```
UPDATE T1 SET DEPTNO = 55
   WHERE LASTNAME = "Miller"
```

Diesmal wird die Anweisung ohne Fehler ausgeführt, weil die LBAC-Berechtigungsnachweise des Benutzers Lhakpa ihm einen Schreibzugriff auf Daten erlauben, die durch den Sicherheitskennsatz geschützt werden, der die Spalte DEPTNO schützt. Dabei spielt es keine Rolle, dass er nicht berechtigt ist, Daten eben dieser Spalte zu lesen. Die Daten der Tabelle T1 sehen nun folgendermaßen aus:

Tabelle 25. Tabelle T1 nach der zweiten Aktualisierung

EMPNO	**LASTNAME**	**DEPTNO** *geschützt durch* **L2**	**PAYSCALE** *geschützt durch* **L3**
1	Rjaibi	11	4
2	Miller	55	7
4	Bird	11	9

Als Nächstes setzt Lhakpa die folgende SQL-Anweisung ab:

```
UPDATE T1 SET DEPTNO = 55, PAYSCALE = 4
   WHERE LASTNAME = "Bird"
```

Die Spalte PAYSCALE wird durch den Sicherheitskennsatz L3 geschützt, und die LBAC-Berechtigungsnachweise des Benutzers Lhakpa erlauben ihm keinen Schreibzugriff auf sie. Da Lhakpa nicht in der Lage, Daten in die Spalte zu schreiben, schlägt die Aktualisierung fehl, sodass keine Daten geändert werden.

Aktualisieren geschützter Zeilen

Wenn Ihre LBAC-Berechtigungsnachweise keinen Lesezugriff auf eine Zeile zulassen, dann verhält sich das System so, als wäre diese Zeile für Sie nicht vorhanden. Daher haben Sie auch keine Möglichkeit, diese Zeile zu aktualisieren. Für Zeilen, die Sie lesen können, benötigen Sie zusätzlich einen Schreibzugriff, um die Zeile aktualisieren zu können.

Wenn Sie versuchen, eine Zeile zu aktualisieren, werden Ihre LBAC-Berechtigungsnachweise für den Schreibzugriff mit dem Sicherheitskennsatz verglichen, der die Zeile schützt. Wenn der Schreibzugriff blockiert wird, schlägt die Aktualisierung fehl und ein Fehler wird zurückgegeben. Wenn der Schreibzugriff nicht blockiert wird, wird die Aktualisierung fortgesetzt.

Die Aktualisierung wird abgesehen von der Behandlung der Spalte mit dem Datentyp DB2SECURITYLABEL auf die gleiche Weise wie eine Aktualisierung an einer nicht geschützten Zeile ausgeführt. Wenn Sie den Wert für diese Spalte nicht explizit festlegen, wird er automatisch auf den Sicherheitskennsatz gesetzt, den Sie für den Schreibzugriff besitzen. Wenn Sie keinen Sicherheitskennsatz für den Schreibzugriff haben, wird ein Fehler zurückgegeben und die Ausführung der Anweisung schlägt fehl.

Wenn die Aktualisierung explizit einen Wert für die Spalte mit dem Datentyp DB2SECURITYLABEL angibt, werden Ihre LBAC-Berechtigungsnachweise erneut überprüft. Wenn die Aktualisierung, die Sie auszuführen versuchen, eine Zeile erstellen würde, auf die Ihre LBAC-Berechtigungsnachweise keinen Schreibzugriff zuließen, hängt die weitere Verarbeitung von der Sicherheitsrichtlinie ab, welche die Tabelle schützt. Wenn die Sicherheitsrichtlinie die Option RESTRICT NOT AUTHORIZED WRITE SECURITY LABEL enthält, schlägt die Aktualisierung fehl und es wird ein Fehler zurückgegeben. Wenn die Sicherheitsrichtlinie die Option RESTRICT NOT AUTHORIZED WRITE SECURITY LABEL nicht enthält oder stattdessen die Option OVERRIDE NOT AUTHORIZED WRITE SECURITY LABEL enthält, wird der Sicherheitskennsatz, den Sie angeben, ignoriert und der Sicherheitskennsatz für den Schreibzugriff verwendet, sofern Sie einen haben. Wenn Sie keinen Sicherheitskennsatz für den Schreibzugriff haben, wird ein Fehler zurückgegeben.

Beispiel:

Nehmen Sie an, die Tabelle T1 wird durch eine Sicherheitsrichtlinie mit dem Namen P1 geschützt und enthält eine Spalte mit dem Namen LABEL, die den Datentyp DB2SECURITYLABEL hat.

Die Tabelle T1 sieht einschließlich ihrer Daten wie folgt aus:

Tabelle 26. Tabelle T1

EMPNO	LASTNAME	DEPTNO	LABEL
1	Rjaibi	11	L1
2	Miller	11	L2
3	Bird	11	L3

Nehmen Sie an, die Benutzerin Jenni besitzt LBAC-Berechtigungsnachweise, die ihr einen Lese- und einen Schreibzugriff auf Daten erlauben, die durch die Sicherheitskennsätze L0 und L1 geschützt werden, jedoch nicht auf Daten, die durch beliebige andere Sicherheitskennsätze geschützt werden. Der Sicherheitskennsatz, den Sie für den Schreib- und den Lesezugriff besitzt, ist jeweils L0. Die vollständigen Details ihrer Berechtigungsnachweise sowie die einzelnen Elemente, die in den Kennsätzen enthalten sind, spielen für dieses Beispiel keine wichtige Rolle.

Jenni setzt die folgende SQL-Anweisung ab:

```
SELECT * FROM T1
```

Jenni wird nur eine Zeile aus der Tabelle angezeigt:

Tabelle 27. Ergebnis der SELECT-Abfrage von Jenni

EMPNO	LASTNAME	DEPTNO	LABEL
1	Rjaibi	11	L1

Die Zeilen, die durch die Kennsätze L2 und L3 geschützt werden, gehören nicht zur Ergebnismenge, weil Jennis LBAC-Berechtigungsnachweise keinen Lesezugriff auf diese Zeilen zulassen. Für Jenni verhält es sich so, als ob diese Zeilen nicht vorhanden wären.

Jenni setzt die folgenden SQL-Anweisungen ab:

```
UPDATE T1 SET DEPTNO = 44 WHERE DEPTNO = 11;
SELECT * FROM T1;
```

Die von der Abfrage zurückgegebene Ergebnismenge sieht folgendermaßen aus:

Tabelle 28. Ergebnis der UPDATE-Anweisung und der SELECT-Abfrage von Jenni

EMPNO	LASTNAME	DEPTNO	LABEL
1	Rjaibi	44	L0

Tatsächlich sehen die vollständigen Daten der Tabelle wie folgt aus:

Tabelle 29. Tabelle T1

EMPNO	LASTNAME	DEPTNO	LABEL
1	Rjaibi	44	L0
2	Miller	11	L2
3	Bird	11	L3

Die Anweisung wurde ohne Fehler ausgeführt, jedoch hat sie nur die erste Zeile geändert. Die zweite und dritte Zeile sind für Jenni nicht lesbar, sodass sie nicht für die Aktualisierung durch die Anweisung ausgewählt wurden, obwohl sie die Bedingung der WHERE-Klausel erfüllen.

Beachten Sie, dass der Wert der Spalte LABEL in der aktualisierten Zeile geändert wurde, obwohl diese Spalte in der UPDATE-Anweisung nicht explizit angegeben war. Die Spalte wurde auf den Sicherheitskennsatz gesetzt, den Jenni für den Schreibzugriff besaß.

Nehmen Sie jetzt an, dass Jenni LBAC-Berechtigungsnachweise für den Lesezugriff auf Daten erteilt werden, die durch einen beliebigen Sicherheitskennsatz geschützt werden. Ihre LBAC-Berechtigungsnachweise für den Schreibzugriff werden nicht geändert. Sie kann auch weiterhin nur Daten schreiben, die durch die Sicherheitskennsätze L0 und L1 geschützt werden.

Jenni setzt die folgende SQL-Anweisung erneut ab:

```
UPDATE T1 SET DEPTNO = 44 WHERE DEPTNO = 11
```

Diesmal schlägt die Aktualisierung wegen der zweiten und der dritten Zeile fehl. Jenni kann diese Zeilen lesen, sodass sie für die Aktualisierung durch die Anweisung ausgewählt werden. Sie ist jedoch nicht berechtigt, in diese Zeilen zu schreiben, weil sie durch die Sicherheitskennsätze L2 und L3 geschützt werden. Die Aktualisierung findet nicht statt, und es wird ein Fehler zurückgegeben.

Jenni setzt nun die folgende SQL-Anweisung ab:

```
UPDATE T1
 SET DEPTNO = 55, LABEL = SECLABEL_BY_NAME( 'P1', 'L2' )
 WHERE LASTNAME = "Rjaibi"
```

Die Funktion SECLABEL_BY_NAME in der Anweisung gibt den Sicherheitskennsatz mit dem Namen L2 zurück. Jenni versucht, den Sicherheitskennsatz, der die erste Zeile schützt, explizit festzulegen. Jennis LBAC-Berechtigungsnachweise erlauben ihr, die erste Zeile zu lesen, sodass diese für die Aktualisierung ausgewählt wird. Ihre LBAC-Berechtigungsnachweise erlauben ihr einen Schreibzugriff auf Zeilen, die durch den Sicherheitskennsatz L0 geschützt werden, sodass sie die Zeile aktualisieren darf. Allerdings würden ihre LBAC-Berechtigungsnachweise ihr keinen Schreibzugriff auf eine Zeile erlauben, die durch den Sicherheitskennsatz L2 geschützt wird. Daher ist sie nicht berechtigt, die Spalte LABEL auf diesen Wert zu setzen. Die Ausführung der Anweisung schlägt unter Rückgabe eines Fehlers fehl. Es werden keine Spalten in der Zeile aktualisiert.

Jenni setzt nun die folgende SQL-Anweisung ab:

```
UPDATE T1 SET LABEL = SECLABEL_BY_NAME( 'P1', 'L1' ) WHERE LASTNAME = "Rjaibi"
```

Die Anweisung wird erfolgreich ausgeführt, weil sie in eine Zeile schreiben könnte, die durch den Sicherheitskennsatz L1 geschützt wird.

Die Tabelle T1 sieht nun folgendermaßen aus:

Tabelle 30. Tabelle T1

EMPNO	LASTNAME	DEPTNO	LABEL
1	Rjaibi	44	L1
2	Miller	11	L2
3	Bird	11	L3

Aktualisieren geschützter Zeilen, die geschützte Spalten enthalten

Wenn Sie versuchen, geschützte Spalten in einer Tabelle mit geschützten Zeilen zu aktualisieren, müssen Ihre LBAC-Berechtigungsnachweise einen Schreibzugriff auf alle geschützten Spalten zulassen, die von der Aktualisierung betroffen sind. Ansonsten schlägt die Aktualisierung fehl und ein Fehler wird zurückgegeben. Dieser Sachverhalt wird im vorausgehenden Abschnitt über das Aktualisieren geschützter

Spalten beschrieben. Wenn Sie berechtigt sind, alle geschützten Spalten, die von der Aktualisierung betroffen sind, zu aktualisieren, können Sie trotzdem nur die Zeilen aktualisieren, auf die Ihnen Ihre LBAC-Berechtigungsnachweise sowohl einen Lesezugriff als auch einen Schreibzugriff ermöglichen. Dieser Sachverhalt wird im vorausgehenden Abschnitt über das Aktualisieren geschützter Zeilen beschrieben. Die Behandlung einer Spalte mit dem Datentyp DB2SECURITYLABEL bleibt gleich, unabhängig davon, ob die Aktualisierung geschützte Spalten betrifft oder nicht.

Wenn die Spalte, die den Datentyp DB2SECURITYLABEL hat, selbst eine geschützte Spalte ist, müssen Ihre LBAC-Berechtigungsnachweise einen Schreibzugriff auf diese Spalte zulassen. Ansonsten können Sie keine der Zeilen in der Tabelle aktualisieren.

Löschen oder Entfernen von LBAC-geschützten Daten

Ihre LBAC-Berechtigungsnachweise entscheiden darüber, ob Sie Daten in LBAC-geschützten Tabellen löschen können.

Löschen geschützter Zeilen

Wenn Ihre LBAC-Berechtigungsnachweise keinen Lesezugriff auf eine Zeile zulassen, dann verhält sich das System so, als wäre diese Zeile für Sie nicht vorhanden. Daher haben Sie auch keine Möglichkeit, sie zu löschen. Zum Löschen einer Zeile, die Sie lesen können, müssen Ihnen Ihre LBAC-Berechtigungsnachweise auch den Schreibzugriff auf diese Zeile erlauben. Zum Löschen einer Zeile in einer Tabelle, die geschützte Spalten enthält, müssen Sie über LBAC-Berechtigungsnachweise verfügen, die Ihnen einen Schreibzugriff auf alle geschützten Spalten in der Tabelle erlauben.

Wenn Sie versuchen, eine Zeile zu löschen, werden Ihre LBAC-Berechtigungsnachweise für den Schreibzugriff mit dem Sicherheitskennsatz verglichen, der die Zeile schützt. Wenn der schützende Sicherheitskennsatz den Schreibzugriff durch Ihre LBAC-Berechtigungsnachweise blockiert, schlägt die Ausführung der DELETE-Anweisung fehl. Darüber hinaus wird ein Fehler zurückgegeben, und keine Zeile wird gelöscht.

Beispiel

Die geschützte Tabelle T1 enthält folgende Zeilen:

LASTNAME	DEPTNO	LABEL
Rjaibi	55	L2
Miller	77	L1
Bird	55	L2
Fielding	77	L3

Nehmen Sie an, die Benutzerin Pat besitzt LBAC-Berechtigungsnachweise, die ihr den in der folgenden Tabelle dargestellten Zugriff erlauben:

Sicherheitskennsatz	Lesezugriff?	Schreibzugriff?
L1	Ja	Ja
L2	Ja	Nein
L3	Nein	Nein

Die exakten Details ihrer LBAC-Berechtigungsnachweise und der Sicherheitskennsätze sind für dieses Beispiel nicht relevant.

Pat setzt die folgende SQL-Anweisung ab:

```
SELECT * FROM T1 WHERE DEPTNO != 999
```

Die Anweisung wird ausgeführt und liefert die folgende Ergebnismenge:

LASTNAME	DEPTNO	LABEL
Rjaibi	55	L2
Miller	77	L1
Bird	55	L2

Die letzte Zeile von T1 ist nicht in den Ergebnissen enthalten, weil Pat keinen Lesezugriff auf diese Zeile besitzt. Es verhält sich so, als ob diese Zeile für Pat nicht vorhanden wäre.

Pat setzt die folgende SQL-Anweisung ab:

```
DELETE FROM T1 WHERE DEPTNO != 999
```

Pat hat keinen Schreibzugriff auf die erste und die dritte Zeile, die beide durch den Sicherheitskennsatz L2 geschützt werden. Obwohl sie also die Zeilen lesen kann, ist sie nicht berechtigt, sie zu löschen. Die Ausführung der DELETE-Anweisung schlägt fehl, und keine Zeilen werden gelöscht.

Pat setzt die folgende SQL-Anweisung ab:

```
DELETE FROM T1 WHERE DEPTNO = 77;
```

Diese Anweisung wird erfolgreich ausgeführt, weil Pat Schreibzugriff auf die Zeile mit dem Wert 'Miller' in der Spalte LASTNAME hat. Dies ist die einzige Zeile, die durch die Anweisung ausgewählt wird. Die Zeile mit dem Wert 'Fielding' in der Spalte LASTNAME wird nicht ausgewählt, weil Pats LBAC-Berechtigungsnachweise keinen Lesezugriff auf diese Zeile zulassen. Diese Zeile wird für die Löschung nie in Betracht gezogen, sodass kein Fehler auftritt.

Die tatsächlichen Zeilen der Tabelle sehen nun folgendermaßen aus:

LASTNAME	DEPTNO	LABEL
Rjaibi	55	L2
Bird	55	L2
Fielding	77	L3

Löschen von Zeilen, die geschützte Spalten enthalten

Zum Löschen einer Zeile in einer Tabelle, die geschützte Spalten enthält, müssen Sie über LBAC-Berechtigungsnachweise verfügen, die Ihnen einen Schreibzugriff auf alle geschützten Spalten in der Tabelle erlauben. Wenn sich in der Tabelle eine Zeile befindet, auf die Ihnen Ihre LBAC-Berechtigungsnachweise keinen Schreibzugriff gewähren, schlägt das Löschen fehl und es wird ein Fehler zurückgegeben.

Wenn die Tabelle sowohl geschützte Spalten als auch geschützte Zeilen enthält, müssen Sie zum Löschen einer bestimmten Zeile über LBAC-Berechtigungsnach-

weise verfügen, die Ihnen einen Schreibzugriff auf jede geschützte Spalte in der Tabelle und außerdem einen Lese- und Schreibzugriff auf die Zeile erlauben, die Sie löschen möchten.

Beispiel

In der geschützten Tabelle T1 wird die Spalte DEPTNO durch den Sicherheitskennsatz L2 geschützt. Die Tabelle T1 enthält die folgenden Zeilen:

LASTNAME	**DEPTNO** *geschützt durch* **L2**	**LABEL**
Rjaibi	55	L2
Miller	77	L1
Bird	55	L2
Fielding	77	L3

Nehmen Sie an, der Benutzer Benny verfügt über LBAC-Berechtigungsnachweise, die ihm den in der folgenden Tabelle dargestellten Zugriff erlauben:

Sicherheitskennsatz	**Lesezugriff?**	**Schreibzugriff?**
L1	Ja	Ja
L2	Ja	Nein
L3	Nein	Nein

Die exakten Details seiner LBAC-Berechtigungsnachweise und der Sicherheitskennsätze sind für dieses Beispiel nicht relevant.

Benny setzt die folgende SQL-Anweisung ab:

```
DELETE FROM T1 WHERE DEPTNO = 77
```

Die Ausführung der Anweisung schlägt fehl, weil Benny keinen Schreibzugriff auf die Spalte DEPTNO besitzt.

Nun werden LBAC-Berechtigungsnachweise des Benutzers Benny geändert, sodass er über den in der folgenden Tabelle dargestellten Zugriff verfügt:

Sicherheitskennsatz	**Lesezugriff?**	**Schreibzugriff?**
L1	Ja	Ja
L2	Ja	Ja
L3	Ja	Nein

Benny setzt die folgende SQL-Anweisung erneut ab:

```
DELETE FROM T1 WHERE DEPTNO = 77
```

Diesmal hat Benny Schreibzugriff auf die Spalte DEPTNO, sodass die Ausführung der DELETE-Anweisung fortgesetzt wird. Die DELETE-Anweisung wählt nur die Zeile aus, die den Wert 'Miller' in der Spalte LASTNAME enthält. Die Zeile mit dem Wert 'Fielding' in der Spalte LASTNAME wird nicht ausgewählt, weil Bennys LBAC-Berechtigungsnachweise keinen

Lesezugriff auf diese Zeile zulassen. Da die Zeile von der Anweisung zum Löschen nicht ausgewählt wird, spielt es keine Rolle, dass Benny keinen Schreibzugriff auf die Zeile besitzt.

Die eine ausgewählte Zeile wird durch den Sicherheitskennsatz L1 geschützt. Bennys LBAC-Berechtigungsnachweise erlauben ihm einen Schreibzugriff auf Daten, die durch den Sicherheitskennsatz L1 geschützt werden, sodass die Löschung erfolgreich ausgeführt wird.

Die tatsächlichen Zeilen in der Tabelle T1 sehen nun folgendermaßen aus:

LASTNAME	**DEPTNO** *geschützt durch* **L2**	**LABEL**
Rjaibi	55	L2
Bird	55	L2
Fielding	77	L3

Löschen geschützter Daten

Sie können eine Spalte, die durch einen Sicherheitskennsatz geschützt wird, nur löschen, wenn Ihre LBAC-Berechtigungsnachweise einen Schreibzugriff auf diese Spalte zulassen.

Eine Spalte mit dem Datentyp DB2SECURITYLABEL kann nicht aus einer Tabelle gelöscht werden. Um diese Spalte zu entfernen, müssen Sie zuerst die Sicherheitsrichtlinie der Tabelle löschen. Wenn Sie die Sicherheitsrichtlinie löschen, ist die Tabelle nicht mehr durch LBAC geschützt und der Datentyp der Spalte wird automatisch von DB2SECURITYLABEL in VARCHAR(128) FOR BIT DATA geändert. Anschließend kann die Spalte gelöscht werden.

Ihre LBAC-Berechtigungsnachweise hindern Sie nicht daran, gesamte Tabellen oder Datenbanken zu löschen, die geschützte Daten enthalten. Wenn Sie unter normalen Umständen die Berechtigung besitzen, eine Tabelle oder Datenbank zu löschen, benötigen Sie dazu keine LBAC-Berechtigungsnachweise, selbst wenn die Datenbank geschützte Daten enthält.

Entfernen des LBAC-Schutzes von Daten

Sie müssen über die Berechtigung SECADM verfügen, um die Sicherheitsrichtlinie von einer Tabelle entfernen zu können. Verwenden Sie die Klausel DROP SECURITY POLICY der Anweisung ALTER TABLE, um die Sicherheitsrichtlinie von einer Tabelle zu entfernen. Dadurch wird automatisch auch der Schutz von allen Zeilen und allen Spalten der Tabelle entfernt.

Entfernen des Schutzes von Zeilen

In einer Tabelle, die geschützte Zeilen enthält, muss jede Zeile durch einen Sicherheitskennsatz geschützt werden. Es gibt keine Möglichkeit, den LBAC-Schutz von einzelnen Zeilen zu entfernen.

Eine Spalte des Typs DB2SECURITYLABEL kann nicht geändert oder entfernt werden, ohne zuvor die Sicherheitsrichtlinie von der Tabelle zu entfernen.

Entfernen des Schutzes von Spalten

Der Schutz einer Spalte kann mithilfe der Klausel DROP COLUMN SECURITY der SQL-Anweisung ALTER TABLE entfernt werden. Um den Schutz von einer Spalte entfernen zu können, müssen Sie neben den normalen Zugriffsrechten und Berechtigungen zum Ändern einer Tabelle auch die LBAC-Berechtigungsnachweise für einen Lese- und Schreibzugriff auf die Spalte besitzen.

Kapitel 6. Verwenden des Systemkatalogs für Sicherheitsinformationen

Informationen über die einzelnen Datenbanken werden automatisch in einer Gruppe von Sichten, dem so genannten Systemkatalog, gepflegt, der beim Erstellen der Datenbank erstellt werden. Dieser Systemkatalog beschreibt Tabellen, Spalten, Indizes, Programme, Zugriffsrechte und andere Objekte.

In den folgenden Sichten und Tabellenfunktionen werden Informationen über Zugriffsrechte für Benutzer, Kennungen für Benutzer, die Zugriffsrechte erteilen sowie Informationen zu Objekteigentumsrechten aufgeführt:

SYSCAT.COLAUTH
Listet die Zugriffsrechte für Spalten auf.

SYSCAT.DBAUTH
Listet die Zugriffsrechte für Datenbanken auf.

SYSCAT.INDEXAUTH
Listet die Zugriffsrechte für Indizes auf.

SYSCAT.MODULEAUTH
Listet die Zugriffsrechte für Module auf.

SYSCAT.PACKAGEAUTH
Listet die Zugriffsrechte für Pakete auf.

SYSCAT.PASSTHRUAUTH
Listet das Zugriffsrecht für Server auf.

SYSCAT.ROLEAUTH
Listet die Zugriffsrechte für Rollen auf.

SYSCAT.ROUTINEAUTH
Listet die Zugriffsrechte für Routinen (Funktionen, Methoden und gespeicherte Prozeduren) auf.

SYSCAT.SCHEMAAUTH
Listet die Zugriffsrechte für Schemata auf.

SYSCAT.SEQUENCEAUTH
Listet die Zugriffsrechte für Sequenzen auf.

SYSCAT.SURROGATEAUTHIDS
Listet die Berechtigungs-IDs auf, für die eine andere Berechtigungs-ID als Ersatz fungieren kann.

SYSCAT.TABAUTH
Listet die Zugriffsrechte für Tabellen und Sichten auf.

SYSCAT.TBSPACEAUTH
Listet die Zugriffsrechte für Tabellenbereiche auf.

SYSCAT.VARIABLEAUTH
Listet die Zugriffsrechte für Variablen auf.

SYSCAT.WORKLOADAUTH
Listet die Zugriffsrechte für Auslastungen auf.

SYSCAT.XSROBJECTAUTH
Listet die Zugriffsrechte für XSR-Objekte auf.

Zugriffsrechte, die den Benutzern vom System verliehen wurden, weisen in der Spalte GRANTOR den Wert SYSIBM auf. SYSADM, SYSMAINT SYSCTRL und SYSMON werden nicht im Systemkatalog aufgelistet.

Durch die Anweisungen CREATE und GRANT werden Zugriffsrechte in den Systemkatalog eingetragen. Benutzer mit den Berechtigungen ACCESSCTRL und SECADM können das Zugriffsrecht SELECT für die Systemkatalogsichten erteilen und entziehen.

Abrufen von Berechtigungsnamen mit erteilten Zugriffsrechten

Über die Verwaltungssicht PRIVILEGES und andere Verwaltungssichten können Sie Informationen über die Berechtigungsnamen abrufen, denen Zugriffsrechte in einer Datenbank erteilt wurden.

Informationen zu diesem Vorgang

Beispiel: Mit der folgenden Abfrage werden alle expliziten Zugriffsrechte und Berechtigungs-IDs, denen sie erteilt wurden, sowie weitere Informationen über die Verwaltungssicht PRIVILEGES abgerufen:

```
SELECT AUTHID, PRIVILEGE, OBJECTNAME, OBJECTSCHEMA, OBJECTTYPE FROM SYSIBMADM.PRIVILEGES
```

In der folgenden Abfrage wird die Verwaltungssicht AUTHORIZATIONIDS verwendet, um nach allen Berechtigungs-IDs zu suchen, denen Zugriffsrechte oder Berechtigungen erteilt wurden, und um die entsprechenden Typen anzuzeigen:

```
SELECT AUTHID, AUTHIDTYPE FROM SYSIBMADM.AUTHORIZATIONIDS
```

Mit der Verwaltungssicht SYSIBMADM.OBJECTOWNERS und der Tabellenfunktion SYSPROC.AUTH_LIST_GROUPS_FOR_AUTHID können Sie nach sicherheitsbezogenen Informationen suchen.

Vor Version 9.1 enthielt keine einzelne Systemkatalogsicht Informationen zu allen Zugriffsrechten. Bei Releases vor Version 9.1 werden mithilfe der folgenden Anweisung sämtliche Berechtigungsnamen mit Zugriffsrechten abgerufen:

```
 SELECT DISTINCT GRANTEE, GRANTEETYPE, 'DATABASE' FROM SYSCAT.DBAUTH
UNION
SELECT DISTINCT GRANTEE, GRANTEETYPE, 'TABLE   ' FROM SYSCAT.TABAUTH
UNION
SELECT DISTINCT GRANTEE, GRANTEETYPE, 'PACKAGE ' FROM SYSCAT.PACKAGEAUTH
UNION
SELECT DISTINCT GRANTEE, GRANTEETYPE, 'INDEX   ' FROM SYSCAT.INDEXAUTH
UNION
SELECT DISTINCT GRANTEE, GRANTEETYPE, 'COLUMN  ' FROM SYSCAT.COLAUTH
UNION
SELECT DISTINCT GRANTEE, GRANTEETYPE, 'SCHEMA  ' FROM SYSCAT.SCHEMAAUTH
UNION
SELECT DISTINCT GRANTEE, GRANTEETYPE, 'SERVER  ' FROM SYSCAT.PASSTHRUAUTH
ORDER BY GRANTEE, GRANTEETYPE, 3
```

Die durch diese Anweisung abgerufene Liste sollte in regelmäßigen Abständen mit Listen von Benutzer- und Gruppennamen verglichen werden, die in der Sicherheitseinrichtung des Systems definiert sind. Auf diese Weise können die Berechtigungsnamen ermittelt werden, die nicht mehr gültig sind.

Anmerkung: Wenn Sie ferne Datenbankclients unterstützen, ist der Berechtigungsname möglicherweise nur auf dem fernen Client und nicht auf Ihrem Datenbankserversystem definiert.

Abrufen aller Namen mit der Berechtigung DBADM

Die folgende Anweisung ruft alle Berechtigungsnamen ab, denen die Berechtigung DBADM direkt erteilt wurde:

Informationen zu diesem Vorgang

```
SELECT DISTINCT GRANTEE, GRANTEETYPE FROM SYSCAT.DBAUTH
   WHERE DBADMAUTH = 'Y'
```

Abrufen der Namen mit Zugriffsberechtigung für eine Tabelle

Über die Verwaltungssicht PRIVILEGES und andere Verwaltungssichten können Sie Informationen über die Berechtigungsnamen abrufen, denen Zugriffsrechte in einer Datenbank erteilt wurden.

Informationen zu diesem Vorgang

Die folgende Anweisung ruft alle Berechtigungsnamen (und deren Typen) ab, die direkt zum Zugriff auf die Tabelle EMPLOYEE mit dem Qualifikationsmerkmal JAMES berechtigt sind:

```
SELECT DISTINCT AUTHID, AUTHIDTYPE FROM SYSIBMADM.PRIVILEGES
    WHERE OBJECTNAME = 'EMPLOYEE' AND OBJECTSCHEMA = 'JAMES'
```

Bei Releases vor Version 9.1 werden mithilfe der folgenden Abfrage dieselben Informationen abgerufen:

```
SELECT DISTINCT GRANTEETYPE, GRANTEE FROM SYSCAT.TABAUTH
   WHERE TABNAME = 'EMPLOYEE'
     AND TABSCHEMA = 'JAMES'
UNION
SELECT DISTINCT GRANTEETYPE, GRANTEE FROM SYSCAT.COLAUTH
   WHERE TABNAME = 'EMPLOYEE'
     AND TABSCHEMA = 'JAMES'
```

Mithilfe der folgenden Anweisung können Sie herausfinden, wer die Tabelle EMPLOYEE mit dem Qualifikationsmerkmal JAMES aktualisieren kann:

```
 SELECT DISTINCT GRANTEETYPE, GRANTEE FROM SYSCAT.TABAUTH
   WHERE TABNAME = 'EMPLOYEE' AND TABSCHEMA = 'JAMES' AND
      (CONTROLAUTH  = 'Y' OR
       UPDATEAUTH IN ('G','Y'))
UNION
SELECT DISTINCT GRANTEETYPE, GRANTEE FROM SYSCAT.DBAUTH
   WHERE DBADMAUTH = 'Y'
UNION
SELECT DISTINCT GRANTEETYPE, GRANTEE FROM SYSCAT.COLAUTH
   WHERE TABNAME = 'EMPLOYEE' AND TABSCHEMA = 'JAMES' AND
   PRIVTYPE = 'U'
```

Diese Anweisung ruft alle Berechtigungsnamen mit der Berechtigung DBADM sowie die Namen ab, denen die Zugriffsrechte CONTROL oder UPDATE direkt erteilt wurden.

Es ist zu beachten, dass einige Berechtigungsnamen Gruppen sein können, nicht nur Einzelbenutzer.

Abrufen aller Benutzern erteilter Zugriffsrechte

Durch Abfragen der Systemkatalogsichten können Benutzer eine Liste der Zugriffsrechte, über die sie verfügen, und eine Liste der Zugriffsrechte, die sie anderen Benutzern erteilt haben, abrufen.

Informationen zu diesem Vorgang

Über die Verwaltungssicht PRIVILEGES und andere Verwaltungssichten können Sie Informationen über die Berechtigungsnamen abrufen, denen Zugriffsrechte in einer Datenbank erteilt wurden. Beispiel: Mit der folgenden Abfrage werden alle Zugriffsrechte abgerufen, die der Berechtigungs-ID der aktuellen Sitzung erteilt wurden:

```
SELECT * FROM SYSIBMADM.PRIVILEGES
 WHERE AUTHID = SESSION_USER AND AUTHIDTYPE = 'U'
```

Das Schlüsselwort SESSION_USER in dieser Anweisung ist ein Sonderregister, das dem Wert des Berechtigungsnamens des aktuellen Benutzers entspricht.

Bei Releases vor Version 9.1 werden mithilfe der Anweisungen in folgenden Beispielen dieselben Informationen bereitgestellt. Zum Beispiel wird durch die folgende Anweisung eine Liste der Datenbankzugriffsrechte abgerufen, die direkt dem einzelnen Berechtigungsnamen JAMES erteilt wurden:

```
SELECT * FROM SYSCAT.DBAUTH
 WHERE GRANTEE = 'JAMES' AND GRANTEETYPE = 'U'
```

Mit der folgenden Anweisung wird eine Liste der Tabellenzugriffsrechte abgerufen, die direkt vom Benutzer JAMES erteilt wurden:

```
SELECT * FROM SYSCAT.TABAUTH
 WHERE GRANTOR  = 'JAMES'
```

Mit der folgenden Anweisung wird eine Liste der einzelnen Spaltenzugriffsrechte abgerufen, die direkt vom Benutzer JAMES erteilt wurden:

```
SELECT * FROM SYSCAT.COLAUTH
   WHERE GRANTOR  = 'JAMES'
```

Schützen der Systemkatalogsicht

Da in den Systemkatalogsichten jedes Objekt in der Datenbank beschrieben wird, möchten Sie möglicherweise, falls Sie über sensible Daten verfügen, den Zugriff auf diese Daten einschränken.

Informationen zu diesem Vorgang

Die folgenden Berechtigungen beinhalten das Zugriffsrecht SELECT für alle Katalogtabellen:

- ACCESSCTRL
- DATAACCESS
- DBADM
- SECADM
- SQLADM

Darüber hinaus beinhalten die folgenden Berechtigungen der Instanzebene die Möglichkeiten Daten in SYSCAT.BUFFERPOOLS, SYSCAT.DBPARTITIONGROUPS, SYSCAT.DBPARTITIONGROUPDEF, SYSCAT.PACKAGES und SYSCAT.TABLES auszuwählen:

- SYSADM
- SYSCTRL
- SYSMAINT
- SYSMON

Sie können den Befehl CREATE DATABASE ... RESTRICTIVE zum Erstellen einer Datenbank verwenden, in der keine Zugriffsrechte automatisch der speziellen Gruppe PUBLIC erteilt werden. In diesem Fall kommt es zu keiner der folgenden normalen Standard-GRANT-Aktionen:

- CREATETAB
- BINDADD
- CONNECT
- IMPLICIT_SCHEMA
- EXECUTE with GRANT für alle Prozeduren im Schema SQLJ
- EXECUTE with GRANT für alle Funktionen und Prozeduren im Schema SYSPROC
- BIND für alle im Schema NULLID erstellten Pakete
- EXECUTE für alle im Schema NULLID erstellten Pakete
- CREATEIN im Schema SQLJ
- CREATEIN im Schema NULLID
- USE für den Tabellenbereich USERSPACE1
- SELECT-Zugriff auf die SYSIBM-Katalogtabellen
- SELECT-Zugriff auf die SYSCAT-Katalogsichten
- SELECT-Zugriff auf die SYSIBMADM-Verwaltungssichten
- SELECT-Zugriff auf die SYSSTAT-Katalogsichten
- UPDATE-Zugriff auf die SYSSTAT-Katalogsichten

Wenn Sie eine Datenbank mit der Option RESTRICTIVE erstellt haben und Sie überprüfen möchten, ob die Berechtigungen eingeschränkt sind, die Sie PUBLIC erteilt haben, können Sie die folgende Abfrage absetzen, um zu prüfen, auf welche Schemata PUBLIC zugreifen kann:

```
SELECT DISTINCT OBJECTSCHEMA FROM SYSIBMADM.PRIVILEGES WHERE AUTHID='PUBLIC'

OBJECTSCHEMA
------------
SYSFUN
SYSIBM
SYSPROC
```

Wenn Sie wissen möchten, welchen Zugriff PUBLIC noch auf SYSIBM hat, können Sie die folgende Abfrage absetzen, um zu überprüfen, welche Zugriffsrechte für SYSIBM erteilt wurden. Die Ergebnisse zeigen, dass nur EXECUTE für bestimmte Prozeduren und Funktionen erteilt wurde.

```
SELECT * FROM SYSIBMADM.PRIVILEGES WHERE OBJECTSCHEMA = 'SYSIBM'

AUTHID       AUTHIDTYPE  PRIVILEGE   GRANTABLE  OBJECTNAME         OBJECTSCHEMA    OBJECTTYPE
---------... ----------  ----------  ---------  ---------------... ------------... ----------
PUBLIC       G           EXECUTE     N          SQL060207192129400 SYSPROC         FUNCTION
PUBLIC       G           EXECUTE     N          SQL060207192129700 SYSPROC         FUNCTION
PUBLIC       G           EXECUTE     N          SQL060207192129701 SYSPROC
```

```
...
PUBLIC      G           EXECUTE     Y          TABLES             SYSIBM          PROCEDURE
PUBLIC      G           EXECUTE     Y          TABLEPRIVILEGES    SYSIBM          PROCEDURE
PUBLIC      G           EXECUTE     Y          STATISTICS         SYSIBM          PROCEDURE
PUBLIC      G           EXECUTE     Y          SPECIALCOLUMNS     SYSIBM          PROCEDURE
PUBLIC      G           EXECUTE     Y          PROCEDURES         SYSIBM          PROCEDURE
PUBLIC      G           EXECUTE     Y          PROCEDURECOLS      SYSIBM          PROCEDURE
PUBLIC      G           EXECUTE     Y          PRIMARYKEYS        SYSIBM          PROCEDURE
PUBLIC      G           EXECUTE     Y          FOREIGNKEYS        SYSIBM          PROCEDURE
PUBLIC      G           EXECUTE     Y          COLUMNS            SYSIBM          PROCEDURE
PUBLIC      G           EXECUTE     Y          COLPRIVILEGES      SYSIBM          PROCEDURE
PUBLIC      G           EXECUTE     Y          UDTS               SYSIBM          PROCEDURE
PUBLIC      G           EXECUTE     Y          GETTYPEINFO        SYSIBM          PROCEDURE
PUBLIC      G           EXECUTE     Y          SQLCAMESSAGE       SYSIBM          PROCEDURE
PUBLIC      G           EXECUTE     Y          SQLCAMESSAGECCSID  SYSIBM          PROCEDURE
```

Anmerkung: Die Verwaltungssicht SYSIBMADM.PRIVILEGES ist ab Version 9.1 des DB2-Datenbankmanagers verfügbar.

Bei Releases vor Version 9.1 des DB2-Datenbankmanagers wird während der Datenbankerstellung das Zugriffsrecht SELECT für die Systemkatalogsichten der Gruppe PUBLIC erteilt. In den meisten Fällen stellt dies kein Sicherheitsproblem dar. Bei sehr sensiblen Daten kann es jedoch zu Problemen kommen, da in diesen Tabellen jedes Objekt der Datenbank beschrieben wird. Wenn dies der Fall ist, kann das Zugriffsrecht SELECT für PUBLIC entzogen werden. Anschließend kann das Zugriffsrecht SELECT nach Bedarf bestimmten Benutzern erteilt werden. Das Erteilen und Widerrufen des Zugriffsrechts SELECT für die Systemkatalogsichten erfolgt genauso wie für jede andere Sicht, jedoch benötigen Sie dafür die Berechtigung ACCESSCTRL oder SECADM.

Sie sollten zumindest in Betracht ziehen, falls kein Benutzer wissen soll, auf welche Objekte andere Benutzer Zugriff haben, den Zugriff auf die folgenden Katalog- und Verwaltungssichten einzuschränken:

- SYSCAT.COLAUTH
- SYSCAT.DBAUTH
- SYSCAT.INDEXAUTH
- SYSCAT.PACKAGEAUTH
- SYSCAT.PASSTHRUAUTH
- SYSCAT.ROUTINEAUTH
- SYSCAT.SCHEMAAUTH
- SYSCAT.SECURITYLABELACCESS
- SYSCAT.SECURITYPOLICYEXEMPTIONS
- SYSCAT.SEQUENCEAUTH
- SYSCAT.SURROGATEAUTHIDS
- SYSCAT.TABAUTH
- SYSCAT.TBSPACEAUTH
- SYSCAT.XSROBJECTAUTH
- SYSIBMADM.AUTHORIZATIONIDS
- SYSIBMADM.OBJECTOWNERS
- SYSIBMADM.PRIVILEGES

Dadurch wird verhindert, dass Informationen über Zugriffsrechte von Benutzern für alle Benutzer mit Zugriff auf die Datenbank verfügbar werden.

Sie sollten auch die Spalten überprüfen, für die die Statistikdaten gesammelt wurden. Einige der im Systemkatalog aufgezeichneten Statistikdaten könnten Datenwerte enthalten, die in Ihrer Umgebung sensible Informationen darstellen. Wenn diese Statistikdaten sensible Daten enthalten, können Sie das Zugriffsrecht SELECT auf die Katalogsichten SYSCAT.COLUMNS und SYSCAT.COLDIST für PUBLIC widerrufen.

Wenn Sie den Zugriff auf die Systemkatalogsichten beschränken möchten, können Sie Sichten definieren, damit jeder Berechtigungsname Informationen über seine eigenen Zugriffsrechte abrufen kann.

Zum Beispiel enthält die folgende Sicht MYSELECTS den Eigner und den Namen jeder Tabelle, für die dem Berechtigungsnamen eines Benutzers direkt das Zugriffsrecht SELECT erteilt wurde:

```
CREATE VIEW MYSELECTS AS
   SELECT TABSCHEMA, TABNAME FROM SYSCAT.TABAUTH
   WHERE GRANTEETYPE = 'U'
     AND GRANTEE = USER
     AND SELECTAUTH = 'Y'
```

Das Schlüsselwort USER in dieser Anweisung ist gleich dem Wert des Berechtigungsnamens der aktuellen Sitzung.

Mit der folgenden Anweisung wird die Sicht für jeden Berechtigungsnamen verfügbar gemacht:

```
GRANT SELECT ON TABLE MYSELECTS TO PUBLIC
```

Außerdem sollten Sie nicht vergessen, das Zugriffsrecht SELECT für die Sicht- und Basistabelle durch Absetzen der folgenden beiden Anweisungen zu widerrufen:

```
REVOKE SELECT ON TABLE SYSCAT.TABAUTH FROM PUBLIC
REVOKE SELECT ON TABLE SYSIBM.SYSTABAUTH FROM PUBLIC
```

Kapitel 7. Firewallunterstützung

Eine *Firewall* ist eine Gruppe zusammengehöriger Programme, die sich auf einem Gateway-Netzserver befinden und dazu dienen, unbefugten Zugriff auf ein System oder ein Netz zu verhindern.

Es gibt vier Arten von Firewalls:

1. Paketfilterungsfirewalls, Screening Router Firewalls oder Firewalls der Netzebene
2. Klassische Proxy-Firewalls der Anwendungsebene
3. Transparente Proxy- oder Circuit-Level-Firewalls
4. SMLI-Firewalls (Stateful Multi-Layer Inspection)

Es sind Firewallprodukte verfügbar, die eine der zuvor genannten Firewallarten implementieren. Viele andere Firewallprodukte implementieren jedoch eine Kombination aus diesen Arten.

Screening-Router-Firewalls

Die Screening-Router-Firewall wird auch als Netz- oder Paketfilterungsfirewall bezeichnet. Die Funktionsweise einer solchen Firewall beruht auf der Überwachung (Screening) ankommender Datenpakete und der Prüfung von Protokollattributen. Zu den getesteten Protokollattributen gehören die Quellen- und Zieladresse, der Typ des Protokolls, der Quellen- und Zielport sowie einige andere protokollspezifische Attribute.

Für alle Firewall-Lösungen (außer SOCKS) müssen Sie sicherstellen, dass alle von der DB2-Datenbank verwendeten Ports für ankommende und abgehende Pakete geöffnet sind. Die DB2-Datenbank verwendet den Port 523 für den DB2-Verwaltungsserver (DAS), der von den DB2-Datenbanktools verwendet wird. Bestimmen Sie die Ports, die von allen Ihren Serverinstanzen verwendet werden, indem Sie mithilfe der Datei 'services' den Servicenamen in der Konfigurationsdatei des Datenbankmanagers auf dem Server mit seiner Portnummer abgleichen.

Proxy-Firewalls der Anwendungsebene

Ein Proxy bzw. ein Proxy-Server ist eine Technik, die als Vermittlungsstelle zwischen einem Web-Client und einem Web-Server fungiert. Eine Proxy-Firewall fungiert als Gateway für Anforderungen, die von Clients ankommen.

Wenn Clientanforderungen durch die Firewall empfangen werden, wird die endgültige Zieladresse durch die Proxysoftware ermittelt. Der Anwendungsproxy übersetzt die Adresse, führt weitere Prüfungen zur Zugriffssteuerung und Protokollfunktionen nach Bedarf aus und stellt die Verbindung zum Server für den Client her.

Das Produkt DB2 Connect auf einer Firewallmaschine kann als Proxy zum Zielserver eingesetzt werden. Darüber hinaus funktioniert ein DB2-Datenbankserver auf der Firewallmaschine, der als Hop-Server zum endgültigen Zielserver eingesetzt wird, wie ein Anwendungsproxy.

Circuit-Level-Firewalls

Die Circuit-Level-Firewall wird auch als transparente Proxy-Firewall bezeichnet.

Eine transparente Proxy-Firewall modifiziert die Anforderung bzw. die Antwort nicht über das hinaus, was für die Proxy-Authentifizierung und -Identifikation erforderlich ist. Ein Bespiel für eine transparente Proxy-Firewall ist SOCKS.

Das DB2-Datenbanksystem unterstützt SOCKS Version 4.

SMLI-Firewalls (Stateful Multi-Layer Inspection)

Die SMLI-Firewall bietet eine fortgeschrittene Form der Paketfilterung, bei der alle sieben Schichten des OSI-Modells (Open System Interconnection) untersucht werden.

Jedes Paket wird untersucht und mit bekannten Status freundlicher Pakete verglichen. Während Screening-Router-Firewalls nur den Paketheader untersuchen, prüfen SMLI-Firewalls das gesamte Paket einschließlich der Daten.

Kapitel 8. Sicherheits-Plug-ins

Die Authentifizierung für das DB2-Datenbanksystem erfolgt mithilfe von *Sicherheits-Plug-ins*. Ein Sicherheits-Plug-in ist eine dynamisch ladbare Bibliothek, die Authentifizierungssicherheitsservices bereitstellt.

Plug-in zum Abrufen von Gruppen
Ruft Informationen zur Gruppenzugehörigkeit für einen bestimmten Benutzer ab.

Plug-in für Benutzer-ID/Kennwort-Authentifizierung
Die folgenden Authentifizierungstypen werden mithilfe eines Plug-ins für die Benutzer-ID/Kennwort-Authentifizierung implementiert:

- `CLIENT`
- `SERVER`
- `SERVER_ENCRYPT`
- `DATA_ENCRYPT`
- `DATA_ENCRYPT_CMP`

Diese Authentifizierungstypen bestimmen, wie und wo die Authentifizierung eines Benutzers erfolgt. Der Authentifizierungstyp, der verwendet werden soll, wird anhand der folgenden Methode ermittelt:

- Bei CONNECT- oder ATTACH-Operationen hat der Wert für den Konfigurationsparameter **srvcon_auth** (sofern er angegeben wurde) Vorrang gegenüber dem Wert des Konfigurationsparameters **authentication**.
- In allen anderen Fällen wird der Wert des Konfigurationsparameters **authentication** verwendet.

Plug-in für GSS-API-Authentifizierung
Der offizielle Name für GSS-API lautet Generic Security Service Application Program Interface Version 2 (IETF RFC2743) und Generic Security Service API Version 2: C-Bindings (IETF RFC2744). Zur Implementierung des GSS-API-Authentifizierungsmechanismus wird in den meisten Fällen das Kerberos-Protokoll verwendet. Die folgenden Authentifizierungstypen werden mithilfe der Plug-ins der GSS-API-Authentifizierung implementiert:

- `KERBEROS`
- `GSSPLUGIN`
- `KRB_SERVER_ENCRYPT`
- `GSS_SERVER_ENCRYPT`

`KRB_SERVER_ENCRYPT` und `GSS_SERVER_ENCRYPT` unterstützen sowohl die GSS-API-Authentifizierung als auch die Benutzer-ID/Kennwort-Authentifizierung. Allerdings ist die GSS-API-Authentifizierung der bevorzugte Authentifizierungstyp. Die clientseitige Kerberos-Unterstützung ist unter den Betriebssystemen Solaris, AIX, HP-UX (nur 64-Bit), Windows und Linux verfügbar. Für Windows-Betriebssysteme ist die Kerberos-Unterstützung standardmäßig aktiviert.

Der DB2-Datenbankmanager unterstützt diese Plug-ins sowohl auf dem Client als auch auf dem Server.

Anmerkung: Authentifizierungstypen bestimmen, wie und wo ein Benutzer authentifiziert wird. Zur Verwendung eines bestimmten Authentifizierungstyps müssen Sie den Wert des Konfigurationsparameters **authentication** des Datenbankmanagers definieren.

Sie können die einzelnen Plug-ins unabhängig voneinander verwenden oder sie zusammen mit den anderen Plug-ins einsetzen. Sie können zum Beispiel ein spezielles serverseitiges Authentifizierungs-Plug-in verwenden, aber die DB2-Standardwerte für die Client- und die Gruppenauthentifizierung übernehmen. Alternativ hierzu können Sie auch lediglich mit einem Plug-in zum Abrufen von Gruppen oder einem Plug-in für die Clientauthentifizierung arbeiten, ohne dass ein serverseitiges Plug-in eingesetzt wird.

Wenn Sie die GSS-API-Authentifizierung verwenden wollen, dann benötigen Sie Plug-ins sowohl auf dem Client als auch auf dem Server.

Das Standardverhalten bei der Authentifizierung besteht in der Verwendung eines Benutzer-ID/Kennwort-Plug-ins, das einen Authentifizierungsmechanismus auf Betriebssystemebene implementiert.

Das DB2-Datenbankprodukt umfasst Plug-ins für das Abrufen von Gruppen, für die Benutzer-ID/Kennwort-Authentifizierung und die GSS-API-Authentifizierung. Sie können das Authentifizierungsverhalten für DB2-Clients und -Server weiter anpassen, indem Sie eigene Plug-ins entwickeln oder geeignete Plug-ins anderer Anbieter erwerben.

Implementierung von Sicherheits-Plug-ins auf DB2-Clients

DB2-Clients können ein Plug-in zum Abrufen von Gruppen und ein Plug-in für die Benutzer-ID/Kennwort-Authentifizierung unterstützen.

Alternativ hierzu können Clients, die mit einem Plug-in für die GSS-API-Authentifizierung arbeiten, feststellen, welches Plug-in verwendet werden sollte. Hierzu wird die Liste der implementierten GSS-API-Plug-ins auf dem DB2-Server durchsucht. Der Name des ersten Authentifizierungs-Plug-ins, der mit dem Namen des Plug-ins für die GSS-API-Authentifizierung übereinstimmt, das auf dem Client implementiert ist, wird ausgewählt. Sie geben die Liste der implementierten GSS-API-Plug-ins des Servers im Konfigurationsparameter **srvcon_gssplugin_list** des Datenbankmanagers an. Die folgende Abbildung veranschaulicht die Infrastruktur der Sicherheits-Plug-ins auf einem DB2-Client:

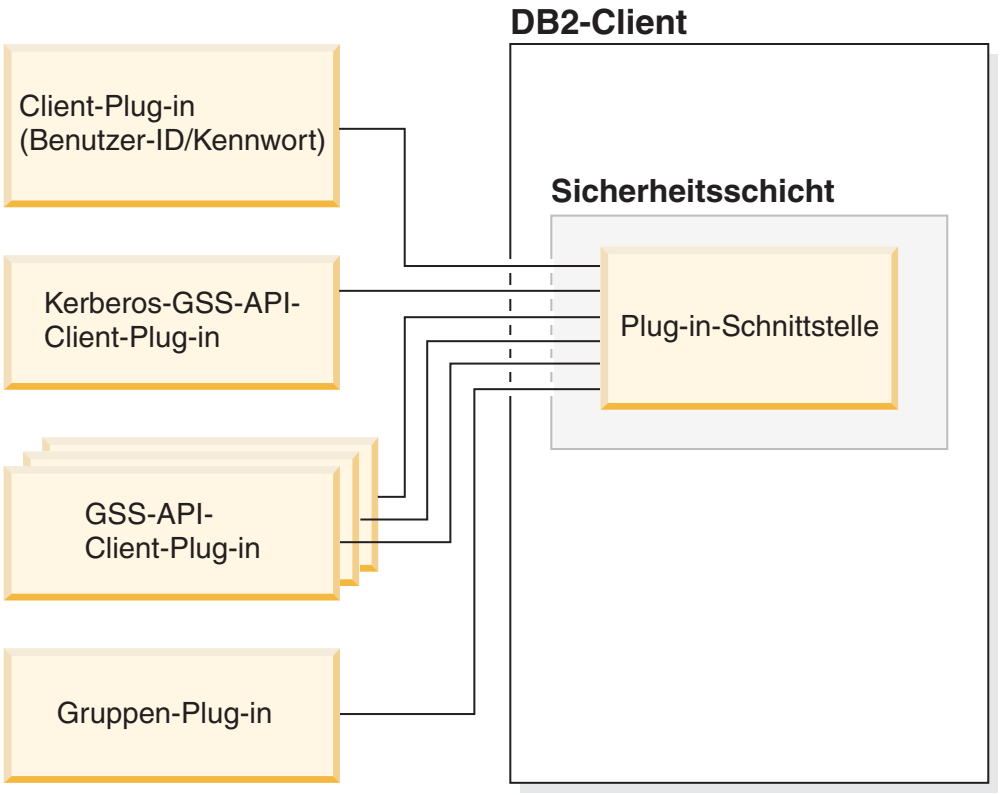


Abbildung 5. Implementierung von Sicherheits-Plug-ins auf DB2-Clients

Implementierung von Sicherheits-Plug-ins auf DB2-Servern

DB2-Server können ein Plug-in zum Abrufen von Gruppen, ein Plug-in zur Benutzer-ID/Kennwort-Authentifizierung und mehrere GSS-API-Plug-ins unterstützen. Sie können die verfügbaren GSS-API-Plug-ins in Form einer Liste mit Werten für den Konfigurationsparameter **srvcon_gssplugin_list** des Datenbankmanagers angeben. Allerdings kann nur ein GSS-API-Plug-in in dieser Liste ein Kerberos-Plug-in sein.

Zusätzlich zur Implementierung der serverseitigen Sicherheits-Plug-ins auf Ihrem Datenbankserver kann es erforderlich sein, auf dem Datenbankserver auch Plug-ins für die Clientauthentifizierung zu implementieren. Wenn Sie Operationen auf Instanzebene wie z. B. die Befehle **db2start** und **db2trc** ausführen, führt der DB2-Datenbankmanager Berechtigungsprüfungen für diese Operationen mithilfe der Plug-ins zur Clientauthentifizierung durch. Aus diesem Grund sollten Sie das Plug-in für die Clientauthentifizierung installieren, das dem Authentifizierungs-Plug-in auf dem Server entspricht. Der Name dieses Plug-ins wird im Konfigurationsparameter **authentication** des Datenbankmanagers auf dem Server angegeben.

Sie können für die Konfigurationsparameter **authentication** und **srvcon_auth** unterschiedliche Werte angeben. In diesem Szenario wird ein Mechanismus für die Authentifizierung von Datenbankverbindungen und der andere Mechanismus für die lokale Berechtigung verwendet.

Normalerweise gehen Sie zur Anwendung dieser Methode wie folgt vor:

- Geben Sie für den Konfigurationsparameter **srvcon_auth** den Wert GSSPLUGIN ein.

- Außerdem müssen Sie für den Konfigurationsparameter **authentication** den Wert SERVER definieren.

Der Konfigurationsparameter **srvcon_auth** wird zum Überschreiben des Authentifizierungstyps verwendet, der von eingehenden Verbindungen benutzt wird. Diese Verbindungen verwenden die Authentifizierungsmethode, die im Konfigurationsparameter **srvcon_auth** angegeben ist. Wenn dieser Wert jedoch nicht angegeben wird, dann wird stattdessen der Wert des Parameters **authentication** verwendet.

Wenn Sie keine Plug-ins zur Clientauthentifizierung auf dem Datenbankserver verwenden, schlagen Operationen auf Instanzebene wie zum Beispiel **db2start** fehl.

Die folgende Abbildung veranschaulicht die Plug-in-Infrastruktur für die Sicherheitsauthentifizierung auf einem DB2-Server:

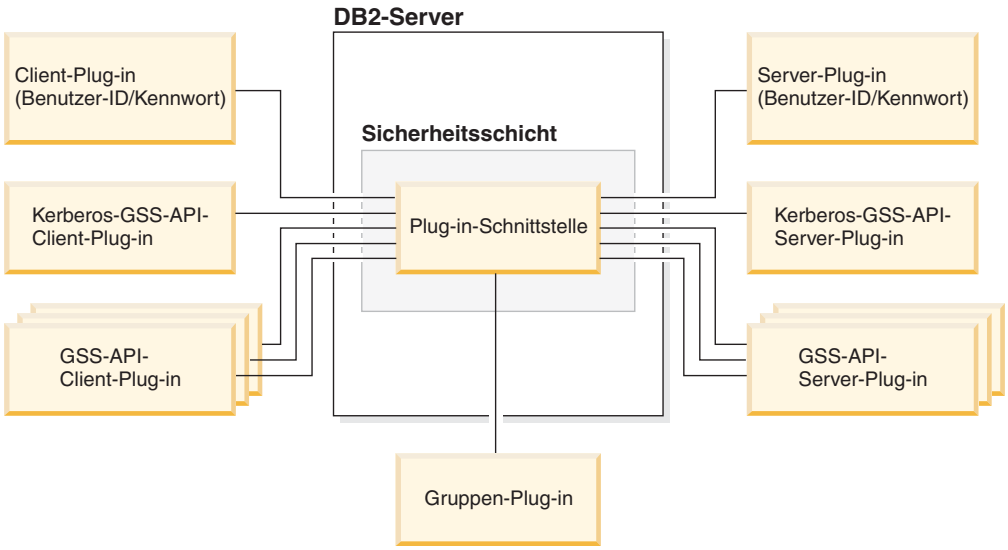


Abbildung 6. Implementierung von Sicherheits-Plug-ins auf DB2-Servern

Aktivieren von Sicherheits-Plug-ins

Sie können die Plug-ins, die von den einzelnen Authentifizierungsmechanismen verwendet werden sollen, angeben, indem Sie die Konfigurationsparameter des Datenbankmanagers entsprechend definieren. In der folgenden Tabelle werden diese Parameter aufgeführt:

Tabelle 31. Konfigurationsparameter des Datenbankmanagers für Sicherheitsauthentifizierungs-Plug-ins

Beschreibung	Parametername
Plug-in für Benutzer-ID/Kennwort-Authentifizierung des Clients	CLNT_PW_PLUGIN
Kerberos-Plug-in des Clients	CLNT_KRB_PLUGIN
Gruppen-Plug-in	GROUP_PLUGIN
GSS-Plug-in für lokale Berechtigung	LOCAL_GSSPLUGIN
Server-Plug-in-Modus	SRV_PLUGIN_MODE
Serverliste der GSS-Plug-ins	SRVCON_GSSPLUGIN_LIST
Plug-in für Benutzer-ID/Kennwort-Authentifizierung des Servers	SRVCON_PW_PLUGIN

Tabelle 31. Konfigurationsparameter des Datenbankmanagers für Sicherheitsauthentifizierungs-Plug-ins (Forts.)

Beschreibung	Parametername
Serververbindungsauthentifizierung	SRVCON_AUTH
Datenbankmanagerauthentifizierung	AUTHENTICATION

Wenn Sie die Werte für diese Parameter nicht angeben, werden für das Abrufen von Gruppen, die Benutzer-ID/Kennwort-Verwaltung und die Kerberos-Authentifizierung (wenn der Parameter **authentication** auf dem Server auf KERBEROS gesetzt ist) die Standard-Plug-ins des DB2-Produkts verwendet. Allerdings wird kein GSS-API-Standard-Plug-in bereitgestellt. Aus diesem Grund müssen Sie bei Angabe des Authentifizierungstyps GSSPLUGIN für den Parameter **authentication** auch ein Plug-in für die GSS-API-Authentifizierung für den Konfigurationsparameter **srvcon_gssplugin_list** angeben.

Laden von Sicherheits-Plug-ins

Alle unterstützten Plug-ins, die durch die Konfigurationsparameter des Datenbankmanagers angegeben werden, werden geladen, wenn der Datenbankmanager gestartet wird.

Während der Ausführung von CONNECT- oder ATTACH-Operationen lädt der DB2-Client ein Plug-in, das auf den Sicherheitsmechanismus abgestimmt ist, den der Client mit dem Server vereinbart hat. Eine Clientanwendung kann das gleichzeitige Laden und Verwenden mehrerer Sicherheits-Plug-ins veranlassen. Diese Situation kann zum Beispiel bei einem Multithread-Programm auftreten, das gleichzeitige Verbindungen zu verschiedenen Datenbanken aus verschiedenen Instanzen unterhält. In diesem Szenario stellt das Clientprogramm eine einleitende Verbindung zum Server A her, der ein GSS-API-Plug-in (G1) benutzt. Der Server A sendet eine Liste der unterstützten Plug-ins an den Client und das passende Plug-in G1 wird auf den Client geladen. Das Clientprogramm verfügt dann über einen anderen Thread, der eine Verbindung zu Server B herstellt, der ein GSS-API-Plug-in (G2) verwendet. Der Client wird über G2 informiert, das dann geladen wird. Beide Plug-ins (G1 und G2) werden nun auf dem Client gleichzeitig benutzt.

Für andere Aktionen als CONNECT- oder ATTACH-Operationen (z. B. Aktualisieren der Datenbankmanagerkonfiguration, Starten und Stoppen des Datenbankmanagers, Aktivieren und Inaktivieren der DB2-Tracefunktion) ist ebenfalls ein Berechtigungsmechanismus erforderlich. Für solche Aktionen lädt das DB2-Clientprogramm ein Plug-in, das von einem anderen Konfigurationsparameter des Datenbankmanagers angegeben wird:

- Wenn Sie für den Konfigurationsparameter **authentication** den Wert GSSPLUGIN angeben, dann verwendet der DB2-Datenbankmanager das Plug-in, das im Konfigurationsparameter **local_gssplugin** angegeben wurde.
- Wenn Sie für den Konfigurationsparameter **authentication** den Wert KERBEROS angeben, dann verwendet der DB2-Datenbankmanager das Plug-in, das im Konfigurationsparameter **clnt_krb_plugin** angegeben wurde.
- Andernfalls verwendet der DB2-Datenbankmanager das Plug-in, das im Konfigurationsparameter **clnt_pw_plugin** angegeben wurde.

Sicherheits-Plug-ins werden für Verbindungen unterstützt, die zum Datenbankserver über die IPv4- und IPv6-Adressprotokolle hergestellt wurden.

Entwickeln von Sicherheits-Plug-ins

Wenn Sie ein Sicherheitsauthentifizierungs-Plug-in entwickeln, müssen Sie die Standardauthentifizierungsfunktionen implementieren, die vom DB2-Datenbankmanager verwendet werden. Die zu implementierende Funktionalität lautet für die drei Plug-in-Typen wie folgt:

Plug-in zum Abrufen von Gruppen

- Sucht die Gruppen, zu denen ein Benutzer gehört, und gibt eine entsprechende Liste zurück.

Plug-in für Benutzer-ID/Kennwort-Authentifizierung

- Identifiziert den Standardsicherheitskontext (nur für Client-Plug-in).
- Überprüft und ändert (optional) ein Kennwort.
- Ermittelt, ob eine bestimmte Zeichenfolge einen gültigen Benutzer darstellt (nur für Server-Plug-in).
- Ändert die Benutzer-ID oder das Kennwort, die bzw. das auf dem Client bereitgestellt wird, bevor es an den Server gesendet wird (nur für Client-Plug-in).
- Gibt die DB2-Berechtigungs-ID zurück, die einem bestimmten Benutzer zugeordnet ist.

Plug-in für GSS-API-Authentifizierung

- Identifiziert den Standardsicherheitskontext (nur für Client-Plug-in).
- Implementiert die erforderlichen GSS-API-Funktionen.
- Generiert die Anfangsberechtigungsnachweise auf der Basis einer Benutzer-ID und eines Kennworts und ändert (optional) ein Kennwort (nur für Client-Plug-in).
- Erstellt und akzeptiert die Sicherheitstickets.
- Gibt die DB2-Berechtigungs-ID zurück, die einem bestimmten GSS-API-Sicherheitskontext zugeordnet ist.

Sie können für eine CONNECT-Anweisung, die Sie über den Befehlszeilenprozessor oder über eine dynamische SQL-Anweisung absetzen, eine Benutzer-ID mit bis zu 255 Zeichen übergeben.

Wichtig: Die Integrität Ihrer DB2-Datenbanksysteminstallation kann beeinträchtigt werden, wenn die Sicherheits-Plug-ins nicht korrekt codiert, geprüft und getestet wurden. Im DB2-Datenbankprodukt wurden Vorkehrungen zur Vermeidung zahlreicher gängiger Fehlertypen getroffen, eine vollständige Integrität kann jedoch nicht garantiert werden, wenn von Benutzern geschriebene Sicherheits-Plug-ins implementiert werden.

Speicherpositionen für Sicherheits-Plug-in-Bibliotheken

Wenn Sie Sicherheits-Plug-ins (entweder durch Eigenentwicklung oder durch Kauf bei einem Fremdanbieter) erwerben, kopieren Sie sie an bestimmte Speicherpositionen auf Ihrem Datenbankserver.

DB2-Clients suchen clientseitige Plug-ins zur Benutzerauthentifizierung im folgenden Verzeichnis:

- UNIX (32 Bit): `$DB2PATH/security32/plugin/client`
- UNIX (64 Bit): `$DB2PATH/security64/plugin/client`
- WINDOWS (32 und 64 Bit): `$DB2PATH\security\plugin\`*`instanzname`*`\client`

Anmerkung: Auf Windows-basierten Plattformen werden die Unterverzeichnisse *instanzname* und *client* nicht automatisch erstellt. Der Instanzeigner muss diese Verzeichnisse manuell erstellen.

Der DB2-Datenbankmanager sucht serverseitige Plug-ins zur Benutzerauthentifizierung im folgenden Verzeichnis:

- UNIX (32 Bit): `$DB2PATH/security32/plugin/server`
- UNIX (64 Bit): `$DB2PATH/security64/plugin/server`
- WINDOWS (32 und 64 Bit): `$DB2PATH\security\plugin\`*`instanzname`*`\server`

Anmerkung: Auf Windows-basierten Plattformen werden die Unterverzeichnisse *instanzname* und *server* nicht automatisch erstellt. Der Instanzeigner muss diese Verzeichnisse manuell erstellen.

Der DB2-Datenbankmanager sucht nach Plug-ins für Gruppen im folgenden Verzeichnis:

- UNIX (32 Bit): `$DB2PATH/security32/plugin/group`
- UNIX (64 Bit): `$DB2PATH/security64/plugin/group`
- WINDOWS (32 und 64 Bit): `$DB2PATH\security\plugin\`*`instanzname`*`\group`

Anmerkung: Auf Windows-basierten Plattformen werden die Unterverzeichnisse *instanzname* und *group* nicht automatisch erstellt. Der Instanzeigner muss diese Verzeichnisse manuell erstellen.

Namenskonventionen für Sicherheits-Plug-ins

Sicherheits-Plug-in-Bibliotheken müssen eine plattformspezifische Dateinamenerweiterung haben. Sicherheits-Plug-in-Bibliotheken, die in C oder C++ geschrieben sind, müssen eine plattformspezifische Dateinamenerweiterung haben:

- Windows: `.dll`.
- AIX: `.a oder .so`. Falls beide Erweiterungen vorhanden sind, wird die Erweiterung `.a` verwendet.
- Linux, HP IPF und Solaris: `.so`.

Anmerkung: Benutzer können Sicherheits-Plug-ins auch mit DB2 Universal JDBC Driver entwickeln.
Nehmen Sie zum Beispiel an, Sie haben eine Sicherheits-Plug-in-Bibliothek mit dem Namen `MyPlugin`. Für die einzelnen unterstützten Betriebssysteme müsste der korrekte Dateiname der Bibliothek wie folgt aussehen:

- Windows (32 Bit): MyPlugin.dll
- Windows (64 Bit): MyPlugin64.dll
- AIX (32 oder 64 Bit): MyPlugin.a oder MyPlugin.so
- SUN (32 oder 64 Bit), Linux (32 oder 64 Bit), HP (32 oder 64 Bit) auf IPF: MyPlugin.so

Anmerkung: Das Suffix "64" ist nur bei Bibliotheksnamen für Sicherheits-Plug-ins auf 64-Bit-Windows-Plattformen erforderlich.

Wenn Sie die Datenbankmanagerkonfiguration mit dem Namen eines Sicherheits-Plug-ins aktualisieren, verwenden Sie den vollen Namen der Bibliothek ohne das Suffix "64" und lassen sowohl die Dateierweiterung als auch den qualifizierten Pfadteil des Namens weg. Unabhängig vom Betriebssystem wird ein Sicherheits-Plug-in mit dem Namen `MyPlugin` wie folgt registriert:

```
UPDATE DBM CFG USING CLNT_PW_PLUGIN MyPlugin
```

Der Sicherheits-Plug-in-Name ist von der Groß-/Kleinschreibung abhängig und muss mit dem Bibliotheksnamen exakt übereinstimmen. DB2-Datenbanksysteme verwenden den Wert aus dem relevanten Konfigurationsparameter des Datenbankmanagers, um den Bibliothekspfad zusammenzusetzten, und verwenden anschließend den Bibliothekspfad, um die Sicherheits-Plug-in-Bibliothek zu laden.

Zur Vermeidung von Plug-in-Namenskonflikten sollten Sie das Plug-in unter Angabe der verwendeten Authentifizierungsmethode und eines kennzeichnenden Symbols für das Unternehmen, von dem das Plug-in geschrieben wurde, benennen. Wenn zum Beispiel das Plug-in, das die Authentifizierungsmethode `einemethode` implementiert, vom Unternehmen `Foo, Inc.` geschrieben wurde, könnte es einen Namen wie `FOOeinemethode.dll` erhalten.

Die maximale Länge eines Plug-in-Namens (ohne Dateierweiterung und das Suffix "64") ist auf 32 Byte begrenzt. Es gibt keine maximale Anzahl von Plug-ins, die vom Datenbankserver unterstützt werden, jedoch ist die maximale Länge der durch Kommata getrennten Plug-in-Liste in der Datenbankmanagerkonfiguration auf 255 Byte begrenzt. Zwei Define-Anweisungen in der Kopfdatendatei `sqlenv.h` geben diese beiden Grenzwerte an:

```
#define SQL_PLUGIN_NAME_SZ     32     /* Plug-in-Name */
#define SQL_SRVCON_GSSPLUGIN_LIST_SZ 255 /* GSS-API-Plug-in-Liste */
```

Die Bibliotheksdateien eines Plug-ins müssen die folgenden Dateiberechtigungen besitzen:

- Eigentum des Instanzeigners
- Lesbar durch alle Benutzer auf dem System
- Ausführbar durch alle Benutzer auf dem System

Unterstützung zweiteiliger Benutzer-IDs in Sicherheits-Plug-ins

Der DB2-Datenbankmanager unter Windows unterstützt die Verwendung zweiteiliger Benutzer-IDs sowie die Zuordnung zwischen zweiteiligen Benutzer-IDs und zweiteiligen Berechtigungs-IDs.

Betrachten Sie zum Beispiel eine zweiteilige Benutzer-ID unter dem Windows-Betriebssystem, die sich aus einer Domäne und einer Benutzer-ID zusammensetzt: `MEDWAY\pieter`. In diesem Beispiel ist `MEDWAY` eine Domäne und `pieter` der Benutzername. In DB2-Datenbanksystemen können Sie angeben, ob diese zweiteilige Benutzer-ID einer einteiligen oder einer zweiteiligen Berechtigungs-ID zugeordnet werden soll.

Die Zuordnung einer zweiteiligen Benutzer-ID zu einer zweiteiligen Berechtigungs-ID wird unterstützt, ist jedoch nicht das Standardverhalten. Standardmäßig werden sowohl einteilige Benutzer-IDs als auch zweiteilige Benutzer-IDs einteiligen Berechtigungs-IDs zugeordnet.

Die Standardzuordnung einer zweiteiligen Benutzer-ID zu einer einteiligen Benutzer-ID ermöglicht einem Benutzer, eine Verbindung zu der Datenbank mit folgenden Befehl herzustellen:

```
db2 connect to db user MEDWAY\pieter using pw
```

Beim Standardverhalten wird die Benutzer-ID `MEDWAY\pieter` in diesem Fall in die Berechtigungs-ID `PIETER` aufgelöst. Wenn die Unterstützung für die Zuordnung ei-

ner zweiteiligen Benutzer-ID zu einer zweiteiligen Berechtigungs-ID aktiviert würde, hieße die Berechtigungs-ID MEDWAY\PIETER.

Zur Aktivierung der Zuordnung zweiteiliger Benutzer-IDs zu zweiteiligen Berechtigungs-IDs in DB2 stellt DB2 zwei Gruppen von Authentifizierungs-Plug-ins zur Verfügung:

- Bei der einen Gruppe werden einteilige Benutzer-IDs und zweiteilige Benutzer-IDs ausschließlich einteiligen Berechtigungs-IDs zuordnet.
- Bei der anderen Gruppe werden einteilige Benutzer-IDs und zweiteilige Benutzer-IDs zweiteiligen Berechtigungs-IDs zugeordnet.

Wenn sich ein Benutzername in Ihrer Arbeitsumgebung mehreren Konten zuordnen lässt, die an verschiedenen Positionen (z. B. als lokales Konto, Domänenkonto und vertraute Domänenkonten) definiert sind, können Sie das Plug-in angeben, das die Zuordnung zweiteiliger Berechtigungs-IDs ermöglicht.

Es wichtig zu beachten, dass eine einteilige Berechtigungs-ID wie zum Beispiel PIETER und eine zweiteilige Berechtigungs-ID, die eine Domäne und eine Benutzer-ID wie zum Beispiel MEDWAY\pieter kombiniert, funktionell verschiedene Berechtigungs-IDs sind. Die Gruppe der Zugriffsrechte, die einer dieser Berechtigungs-IDs erteilt ist, kann sich von der Gruppe der Zugriffsrechte, die der anderen Berechtigungs-ID erteilt sind, völlig unterscheiden. Daher muss mit einteiligen und zweiteiligen Berechtigungs-IDs sorgfältig gearbeitet werden.

In der folgenden Tabelle sind die Arten der von DB2-Datenbanksystemen bereitgestellten Plug-ins sowie die Plug-in-Namen für die jeweiligen Authentifizierungsimplementierungen aufgeführt.

Tabelle 32. DB2-Sicherheits-Plug-ins

Authentifizierungstyp	Name des Plug-ins für einteilige Benutzer-IDs	Name des Plug-ins für zweiteilige Benutzer-IDs
Benutzer-ID/Kennwort (Client)	IBMOSauthclient	IBMOSauthclientTwoPart
Benutzer-ID/Kennwort (Server)	IBMOSauthserver	IBMOSauthserverTwoPart
Kerberos	IBMkrb5	IBMkrb5TwoPart

Anmerkung: Auf 64-Bit-Windows-Plattformen werden die Zeichen "64" an die hier aufgeführten Plug-in-Namen angehängt.

Wenn Sie einen Authentifizierungstyp angeben, der ein Benutzer-ID/Kennwort- oder Kerberos-Plug-in erfordert, werden standardmäßig die Plug-ins verwendet, die in der Spalte "Name des Plug-ins für einteilige Benutzer-IDs" der obigen Tabelle aufgeführt sind.

Wenn zweiteilige Benutzer-IDs zweiteiligen Berechtigungs-IDs zugeordnet werden sollen, müssen Sie angeben, dass das Plug-in für zweiteilige Benutzer-IDs, das nicht das Standard-Plug-in ist, verwendet wird. Sicherheits-Plug-ins werden auf der Instanzebene angegeben, indem die sicherheitsbezogenen Konfigurationsparameter des Datenbankmanagers wie folgt eingestellt werden:

Für die Serverauthentifizierung, bei der zweiteilige Benutzer-IDs zweiteiligen Berechtigungs-IDs zugeordnet werden, müssen Sie die folgenden Parametereinstellungen definieren:

- srvcon_pw_plugin mit dem Wert IBMOSauthserverTwoPart
- clnt_pw_plugin mit dem Wert IBMOSauthclientTwoPart

Für die Clientauthentifizierung, bei der zweiteilige Benutzer-IDs zweiteiligen Berechtigungs-IDs zugeordnet werden, müssen Sie die folgenden Parametereinstellungen definieren:

- srvcon_pw_plugin mit dem Wert IBMOSauthserverTwoPart
- clnt_pw_plugin mit dem Wert IBMOSauthclientTwoPart

Für die Kerberos-Authentifizierung, bei der zweiteilige Benutzer-IDs zweiteiligen Berechtigungs-IDs zugeordnet werden, müssen Sie die folgenden Parametereinstellungen definieren:

- srvcon_gssplugin_list mit dem Wert IBMOSkrb5TwoPart
- clnt_krb_plugin mit dem Wert IBMkrb5TwoPart

Die Bibliotheken der Sicherheits-Plug-ins akzeptieren zweiteilige Benutzer-IDs, die in einem mit der Microsoft Windows-SAM-Datenbank (Security Account Manager) kompatiblen Format angegeben werden, wie zum Beispiel das Format: *domäne\benutzer-id*. Sowohl der Domänenname als auch die Benutzer-ID werden von der DB2-Authentifizierung und den Authentifizierungsprozessen beim Verbindungsaufbau verwendet.

Sie sollten in Betracht ziehen, die Plug-ins mit zweiteiligen Berechtigungs-IDs zu implementieren, wenn Sie neue Datenbanken erstellen, um Konflikte mit einteiligen Berechtigungs-IDs in vorhandenen Datenbanken zu vermeiden. Neue Datenbanken, die zweiteilige Berechtigungs-IDs verwenden, müssen in einer Instanz erstellt werden, die von den Datenbanken mit einteiligen Berechtigungs-IDs getrennt ist.

Versionssteuerung für Sicherheits-Plug-in-APIs

Das DB2-Datenbanksystem unterstützt eine Versionsnummerierung der Sicherheits-Plug-in-APIs. Solche Versionsnummern sind ganzzahlige Werte, die bei 1 für DB2 UDB Version 8.2 beginnen.

Die Versionsnummer, die DB2 an die Sicherheits-Plug-in-APIs übergibt, ist die höchste Versionsnummer der API, die DB2 unterstützen kann. Sie entspricht der Versionsnummer der Struktur. Wenn das Plug-in eine höhere API-Version unterstützen kann, muss sie Funktionszeiger für die Version zurückgeben, die DB2 angefordert hat. Wenn das Plug-in nur eine niedrigere Version der API unterstützt, sollte das Plug-in die Funktionszeiger für die niedrigere Version angeben. In beiden Fällen sollten die Sicherheits-Plug-in-APIs die unterstützte Versionsnummer für die API im Feld 'version' der Funktionsstruktur zurückgeben.

Für DB2 ändern sich die Versionsnummern von Sicherheits-Plug-ins nur, wenn dies erforderlich ist (z. B. wenn es Änderungen an den Parametern der APIs gibt). Versionsnummern werden nicht automatisch mit DB2-Releasenummern geändert.

Hinweise zu 32- und 64-Bit-Sicherheits-Plug-ins

Im Allgemeinen verwendet eine 32-Bit-DB2-Instanz das 32-Bit-Sicherheits-Plug-in und eine 64-Bit-DB2-Instanz das 64-Bit-Sicherheits-Plug-in. Allerdings unterstützt DB2 in einer 64-Bit-Instanz 32-Bit-Anwendungen, die eine 32-Bit-Plug-in-Bibliothek erfordern.

Eine Datenbankinstanz, in der sowohl 32-Bit-Anwendungen als auch 64-Bit-Anwendungen ausgeführt werden können, wird als Hybridinstanz bezeichnet. Wenn Sie eine Hybridinstanz haben und 32-Bit-Anwendungen ausführen wollen, stellen Sie sicher, dass die erforderlichen 32-Bit-Sicherheits-Plug-ins im Verzeichnis für 32-Bit-Plug-ins verfügbar sind. In 64-Bit-DB2-Instanzen unter Linux- und UNIX-Betriebssystemen (ausgenommen Linux auf IPF) sind die Verzeichnisse `security32` und `security64` vorhanden. In einer 64-Bit-DB2-Instanz unter Windows auf x64 oder IPF befinden sich die 32- und die 64-Bit-Sicherheits-Plug-ins im selben Verzeichnis, jedoch sind die Namen der 64-Bit-Plug-ins durch das Suffix "64" gekennzeichnet.

Wenn Sie von einer 32-Bit-Instanz auf eine 64-Bit-Instanz aufrüsten wollen, sollten Sie Versionen Ihrer Sicherheits-Plug-ins erwerben, die für 64-Bit-Systeme rekompiliert sind.

Wenn Sie Ihre Sicherheits-Plug-ins bei einem Anbieter erworben haben, der keine 64-Bit-Plug-in-Bibliotheken zur Verfügung stellt, können Sie ein 64-Bit-Stub implementieren, das eine 32-Bit-Anwendung ausführt. In diesem Fall ist das Sicherheits-Plug-in eher ein externes Programm und keine Bibliothek mehr.

Fehlerbestimmung für Sicherheits-Plug-ins

Probleme mit Sicherheits-Plug-ins werden auf zwei Arten zurückgemeldet: durch SQL-Fehler und durch das Protokoll mit Benachrichtigungen für die Systemverwaltung.

Die folgenden SQLCODE-Werte beziehen sich auf Sicherheits-Plug-ins:

- SQLCODE -1365 wird zurückgegeben, wenn ein Plug-in-Fehler bei der Verarbeitung von **db2start** oder **db2stop** auftritt.
- SQLCODE -1366 wird zurückgegeben, wenn ein Problem mit der lokalen Berechtigung vorliegt.
- SQLCODE -30082 wird für alle verbindungsbezogenen Plug-in-Fehler zurückgegeben.

Die Protokolle mit Benachrichtigungen für die Systemverwaltung sind eine gute Ressource für das Debugging und Verwalten von Sicherheits-Plug-ins. Ein Protokoll mit Benachrichtigungen für die Systemverwaltung unter UNIX können Sie der Datei `sqllib/db2dump/`*`instanzname`*`.N.nfy` entnehmen. Zum Anzeigen eines Protokolls mit Benachrichtigungen für die Systemverwaltung unter Windows-Betriebssystemen verwenden Sie das Tool 'Ereignisanzeige'. Das Tool 'Ereignisanzeige' finden Sie, indem Sie von der Schaltfläche 'Start' des Windows-Betriebssystems aus die Optionen `Einstellungen -> Systemsteuerung -> Verwaltung -> Ereignisanzeige` auswählen. Die folgenden Werte im Protokoll mit Benachrichtigungen für die Systemverwaltung beziehen sich auf Sicherheits-Plug-ins:

- 13000 gibt an, dass ein Aufruf der API für das GSS-API-Sicherheits-Plug-in mit einem Fehler fehlgeschlagen ist und eine optionale Fehlernachricht zurückgegeben hat.

  ```
  SQLT_ADMIN_GSS_API_ERROR (13000)
  Das Plug-in "plug-in-name" hat den Fehlercode "fehlercode"
  von der GSS-API (Generic Security Service) "name-der-gss-api" mit der
  Fehlernachricht "fehlernachricht" erhalten.
  ```

- 13001 gibt an, dass ein Aufruf der API für das DB2-Sicherheits-Plug-in mit einem Fehler fehlgeschlagen ist und eine optionale Fehlernachricht zurückgegeben hat.

```
SQLT_ADMIN_PLUGIN_API_ERROR(13001)
Das Plug-in "plug-in-name" hat den Fehlercode "fehlercode"
von der API des DB2-Sicherheits-Plug-in "name-der-gss-api" mit der
Fehlernachricht "fehlernachricht" erhalten.
```

- 13002 gibt an, dass DB2 ein Plug-in nicht entladen konnte.

```
SQLT_ADMIN_PLUGIN_UNLOAD_ERROR (13002)
Das Plug-in "plug-in-name" konnte nicht entladen werden.
Es sind keine weiteren Schritte erforderlich.
```

- 13003 weist auf einen falschen Namen des Principals hin.

```
SQLT_ADMIN_INVALID_PRIN_NAME (13003)
Der für "plug-in-name" verwendete Name des Principals "name-des-principals"
ist ungültig. Stellen Sie sicher, dass der Name des Principals gültig ist und
dass er ein Format verwendet, das vom Sicherheits-Plug-in erkannt wird.
```

- 13004 gibt an, dass der Plug-in-Name nicht gültig ist. Pfadtrennzeichen (unter UNIX "/", unter Windows "\") sind im Plug-in-Namen nicht zulässig.

```
SQLT_ADMIN_INVALID_PLGN_NAME (13004)
Der Plug-in-Name "plug-in-name" ist ungültig. Stellen Sie sicher,
dass ein gültiger Plug-in-Name angegeben wurde.
```

- 13005 gibt an, dass das Sicherheits-Plug-in nicht geladen werden konnte. Stellen Sie sicher, dass sich das Plug-in im richtigen Verzeichnis befindet und dass die entsprechenden Konfigurationsparameter des Datenbankmanagers aktualisiert wurden.

```
SQLT_ADMIN_PLUGIN_LOAD_ERROR (13005)
Das Plug-in "plug-in-name" konnte nicht geladen werden. Prüfen Sie, ob das Plug-in
vorhanden ist und ob die Verzeichnisposition und die Dateiberechtigungen gültig sind.
```

- 13006 gibt an, dass von einem Sicherheits-Plug-in ein unerwarteter Fehler festgestellt wurde. Sammeln Sie alle Unterstützungsinformationen (**db2support**), zeichnen Sie einen Trace auf (**db2trc**), falls möglich, und rufen Sie anschließend die IBM Unterstützungsfunktion an, um weitere Hilfe zu erhalten.

```
SQLT_ADMIN_PLUGIN_UNEXP_ERROR (13006)
Das Plug-in "plug-in-name" hat einen unerwarteten Fehler festgestellt.
Hilfe erhalten Sie bei der IBM Unterstützungsfunktion.
```

Anmerkung: Wenn Sie Sicherheits-Plug-ins auf einem 64-Bit-Windows-Datenbankserver verwenden und einen Ladefehler für ein Sicherheits-Plug-in empfangen, lesen Sie den Abschnitt mit den Hinweisen zu 32- und 64-Bit-Sicherheits-Plug-ins sowie den Abschnitt zu den Namenskonventionen für Sicherheits-Plug-ins. Die Plug-in-Bibliothek für die 64-Bit-Plattform erfordert das Suffix "64" am Ende des Namens der Bibliothek, während der Eintrag in den Konfigurationsparametern des Datenbankmanagers für Sicherheits-Plug-ins dieses Suffix nicht angeben darf.

Aktivieren von Plug-ins

Implementieren eines Plug-ins zum Abrufen von Gruppen

Zur Anpassung des Verhaltens des DB2-Sicherheitssystems beim Abrufen von Gruppen können Sie ein eigenes Plug-ins zum Abrufen von Gruppen entwickeln oder ein Plug-in bei einem Fremdanbieter erwerben.

Vorbereitende Schritte

Nachdem Sie ein Plug-in zum Abrufen von Gruppen erworben haben, das für Ihr Datenbankverwaltungssystem geeignet ist, können Sie es implementieren.

Vorgehensweise

- Führen Sie die folgenden Schritte aus, um ein Plug-in zum Abrufen von Gruppen auf dem Datenbankserver zu implementieren:
  1. Kopieren Sie die Bibliothek des Plug-ins zum Abrufen von Gruppen in das Gruppen-Plug-in-Verzeichnis des Servers (group).
  2. Aktualisieren Sie den Konfigurationsparameter **group_plugin** des Datenbankmanagers mit dem Namen des Plug-ins.
- Führen Sie die folgenden Schritte aus, um ein Plug-in zum Abrufen von Gruppen auf Datenbankclients zu implementieren:
  1. Kopieren Sie die Bibliothek des Plug-ins zum Abrufen von Gruppen in das Gruppen-Plug-in-Verzeichnis des Clients (group).
  2. Aktualisieren Sie auf dem Datenbankclient den Konfigurationsparameter **group_plugin** des Datenbankmanagers mit dem Namen des Plug-ins.

Implementieren eines Benutzer-ID/Kennwort-Plug-ins

Zur Anpassung des Benutzer-ID/Kennwort-Authentifizierungsverhaltens des DB2-Sicherheitssystems können Sie eigene Plug-ins für die Benutzer-ID/Kennwort-Authentifizierung entwickeln oder ein Plug-in bei einem Fremdanbieter erwerben.

Vorbereitende Schritte

Abhängig von ihrer geplanten Verwendung müssen alle Benutzer-ID/Kennwort-Authentifizierungs-Plug-ins entweder im Client-Plug-in-Verzeichnis oder im Server-Plug-in-Verzeichnis platziert werden. Wenn ein Plug-in im Client-Plug-in-Verzeichnis platziert wird, wird es für die Überprüfung bei der lokalen Authentifizierung sowie zur Prüfung des Clients verwendet, wenn dieser versucht, eine Verbindung zum Server herzustellen. Wenn das Plug-in im Server-Plug-in-Verzeichnis platziert wird, wird es verwendet, um eingehende Verbindungen zum Server zu verarbeiten und zu überprüfen, ob eine Berechtigungs-ID vorhanden und gültig ist, wenn die Anweisung GRANT ohne Angabe des Schlüsselworts USER oder GROUP abgesetzt wird. In den meisten Fällen erfordert die Benutzer-ID/Kennwort-Authentifizierung nur ein serverseitiges Plug-in. Es ist möglich, auch wenn dies allgemein für weniger sinnvoll erachtet wird, nur ein clientseitiges Plug-in zur Benutzer-ID/Kennwort-Authentifizierung zu haben. Es ist möglich, wenn auch recht ungewöhnlich, übereinstimmende Benutzer-ID/Kennwort-Plug-ins auf dem Client und auf dem Server vorauszusetzen.

Anmerkung: Sie müssen den DB2-Server und alle Anwendungen, die die Plug-ins verwenden, stoppen, bevor Sie eine *neue* Version eines *vorhandenen* Plug-ins implementieren. Undefiniertes Verhalten, einschließlich Traps, kann auftreten, wenn ein Prozess ein Plug-in zu dem Zeitpunkt verwendet, zu dem eine neue Version (mit demselben Namen) über das Plug-in kopiert wird. Diese Einschränkung gilt nicht, wenn Sie ein Plug-in zum ersten Mal implementieren oder wenn das Plug-in nicht im Gebrauch ist.
Nachdem Sie Benutzer-ID/Kennwort-Authentifizierungs-Plug-ins erworben haben, die für Ihr Datenbankverwaltungssystem geeignet sind, können Sie sie implementieren.

Vorgehensweise

- Führen Sie die folgenden Schritte auf dem Datenbankserver aus, um ein Benutzer-ID/Kennwort-Authentifizierungs-Plug-in auf dem Datenbankserver zu implementieren:

1. Kopieren Sie die Bibliothek des Benutzer-ID/Kennwort-Authentifizierungs-Plug-ins in das Server-Plug-in-Verzeichnis.
2. Aktualisieren Sie den Konfigurationsparameter **srvcon_pw_plugin** des Datenbankmanagers mit dem Namen des Server-Plug-ins. Dieses Plug-in wird vom Server verwendet, wenn er CONNECT- und **ATTACH**-Anforderungen verarbeitet.
3. Anschließend haben Sie zwei Möglichkeiten:
   – Setzen Sie den Konfigurationsparameter **srvcon_auth** des Datenbankmanagers auf den Authentifizierungstyp CLIENT, SERVER, SERVER_ENCRYPT, DATA_ENCRYPT oder DATA_ENCRYPT_CMP. Alternativ:
   – Setzen Sie den Konfigurationsparameter **srvcon_auth** des Datenbankmanagers auf den Wert NOT_SPECIFIED und den Parameter **authentication** auf den Authentifizierungstyp CLIENT, SERVER, SERVER_ENCRYPT, DATA_ENCRYPT oder DATA_ENCRYPT_CMP.

- Führen Sie die folgenden Schritte auf jedem Client aus, um ein Benutzer-ID/Kennwort-Authentifizierungs-Plug-in auf Datenbankclients zu implementieren:
  1. Kopieren Sie die Bibliothek des Benutzer-ID/Kennwort-Authentifizierungs-Plug-ins in das Client-Plug-in-Verzeichnis.
  2. Aktualisieren Sie den Konfigurationsparameter **clnt_pw_plugin** des Datenbankmanagers mit dem Namen des Client-Plug-ins. Dieses Plug-in wird unabhängig davon, wo die Authentifizierung stattfindet, geladen und aufgerufen, nicht nur wenn der Datenbankkonfigurationsparameter **authentication** auf den Wert CLIENT gesetzt ist.
- Führen Sie die folgenden Schritte zur Implementierung der Berechtigung auf einem Client, Server oder Gateway durch ein Benutzer-ID/Kennwort-Authentifizierungs-Plug-in auf jedem Client, Server oder Gateway aus:
  1. Kopieren Sie die Bibliothek des Benutzer-ID/Kennwort-Authentifizierungs-Plug-ins in das Client-Plug-in-Verzeichnis auf dem Client, Server oder Gateway.
  2. Aktualisieren Sie den Konfigurationsparameter **clnt_pw_plugin** des Datenbankmanagers mit dem Namen des Plug-ins.
  3. Setzen Sie den Konfigurationsparameter **authentication** des Datenbankmanagers auf den Wert CLIENT, SERVER, SERVER_ENCRYPT, DATA_ENCRYPT oder DATA_ENCRYPT_CMP.

Implementieren eines GSS-API-Plug-ins

Zur Anpassung des Authentifizierungsverhaltens des DB2-Sicherheitssystems können Sie eigene Plug-ins für die Authentifizierung mithilfe der GSS-API entwickeln oder ein Plug-in bei einem Fremdanbieter erwerben.

Vorbereitende Schritte

Bei anderen Plug-in-Typen als Kerberos müssen die Plug-in-Namen auf dem Client und dem Server sowie der Plug-in-Typ übereinstimmen. Die Plug-ins auf dem Client und dem Server brauchen nicht von demselben Anbieter zu stammen, jedoch müssen sie kompatible GSS-API-Token generieren und verarbeiten. Jede Kombination von Kerberos-Plug-ins, die auf dem Client und auf dem Server implementiert sind, ist akzeptabel, da Kerberos-Plug-ins standardisiert sind. Verschiedene Implementierungen weniger standardisierter GSS-API-Mechanismen, wie zum Beispiel *x.509*-Zertifikate, sind jedoch möglicherweise nur teilweise mit DB2-Datenbanksystemen kompatibel. Abhängig von ihrer geplanten Verwendung müssen alle GSS-API-Authentifizierungs-Plug-ins entweder im Client-Plug-in-Verzeichnis oder im Server-Plug-in-Verzeichnis platziert werden. Wenn ein Plug-in im Client-Plug-in-

Verzeichnis platziert wird, wird es für die Überprüfung bei der lokalen Authentifizierung sowie bei Versuchen eines Clients, eine Verbindung mit dem Server herzustellen, verwendet. Wenn das Plug-in im Server-Plug-in-Verzeichnis platziert wird, wird es verwendet, um eingehende Verbindungen zum Server zu verarbeiten und zu überprüfen, ob eine Berechtigungs-ID vorhanden und gültig ist, wenn die Anweisung GRANT ohne Angabe des Schlüsselworts USER oder GROUP abgesetzt wird.

Anmerkung: Sie müssen den DB2-Server und alle Anwendungen, die die Plug-ins verwenden, stoppen, bevor Sie eine *neue* Version eines *vorhandenen* Plug-ins implementieren. Undefiniertes Verhalten, einschließlich Traps, kann auftreten, wenn ein Prozess ein Plug-in zu dem Zeitpunkt verwendet, zu dem eine neue Version (mit demselben Namen) über das Plug-in kopiert wird. Diese Einschränkung gilt nicht, wenn Sie ein Plug-in zum ersten Mal implementieren oder wenn das Plug-in nicht im Gebrauch ist.

Nachdem Sie GSS-API-Authentifizierungs-Plug-ins erworben haben, die für Ihr Datenbankverwaltungssystem geeignet sind, können Sie sie implementieren.

Vorgehensweise

- Führen Sie die folgenden Schritte auf dem Server aus, um ein GSS-API-Authentifizierungs-Plug-in auf dem Datenbankserver zu implementieren:
  1. Kopieren Sie die Bibliothek des GSS-API-Authentifizierungs-Plug-ins in das Server-Plug-in-Verzeichnis. Sie können viele GSS-API-Plug-ins in dieses Verzeichnis kopieren.
  2. Aktualisieren Sie den Konfigurationsparameter **srvcon_gssplugin_list** des Datenbankmanagers mit einer geordneten, durch Kommata begrenzten Liste der Namen der im GSS-API-Plug-in-Verzeichnis installierten Plug-ins.
  3. Anschließend haben Sie zwei Möglichkeiten:
     - Setzen Sie den Konfigurationsparameter **srvcon_auth** des Datenbankmanagers auf den Wert GSSPLUGIN oder GSS_SERVER_ENCRYPT, um den Server zur Verwendung der Authentifizierungsmethode GSSAPI PLUGIN zu konfigurieren. Alternativ:
     - Setzen Sie den Konfigurationsparameter **srvcon_auth** des Datenbankmanagers auf den Wert NOT_SPECIFIED und den Konfigurationsparameter **authentication** auf den Wert GSSPLUGIN oder GSS_SERVER_ENCRYPT, um den Server zur Verwendung der Authentifizierungsmethode GSSAPI PLUGIN zu konfigurieren.
- Führen Sie die folgenden Schritte auf jedem Client aus, um ein GSS-API-Authentifizierungs-Plug-in auf Datenbankclients zu implementieren:
  1. Kopieren Sie die Bibliothek des GSS-API-Authentifizierungs-Plug-ins in das Client-Plug-in-Verzeichnis. Sie können viele GSS-API-Plug-ins in dieses Verzeichnis kopieren. Der Client wählt ein GSS-API-Plug-in zur Authentifizierung während einer CONNECT- oder ATTACH-Operation aus, indem er das erste, in der Plug-in-Liste des Servers enthaltene GSS-API-Plug-in verwendet, das auf dem Client verfügbar ist.
  2. Optional: Katalogisieren Sie die Datenbanken, auf die der Client zugreifen soll, indem Sie angeben, dass der Client nur ein GSS-API-Authentifizierungs-Plug-in als Authentifizierungsverfahren akzeptiert. Beispiel:

     ```
     CATALOG DB testdb AT NODE testnode AUTHENTICATION GSSPLUGIN
     ```
- Führen Sie die folgenden Schritte zur Implementierung der Berechtigung auf einem Client, Server oder Gateway durch ein GSS-API-Authentifizierungs-Plug-in aus:

1. Kopieren Sie die Bibliothek des GSS-API-Authentifizierungs-Plug-ins in das Client-Plug-in-Verzeichnis auf dem Client, Server oder Gateway.
2. Aktualisieren Sie den Konfigurationsparameter **local_gssplugin** des Datenbankmanagers mit dem Namen des Plug-ins.
3. Setzen Sie den Konfigurationsparameter **authentication** des Datenbankmanagers auf den Wert GSSPLUGIN oder GSS_SERVER_ENCRYPT.

Implementieren eines Kerberos-Plug-ins

Zur Anpassung des Kerberos-Authentifizierungsverhaltens des DB2-Sicherheitssystems können Sie eigene Plug-ins für die Kerberos-Authentifizierung entwickeln oder ein Plug-in eines anderen Anbieters erwerben.

Vorbereitende Schritte

Wenn Sie eine neue Version eines vorhandenen Plug-ins implementieren wollen, müssen Sie den DB2-Server und alle Anwendungen stoppen, die das betreffende Plug-in benutzen. Undefiniertes Verhalten (einschließlich Traps) kann auftreten, wenn ein Prozess ein Plug-in zu dem Zeitpunkt verwendet, zu dem eine neue Version des Plug-ins (mit dem gleichen Namen) implementiert wird.

Informationen zu diesem Vorgang

Das Plug-in für die Kerberos-Authentifizierung kann auf einem Datenbankserver oder einem Datenbankclient implementiert werden.

Vorgehensweise

- Führen Sie die folgenden Schritte auf dem Server aus, um ein Kerberos-Authentifizierungs-Plug-in auf dem Datenbankserver zu implementieren:
  1. Kopieren Sie die Bibliothek des Kerberos-Authentifizierungs-Plug-ins in das Server-Plug-in-Verzeichnis.
  2. Aktualisieren Sie die Einstellung des Konfigurationsparameters **srvcon_gssplugin_list** des Datenbankmanagers, bei dem es sich um eine geordnete, durch Kommas begrenzte Liste handelt, und fügen Sie dabei den Namen des Kerberos-Server-Plug-ins ein. Nur ein Plug-in in dieser Liste darf ein Kerberos-Plug-in sein. Wenn in der Liste kein Kerberos-Plug-in vorhanden ist, dann wird ein Fehler zurückgegeben. Wenn in der Liste mehrere Kerberos-Plug-ins vorhanden sind, dann wird ebenfalls ein Fehler zurückgegeben. Wenn der Wert des Konfigurationsparameters leer ist und der Konfigurationsparameter **authentication** auf den Wert KERBEROS oder KRB_SVR_ENCRYPT gesetzt ist, wird das DB2-Standard-Kerberos-Plug-in IBMkrb5 verwendet.
  3. Falls erforderlich, müssen Sie den Wert des Konfigurationsparameters **srvcon_auth** des Datenbankmanagers definieren. Wenn Sie ein Kerberos-Plug-in implementieren wollen, dann lauten die zulässigen Werte für den Konfigurationsparameter **srvcon_auth** des Datenbankmanagers wie folgt:
     - KERBEROS
     - KRB_SERVER_ENCRYPT
     - GSSPLUGIN
     - GSS_SERVER_ENCRYPT
     - Leer. Dies ist jedoch nur möglich, wenn der Konfigurationsparameter **authentication** auf einen der vorherigen Werte in dieser Liste gesetzt ist.

- Führen Sie die folgenden Schritte auf dem Client aus, um ein Kerberos-Authentifizierungs-Plug-in auf einem Datenbankclient zu implementieren:
  1. Kopieren Sie die Bibliothek des Kerberos-Authentifizierungs-Plug-ins in das Client-Plug-in-Verzeichnis.
  2. Geben Sie für den Konfigurationsparameter **clnt_krb_plugin** des Datenbankmanagers den Namen des Kerberos-Plug-ins an. Wenn der Wert des Konfigurationsparameters **clnt_krb_plugin** leer ist, kann der Client die Kerberos-Authentifizierung nicht benutzen. Unter Windows lautet der Standardwert IBMkrb5. Er muss nur für ein angepasstes Kerberos-Plug-in geändert werden. Unter UNIX muss der Wert definiert werden, da der Standardwert hier leer ist. Führen Sie die folgenden Schritte zur Implementierung der Berechtigung auf einem Client, Server oder Gateway durch ein Kerberos-Authentifizierungs-Plug-in aus:
     a. Kopieren Sie die Bibliothek des Kerberos-Authentifizierungs-Plug-ins in das Client-Plug-in-Verzeichnis auf dem Client, Server oder Gateway.
     b. Geben Sie für den Konfigurationsparameter **clnt_krb_plugin** des Datenbankmanagers den Namen des Plug-ins an.
     c. Geben Sie für den Konfigurationsparameter **authentication** des Datenbankmanagers den Wert KERBEROS oder KRB_SERVER_ENCRYPT an.
  3. Optional: Katalogisieren Sie die Datenbanken, auf die der Client zugreifen soll, indem Sie angeben, dass der Client nur ein Kerberos-Authentifizierungs-Plug-in verwendet. Im folgenden Beispiel wird die Datenbank testdb katalogisiert:

     ```
     CATALOG DB testdb AT NODE testnode AUTHENTICATION KERBEROS
             TARGET PRINCIPAL service/host@REALM
     ```

Unterstützung für LDAP-basierte Authentifizierung und Gruppensuchfunktion

Der DB2-Datenbankmanager und DB2 Connect unterstützen die LDAP-basierte Funktionalität zur Authentifizierung und Gruppensuche durch den Einsatz von LDAP-Sicherheits-Plug-in-Modulen (LDAP - Lightweight Directory Access Protocol) sowie durch transparente Authentifizierung über LDAP.

Die Unterstützung für die LDAP-basierte Authentifizierung wurde unter dem Betriebssystem AIX erweitert. Ab DB2 V9.7 Fixpack 1 wurde die transparente LDAP-Unterstützung auch auf die Betriebssysteme Linux, HP-UX und Solaris mit den Versionsständen erweitert, die das DB2-Produkt unterstützt. LDAP ermöglicht nun eine zentrale Verwaltung der Benutzerauthentifizierung und der Gruppenzugehörigkeit durch eine transparente LDAP-Authentifizierung. Sie können DB2-Instanzen zur Authentifizierung von Benutzern und zur Anforderung der Gruppen dieser Benutzer durch das Betriebssystem konfigurieren. Das Betriebssystem führt die Authentifizierung wiederum über einen LDAP-Server aus. Zur Aktivierung der transparenten LDAP-Authentifizierung setzen Sie die Registrierdatenbankvariable **DB2AUTH** auf den Wert OSAUTHDB. Die folgenden Betriebssysteme werden unterstützt:

- AIX
- HP-UX
- Linux
- Solaris

Eine andere Option zur Implementierung der LDAP-basierten Authentifizierung ist die Verwendung von LDAP-Sicherheits-Plug-ins. LDAP-Sicherheits-Plug-in-Module bieten dem DB2-Datenbankmanager die Möglichkeit, Benutzer, die in einem LDAP-

Verzeichnis definiert sind, zu authentifizieren, ohne dass die Benutzer und Gruppen dazu im Betriebssystem mit demselben Versionsstand definiert sein müssen, der auch vom DB2-Produkt unterstützt wird. Die folgenden Betriebssysteme werden unterstützt:

- AIX
- HP-UX auf Itanium-basierten HP Integrity Series-Systemen (IA-64)
- Linux auf IA32-, x64- oder zSeries-Hardware
- Solaris
- Windows

Die folgenden LDAP-Server werden zur Verwendung mit Sicherheits-Plug-in-Modulen unterstützt:

- IBM Lotus Domino LDAP Server Version 8.0 und höher
- IBM Tivoli Directory Server (ITDS) Version 6.2 (mit GSKit 7.0.4.20 und höher) und höher
- Microsoft Active Directory (MSAD) Version 2008 und höher
- Novell eDirectory Version 8.8 und höher
- OpenLDAP-Server Version 2.4 und höher
- Sun Java System Directory Server Enterprise Edition Version 5.2 FP4 und höher
- z/OS Integrated Security Services LDAP Server Version V1R6 und höher

Anmerkung: Wenn Sie die LDAP-Plug-in-Module verwenden, müssen alle Benutzer, die der Datenbank zugeordnet sind, im LDAP-Server definiert sein. Dies schließt sowohl die DB2-Instanzeigner-ID als auch den abgeschirmten Benutzer mit ein. (Diese Benutzer werden in der Regel im Betriebssystem definiert, müssen jedoch auch in LDAP definiert werden.) In ähnlicher Weise gilt, dass bei Verwendung des LDAP-Gruppen-Plug-in-Moduls alle Gruppen, die zur Authentifizierung erforderlich sind, im LDAP-Server definiert sein müssen. Dies schließt die Gruppen SYSADM, SYSMAINT, SYSCTRL und SYSMON mit ein, die in der Datenbankmanagerkonfiguration definiert sind.

DB2-Sicherheits-Plug-in-Module sind für die serverseitige Authentifizierung, die clientseitige Authentifizierung und die Gruppensuche verfügbar. Diese Funktionen werden nachfolgend beschrieben. Abhängig von Ihrer speziellen Umgebung müssen Sie einen, zwei oder alle drei Typen von Plug-in verwenden.

Gehen Sie zur Verwendung von DB2-Sicherheits-Plug-in-Modulen wie folgt vor:

1. Entscheiden Sie, ob Sie Server-, Client- oder Gruppen-Plug-in-Module oder eine Kombination dieser Module benötigen.
2. Konfigurieren Sie die Plug-in-Module, indem Sie Werte in der Konfigurationsdatei für IBM LDAP-Sicherheits-Plug-ins (Standardname: `IBMLDAPSecurity.ini`) definieren. Sie müssen sich wahrscheinlich bei Ihrem LDAP-Administrator nach geeigneten Werten erkundigen.
3. Aktivieren Sie die Plug-in-Module.
4. Testen Sie den Verbindungsaufbau mit verschiedenen LDAP-Benutzer-IDs.

Plug-in zur Serverauthentifizierung

Das Plug-in-Modul zur Serverauthentifizierung führt eine serverseitige Prüfung von Benutzer-IDs und Kennwörtern aus, die von Clients in CONNECT- und ATTACH-Anweisungen angegeben werden. Es stellt außerdem eine Methode bereit, LDAP-Benutzer-IDs DB2-Berechtigungs-IDs zuzuordnen, wenn dies erforderlich ist.

Das Server-Plug-in-Modul ist im Allgemeinen erforderlich, wenn Benutzer für den DB2-Datenbankmanager mit ihrer LDAP-Benutzer-ID und dem zugehörigen Kennwort authentifiziert werden sollen.

Plug-in zur Clientauthentifizierung

Das Plug-in-Modul zur Clientauthentifizierung wird verwendet, wenn die Prüfung von Benutzer-ID und Kennwort auf dem Clientsystem stattfindet. Das heißt, wenn der DB2-Server mit der Einstellung CLIENT des Parameters SRVCON_AUTH oder AUTHENTICATION konfiguriert ist. Der Client prüft alle Benutzer-IDs und Kennwörter, die in CONNECT- oder ATTACH-Anweisungen angegeben werden, und sendet die Benutzer-ID an den DB2-Server. Beachten Sie, dass sich die Clientauthentifizierung nur schwer schützen lässt und im Allgemeinen nicht empfohlen wird.

Das Plug-in-Modul zur Clientauthentifizierung kann außerdem erforderlich sein, wenn sich die Benutzer-IDs des lokalen Betriebssystems auf dem Datenbankserver von den DB2-Berechtigungs-IDs, die diesen Benutzern zugeordnet sind, unterscheiden. Mithilfe des clientseitigen Plug-ins können Sie Benutzer-IDs des lokalen Betriebssystems DB2-Berechtigungs-IDs zuordnen, bevor Sie Berechtigungsprüfungen für lokale Befehle auf dem Datenbankserver (z. B. für `db2start`) durchführen.

Plug-in zum Abrufen von Gruppen

Das Plug-in-Modul zum Abrufen von Gruppen ruft Informationen zur Gruppenzugehörigkeit für einen bestimmten Benutzer aus dem LDAP-Server ab. Es ist erforderlich, wenn Sie Ihre Gruppendefinitionen in LDAP speichern wollen. Das gängigste Szenario weist folgende Merkmale auf:

- Alle Benutzer und Gruppen sind im LDAP-Server definiert.
- Alle Benutzer, die lokal auf dem Datenbankserver definiert sind, sind mit derselben Benutzer-ID im LDAP-Server definiert (einschließlich des Instanzeigners und des abgeschirmten Benutzers).
- Die Kennwortprüfung findet auf dem DB2-Server statt (d. h. in der Konfigurationsdatei des Datenbankmanagers des Servers ist der Parameter AUTHENTICATION oder SRVCON_AUTH auf den Wert SERVER, SERVER_ENCRYPT oder DATA_ENCRYPT gesetzt).

Im Regelfall ist es ausreichend, nur das Plug-in-Modul zur Serverauthentifizierung und das Plug-in-Modul zum Abrufen von Gruppen auf dem Server zu installieren. DB2-Clients benötigen im Normalfall keine Installation des LDAP-Plug-in-Moduls.

Es ist möglich, nur das Plug-in-Modul zum Abrufen von LDAP-Gruppen in Kombination mit einer anderen Form von Authentifizierungs-Plug-in (z. B. Kerberos) zu verwenden. In diesem Fall werden dem Plug-in-Modul zum Abrufen von LDAP-Gruppen die DB2-Berechtigungs-IDs übergeben, die einem Benutzer zugeordnet sind. Das Plug-in-Modul durchsucht das LDAP-Verzeichnis nach einem Benutzer mit einem entsprechenden Attribut AUTHID_ATTRIBUTE und ruft anschließend die Gruppen ab, die diesem Benutzerobjekt zugeordnet sind.

Konfigurieren von transparentem LDAP für Authentifizierung und Gruppensuche (AIX)

Ab DB2 V9.7 werden die transparente LDAP-basierte Authentifizierung und die Gruppensuche unter dem Betriebssystem AIX unterstützt. Zur Aktivierung dieser Unterstützung sind bestimmte Konfigurationsschritte erforderlich.

Vorbereitende Schritte

Bei diesen Schritten wird vorausgesetzt, dass der LDAP-Server dem RFC-Standard 2307 entspricht und für die Speicherung von Benutzer- und Gruppeninformationen konfiguriert ist.

Vorgehensweise

1. Führen Sie die folgenden Schritte aus, um Ihr AIX-Clientsystem für LDAP zu konfigurieren:
   a. Melden Sie sich als Benutzer mit Rootberechtigung an.
   b. Stellen Sie sicher, dass die LDAP-Clientdateigruppe auf Ihrem AIX-System installiert ist. AIX arbeiten mit allen drei Versionen der LDAP-Clients: ITDS V5.2, das im Lieferumfang von AIX V5.3 enthalten ist, ITDS V6.1, das im Lieferumfang von AIX V6.1 enthalten ist und ITDS V6.2, das im Lieferumfang des AIX-Erweiterungspakets enthalten ist. Im Folgenden sind die ITDS V5.2-Dateigruppen aufgelistet, die auf einem AIX V5.3-System installiert werden:

```
$ lslpp -l "ldap*"
  Dateigruppe                Version  Status     Beschreibung
  ----------------------------------------------------------------------------
Pfad: /usr/lib/objrepos
  ldap.client.adt            5.2.0.0  COMMITTED  Verzeichnis Client SDK
  ldap.client.rte            5.2.0.0  COMMITTED  Verzeichnis Client Runtime (No
                                                 SSL)
  ldap.html.en_US.config     5.2.0.0  COMMITTED  Verzeichnis Install/Config
                                                 Gd-U.S. English
  ldap.html.en_US.man        5.2.0.0  COMMITTED  Verzeichnis Man Pages - U.S.
                                                 English  ldap.msg.en_US
                             5.2.0.0  COMMITTED  Verzeichnis Messages - U.S.
                                                 English
Pfad: /etc/objrepos
  ldap.client.rte            5.2.0.0  COMMITTED  Verzeichnis Client Runtime
                                                 (No SSL)
```

   c. Konfigurieren Sie den Client mit dem Befehl **mksecldap** und der Option **-c**. Weitere Informationen zur Verwendung des Befehls **mksecldap** zum Konfigurieren des Clients finden Sie unter http://publib.boulder.ibm.com/infocenter/pseries/v5r3/index.jsp?topic=/com.ibm.aix.security/doc/security/setup_ldap_sec_info_server.htm.
   d. Aktualisieren Sie die Standardzeilengruppe in der Datei `/etc/security/user`.

      Nachdem Sie sich vergewissert haben, dass LDAP ordnungsgemäß konfiguriert und das LDAP-Verzeichnis mit Benutzern gefüllt ist, müssen Sie den Standardbenutzer für die Verwendung von LDAP festlegen. Dadurch wird sichergestellt, dass Sie sich mit jeder nicht eingeschränkten Benutzer-ID aus dem LDAP-Verzeichnis bei dem AIX-Client anmelden können.

      Mit den Attributen **SYSTEM** und **REGISTRY** in der Datei `/etc/security/user` können die Authentifizierungsmethode und die Datenbank für die Benutzerverwaltung angegeben werden. Um die Authentifizierung und Benutzerverwaltung über LDAP zu aktivieren, setzen Sie die Attribute **SYSTEM** und **REGISTRY** in der Standardzeilengruppe auf 'LDAP'. Beispiel:

```
chsec -f /etc/security/user -s default -a "SYSTEM=LDAP or files"
chsec -f /etc/security/user -s default -a "REGISTRY=LDAP"
```

      Weitere Informationen zu den Attributen **SYSTEM** und **REGISTRY** in der Zeilengruppe finden Sie unter http://publib.boulder.ibm.com/infocenter/aix/v6r1/index.jsp?topic=/com.ibm.aix.files/doc/aixfiles/user.htm?.

Weitere Informationen enthält das Redbook mit dem Titel 'Integrating AIX into Heterogeneous LDAP Environments' unter folgender Adresse: http://www.redbooks.ibm.com/abstracts/sg247165.html

2. Gehen Sie wie folgt vor, um die transparente LDAP-Authentifizierung in Ihrer DB2-Instanz zu konfigurieren:
   a. Setzen Sie die Registrierdatenbankvariable **DB2AUTH** auf den Wert OSAUTHDB. Führen Sie als Benutzer mit SYSADM-Berechtigung den Befehl **db2set DB2AUTH=OSAUTHDB** aus.
   b. Legen Sie mit dem Befehl **UPDATE DBM CFG** für die Authentifizierung für die Datenbankserverinstanz einen der folgenden Werte fest:
      - SERVER
      - SERVER_ENCRYPT
      - DATA_ENCRYPT
   c. Stellen Sie sicher, dass Sie die Standardwerte `Client Userid-Password Plugin (clnt_pw_plugin)`, `Server Userid-Password Plugin (srvcon_pw_plugin)` und `Group Plugin (group_plugin)` verwenden.
   d. Starten Sie die DB2-Instanz erneut.

Aspekte für die Verwendung verschiedener Authentifizierungsmethoden

Bei der transparenten LDAP-basierten Authentifizierung und der Gruppensuche unter AIX wird auch die Kerberos-Authentifizierung unterstützt.

AIX wurde modifiziert und ermöglicht jetzt auch die Verwendung der Kerberos-Authentifizierung mit transparentem LDAP. Die folgenden Ergänzungen in `/usr/lib/security/methods.cfg` und `/etc/security/users` sind erforderlich, um Konten an verschiedenen Positionen zu verwalten und verschiedene Authentifizierungsmethoden (z. B. Kerberos) zu verwenden.

In `/usr/lib/security/methods.cfg` muss Folgendes angegeben sein, damit die Authentifizierung über Dateien, LDAP und Kerberos verwendet werden kann.

Anmerkung: KRB5A ermöglicht die Verwendung von Microsoft Active Directory als Kerberos Key-Distribution-Center (KDC).

Für LDAP:

```
program = /usr/lib/security/LDAP
program_64 =/usr/lib/security/LDAP64
```

Für KRB5A:

```
program = /usr/lib/security/KRB5A
program_64 = /usr/lib/security/KRB5A_64
options = tgt_verify=no,authonly,is_kadmind_compat=no
```

Für KRB5:

```
program = /usr/lib/security/KRB5
program_64 = /usr/lib/security/KRB5_64
options = kadmind=no
```

Für KRB5Afiles:

```
options = db=BUILTIN,auth=KRB5A
```

Für KRB5files:

```
options = db=BUILTIN,auth=KRB5
```

Für KRB5ALDAP:

```
options = db=LDAP,auth=KRB5A
```

Für KRB5LDAP:

```
options = db=LDAP,auth=KRB5
```

Beispiel

Das folgende Beispiel zeigt vier mit verschiedenen Methoden verwaltete Konten. Jedes Konto verwendet unterschiedliche Authentifizierungsmethoden.

Wenn das Konto für den Benutzer 'frank' in einer Datei gespeichert ist und über Dateien authentifiziert wird, müsste der Abschnitt für den Benutzer 'frank' im Verzeichnis /etc/security/users wie folgt lauten:

```
frank:
        SYSTEM = files
        registry = files
```

Wenn das Konto für die Benutzerin 'karen' in einer Datei gespeichert ist und über Kerberos authentifiziert wird, müsste der Abschnitt für die Benutzerin 'karen' im Verzeichnis /etc/security/users wie folgt lauten:

```
karen:
        SYSTEM = KRB5files
        registry = KRB5files
```

Wenn das Konto des Benutzers 'luke' in LDAP gespeichert ist und über Kerberos authentifiziert wird, müsste der Abschnitt für den Benutzer 'luke' im Verzeichnis /etc/security/users wie folgt lauten:

```
luke:
        SYSTEM = KRB5LDAP
        registry = KRB5LDAP
```

Wenn das Konto der Benutzerin 'lucy' in LDAP gespeichert ist und über LDAP authentifiziert wird, müsste der Abschnitt für die Benutzerin 'lucy' im Verzeichnis /etc/security/users wie folgt lauten.

```
lucy:
        SYSTEM = LDAP
        registry = LDAP
```

Mit dem folgenden Befehl können Sie feststellen, ob ein Benutzer in LDAP definiert ist:

```
$ lsuser -R LDAP lucy
lucy id=1234 pgrp=staff groups=staff home=/home/lucy shell=/bin/ksh registry=LDAP
```

Konfigurieren von transparentem LDAP für Authentifizierung und Gruppensuche (Linux)

Um ab DB2 Version 9.7 Fixpack 1 sicherzustellen, dass der DB2-Datenbankserver die LDAP-basierte Authentifizierung auf dem Betriebssystem Linux transparent verwendet, verwenden Sie Pluggable Authentication Modules (PAM). Ihr LDAP-Server muss bereits so konfiguriert sein, dass er Benutzer- und Gruppeninformationen speichert.

Vorbereitende Schritte

Um die Unterstützung für transparentes LDAP bei der DB2-Datenbank zu aktivieren, führen Sie folgende Aktivitäten aus:

1. Konfigurieren Sie Ihr Betriebssystem zur Benutzerauthentifizierung unter Verwendung von PAM (Pluggable Authentication Module).
2. Konfigurieren Sie Ihre DB2-Instanz.

Bei diesen Schritten wird vorausgesetzt, dass der LDAP-Server dem RFC-Standard 2307 entspricht.

Vorgehensweise

1. Führen Sie folgende Schritte aus, um Ihr Betriebssystem für LDAP und PAM zu konfigurieren:
   a. Melden Sie sich als Benutzer mit Rootberechtigung an.
   b. Stellen Sie sicher, dass die Pakete `nss_ldap` und `pam_ldap` installiert sind. Diese beiden Pakete sind als `libnss_ldap.so` und `libpam_ldap.so` im Verzeichnis `/lib(64)` oder `/usr/lib(64)` enthalten.
   c. Richten Sie Ihr Betriebssystem so ein, dass es als LDAP-Clientmaschine fungiert, indem Sie an der Datei `/etc/ldap.conf` die erforderlichen Änderungen vornehmen, um das *Binden* des Betriebssystems an den LDAP-Server zu ermöglichen. Das folgende Beispiel zeigt, wie die Datei `/etc/ldap.conf` aussehen könnte:

```
host <host>             # Adresse des LDAP-Servers
base <basis>             # DN der Suchbasis
rootbinddn <binddn>     # Binde-DN für das Binden an LDAP
ldap_version 3             # LDAP-Version
pam_login_attribute uid    # Benutzer-ID-Attribut für PAM-Benutzersuche
nss_base_group <gruppe>  # nsswitch-Konfiguration der Gruppensuche
```

   d. Definieren Sie Ihr Kennwort in der Datei `/etc/ldap.secret`. Nur der Rootbenutzer darf diese Datei lesen oder ändern können.
   e. Erstellen oder ändern Sie die PAM-Konfigurationsdatei im Verzeichnis `/etc/pam.d/db2`. Nur der Rootbenutzer darf diese Datei lesen oder ändern können.

```
auth     sufficient pam_unix2.so
auth     required   pam_ldap.so     use_first_pass
account  sufficient pam_unix2.so
account  required   pam_ldap.so
password required   pam_pwcheck.so
password sufficient pam_unix2.so    use_authtok use_first_pass
password required   pam_ldap.so     use_first_pass
session  required pam_unix2.so
```

   Für Red Hat Enterprise Linux 5 ändern Sie die Konfigurationsdatei wie folgt:

```
#%PAM-1.0

auth     required   /lib/security/$ISA/pam_env.so
auth     sufficient /lib/security/$ISA/pam_unix.so likeauth nullok
auth    sufficient  /lib/security/$ISA/pam_ldap.so use_first_pass
auth     required   /lib/security/$ISA/pam_deny.so
account  required   /lib/security/$ISA/pam_unix.so
account  sufficient /lib/security/$ISA/pam_succeed_if.so uid < 100 quiet
account  sufficient /lib/security/$ISA/pam_ldap.so
account  required   /lib/security/$ISA/pam_permit.so
password requisite  /lib/security/$ISA/pam_cracklib.so retry=3
  dcredit=-1 ucredit=-1
password sufficient /lib/security/$ISA/pam_unix.so nullok use_authtok
  md5 shadowremember=3
```

```
password sufficient /lib/security/$ISA/pam_ldap.so  use_first_pass
password required   /lib/security/$ISA/pam_deny.so
session  required   /lib/security/$ISA/pam_limits.so
session  required   /lib/security/$ISA/pam_unix.so
```

f. Richten Sie Ihr Linux-System für das Ausführen der Gruppensuche durch LDAP ein. Lokalisieren Sie die Einträge **group** und **passwd** in der Datei /etc/nsswitch.conf und stellen Sie sicher, dass ldap als Suchmethode eingegeben ist. Hier finden Sie Beispieleinträge für **group** und **passwd**:

```
group:          files ldap
passwd:         files ldap
```

2. Um Ihre DB2-Instanz für die Nutzung der transparenten LDAP-Authentifizierung zu konfigurieren, führen Sie folgende Aktivitäten durch:
   a. Setzen Sie die Registrierdatenbankvariable **DB2AUTH** auf den Wert OSAUTHDB. Setzen Sie als Benutzer mit SYSADM-Berechtigung den folgenden Befehl ab:

      ```
      db2set DB2AUTH=OSAUTHDB
      ```

   b. Legen Sie für die Authentifizierung auf dem Server einen der folgenden Werte fest:
      - SERVER
      - SERVER_ENCRYPT
      - DATA_ENCRYPT

   c. Stellen Sie sicher, dass Sie die Standardwerte Client Userid-Password Plugin (clnt_pw_plugin), Server Userid-Password Plugin (srvcon_pw_plugin) und Group Plugin (group_plugin) verwenden.
   d. Starten Sie die DB2-Instanz erneut.

Konfigurieren von transparentem LDAP für Authentifizierung und Gruppensuche (HP-UX)

Um ab DB2 V9.7 Fixpack 1 sicherzustellen, dass Ihr DB2-Datenbankserver die LDAP-basierte Authentifizierung auf dem Betriebssystem HP-UX transparent verwendet, müssen Sie Pluggable Authentication Modules (PAM) einsetzen. Ihr LDAP-Server muss bereits so konfiguriert sein, dass er Benutzer- und Gruppeninformationen speichert.

Vorbereitende Schritte

Bei dieser Prozedur wird vorausgesetzt, dass der LDAP-Server dem RFC-Standard 2307 entspricht.

Vorgehensweise

1. Wenn Sie IBM Tivoli Directory Server (ITDS) Version 6.1 verwenden, richten Sie den LDAP-Server ein, bevor das HP-UX-System eine Verbindung zu ihm herstellen kann. Gehen Sie wie folgt vor, um Ihren LDAP-Server auf dem Betriebssystem HP-UX zu konfigurieren:
   a. Melden Sie sich beim LDAP-Server als Benutzer mit Rootberechtigung an.
   b. Setzen Sie den Befehl **idsldapadd** wie folgt ab:

      ```
      idsldapadd -D <root> -w <kennwort> -h <hostname> -p <port> -c -i
      duaconfigschema.ldif
      Dabei gilt:
      <root> - Binde-DN für das Binden an LDAP
      <kennwort> - Kennwort für Binde-DN
      <hostname> - Hostname des LDAP-Servers
      <port> - Port des LDAP-Servers. Standardmäßig wird 389 verwendet.
      <schema.ldif> - LDIF-Datei mit DUAConfigProfle-Schema
      ```

      Die in Datei duaconfigschema.ldif aufgeführte Objektklasse wird automatisch zum LDAP-Server hinzugefügt. Dazu wird bei Einsatz von Netscape

oder Red Hat Directory Server das LDAP-UX-Setup-Programm verwendet. Bei Einsatz von ITDS muss die Objektklasse manuell hinzugefügt werden, bevor das LDAP-UX-Setup-Programm für den HP-UX-Client ausgeführt wird.

2. Führen Sie folgende Schritte aus, um Ihr Betriebssystem für LDAP und PAM zu konfigurieren:
   a. Melden Sie sich als Benutzer mit Rootberechtigung an.
   b. Installieren Sie den LDAP-UX Client Service und führen Sie das LDAP-UX-Setup-Programm aus. Es erscheint folgende Anzeige:

      ```
      [ctrl-B]=Go Back screen 2
      Hewlett-Packard Company
      LDAP-UX Client Services Setup Program
      ------------------------------------------------------------------------
      Select which Directory Server you want to connect to:
      1. Netscape or Red Hat Directory
      2. Windows 2000/2003/2003 R2 Active Directory
      To accept the default shown in brackets, press the Return key.
      Directory Server: [1]:
      ```

      Wählen Sie Option 1 aus, als wollten Sie eine Verbindung zu Netscape oder Red Hat Directory Server herstellen, und befolgen Sie die angezeigten Anweisungen. Weitere Informationen zur Installation von LDAP-UX finden Sie in der Veröffentlichung *LDAP-UX Client Services B.04.15 Administrator's Guide*.
   c. Bearbeiten Sie die PAM-Konfigurationsdatei im Verzeichnis /etc/pam.conf. Fügen Sie folgenden Text in die Datei ein:

      ```
      db2 auth required       libpam_hpsec.so.1
      db2 auth sufficient         libpam_unix.so.1
      db2 auth required       libpam_ldap.so.1 use_first_pass
      ```

      Die bisherige Konfiguration prüft zunächst Benutzer-ID und Kennwort gegen die entsprechenden Informationen des lokalen Dateisystems. Eine LDAP-Suche wird nur ausgeführt, wenn der Benutzer nicht gefunden wird oder wenn die Authentifizierung beim lokalen Dateisystem fehlschlägt.
   d. Richten Sie Ihr HP-UX-System für das Ausführen der Gruppensuche durch LDAP ein. Lokalisieren Sie die Einträge **group** und **passwd** in der Datei /etc/nsswitch.conf und stellen Sie sicher, dass ldap als Suchmethode eingegeben ist. Hier finden Sie Beispieleinträge für **group** und **passwd**:

      ```
      group:          files ldap
      passwd:         files ldap
      ```

3. Um Ihre DB2-Instanz für die Nutzung der transparenten LDAP-Authentifizierung zu konfigurieren, führen Sie folgende Aktivitäten durch:
   a. Setzen Sie die Registrierdatenbankvariable **DB2AUTH** auf den Wert OSAUTHDB. Setzen Sie als Benutzer mit SYSADM-Berechtigung den folgenden Befehl ab:

      ```
      db2set DB2AUTH=OSAUTHDB
      ```
   b. Legen Sie für die Authentifizierung auf dem Server einen der folgenden Werte fest:
      - SERVER
      - SERVER_ENCRYPT
      - DATA_ENCRYPT
   c. Stellen Sie sicher, dass Sie die Standardwerte Client Userid-Password Plugin (clnt_pw_plugin), Server Userid-Password Plugin (srvcon_pw_plugin) und Group Plugin (group_plugin) verwenden.
   d. Starten Sie die DB2-Instanz erneut.

Konfigurieren von transparentem LDAP für Authentifizierung und Gruppensuche (Solaris)

Um ab DB2 V9.7 Fixpack 1 sicherzustellen, dass Ihr DB2-Datenbankserver die LDAP-basierte Authentifizierung auf dem Betriebssystem Solaris transparent verwendet, müssen Sie Pluggable Authentication Modules (PAM) einsetzen. Ihr LDAP-Server muss bereits so konfiguriert sein, dass er Benutzer- und Gruppeninformationen speichert.

Vorbereitende Schritte

Bei dieser Prozedur wird vorausgesetzt, dass der LDAP-Server dem RFC-Standard 2307 entspricht.

Informationen zu diesem Vorgang

In dieser Task werden die Schritte beschrieben, die für Solaris 10 gültig sind. Die Anweisungen können für andere Versionen des Betriebssystems Solaris leicht abweichen.

Vorgehensweise

1. Führen Sie folgende Schritte aus, um Ihr Betriebssystem für LDAP und PAM zu konfigurieren:
   a. Melden Sie sich als Benutzer mit Rootberechtigung an.
   b. Stellen Sie sicher, dass die Pakete `nss_ldap` und `pam_ldap` installiert sind. Diese beiden Pakete stehen als `nss_ldap.so` und `pam_ldap.so` in den Verzeichnissen `/usr/lib` bzw. `/usr/lib/security`.
   c. Richten Sie Ihr Betriebssystem so ein, dass es als LDAP-Clientmaschine fungiert. Sie können die Schnittstelle `ldapclient(1M)` verwenden, um den Befehl **`ldapclient`** abzusetzen. Das folgende Beispiel zeigt die mögliche Ausgabe:

```
ldapclient manual -a credentialLevel=proxy \
-a authenticationMethod=simple \
-a proxyDN=<root> \
-a proxyPassword=<kennwort> \
-a defaultSearchBase=<basis> \
-a serviceSearchDescriptor=group:<gruppe> \
-a domainName=<domäne> \
-a defaultServerList=<IP>
```

   Dabei gilt:

   <root> Binde-DN für das Binden an LDAP. Dies ist der DN des Benutzereintrags im LDAP-Server, der berechtigt ist, den LDAP-Server nach Benutzerkonten und -gruppen zu durchsuchen.

   <kennwort>
   Kennwort für Binde-DN

   <basis>
   DN der Suchbasis. (Muss eine Ebene über dem Benutzer- und Gruppeneintrag liegen.)

   <gruppe>
   Basis-DN für die Position, an der die Gruppeninformationen gespeichert sind

   <domäne>
   Domänenname für den LDAP-Server

<IP> IP-Adresse für den LDAP-Server

Weitere Informationen finden Sie im Handbuch zu `ldapclient(1M)`.

d. Bearbeiten Sie die PAM-Konfigurationsdatei im Verzeichnis `/etc/pam.conf`. Fügen Sie folgenden Text in die Datei ein:

```
db2 auth requisite          pam_authtok_get.so.1
db2 auth required           pam_unix_cred.so.1
db2 auth sufficient         pam_unix_auth.so.1
db2 auth required           pam_ldap.so.1
```

Die bisherige Konfiguration prüft zunächst Benutzer-ID und Kennwort gegen die entsprechenden Informationen des lokalen Dateisystems. Eine LDAP-Suche wird nur ausgeführt, wenn der Benutzer nicht gefunden wird oder wenn die Authentifizierung beim lokalen Dateisystem fehlschlägt.

e. Richten Sie Ihr Solaris-System für das Ausführen der Gruppensuche durch LDAP ein. Lokalisieren Sie die Einträge **group** und **passwd** in der Datei `/etc/nsswitch.conf` und stellen Sie sicher, dass `ldap` als Suchmethode eingegeben ist. Hier finden Sie Beispieleinträge für **group** und **passwd**:

```
group:          files ldap
passwd:         files ldap
```

2. Um Ihre DB2-Instanz für die Nutzung der transparenten LDAP-Authentifizierung zu konfigurieren, führen Sie folgende Schritte aus:
   a. Setzen Sie die Registrierdatenbankvariable **DB2AUTH** auf den Wert `OSAUTHDB`. Setzen Sie als Benutzer mit `SYSADM`-Berechtigung den folgenden Befehl ab:

      ```
      db2set DB2AUTH=OSAUTHDB
      ```

   b. Legen Sie für die Authentifizierung auf dem Server einen der folgenden Werte fest:
      - SERVER
      - SERVER_ENCRYPT
      - DATA_ENCRYPT

   c. Stellen Sie sicher, dass Sie die Standardwerte `Client Userid-Password Plugin (clnt_pw_plugin)`, `Server Userid-Password Plugin (srvcon_pw_plugin)` und `Group Plugin (group_plugin)` verwenden.
   d. Starten Sie die DB2-Instanz erneut.

Konfigurieren der LDAP-Plug-in-Module

Zur Konfiguration der LDAP-Plug-in-Module (LDAP, Lightweight Directory Access Protocol) müssen Sie Ihre Konfigurationsdatei für IBM LDAP-Sicherheits-Plug-ins aktualisieren, um sie an Ihre Umgebung anzupassen. In den meisten Fällen müssen Sie sich bei Ihrem LDAP-Administrator über die erforderlichen Konfigurationswerte informieren.

Die Konfigurationsdatei für IBM LDAP-Sicherheits-Plug-ins hat standardmäßig folgenden Namen und folgende Position:

- Unter UNIX: `INSTHOME/sqllib/cfg/IBMLDAPSecurity.ini`
- Unter Windows: `%DB2PATH%\cfg\IBMLDAPSecurity.ini`

Optional können Sie die Position dieser Datei mit der Umgebungsvariablen 'DB2LDAPSecurityConfig' angeben. Unter Windows sollten Sie die Variable 'DB2LDAPSecurityConfig' in der globalen Systemumgebung definieren, um sicherzustellen, dass sie vom DB2-Service berücksichtigt wird.

Die folgenden Tabellen enthalten Informationen, die Ihnen bei der Bestimmung geeigneter Konfigurationswerte helfen.

Tabelle 33. Serverbezogene Werte

Parameter	Beschreibung
LDAP_HOST	Der Name Ihres bzw. Ihrer LDAP-Server. Dieser Wert ist eine durch Leerzeichen getrennte Liste von Hostnamen oder IP-Adressen (mit optionaler Portnummer) für LDAP-Server. Beispiel: `host1[:port] [host2:[port2] ... ]` Die Standardportnummer ist 389, oder 636, wenn SSL aktiviert ist.
ENABLE_SSL	Zum Aktivieren der SSL-Unterstützung setzen Sie ENABLE_SSL auf TRUE. (Das GSKit muss installiert sein.) Dieser Parameter ist optional; sein Standardwert ist FALSE (keine SSL-Unterstützung).
SSL_KEYFILE	Der Pfad für die SSL-Schlüsselringdatei. Eine Schlüsseldatei (keyfile) ist nur erforderlich, wenn der LDAP-Server ein Zertifikat verwendet, dem von Ihrer GSKit-Installation nicht automatisch vertraut wird. Beispiel: `SSL_KEYFILE = /home/db2inst1/IBMLDAPSecurity.kdb`
SSL_PW	Das Kennwort für die SSL-Schlüsselringdatei. Beispiel: `SSL_PW = kennwort-für-schlüsselringdatei`

Tabelle 34. Benutzerbezogene Werte

Parameter	Beschreibung
USER_ OBJECTCLASS	Die LDAP-Objektklasse für Benutzer. Setzen Sie USER_OBJECTCLASS im Allgemeinen auf `inetOrgPerson` (den Benutzer für Microsoft Active Directory) Beispiel: `USER_OBJECTCLASS = inetOrgPerson`
USER_BASEDN	Der LDAP-Basis-DN für die Suche nach Benutzern. Wenn nicht angegeben, beginnen Benutzersuchen im Stammelement des LDAP-Verzeichnisses. Einige LDAP-Server setzen voraus, dass Sie einen Wert für diesen Parameter angeben. Beispiel: `USER_BASEDN = o=ibm`

USERID_ ATTRIBUTE	Das LDAP-Benutzerattribut, das die Benutzer-ID darstellt.
	Das Attribut USERID_ATTRIBUTE wird mit USER_OBJECTCLASS und USER_BASEDN kombiniert (falls angegeben), um einen LDAP-Suchfilter zu bilden, wenn ein Benutzer eine DB2-Anweisung CONNECT mit einer nicht qualifizierten Benutzer-ID absetzt.
	Beispiel: Bei `USERID_ATTRIBUTE = uid` wird die folgende Anweisung abgesetzt:
	`db2 connect to MYDB user bob using bobpass`
	In diesem Fall wird der folgende Suchfilter gebildet:
	`&(objectClass=inetOrgPerson)(uid=bob)`

AUTHID_ ATTRIBUTE	Das LDAP-Benutzerattribut, das die DB2-Berechtigungs-ID darstellt. Dieser Wert ist in der Regel gleich USERID_ATTRIBUTE. Beispiel: `AUTHID_ATTRIBUTE = uid`

Tabelle 35. Gruppenbezogene Werte

Parameter	Beschreibung
GROUP_ OBJECTCLASS	Die LDAP-Objektklasse für Gruppen. Dieser Wert ist in der Regel `groupOfNames` oder `groupOfUniqueNames` (für Microsoft Active Directory ist dies `group`). Beispiel: `GROUP_OBJECTCLASS = groupOfNames`

Tabelle 35. Gruppenbezogene Werte (Forts.)

Parameter	Beschreibung
GROUP_BASEDN	Der LDAP-Basis-DN für die Suche nach Gruppen. Wenn nicht angegeben, beginnen Gruppensuchen im Stammelement des LDAP-Verzeichnisses. Einige LDAP-Server setzen voraus, dass Sie einen Wert für diesen Parameter angeben. Beispiel: `GROUP_BASEDN = o=ibm`
GROUPNAME_ ATTRIBUTE	Das LDAP-Gruppenattribut, das den Namen der Gruppe darstellt. Beispiel: `GROUPNAME_ATTRIBUTE = cn`

GROUP_LOOKUP_ METHOD	Bestimmt die Methode zur Suche der Gruppenzugehörigkeiten für einen Benutzer. Mögliche Werte:
	• SEARCH_BY_DN: Gibt an, dass nach Gruppen zu suchen ist, die den Benutzer als Mitglied auflisten. Die Zugehörigkeit wird durch das Gruppenattribut angegeben, das als GROUP_LOOKUP_ATTRIBUTE definiert ist (in der Regel `member` oder `uniqueMember`).
	• USER_ATTRIBUTE: In diesem Fall werden die Gruppen eines Benutzers als Attribute des Benutzerobjekts selbst aufgelistet. Diese Einstellung gibt an, dass nach dem Benutzerattribut zu suchen ist, das als GROUP_LOOKUP_ATTRIBUTE definiert ist, um die Gruppen des Benutzers abzurufen (in der Regel `memberOf` für Microsoft Active Directory oder `ibm-allGroups` für IBM Tivoli Directory Server).
	Zum Beispiel:
	`GROUP_LOOKUP_METHOD = SEARCH_BY_DN`
	`GROUP_LOOKUP_METHOD = USER_ATTRIBUTE`

GROUP_LOOKUP_ ATTRIBUTE	Der Name des Attributs, das zur Ermittlung der Gruppenzugehörigkeit wie für GROUP_LOOKUP_METHOD beschrieben verwendet wird.
	
	Zum Beispiel:
	`GROUP_LOOKUP_ATTRIBUTE = member`
	`GROUP_LOOKUP_ATTRIBUTE = ibm-allGroups`

NESTED_GROUPS	Wenn NESTED_GROUPS auf TRUE gesetzt ist, sucht der DB2-Datenbankmanager rekursiv nach Gruppenzugehörigkeiten, indem er versucht, die Gruppenzugehörigkeiten für jede gefundene Gruppe zu ermitteln. Zyklen (Beispiel: A gehört zu B und B gehört zu A) werden ordnungsgemäß verarbeitet. Dieser Parameter ist optional und hat den Standardwert FALSE.

Tabelle 36. Verschiedene Werte

Parameter	Beschreibung

SEARCH_DN, SEARCH_PW	Wenn Ihr LDAP-Server keinen anonymen Zugriff unterstützt oder wenn ein anonymer Zugriff für die Suche nach Benutzern oder Gruppen nicht ausreicht, können Sie optional einen definierten Namen (DN) und ein Kennwort definieren, die zur Ausführung von Suchen verwendet werden.
	
	Zum Beispiel:
	`SEARCH_DN = cn=root`
	`SEARCH_PW = rootkennwort`

Tabelle 36. Verschiedene Werte (Forts.)

Parameter	Beschreibung
DEBUG	Setzen Sie DEBUG auf den Wert TRUE, wenn zusätzliche Informationen in die **db2diag**-Protokolldateien geschrieben werden sollen, um ein Debugging für LDAP zu unterstützen. Die meisten Zusatzinformationen werden bei folgender Stufe protokolliert: DIAGLEVEL 4 (INFO). DEBUG hat den Standardwert FALSE.

Aktivieren der LDAP-Plug-in-Module

Kompilierte binäre LDAP-Plug-in-Module befinden sich in Ihrem DB2-Instanzverzeichnis.

Die folgenden Tabellen zeigen, wo sich die LDAP-Plug-in-Module in Ihrer DB2-Instanz befinden.

Tabelle 37. Für UNIX- und Linux-Systeme (64 Bit)

Typ des Plug-in-Moduls	Position
Server	`/sqllib/security64/plugin/IBM/server`
Client	`/sqllib/security64/plugin/IBM/client`
Gruppe	`/sqllib/security64/plugin/IBM/group`

Tabelle 38. Für UNIX- und Linux-Systeme (32 Bit)

Typ des Plug-in-Moduls	Position
Server	`/sqllib/security32/plugin/IBM/server`
Client	`/sqllib/security32/plugin/IBM/client`
Gruppe	`/sqllib/security32/plugin/IBM/group`

Tabelle 39. Für Windows-Systeme (64- und 32 Bit)

Typ des Plug-in-Moduls	Position
Server	`%DB2PATH%\security\plugin\IBM\`*instanzname*`\server`
Client	`%DB2PATH%\security\plugin\IBM\`*instanzname*`\client`
Gruppe	`%DB2PATH%\security\plugin\IBM\`*instanzname*`\group`

Anmerkung: Die Plug-in-Module für 64-Bit-Windows-Systeme haben die Ziffern "64" im Dateinamen.

Verwenden Sie den DB2-Befehlszeilenprozessor, um die Datenbankmanagerkonfiguration zur Aktivierung der erforderlichen Plug-in-Module zu aktualisieren:

- Für das Server-Plug-in-Modul:
  ```
  UPDATE DBM CFG USING SRVCON_PW_PLUGIN IBMLDAPauthserver
  ```
- Für das Client-Plug-in-Modul:
  ```
  UPDATE DBM CFG USING CLNT_PW_PLUGIN IBMLDAPauthclient
  ```

- Für das Gruppen-Plug-in-Modul:

```
UPDATE DBM CFG USING GROUP_PLUGIN IBMLDAPgroups
```

Beenden Sie alle laufenden Back-End-Prozesse des DB2-Befehlszeilenprozessors, indem Sie den Befehl **db2 terminate** ausführen. Führen Sie anschließend die Befehle **db2stop** und **db2start** aus, um die Instanz zu stoppen und erneut zu starten.

Verbindungsaufbau mit einer LDAP-Benutzer-ID

Nachdem die LDAP-Sicherheits-Plug-ins in einer DB2-Instanz konfiguriert wurden, kann ein Benutzer eine Verbindung zu den Datenbanken mit einer Reihe verschiedener Benutzerzeichenfolgen herstellen.

Die Position eines Objekts in einem LDAP-Verzeichnis (Lightweight Directory Access Protocol) wird durch den definierten Namen (DN) festgelegt. Ein definierter Name (DN) besteht in der Regel aus mehreren Teilen, die eine Art von Hierarchie darstellen. Beispiel:

```
cn=John Smith, ou=Sales, o=WidgetCorp
```

Die *Benutzer-ID* eines Benutzers wird durch ein Attribut definiert, das dem Benutzerobjekt zugeordnet ist (in der Regel das Attribut **uid**). Dabei kann es sich um eine einfache Zeichenfolge (z. B. jsmith) oder um eine Zeichenfolge nach Art einer E-Mail-Adresse (z. B. jsmith@sales.widgetcorp.com), die einen Teil der Organisationshierarchie widerspiegelt, handeln.

Die *Berechtigungs-ID* eines Benutzers in DB2 ist der Name, der diesem Benutzer innerhalb der DB2-Datenbank zugeordnet wird.

In der Vergangenheit wurden Benutzer in der Regel im Hostbetriebssystem des Servers definiert, und die Benutzer-ID und die Berechtigungs-ID stimmten überein (obwohl für die Berechtigungs-ID normalerweise Großbuchstaben verwendet werden). Die DB2-LDAP-Plug-in-Module geben Ihnen die Möglichkeit, der Benutzer-ID und der Berechtigungs-ID verschiedene Attribute des LDAP-Benutzerobjekts zuzuordnen. In den meisten Fällen können die Benutzer-ID und die Berechtigungs-ID dieselbe Zeichenfolge sein, und Sie können denselben Attributnamen für USERID_ATTRIBUTE und AUTHID_ATTRIBUTE verwenden. Wenn das Benutzer-ID-Attribut in Ihrer Umgebung in der Regel zusätzliche Informationen enthält, die nicht auf die Berechtigungs-ID übertragen werden sollen, können Sie ein anderes Attribut AUTHID_ATTRIBUTE in der Plug-in-Initialisierungsdatei konfigurieren. Der Wert des Attributs AUTHID_ATTRIBUTE wird aus dem Server abgerufen und als interne DB2-Darstellung des Benutzers verwendet.

Wenn Ihre LDAP-Benutzer-IDs zum Beispiel wie E-Mail-Adressen (z. B. jsmith@sales.widgetcorp.com) aussehen und Sie die Verwendung nur des Benutzerteils (jsmith) als DB2-Berechtigungs-ID vorziehen, können Sie wie folgt vorgehen:

1. Ordnen Sie allen Benutzerobjekten auf Ihrem LDAP-Server ein neues Attribut zu, das nur den kürzeren Namen enthält.
2. Konfigurieren Sie das Attribut AUTHID_ATTRIBUTE mit dem Namen dieses neuen Attributs.

Benutzer können anschließend eine Verbindung zu einer DB2-Datenbank herstellen, indem sie ihre vollständige LDAP-Benutzer-ID und das Kennwort angeben. Beispiel:

```
db2 connect to MYDB user 'jsmith@sales.widgetcorp.com' using 'kennwort'
```

Intern verwendet der DB2-Datenbankmanager für den Benutzer jedoch den kurzen Namen, der mithilfe des Attributs AUTHID_ATTRIBUTE abgerufen wird (in diesem Beispiel: jsmith).

Nach dem Aktivieren und Konfigurieren eines LDAP-Plug-in-Moduls kann ein Benutzer eine Verbindung zu einer DB2-Datenbank mithilfe einer Reihe verschiedener Zeichenfolgen herstellen:

- Ein vollständiger definierter Name (DN). Zum Beispiel:

  ```
  connect to MYDB user 'cn=John Smith, ou=Sales, o=WidgetCorp'
  ```

- Ein Teil eines definierten Namens (Teil-DN), vorausgesetzt, dass eine Suche im LDAP-Verzeichnis mit dem Teil-DN und dem entsprechenden Basis-DN (falls definiert) genau eine Übereinstimmung liefert. Zum Beispiel:

  ```
  connect to MYDB user 'cn=John Smith'
  connect to MYDB user uid=jsmith
  ```

- Eine einfache Zeichenfolge (ohne Gleichheitszeichen). Die Zeichenfolge wird mit dem Wert des Attributs USERID_ATTRIBUTE qualifiziert und wie ein Teil-DN behandelt. Zum Beispiel:

  ```
  connect to MYDB user jsmith
  ```

Anmerkung: Jede Zeichenfolge, die in einer CONNECT-Anweisung oder einem **ATTACH**-Befehl angegeben wird, muss in einfache Anführungszeichen gesetzt werden, wenn sie Leerzeichen oder Sonderzeichen enthält.

Hinweise zur Gruppensuchfunktion

Informationen zu Gruppenzugehörigkeiten werden in einem LDAP-Server in der Regel als Attribut des Benutzerobjekts oder als Attribut des Gruppenobjekts dargestellt:

- Als Attribut des Benutzerobjekts

  Jedes Benutzerobjekt hat ein Attribut mit dem Namen GROUP_LOOKUP_ATTRIBUTE, das Sie abfragen können, um alle Gruppenzugehörigkeiten für den bestimmten Benutzer abzurufen.

- Als Attribut des Gruppenobjekts

  Jedes Gruppenobjekt hat ein Attribut, das ebenfalls den Namen GROUP_LOOKUP_ATTRIBUTE hat, mit dessen Hilfe Sie alle Benutzerobjekte auflisten können, die Mitglieder der Gruppe sind. Sie können die Gruppen für einen bestimmten Benutzer aufzählen, indem Sie nach allen Gruppen suchen, die das Benutzerobjekt als Mitglied auflisten.

Viele LDAP-Server können auf beide Arten konfiguriert werden, und manche unterstützen beide Methoden gleichzeitig. Wenden Sie sich an Ihren LDAP-Administrator, um zu erfahren, wie Ihr LDAP-Server konfiguriert ist.

Bei der Konfiguration der LDAP-Plug-in-Module können Sie mit dem Parameter GROUP_LOOKUP_METHOD angeben, wie die Gruppensuche ausgeführt werden soll:

- Wenn Sie das Attribut GROUP_LOOKUP_ATTRIBUTE des Benutzerobjekts zum Ermitteln von Gruppenzugehörigkeiten verwenden müssen, konfigurieren Sie GROUP_LOOKUP_METHOD = USER_ATTRIBUTE.
- Wenn Sie das Attribut GROUP_LOOKUP_ATTRIBUTE des Gruppenobjekts zum Ermitteln von Gruppenzugehörigkeiten verwenden müssen, konfigurieren Sie GROUP_LOOKUP_METHOD = SEARCH_BY_DN.

Viele LDAP-Server verwenden das Attribut GROUP_LOOKUP_ATTRIBUTE des Gruppenobjekts zur Ermittlung von Zugehörigkeiten. Sie können wie im folgenden Beispiel gezeigt konfiguriert werden:

```
GROUP_LOOKUP_METHOD = SEARCH_BY_DN
GROUP_LOOKUP_ATTRIBUTE = groupOfNames
```

Microsoft Active Directory speichert Gruppenzugehörigkeiten in der Regel als Benutzerattribut und kann daher wie im folgenden Beispiel gezeigt konfiguriert werden:

```
GROUP_LOOKUP_METHOD = USER_ATTRIBUTE
GROUP_LOOKUP_ATTRIBUTE = memberOf
```

IBM Tivoli Directory Server unterstützt beide Methoden gleichzeitig. Zum Abfragen der Gruppenzugehörigkeiten für einen Benutzer können Sie das spezielle Benutzerattribut **ibm-allGroups** wie im folgenden Beispiel verwenden:

```
GROUP_LOOKUP_METHOD = USER_ATTRIBUTE
GROUP_LOOKUP_ATTRIBUTE = ibm-allGroups
```

Andere LDAP-Server können ähnliche spezielle Attribute zur Verfügung stellen, um das Abrufen von Gruppenzugehörigkeiten zu unterstützen. Im Allgemeinen ist das Abrufen von Zugehörigkeiten über ein Benutzerattribut schneller als die Suche nach Gruppen, die den gewünschten Benutzer als Mitglied auflisten.

Fehlerbehebung beim Authentifizieren von LDAP-Benutzern oder beim Abrufen von Gruppen

Wenn Sie Probleme bei der Authentifizierung von LDAP-Benutzern oder beim Abrufen der Gruppen von LDAP-Benutzern feststellen, liefern die **db2diag**-Protokolldateien und das Verwaltungsprotokoll hilfreiche Informationen für die Fehlerbehebung.

Die LDAP-Plug-in-Module protokollieren LDAP-Rückkehrcodes, Suchfilter und andere nützliche Daten in der Regel dann, wenn ein Fehler auftritt. Wenn Sie die Option DEBUG in der LDAP-Plug-in-Konfigurationsdatei aktivieren, protokollieren die Plug-in-Module darüber hinaus weitere Informationen in den **db2diag**-Protokolldateien. Obwohl diese Option bei der Fehlerbehebung eine Hilfe sein kann, wird sie aufgrund des Aufwands, der mit dem Schreiben der zusätzlichen Daten in eine Datei verbunden ist, nicht zur dauerhaften Verwendung auf Produktionssystemen empfohlen.

Stellen Sie sicher, dass der Konfigurationsparameter **diaglevel** im Datenbankmanager auf den Wert 4 gesetzt ist, sodass alle Nachrichten aus den LDAP-Plug-in-Modulen erfasst werden.

Schreiben von Sicherheits-Plug-ins

Laden von Sicherheits-Plug-ins in DB2

Sicherheits-Plug-ins müssen über eine ordnungsgemäß eingerichtete Initialisierungsfunktion verfügen, damit das DB2-Datenbanksystem die erforderlichen Informationen zum Aufrufen der Funktionen des Sicherheits-Plug-ins erhält.

Jede Plug-in-Bibliothek muss eine Initialisierungsfunktion mit einem bestimmten Namen enthalten, der durch den Plug-in-Typ festgelegt wird:

- Serverseitiges Authentifizierungs-Plug-in: `db2secServerAuthPluginInit()`

- Clientseitiges Authentifizierungs-Plug-in: `db2secClientAuthPluginInit()`
- Gruppen-Plug-in: `db2secGroupPluginInit()`

Diese Funktion wird als Plug-in-Initialisierungsfunktion bezeichnet. Die Plug-in-Initialisierungsfunktion initialisiert das angegebene Plug-in und stellt Informationen bereit, die DB2 zum Aufrufen der Funktionen des Plug-ins benötigt. Die Plug-in-Initialisierungsfunktion akzeptiert die folgenden Parameter:

- Die höchste Versionsnummer der Funktionszeigerstruktur, die von der DB2-Instanz, die das Plug-in aufruft, unterstützt werden kann
- Ein Zeiger auf eine Struktur, die Zeiger auf alle APIs enthält, die implementiert werden müssen
- Ein Zeiger auf eine Funktion, die Protokollnachrichten in die **db2diag**-Protokolldateien einfügt
- Ein Zeiger auf eine Fehlernachrichtenzeichenfolge
- Die Länge der Fehlernachricht

Die Funktionskennung für die Initialisierungsfunktion eines Plug-ins zum Abrufen von Gruppen sieht wie folgt aus:

```
SQL_API_RC SQL_API_FN db2secGroupPluginInit(
  db2int32 version,
  void *group_fns,
  db2secLogMessage *logMessage_fn,
  char **errormsg,
  db2int32  *errormsglen);
```

Anmerkung: Wenn die Plug-in-Bibliothek als C++ kompiliert wird, müssen alle Funktionen mit `extern "C"` deklariert werden. DB2 greift auf das dynamische Ladeprogramm des zugrunde liegenden Betriebssystems zurück, um die C++-Konstruktoren und -Destruktoren zu verarbeiten, die in einer benutzerdefinierten C++-Plug-in-Bibliothek verwendet werden.

Die Initialisierungsfunktion ist die einzige Funktion in der Plug-in-Bibliothek, die einen vorgeschriebenen Funktionsnamen verwendet. Die anderen Plug-in-Funktionen werden über Funktionszeiger angegeben, die aus der Initialisierungsfunktion zurückgegeben werden. Server-Plug-ins werden geladen, wenn der DB2-Server gestartet wird. Client-Plug-ins werden geladen, wenn sie auf dem Client erforderlich sind. Unmittelbar nach dem Laden einer Plug-in-Bibliothek löst DB2 die Position dieser Initialisierungsfunktion auf und ruft sie auf. Diese Funktion hat die folgende spezielle Aufgabe:

- Sie setzt den Funktionszeiger in einen Zeiger auf eine entsprechende Funktionsstruktur um.
- Sie gibt die Zeiger auf die anderen Funktionen in der Bibliothek an.
- Sie gibt die Versionsnummer der Funktionszeigerstruktur an, die zurückgegeben wird.

DB2 kann die Plug-in-Initialisierungsfunktion potenziell mehrmals aufrufen. Dieser Fall kann eintreten, wenn eine Anwendung die DB2-Clientbibliothek dynamisch lädt, sie wieder entlädt und erneut lädt und anschließend Authentifizierungsfunktionen aus einem Plug-in sowohl vor als auch nach dem erneuten Laden ausführt. In dieser Situation wird die Plug-in-Bibliothek möglicherweise nicht entladen und dann erneut geladen. Allerdings ist dieses Verhalten je nach Betriebssystem unterschiedlich.

Ein weiteres Beispiel für einen Fall, in dem DB2 mehrere Aufrufe an eine Plug-in-Initialisierungsfunktion absetzt, ist die Ausführung von gespeicherten Prozedu-

ren oder Systemaufrufen in einer Umgebung mit föderierten Datenbanken, in denen der Datenbankserver selbst als Client fungieren kann. Wenn sich die Client- und Server-Plug-ins auf dem Datenbankserver in derselben Datei befinden, könnte DB2 die Plug-in-Initialisierungsfunktion zweimal aufrufen.

Wenn das Plug-in erkennt, dass die Funktion `db2secGroupPluginInit` mehr als einmal aufgerufen wurde, sollte es dieses Ereignis so behandeln, als wäre es angewiesen worden, die Verarbeitung zu beenden und die Plug-in-Bibliothek zu reinitialisieren. Die Plug-in-Initialisierungsfunktion sollte selbst alle Bereinigungsaufgaben ausführen, die von einem Aufruf von `db2secPluginTerm` ausgeführt würden, bevor sie erneut eine Gruppe von Funktionszeigern zurückgibt.

Auf einem DB2-Server, der unter einem UNIX- oder Linux-basierten Betriebssystem ausgeführt wird, kann DB2 Plug-in-Bibliotheken potenziell mehr als einmal in verschiedenen Prozessen laden und initialisieren.

Einschränkungen für die Entwicklung von Sicherheits-Plug-in-Bibliotheken

Bei der Entwicklung von Plug-in-Bibliotheken sind verschiedene Einschränkungen zu beachten.

Die folgenden Einschränkungen sind bei der Entwicklung von Plug-in-Bibliotheken zu beachten:

C-Verlinkung
Plug-in-Bibliotheken müssen mit der C-Verlinkung (C-linkage) verlinkt werden. Headerdateien, die Prototypen definieren, Datenstrukturen, die die Plug-ins implementieren, sowie Fehlercodedefinitionen werden nur für C/C++ zur Verfügung gestellt. Funktionen, die von DB2 zur Ladezeit aufgelöst werden, müssen mit `extern "C"` deklariert werden, wenn die Plug-in-Bibliothek als C++ kompiliert wird.

Keine Unterstützung von .NET Common Language Runtime
.NET Common Language Runtime (CLR) wird für die Kompilierung und Verlinkung von Quellcode für Plug-in-Bibliotheken nicht unterstützt.

Signalroutinen
Plug-in-Bibliotheken dürfen keine Signalroutinen (Signal Handler) installieren oder die Signalmaske ändern, da dies zu einer Kollision mit den Signalroutinen von DB2 führt. Eine Kollision der DB2-Signalroutinen könnte ernste Folgen für die Fähigkeit von DB2 haben, Fehler zu melden und zu beheben, was auch Traps im Plug-in-Code selbst betrifft. Plug-in-Bibliotheken sollten nie C++-Ausnahmebedingungen auslösen, da dies ebenfalls mit der Fehlerbehandlung in DB2 kollidieren könnte.

Threadsicherheit
Plug-in-Bibliotheken müssen threadsicher und simultan verwendbar sein. Die Plug-in-Initialisierungsfunktion ist die einzige API, die nicht simultan verwendbar sein muss. Die Plug-in-Initialisierungsfunktion könnte potenziell von verschiedenen Prozessen aufgerufen werden. In diesem Fall bereinigt das Plug-in alle genutzten Ressourcen und reinitialisiert sich selbst.

Exitsteuerroutinen und Überschreiben von Aufrufen der C-Standardbibliotheken und von Betriebssystemaufrufen
Plug-in-Bibliotheken dürfen Aufrufe der C-Standardbibliotheken und Betriebssystemaufrufe nicht überschreiben. Plug-in-Bibliotheken dürfen außerdem keine Exitsteuerroutinen (Exit Handler) oder Steuerroutinen

pthread_atfork installieren. Die Verwendung von Exitsteuerroutinen wird nicht empfohlen, weil sie aus dem Speicher entladen werden können, bevor das Programm beendet wird.

Bibliotheksabhängigkeiten
Unter Linux oder UNIX können die Prozesse, die die Plug-in-Bibliotheken laden, setuid oder setgid sein. Dies bedeutet, dass sie nicht auf die Umgebungsvariablen $LD_LIBRARY_PATH, $SHLIB_PATH oder $LIBPATH zurückgreifen können, um abhängige Bibliotheken zu finden. Daher sollten die Plug-in-Bibliotheken nicht von weiteren Bibliotheken abhängig sein, sofern auf solche abhängigen Bibliotheken nicht aufgrund anderer Situationen wie zum Beispiel der folgenden zugegriffen werden kann:

- Die Bibliotheken befinden sich im Verzeichnis /lib oder /usr/lib.
- Die Verzeichnisse, in denen sich die Bibliotheken befinden, sind betriebssystemweit bekannt (z. B. durch Angabe in der Datei ld.so.conf unter Linux).
- Die Bibliotheken sind in RPATH in der Plug-in-Bibliothek selbst angegeben.

Diese Einschränkung gilt nicht für Windows-Systeme.

Symbolkollisionen
Plug-in-Bibliotheken sollten nach Möglichkeit mit allen verfügbaren Optionen kompiliert und verlinkt werden, die die Wahrscheinlichkeit von Symbolkollisionen verringern, wie zum Beispiel mit den Optionen, die nicht gebundene externe symbolische Referenzen reduzieren. Zum Beispiel kann die Verwendung der Linkeroption "-Bsymbolic" unter HP, Solaris und Linux dabei helfen, Probleme im Zusammenhang mit Symbolkollisionen zu vermeiden. Allerdings darf für Plug-ins, die unter AIX geschrieben werden, die Linkeroption "-brtl" weder explizit noch implizit verwendet werden.

32- und 64-Bit-Anwendungen
32-Bit-Anwendungen müssen mit 32-Bit-Plug-ins arbeiten. 64-Bit-Anwendungen müssen mit 64-Bit-Plug-ins arbeiten. Weitere Details finden Sie in den Hinweisen zu 32- und 64-Bit-Sicherheits-Plug-ins.

Textzeichenfolgen
Es ist nicht garantiert, dass Eingabetextzeichenfolgen auf null enden, und es ist nicht erforderlich, dass Ausgabezeichenfolgen auf null enden. Stattdessen werden ganzzahlige Längenwerte für alle Eingabezeichenfolgen und Zeiger auf ganzzahlige Werte für zurückzugebende Längen angegeben.

Übergeben von Parametern für die Berechtigungs-ID
Ein Parameter für die Berechtigungs-ID (authid), der von DB2 an ein Plug-in (als Eingabeparameter 'authid') übergeben wird, enthält eine Berechtigungs-ID in Großbuchstaben ohne auffüllende Leerzeichen. Ein Parameter 'authid', den ein Plug-in an DB2 (als Ausgabeparameter 'authid') zurückgibt, erfordert keine besondere Behandlung. Die Berechtigungs-ID wird von DB2 entsprechend dem internen DB2-Standard in Großbuchstaben umgesetzt und mit Leerzeichen aufgefüllt.

Größenbegrenzungen für Parameter
Die Plug-in-APIs arbeiten mit folgenden Werten als Längenbegrenzungen für Parameter:

```
#define DB2SEC_MAX_AUTHID_LENGTH 255
#define DB2SEC_MAX_USERID_LENGTH 255
#define DB2SEC_MAX_USERNAMESPACE_LENGTH 255
#define DB2SEC_MAX_PASSWORD_LENGTH 255
#define DB2SEC_MAX_DBNAME_LENGTH 128
```

Eine bestimmte Plug-in-Implementierung erfordert oder erzwingt möglicherweise kleinere Maximallängen für die Berechtigungs-IDs, Benutzer-IDs und Kennwörter. Insbesondere sind die Plug-ins für die Betriebssystemauthentifizierung, die mit DB2-Datenbanksystemen geliefert werden, auf die Begrenzungen für die Maximallängen von Benutzernamen, Gruppennamen und Namensbereichen beschränkt, die durch das Betriebssystem festgelegt sind, wenn diese Betriebssystembegrenzungen niedriger sind als die zuvor angegebenen Werte.

Dateinamenerweiterungen für Sicherheits-Plug-in-Bibliotheken unter AIX
Auf AIX-Systemen können Sicherheits-Plug-in-Bibliotheken die Dateinamenerweiterung *.a* oder *.so* haben. Der Mechanismus, der zum Laden der Plug-in-Bibliothek verwendet wird, hängt von der Dateinamenerweiterung ab:

- Plug-in-Bibliotheken mit der Dateinamenerweiterung *.a* werden als Archive betrachtet, die gemeinsam genutzte Objektteildateien (Member) enthalten. Diese Teildateien müssen den Namen *shr.o* (32 Bit) oder *shr64.o* (64 Bit) haben. Ein einzelnes Archiv kann sowohl die 32-Bit-Teildateien als auch die 64-Bit-Teildateien enthalten, sodass es auf beiden Typen von Plattformen implementiert werden kann.

  Zum Beispiel kann eine Plug-in-Bibliothek im Stil eines 32-Bit-Archivs wie folgt erstellt werden:

  ```
  xlc_r -qmkshrobj -o shr.o MyPlugin.c -bE:MyPlugin.exp
  ar rv MyPlugin.a shr.o
  ```

- Plug-in-Bibliotheken mit der Dateinamenerweiterung *.so* werden als dynamisch ladbare gemeinsam genutzte Objekte betrachtet. Ein solches Objekt ist entweder ein 32- oder ein 64-Bit-Objekt, je nachdem, welche Compiler- und Linkeroptionen bei der Erstellung verwendet werden. Eine 32-Bit-Plug-in-Bibliothek kann zum Beispiel wie folgt erstellt werden:

  ```
  xlc_r -qmkshrobj -o MyPlugin.so MyPlugin.c -bE:MyPlugin.exp
  ```

Auf allen anderen Plattformen als AIX werden Sicherheits-Plug-in-Bibliotheken immer als dynamisch ladbare gemeinsam genutzte Objekte aufgefasst.

Verzweigung (Fork)
Plug-in-Bibliothek sollten keine Verzweigungen enthalten, da Dateideskriptoren und Sockets im untergeordneten Prozess dupliziert werden, was zu Blockierungen oder falschem Verhalten führen kann. Insbesondere können Sperrenkonflikte aufgrund falscher Dateien auftreten, wenn das untergeordnete Element verzweigt wurde, als ein offener Dateideskriptor für eine solche Datei vorhanden war. Darüber hinaus besteht die Möglichkeit, dass die Verzweigung eine Vielzahl anderer Ressourcen wie beispielsweise Semaphore übernimmt.

Einschränkungen für Sicherheits-Plug-ins

Bei der Verwendung von Sicherheits-Plug-ins sind verschiedene Einschränkungen zu beachten:

Einschränkungen für die Unterstützung in der DB2-Datenbankfamilie

Sie können kein GSS-API-Plug-in zur Authentifizierung von Verbindungen zwischen DB2-Clients unter Linux, UNIX und Windows und einem anderen Server der DB2-Produktfamilie, wie zum Beispiel DB2 für z/OS, verwenden. Darüber hinaus können auch keine Verbindungen von einem anderen Produkt der DB2-Datenbankfamilie, das als Client fungiert, zu einem DB2-Server unter Linux, UNIX oder Windows authentifizieren.

Wenn Sie einen DB2-Client unter Linux, UNIX oder Windows verwenden, um eine Verbindung zu anderen Servern der DB2-Datenbankfamilie herzustellen, können Sie clientseitige Benutzer-ID/Kennwort-Plug-ins (z. B. das von IBM bereitgestellte Plug-in zur betriebssystembasierten Authentifizierung) verwenden oder Sie können ein eigenes Benutzer-ID/Kennwort-Plug-in schreiben. Sie können auch die integrierten Kerberos-Plug-ins verwenden oder eigene implementieren.

Mit einem DB2-Client unter Linux, UNIX oder Windows sollten Sie eine Datenbank nicht mit dem Authentifizierungstyp GSSPLUGIN katalogisieren.

Einschränkungen für die Berechtigungs-ID. Ab Version 9.5 des DB2-Datenbanksystems können Sie eine 128-Byte-Berechtigungs-ID verwenden. Wenn die Berechtigungs-ID jedoch als Benutzer-ID oder Gruppenname des Betriebssystems interpretiert wird, gelten die Einschränkungen des Betriebssystems (z. B. eine Begrenzung auf 8 oder 30 Zeichen für Benutzer-IDs und auf 30 Zeichen für Gruppennamen). Daher können Sie zwar eine 128-Byte-Berechtigungs-ID erteilen, jedoch ist es nicht möglich, als Benutzer, der diese Berechtigungs-ID hat, eine Verbindung herzustellen. Wenn Sie ein eigenes Sicherheits-Plug-in schreiben, sind Sie in der Lage, die erweiterten Größen für die Berechtigungs-ID voll auszunutzen. Zum Beispiel können Sie an Ihr Sicherheits-Plug-in eine 30-Byte-Benutzer-ID übergeben, und das Plug-in kann eine 128-Byte-Berechtigungs-ID während der Authentifizierung zurückgeben, mit der Sie eine Verbindung herstellen können.

Einschränkungen der Unterstützung durch InfoSphere Federation Server

DB2 II unterstützt die Verwendung delegierter Berechtigungsnachweise aus einem GSS_API-Plug-in nicht, um abgehende Verbindungen zu Datenquellen herzustellen. Verbindungen zu Datenquellen müssen weiterhin den Befehl CREATE USER MAPPING verwenden.

Einschränkungen der Unterstützung durch den Datenbankverwaltungsserver (DAS)

Der DB2-Verwaltungsserver (DAS) unterstützt Plug-ins nicht. Der DAS unterstützt nur das Authentifizierungsverfahren über das Betriebssystem.

Sicherheits-Plug-in-Problem und Einschränkung für DB2-Clients (Windows)

Wenn Sie Sicherheits-Plug-ins entwickeln, die in DB2-Clients unter Windows-Betriebssystemen implementiert werden sollen, führen Sie das Entladen Sie von Hilfsbibliotheken aus dem Speicher nicht in der Beendigungsfunktion des Plug-ins aus. Diese Einschränkung gilt für alle Typen von Client-Sicherheits-Plug-ins, einschließlich Gruppen-, Benutzer-ID/Kennwort-, Kerberos- und GSS-API-Plug-ins. Da diese Beendigungs-API wie zum Beispiel 'db2secPluginTerm',

'db2secClientAuthPluginTerm' und 'db2secServerAuthPluginTerm' auf keiner Windows-Plattform aufgerufen werden, müssen Sie die entsprechende Ressourcenbereinigung selbst durchführen.

Diese Einschränkung steht im Zusammenhang mit Bereinigungsproblemen, die mit dem Entladen von DDL-Dateien unter Windows verbunden sind.

Laden von Plug-in-Bibliotheken mit der Erweiterung .a oder .so unter AIX

Unter AIX können Sicherheits-Plug-in-Bibliotheken die Dateinamenerweiterung `.a` oder `.so` haben. Der Mechanismus, der zum Laden der Plug-in-Bibliothek verwendet wird, hängt von der Dateinamenerweiterung ab:

- Plug-in-Bibliotheken mit der Dateinamenerweiterung `.a`

  Plug-in-Bibliotheken mit der Dateinamenerweiterung .a werden als Archive betrachtet, die gemeinsam genutzte Objektteildateien (Member) enthalten. Diese Teildateien müssen den Namen `shr.o` (32 Bit) oder `shr64.o` (64 Bit) haben. Ein einzelnes Archiv kann sowohl die 32-Bit-Teildateien als auch die 64-Bit-Teildateien enthalten, sodass es auf beiden Typen von Plattformen implementiert werden kann.

  Zum Beispiel kann eine Plug-in-Bibliothek im Stil eines 32-Bit-Archivs wie folgt erstellt werden:

  ```
  xlc_r -qmkshrobj -o shr.o MyPlugin.c -bE:MyPlugin.exp
  ar rv MyPlugin.a shr.o
  ```

- Plug-in-Bibliotheken mit der Dateinamenerweiterung `.so`

  Plug-in-Bibliotheken mit der Dateinamenerweiterung .so werden als dynamisch ladbare gemeinsam genutzte Objekte betrachtet. Ein solches Objekt ist entweder ein 32- oder ein 64-Bit-Objekt, je nachdem, welche Compiler- und Linkeroptionen bei der Erstellung verwendet werden. Eine 32-Bit-Plug-in-Bibliothek kann zum Beispiel wie folgt erstellt werden:

  ```
  xlc_r -qmkshrobj -o MyPlugin.so MyPlugin.c -bE:MyPlugin.exp
  ```

Auf allen anderen Plattformen als AIX werden Sicherheits-Plug-in-Bibliotheken immer als dynamisch ladbare gemeinsam genutzte Objekte aufgefasst.

GSS-API-Sicherheits-Plug-ins unterstützen keine Verschlüsselung und Signierung von Nachrichten

Funktionen zur Verschlüsselung und Signierung von Nachrichten sind in GSS-API-Sicherheits-Plug-ins nicht verfügbar.

Rückkehrcodes für Sicherheits-Plug-ins

Alle Sicherheits-Plug-in-AIPs (API - Anwendungsprogrammierschnittstelle) müssen einen ganzzahligen Wert (Integer) zurückgeben, um Erfolg oder Fehler bei der Ausführung der API anzuzeigen. Der Rückkehrcodewert `0` gibt an, dass die API erfolgreich ausgeführt wurde. Alle negativen Rückkehrcodes mit Ausnahme von `-3`, `-4` und `-5` geben an, dass die API einen Fehler festgestellt hat.

Alle negativen Rückkehrcodes, die von den Sicherheits-Plug-in-APIs zurückgegeben werden, werden dem SQLCODE-Wert -1365, SQLCODE-Wert -1366 oder SQLCODE-Wert -30082 zugeordnet. Davon ausgenommen sind die Rückkehrcodes `-3`, `-4` und `-5`. Die Werte `-3`, `-4` und `-5` dienen zur Angabe, ob eine Berechtigungs-ID einen gültigen Benutzer oder eine gültige Gruppe darstellt.

Alle Rückkehrcodes von Sicherheits-Plug-in-APIs sind in der Datei db2secPlugin.h definiert, die sich im DB2-INCLUDE-Verzeichnis befindet: SQLLIB/include.

Details in Bezug auf alle Rückkehrcodes von Sicherheits-Plug-ins werden in der folgenden Tabelle dargestellt:

Tabelle 40. Rückkehrcodes von Sicherheits-Plug-ins

Rückkehrcode	DEFINE-Wert	Bedeutung	Relevante APIs
0	DB2SEC_PLUGIN_OK	Die Plug-in-API wurde erfolgreich ausgeführt.	Alle
-1	DB2SEC_PLUGIN_UNKNOWNERROR	Die Plug-in-API hat einen unerwarteten Fehler festgestellt.	Alle
-2	DB2SEC_PLUGIN_BADUSER	Die als Eingabe übergebene Benutzer-ID ist nicht definiert.	db2secGenerateInitialCred db2secValidatePassword db2secRemapUserid db2secGetGroupsForUser
-3	DB2SEC_PLUGIN_INVALIDUSERORGROUP	Kein solcher Benutzer bzw. keine solche Gruppe vorhanden.	db2secDoesAuthIDExist db2secDoesGroupExist
-4	DB2SEC_PLUGIN_USERSTATUSNOTKNOWN	Unbekannter Benutzerstatus. Dieser Wert wird von DB2 nicht als Fehler behandelt, sondern dient bei der Verarbeitung einer Anweisung GRANT zur Ermittlung, ob eine Berechtigungs-ID (authid) einen Benutzer oder eine Betriebssystemgruppe darstellt.	db2secDoesAuthIDExist
-5	DB2SEC_PLUGIN_GROUPSTATUSNOTKNOWN	Unbekannter Gruppenstatus. Dieser Wert wird von DB2 nicht als Fehler behandelt, sondern dient bei der Verarbeitung einer Anweisung GRANT zur Ermittlung, ob eine Berechtigungs-ID (authid) einen Benutzer oder eine Betriebssystemgruppe darstellt.	db2secDoesGroupExist
-6	DB2SEC_PLUGIN_UID_EXPIRED	Benutzer-ID abgelaufen.	db2secValidatePassword db2GetGroupsForUser db2secGenerateInitialCred
-7	DB2SEC_PLUGIN_PWD_EXPIRED	Kennwort abgelaufen.	db2secValidatePassword db2GetGroupsForUser db2secGenerateInitialCred
-8	DB2SEC_PLUGIN_USER_REVOKED	Benutzer widerrufen.	db2secValidatePassword db2GetGroupsForUser
-9	DB2SEC_PLUGIN_USER_SUSPENDED	Benutzer ausgesetzt (gesperrt).	db2secValidatePassword db2GetGroupsForUser

Tabelle 40. Rückkehrcodes von Sicherheits-Plug-ins (Forts.)

Rückkehr-code	DEFINE-Wert	Bedeutung	Relevante APIs
-10	DB2SEC_PLUGIN_BADPWD	Falsches Kennwort.	db2secValidatePassword db2secRemapUserid db2secGenerateInitialCred
-11	DB2SEC_PLUGIN _BAD_NEWPASSWORD	Falsches neues Kennwort.	db2secValidatePassword db2secRemapUserid
-12	DB2SEC_PLUGIN _CHANGEPASSWORD _NOTSUPPORTED	Kennwortänderung nicht unterstützt.	db2secValidatePassword db2secRemapUserid db2secGenerateInitialCred
-13	DB2SEC_PLUGIN_NOMEM	Speicherzuordnungsversuch des Plug-ins wegen unzureichender Speicherkapazität fehlgeschlagen.	Alle
-14	DB2SEC_PLUGIN_DISKERROR	Plug-in hat einen Plattenfehler festgestellt.	Alle
-15	DB2SEC_PLUGIN_NOPERM	Versuch des Zugriffs auf eine Datei durch das Plug-in wegen falscher Berechtigungen für die Datei fehlgeschlagen.	Alle
-16	DB2SEC_PLUGIN_NETWORKERROR	Plug-in hat einen Netzfehler festgestellt.	Alle
-17	DB2SEC_PLUGIN _CANTLOADLIBRARY	Plug-in kann die erforderliche Bibliothek nicht laden.	db2secGroupPluginInit db2secClientAuthPluginInit db2secServerAuthPluginInit
-18	DB2SEC_PLUGIN_CANT _OPEN_FILE	Plug-in kann eine Datei nicht öffnen und lesen, wobei die Ursache weder eine fehlende Datei noch eine falsche Dateiberechtigung ist.	Alle
-19	DB2SEC_PLUGIN_FILENOTFOUND	Plug-in kann eine Datei nicht öffnen und lesen, weil die Datei im Dateisystem fehlt.	Alle
-20	DB2SEC_PLUGIN _CONNECTION_DISALLOWED	Das Plug-in verweigert den Verbindungsaufbau wegen der Einschränkung, zu welcher Datenbank eine Verbindung zulässig ist, oder die TCP/IP-Adresse kann keine Verbindung zu einer bestimmten Datenbank herstellen.	Alle serverseitigen Plug-in-APIs
-21	DB2SEC_PLUGIN_NO_CRED	Nur GSS-API-Plug-in: Anfangsberechtigungsnachweis für Client fehlt.	db2secGetDefaultLoginContext db2secServerAuthPluginInit
-22	DB2SEC_PLUGIN_CRED_EXPIRED	Nur GSS-API-Plug-in: Clientberechtigungsnachweis ist abgelaufen.	db2secGetDefaultLoginContext db2secServerAuthPluginInit
-23	DB2SEC_PLUGIN _BAD_PRINCIPAL_NAME	Nur GSS-API-Plug-in: Der Name des Principals ist ungültig.	db2secProcessServer PrincipalName

Tabelle 40. Rückkehrcodes von Sicherheits-Plug-ins (Forts.)

Rückkehr-code	DEFINE-Wert	Bedeutung	Relevante APIs
-24	DB2SEC_PLUGIN _NO_CON_DETAILS	Dieser Rückkehrcode wird vom db2secGetConDetails-Callback (z. B. von DB2 an das Plug-in) zurückgegeben, um mitzuteilen, dass DB2 die TCP/IP-Adresse des Clients nicht ermitteln kann.	db2secGetConDetails
-25	DB2SEC_PLUGIN _BAD_INPUT_PARAMETERS	Beim Aufruf der Plug-in-API sind einige Parameter nicht gültig oder fehlen.	Alle
-26	DB2SEC_PLUGIN _INCOMPATIBLE_VER	Die vom Plug-in gemeldete Version der APIs ist nicht mit DB2 kompatibel.	db2secGroupPluginInit db2secClientAuthPluginInit db2secServerAuthPluginInit
-27	DB2SEC_PLUGIN_PROCESS_LIMIT	Nicht genügend Ressourcen für das Plug-in zur Erstellung eines neuen Prozesses verfügbar.	Alle
-28	DB2SEC_PLUGIN_NO_LICENSES	Das Plug-in hat ein Benutzerlizenzproblem festgestellt. Es besteht die Möglichkeit, dass die Lizenz des zugrunde liegenden Mechanismus ihre Obergrenze erreicht hat.	Alle
-29	DB2SEC_PLUGIN_ROOT_NEEDED	Das Plug-in versucht, eine Anwendung auszuführen, für die Rootberechtigungen erforderlich sind.	Alle
-30	DB2SEC_PLUGIN_UNEXPECTED_ SYSTEM_ERROR	Das Plug-in hat einen unerwarteten Systemfehler festgestellt. Möglicherweise wird die aktuelle Systemkonfiguration nicht unterstützt.	Alle

Fehlernachrichtenbehandlung für Sicherheits-Plug-ins

Wenn in einer Sicherheits-Plug-in-API ein Fehler auftritt, kann die API eine ASCII-Textzeichenfolge im Feld errormsg zurückgeben, um eine genauere Beschreibung des Problems als den Rückkehrcode bereitzustellen.

Die Zeichenfolge im Feld errormsg könnte zum Beispiel den Text "Datei /home/db2inst1/mypasswd.txt ist nicht vorhanden." enthalten. DB2 schreibt die gesamte Zeichenfolge in das DB2-Protokoll mit Benachrichtigungen für die Systemverwaltung und fügt darüber hinaus eine abgeschnittene Version als Token in einige SQL-Nachrichten ein. Da Token in SQL-Nachrichten nur eine begrenzte Länge haben können, sollten diese Nachrichten kurz gehalten werden, und wichtige Informationen dieser Nachrichten sollten im Anfang der Zeichenfolge enthalten sein. Als Debughilfe sollten Sie in Betracht ziehen, den Namen des Sicherheits-Plug-ins in die Fehlernachricht aufzunehmen.

Bei nicht dringenden Fehlern, zum Beispiel Fehlern aufgrund abgelaufener Kennwörter, wird die Zeichenfolge in errormsg nur dann in das Protokoll geschrieben, wenn der Konfigurationsparameter DIAGLEVEL des Datenbankmanagers auf den Wert 4 gesetzt ist.

Der Speicher für diese Fehlernachrichten muss vom Sicherheits-Plug-in zugeordnet werden. Daher müssen die Plug-ins auch eine API zur Freigabe dieses Speichers verwenden: `db2secFreeErrormsg`.

Das Feld `errormsg` wird von DB2 nur überprüft, wenn eine API einen Fehler ungleich null zurückgibt. Daher sollte das Plug-in keinen Speicher für diese zurückgegebene Fehlernachricht zuordnen, wenn kein Fehler aufgetreten ist.

Bei der Initialisierung wird ein Zeiger (`logMessage_fn`) für die Protokollierungsfunktion ('db2secLogMessage') an das Gruppen-, Client- oder Server-Plug-in übergeben. Die Plug-ins können die Funktion zum Protokollieren beliebiger Debugging-Informationen in den **db2diag**-Protokolldateien verwenden. Beispiel:

```
// Protokollieren einer Nachricht 'Init erfolgreich'
(*(logMessage_fn))(DB2SEC_LOG_CRITICAL,
                   "db2secGroupPluginInit erfolgreich",
                    strlen("db2secGroupPluginInit erfolgreich"));
```

Weitere detaillierte Informationen zu den Parametern für die Funktion `db2secLogMessage` finden Sie in Beschreibungen der Initialisierungs-APIs für die einzelnen Plug-in-Typen.

Aufrufreihenfolgen für die APIs der Sicherheits-Plug-ins

Die Reihenfolge, in der die APIs der Sicherheits-Plug-ins vom DB2-Datenbankmanager aufgerufen werden, ist von dem Szenario abhängig, in dem die API aufgerufen wird.

Im Folgenden werden die Hauptszenarios aufgeführt, in denen der DB2-Datenbankmanager APIs für Sicherheits-Plug-ins aufruft:

- Auf einem Client zur Herstellung einer Datenbankverbindung (implizit und explizit)
  - CLIENT
  - Serverbasiert (SERVER, SERVER_ENCRYPT, DATA_ENCRYPT)
  - GSSAPI und Kerberos
- Auf einem Client, Server oder Gateway zur lokalen Autorisierung
- Auf einem Server zur Herstellung einer Datenbankverbindung
- Auf einem Server für eine Anweisung GRANT
- Auf einem Server zum Abrufen einer Liste der Gruppen, zu denen eine Berechtigungs-ID gehört

Anmerkung: Die DB2-Datenbankserver behandeln Datenbankaktionen, die lokale Berechtigungen erfordern, wie zum Beispiel **db2start**, **db2stop** und **db2trc**, wie Clientanwendungen.

Für jede dieser Operationen ist die Reihenfolge unterschiedlich, in der der DB2-Datenbankmanager die Sicherheits-Plug-in-APIs aufruft. Im Folgenden werden die Reihenfolgen der APIs, die vom DB2-Datenbankmanager aufgerufen werden, für jedes dieser Szenarios aufgeführt.

CLIENT - implizit

Wenn der vom Benutzer konfigurierte Authentifizierungstyp CLIENT ist, ruft die DB2-Clientanwendung die folgenden Sicherheits-Plug-in-APIs auf:

- db2secGetDefaultLoginContext();
- db2secValidatePassword();
- db2secFreetoken();

Bei einer impliziten Authentifizierung, d. h. wenn Sie die Verbindung herstellen, ohne eine bestimmte Benutzer-ID oder Kennwort anzugeben, wird die API db2secValidatePassword aufgerufen, wenn Sie ein Plug-in zur Benutzer-ID/Kennwort-Authentifizierung verwenden. Diese API ermöglicht Plug-in-Entwicklern, die implizite Authentifizierung zu unterbinden, falls dies erforderlich ist.

CLIENT - explizit
Bei einer expliziten Authentifizierung, d. h., wenn Sie die Verbindung zu einer Datenbank unter Angabe der Benutzer-ID und des Kennworts herstellen, ruft die DB2-Clientanwendung, wenn der Konfigurationsparameter **authentication** des Datenbankmanagers auf den Wert CLIENT gesetzt ist, die folgenden Sicherheits-Plug-in-APIs mehrere Male auf, falls die Implementierung dies erfordert:

- db2secRemapUserid();
- db2secValidatePassword();
- db2secFreeToken();

Serverbasierte Authentifizierung (SERVER, SERVER_ENCRYPT, DATA_ENCRYPT) - implizit
Bei einer impliziten Authentifizierung, wenn der Client und der Server eine Benutzer-ID/Kennwort-Authentifizierung vereinbart haben (z. B. bei Einstellung des Parameters **srvcon_auth** auf dem Server auf den Wert SERVER, SERVER_ENCRYPT, DATA_ENCRYPT oder DATA_ENCRYPT_CMP), ruft die Clientanwendung die folgenden Sicherheits-Plug-in-APIs auf:

- db2secGetDefaultLoginContext();
- db2secFreeToken();

Serverbasierte Authentifizierung (SERVER, SERVER_ENCRYPT, DATA_ENCRYPT) - explizit
Bei einer expliziten Authentifizierung, wenn der Client und der Server eine Benutzer-ID/Kennwort-Authentifizierung vereinbart haben (z. B. bei Einstellung des Parameters **srvcon_auth** auf dem Server auf den Wert SERVER, SERVER_ENCRYPT, DATA_ENCRYPT oder DATA_ENCRYPT_CMP), ruft die Clientanwendung die folgenden Sicherheits-Plug-in-APIs auf:

- db2secRemapUserid();

GSSAPI und Kerberos - implizit
Bei einer impliziten Authentifizierung, wenn der Client und der Server eine GSS-API- oder Kerberos-Authentifizierung vereinbart haben (z. B. bei Einstellung des Parameters **srvcon_auth** auf dem Server auf den Wert KERBEROS,KRB_SERVER_ENCRYPT, GSSPLUGIN oder GSS_SERVER_ENCRYPT), ruft die Clientanwendung die folgenden Sicherheits-Plug-in-APIs auf. (Der Aufruf der API gss_init_sec_context() verwendet GSS_C_NO_CREDENTIAL als Eingabeberechtigungsnachweis.)

- db2secGetDefaultLoginContext();
- db2secProcessServerPrincipalName();
- gss_init_sec_context();
- gss_release_buffer();
- gss_release_name();
- gss_delete_sec_context();
- db2secFreeToken();

Mit der Unterstützung für mehrere GSS-API-Abläufe kann die API gss_init_sec_context() mehrfach aufgerufen werden, wenn die Implementierung dies erfordert.

GSSAPI und Kerberos - explizit

Wenn der vereinbarte Authentifizierungstyp GSS-API oder Kerberos ist, ruft die Clientanwendung die folgenden Sicherheits-Plug-in-APIs für GSS-API-Plug-ins in der angegebenen Reihenfolge auf. Diese APIs gelten für die implizite und die explizite Authentifizierung, sofern nicht anders angegeben.

- db2secProcessServerPrincipalName();
- db2secGenerateInitialCred(); (nur für explizite Authentifizierung)
- gss_init_sec_context();
- gss_release_buffer ();
- gss_release_name();
- gss_release_cred();
- db2secFreeInitInfo();
- gss_delete_sec_context();
- db2secFreeToken();

Die API gss_init_sec_context() kann mehrmals aufgerufen werden, wenn ein Token zur gegenseitigen Authentifizierung vom Server zurückgegeben wird und die Implementierung dies erfordert.

Auf einem Client, Server oder Gateway zur lokalen Autorisierung

Für eine lokale Berechtigung ruft der verwendete DB2-Befehl die folgenden Sicherheits-Plug-in-APIs auf:

- db2secGetDefaultLoginContext();
- db2secGetGroupsForUser();
- db2secFreeToken();
- db2secFreeGroupList();

Diese APIs werden für beide Verfahren, das der Benutzer-ID/Kennwort-Authentifizierung und das der GSS-API-Authentifzierung, aufgerufen.

Auf einem Server zur Herstellung einer Datenbankverbindung

Für eine Datenbankverbindung auf dem Datenbankserver ruft der Prozess oder Thread des DB2-Agenten die folgenden Sicherheits-Plug-in-APIs für das Benutzer-ID/Kennwort-Authentifizierungsverfahren auf:

- db2secValidatePassword(); (nur wenn der Datenbankkonfigurationsparameter **authentication** nicht den Wert CLIENT hat)
- db2secGetAuthIDs();
- db2secGetGroupsForUser();
- db2secFreeToken();
- db2secFreeGroupList();

Für eine CONNECT-Verbindung zu einer Datenbank ruft der Prozess oder Thread des DB2-Agenten die folgenden Sicherheits-Plug-in-APIs für das GSS-API-Authentifizierungsverfahren auf:

- gss_accept_sec_context();
- gss_release_buffer();
- db2secGetAuthIDs();
- db2secGetGroupsForUser();

- gss_delete_sec_context();
- db2secFreeGroupListMemory();

Auf einem Server für eine Anweisung GRANT

Für eine Anweisung GRANT, in der das Schlüsselwort USER oder GROUP nicht angegeben ist (z. B. "`GRANT CONNECT ON DATABASE TO benutzer1`") muss der Prozess oder Thread des DB2-Agenten feststellen können, ob es sich bei `benutzer1` um einen Benutzer, um eine Gruppe oder um beides handelt. Daher ruft der Prozess bzw. Thread des DB2-Agenten die folgenden Sicherheits-Plug-in-APIs auf:

- db2secDoesGroupExist();
- db2secDoesAuthIDExist();

Auf einem Server zum Abrufen einer Liste der Gruppen, zu denen eine Berechtigungs-ID gehört

Wenn Sie von Ihrem Datenbankserver eine Liste der Gruppen abrufen müssen, zu denen eine Berechtigungs-ID gehört, ruft der Prozess oder Thread des DB2-Agenten die folgende Sicherheits-Plug-in-API nur mit der Berechtigungs-ID als Eingabe auf:

- db2secGetGroupsForUser();

Token aus anderen Sicherheits-Plug-ins sind nicht beteiligt.

Kapitel 9. Sicherheits-Plug-in-APIs

Das DB2-Datenbanksystem stellt APIs bereit, mit denen Sie vorhandene Plug-in-Module modifizieren bzw. neue Sicherheits-Plug-in-Module erstellen können, um die Funktionalität zur Authentifizierung und Gruppensuche Ihres DB2-Datenbanksystems an Ihre Verhältnisse anzupassen.

Wenn Sie ein Sicherheits-Plug-in-Modul entwickeln, müssen Sie die Standardfunktionen zur Authentifizierung bzw. zur Ermittlung von Gruppenzugehörigkeiten implementieren, die vom DB2 aufgerufen werden sollen. Für die drei verfügbaren Typen von Plug-in-Modulen müssen Sie die folgende Funktionalität implementieren:

Abrufen von Gruppen
Ruft Informationen zur Gruppenzugehörigkeit für einen angegebenen Benutzer ab und bestimmt, ob eine angegebene Zeichenfolge einen gültigen Gruppennamen darstellt.

Benutzer-ID/Kennwort-Authentifizierung
Ein Authentifizierungsverfahren, das den Standardsicherheitskontext angibt (nur Client), ein Kennwort prüft und optional ändert, feststellt, ob eine angegebene Zeichenfolge einen gültigen Benutzer darstellt (nur Server), die auf dem Client angegebene Benutzer-ID bzw. das Kennwort vor dem Senden an den Server ändert (nur Client) sowie die einem angegebenen Benutzer zugeordnete DB2-Berechtigungs-ID zurückgibt.

GSS-API-Authentifizierung
Ein Authentifizierungsverfahren, das die erforderlichen GSS-API-Funktionen implementiert, den Standardsicherheitskontext angibt (nur Clientseite), die Anfangsberechtigungsnachweise auf der Basis der Benutzer-ID und des Kennworts generiert und optional das Kennwort ändert (nur Clientseite), Sicherheitstickets erstellt und akzeptiert sowie die einem angegebenen GSS-API-Sicherheitskontext zugeordnete DB2-Berechtigungs-ID zurückgibt.

In der folgenden Liste werden einige der in den Beschreibungen der Plug-in-APIs verwendeten Begriffe definiert.

Plug-in
Eine dynamisch ladbare Bibliothek, die von DB2 geladen wird, um auf benutzerdefinierte Funktionen zur Authentifizierung und Gruppenzugehörigkeitsermittlung zuzugreifen.

Implizite Authentifizierung
Eine Verbindung zu einer Datenbank, bei der keine Benutzer-ID bzw. kein Kennwort angegeben wird.

Explizite Authentifizierung
Eine Verbindung zu einer Datenbank, bei der sowohl die Benutzer-ID als auch das Kennwort angegeben werden.

Berechtigungs-ID (authid)
Eine interne Kennung, die einen Einzelbenutzer oder eine Gruppe darstellt, dem bzw. der Berechtigungen und Zugriffsrechte innerhalb der Datenbank erteilt sind. Intern wird eine DB2-Berechtigungs-ID durchgehend in Großbuchstaben umgesetzt und ist mindestens acht Zeichen lang (ggf. mit Leerzeichen auf die Länge von acht Zeichen aufgefüllt). Gegenwärtig setzt DB2

Berechtigungs-IDs, Benutzer-IDs, Kennwörter, Gruppennamen, Namensbereiche und Domänennamen voraus, die im 7-Bit-ASCII-Code darstellbar sind.

Lokale Berechtigung
Eine Berechtigungsfunktion, die in Bezug auf den Server oder Client, der sie implementiert, lokal ist und prüft, ob ein Benutzer zur Ausführung einer Aktion (abgesehen von der Herstellung einer Verbindung zur Datenbank) berechtigt ist, wie zum Beispiel zum Starten und Stoppen des Datenbankmanagers, zum Aktivieren und Inaktivieren der DB2-Tracefunktion oder zum Aktualisieren der Datenbankmanagerkonfiguration.

Namensbereich
Eine Sammlung bzw. Gruppierung von Benutzern, innerhalb deren die Kennungen für einzelne Benutzer eindeutig sein müssen. Allgemeine Beispiele sind Windows-Domänen und Kerberos-Realms. Zum Beispiel müssen innerhalb der Windows-Domäne "usa.company.com" alle Benutzernamen eindeutig sein. Beispiel: "benutzer1@usa.company.com". Dieselbe Benutzer-ID in einer anderen Domäne, zum Beispiel "benutzer1@canada.company.com", bezieht sich hingegen auf eine andere Person. Eine vollständig qualifizierte Benutzerkennung besteht aus einem Paar aus Benutzer-ID und Namensbereich. Beispiel: "benutzer@domäne.name" oder "domäne\benutzer".

Eingabe
Gibt an, dass DB2 den Wert für den jeweiligen API-Parameter des Sicherheits-Plug-ins bereitstellt.

Ausgabe
Gibt an, dass die Sicherheits-Plug-in-API den Wert für den API-Parameter bereitstellt.

APIs für Plug-ins zum Abrufen von Gruppen

Für das Plug-in-Modul zum Abrufen von Gruppen müssen Sie die folgenden APIs implementieren:

- `db2secGroupPluginInit`

  Anmerkung: Die Anwendungsprogrammierschnittstelle (API) 'db2secGroupPluginInit' empfängt als Eingabe den Zeiger `*logMessage_fn` auf eine API mit dem folgenden Prototyp:

  ```
  SQL_API_RC (SQL_API_FN db2secLogMessage)
  (
  db2int32 level,
  void    *data,
  db2int32 length
  );
  ```

  Die API 'db2secLogMessage' ermöglicht dem Plug-in, Nachrichten in den **db2diag**-Protokolldateien zu Debug- oder Informationszwecken zu protokollieren. Diese API wird vom DB2-Datenbanksystem bereitgestellt, sodass Sie sie nicht zu implementieren brauchen.
- `db2secPluginTerm`
- `db2secGetGroupsForUser`
- `db2secDoesGroupExist`
- `db2secFreeGroupListMemory`
- `db2secFreeErrormsg`

- Die einzige API, die extern auflösbar sein muss, ist die API db2secGroupPluginInit. Diese API empfängt den Parameter void *, der in den folgenden Typ umgesetzt werden muss:

```
typedef struct db2secGroupFunctions_1
{
db2int32 version;
db2int32 plugintype;
SQL_API_RC (SQL_API_FN * db2secGetGroupsForUser)
(
const char *authid,
db2int32    authidlen,
const char *userid,
db2int32    useridlen,
const char *usernamespace,
db2int32    usernamespacelen,
db2int32    usernamespacetype,
const char *dbname,
db2int32    dbnamelen,
const void *token,
db2int32    tokentype,
db2int32    location,
const char *authpluginname,
db2int32    authpluginnamelen,
void      **grouplist,
db2int32   *numgroups,
char    **errormsg,
db2int32 *errormsglen
);

SQL_API_RC (SQL_API_FN * db2secDoesGroupExist)
(
const char *groupname,
db2int32    groupnamelen,
char      **errormsg,
db2int32 *errormsglen
);

SQL_API_RC (SQL_API_FN * db2secFreeGroupListMemory)
(
void     *ptr,
char    **errormsg,
db2int32 *errormsglen
);

SQL_API_RC (SQL_API_FN * db2secFreeErrormsg)
(
char *msgtobefree
);

SQL_API_RC (SQL_API_FN * db2secPluginTerm)
(
char    **errormsg,
db2int32 *errormsglen
);

} db2secGroupFunctions_1;
```

Die API db2secGroupPluginInit weist die Adressen für die übrigen extern verfügbaren Funktionen zu.

Anmerkung: Die Erweiterung _1 gibt an, dass dies die Struktur ist, die der Version 1 der API entspricht. Nachfolgende Schnittstellenversionen werden die Erweiterungen _2, _3 usw. haben.

db2secDoesGroupExist (API) - Vorhandensein einer Gruppe überprüfen

Stellt fest, ob eine Berechtigungs-ID (authid) eine Gruppe darstellt.

Wenn der Parameter **groupname** vorhanden ist, muss die API den Wert DB2SEC_PLUGIN_OK zurückgeben können, um den Erfolg zu melden. Sie muss außerdem den Wert DB2SEC_PLUGIN_INVALIDUSERORGROUP zurückgeben können, wenn der Gruppenname nicht gültig ist. Es ist zulässig, dass die API den Wert DB2SEC_PLUGIN_GROUPSTATUSNOTKNOWN zurückgibt, wenn sich nicht feststellen lässt, ob die Eingabe eine gültige Gruppe angibt. Wenn der Wert für ungültige Gruppe (DB2SEC_PLUGIN_INVALIDUSERORGROUP) oder für unbekannte Gruppe (DB2SEC_PLUGIN_GROUPSTATUSNOTKNOWN) zurückgegeben wird, kann DB2 Database for Linux, UNIX and Windows möglicherweise nicht feststellen, ob die Berechtigungs-ID eine Gruppe oder einen Benutzer darstellt, wenn die Anweisung GRANT ohne das Schlüsselwort USER oder GROUP abgesetzt wird. In einem solchen Fall wird ein Fehler mit dem SQLCODE-Wert -569 und dem SQLSTATE-Wert 56092 an den Benutzer zurückgegeben.

API- und Datenstruktursyntax

```
SQL_API_RC ( SQL_API_FN *db2secDoesGroupExist)
                      ( const char *groupname,
                        db2int32 groupnamelen,
                        char      **errormsg,
                        db2int32 *errormsglen );
```

Parameter der API 'db2secDoesGroupExist'

groupname
Eingabe. Eine in Großbuchstaben umgesetzte Berechtigungs-ID (authid) ohne folgende Leerzeichen.

groupnamelen
Eingabe. Die Länge (in Byte) des Werts des Parameters **groupname**.

errormsg
Ausgabe. Ein Zeiger auf die Adresse einer vom Plug-in zugeordneten ASCII-Fehlernachrichtenzeichenfolge, die in diesem Parameter zurückgegeben werden kann, wenn die API 'db2secDoesGroupExist' nicht erfolgreich ausgeführt wird.

errormsglen
Ausgabe. Ein Zeiger auf einen ganzzahligen Wert (Integer), der die Länge (in Byte) der Fehlernachrichtenzeichenfolge im Parameter **errormsg** angibt.

db2secFreeErrormsg (API) - Speicher für Fehlernachricht freigeben

Gibt den Speicher frei, der zur Aufnahme einer Fehlernachricht aus einem früheren API-Aufruf verwendet wurde. Dies ist die einzige API, die keine Fehlernachricht zurückgibt. Wenn diese API einen Fehler zurückgibt, protokolliert DB2 diesen Fehler und setzt die Verarbeitung fort.

API- und Datenstruktursyntax

```
SQL_API_RC ( SQL_API_FN *db2secFreeErrormsg)
                      ( char *errormsg );
```

Parameter der API 'db2secFreeErrormsg'

errormsg
Eingabe. Ein Zeiger auf den Speicher der Fehlernachricht, der von einem früheren API-Aufruf zugeordnet wurde.

db2secFreeGroupListMemory (API) - Speicher für Gruppenliste freigeben

Gibt den Speicher frei, der zur Aufnahme der Liste von Gruppen aus einem früheren Aufruf der API 'db2secGetGroupsForUser' verwendet wurde.

API- und Datenstruktursyntax

```
SQL_API_RC ( SQL_API_FN *db2secFreeGroupListMemory)
                      ( void *ptr,
                        char      **errormsg,
                        db2int32 *errormsglen );
```

Parameter der API 'db2secFreeGroupListMemory'

ptr Eingabe. Zeiger auf den freizugebenden Speicher.

errormsg
Ausgabe. Ein Zeiger auf die Adresse einer vom Plug-in zugeordneten ASCII-Fehlernachrichtenzeichenfolge, die in diesem Parameter zurückgegeben werden kann, wenn die API 'db2secFreeGroupListMemory' nicht erfolgreich ausgeführt wird.

errormsglen
Ausgabe. Ein Zeiger auf einen ganzzahligen Wert (Integer), der die Länge (in Byte) der Fehlernachrichtenzeichenfolge im Parameter **errormsg** angibt.

db2secGetGroupsForUser (API) - Liste von Gruppen für Benutzer abrufen

Gibt eine Liste von Gruppen zurück, zu denen ein Benutzer gehört.

API- und Datenstruktursyntax

```
SQL_API_RC ( SQL_API_FN *db2secGetGroupsForUser)
                      ( const char *authid,
                        db2int32 authidlen,
                        const char *userid,
                        db2int32 useridlen,
                        const char *usernamespace,
                        db2int32 usernamespacelen,
                        db2int32 usernamespacetype,
                        const char *dbname,
                        db2int32 dbnamelen,
                        void *token,
                        db2int32 tokentype,
                        db2int32 location,
                        const char *authpluginname,
                        db2int32 authpluginnamelen,
                        void      **grouplist,
                        db2int32 *numgroups,
                        char      **errormsg,
                        db2int32 *errormsglen );
```

Parameter der API 'db2secGetGroupsForUser'

authid Eingabe. Dieser Parameterwert ist eine SQL-Berechtigungs-ID. Dies bedeutet, dass sie von DB2 Database for Linux, UNIX and Windows in eine Zei-

chenfolge in Großbuchstaben ohne folgende Leerzeichen konvertiert wird. DB2 Database for Linux, UNIX and Windows stellt immer einen Nichtnullwert für den Parameter **authid** bereit. Die API muss eine Liste von Gruppen zurückgeben können, zu denen die Berechtigungs-ID gehört, ohne von den anderen Eingabeparametern abhängig zu sein. Es ist zulässig, eine verkürzte oder leere Liste zuzurückzugeben, wenn sich keine Gruppen bestimmen lassen.

Wenn ein Benutzer nicht vorhanden ist, muss die API den Rückkehrcode DB2SEC_PLUGIN_BADUSER zurückgeben. DB2 Database for Linux, UNIX and Windows behandelt den Fall eines nicht vorhandenen Benutzers nicht als Fehler, da es zulässig ist, wenn eine Berechtigungs-ID keine zugeordneten Gruppen besitzt. Zum Beispiel kann die API 'db2secGetAuthids' eine Berechtigungs-ID zurückgeben, die im Betriebssystem nicht vorhanden ist. Die Berechtigungs-ID ist zwar keinen Gruppen zugeordnet, dennoch können ihr Zugriffsrechte direkt erteilt werden.

Wenn die API nur anhand der Berechtigungs-ID keine vollständige Liste der Gruppen zurückgeben kann, gelten einige Einschränkungen für bestimmte SQL-Funktionen in Bezug auf die Unterstützung von Gruppen. Eine Liste der möglichen Problemszenarios finden Sie unter der Überschrift 'Hinweise' in diesem Abschnitt.

authidlen
Eingabe. Die Länge (in Byte) des Werts des Parameters **authid**. Der DB2-Datenbankmanager stellt immer einen Nichtnullwert für den Parameter **authidlen** bereit.

userid Eingabe. Dieser Wert gibt die Benutzer-ID an, die der Berechtigungs-ID (**authid**) entspricht. Wenn diese API auf dem Server in einem Szenario ohne Verbindung aufgerufen wird, wird dieser Parameter von DB2 Database for Linux, UNIX and Windows nicht gefüllt.

useridlen
Eingabe. Die Länge (in Byte) des Werts des Parameters **userid**.

usernamespace
Eingabe. Der Namensbereich, aus dem die Benutzer-ID abgerufen wurde. Wenn die Benutzer-ID nicht verfügbar ist, wird dieser Parameter vom DB2-Datenbankmanager nicht gefüllt.

usernamespacelen
Eingabe. Die Länge (in Byte) des Werts des Parameters **usernamespace**.

usernamespacetype
Eingabe. Der Typ des Namensbereichs. Folgende Werte für den Parameter **usernamespacetype** (in der Datei db2secPlugin.h definiert) sind gültig:

- DB2SEC_NAMESPACE_SAM_COMPATIBLE: Entspricht einer Benutzernamensdarstellung des Formats 'domäne\meinname'.
- DB2SEC_NAMESPACE_USER_PRINCIPAL: Entspricht einer Benutzernamensdarstellung des Formats 'meinname@domäne.ibm.com'.

Gegenwärtig unterstützt das DB2-Datenbanksystem nur den Wert DB2SEC_NAMESPACE_SAM_COMPATIBLE. Wenn die Benutzer-ID nicht verfügbar ist, wird der Parameter **usernamespacetype** auf den Wert DB2SEC_USER_NAMESPACE_UNDEFINED (in der Datei db2secPlugin.h definiert) gesetzt.

dbname
Eingabe. Der Name der Datenbank, zu der die Verbindung hergestellt wird. Dieser Parameter kann in einem Szenario ohne Verbindung den Wert NULL haben.

dbnamelen
Eingabe. Die Länge (in Byte) des Werts des Parameters **dbname**. Dieser Parameter wird auf den Wert `0` gesetzt, wenn der Parameter **dbname** den Wert NULL in einem Szenario ohne Verbindung hat.

token Eingabe. Ein Zeiger auf Daten, die vom Authentifizierungs-Plug-in bereitgestellt werden. Dieser Parameter wird von DB2 Database for Linux, UNIX and Windows nicht verwendet. Er bietet dem Autor des Plug-ins die Möglichkeit, Benutzer- und Gruppeninformationen zu koordinieren. Dieser Parameter wird möglicherweise nicht in allen Fällen bereitgestellt (z. B. in einem Szenario ohne Verbindung). In einem solchen Fall hat er den Wert NULL. Wenn das verwendete Authentifizierungs-Plug-in auf der GSS-API basiert, wird das Token auf die Kontextkennung (`gss_ctx_id_t`) der GSS-API gesetzt.

tokentype
Eingabe. Gibt den Typ von Daten an, die vom Authentifizierungs-Plug-in bereitgestellt werden. Wenn das verwendete Authentifizierungs-Plug-in auf der GSS-API basiert, wird das Token auf die Kontextkennung (`gss_ctx_id_t`) der GSS-API gesetzt. Wenn das verwendete Authentifizierungs-Plug-in auf Benutzer-ID und Kennwort basiert, gibt dieser Parameter einen generischen Typ an. Folgende Werte für den Parameter **tokentype** (in der Datei `db2secPlugin.h` definiert) sind gültig:

- `DB2SEC_GENERIC`: Gibt an, dass das Token von einem Plug-in auf Benutzer-ID/Kennwort-Basis stammt.
- `DB2SEC_GSSAPI_CTX_HANDLE`: Gibt an, dass das Token von einem Plug-in auf GSS-API-Basis (einschließlich Kerberos) stammt.

location
Eingabe. Gibt an, ob DB2 Database for Linux, UNIX and Windows diese API auf der Clientseite oder der Serverseite aufruft. Folgende Werte für den Parameter 'location' (in der Datei `db2secPlugin.h` definiert) sind gültig:

- `DB2SEC_SERVER_SIDE`: Die API ist auf dem Datenbankserver aufzurufen.
- `DB2SEC_CLIENT_SIDE`: Die API ist auf einem Client aufzurufen.

authpluginname
Eingabe. Der Name des Authentifizierungs-Plug-ins, das die Daten im Token bereitgestellt hat. Die API 'db2secGetGroupsForUser' könnte diese Informationen zur Feststellung der korrekten Gruppenzugehörigkeiten verwenden. Dieser Parameter wird von DB2 Database for Linux, UNIX and Windows möglicherweise nicht gefüllt, wenn die Berechtigungs-ID (**authid**) nicht authentifiziert wurde (z. B. wenn die Berechtigungs-ID nicht dem momentan verbundenen Benutzer entspricht).

authpluginnamelen
Eingabe. Die Länge (in Byte) des Werts des Parameters **authpluginname**.

grouplist
Ausgabe. Die Liste der Gruppen, zu denen der Benutzer gehört. Die Liste der Gruppen muss als Zeiger auf eine Speichersektion zurückgegeben werden, die vom Plug-in zugeordnet wird und verkettete VARCHAR-Werte enthält. (Ein VARCHAR-Wert ist eine Zeichenfeldgruppe, in der das erste Byte die Anzahl der darauf folgenden Byte angibt.) Die Länge ist ein Zei-

chen (CHAR) ohne Vorzeichen (1 Byte). Dies begrenzt die maximale Länge eines Gruppennamens auf 255 Zeichen. Beispiel: "\006GROUP1\ 007MYGROUP\008MYGROUP3". Jeder Gruppenname muss eine gültige DB2-Berechtigungs-ID (authid) sein. Der Speicher für diese Feldgruppe muss vom Plug-in zugeordnet werden. Das Plug-in muss deshalb eine API angeben, zum Beispiel die API 'db2secFreeGroupListMemory', die von DB2 Database for Linux, UNIX and Windows aufgerufen wird, um den Speicher freizugeben.

numgroups
Ausgabe. Die Anzahl der Gruppen, die im Parameter **grouplist** enthalten sind.

errormsg
Ausgabe. Ein Zeiger auf die Adresse einer vom Plug-in zugeordneten ASCII-Fehlernachrichtenzeichenfolge, die in diesem Parameter zurückgegeben werden kann, wenn die API 'db2secGetGroupsForUser' nicht erfolgreich ausgeführt wird.

errormsglen
Ausgabe. Ein Zeiger auf einen ganzzahligen Wert (Integer), der die Länge (in Byte) der Fehlernachrichtenzeichenfolge im Parameter **errormsg** angibt.

Hinweise

In der folgenden Liste werden Problemszenarios aufgeführt, die auftreten können, wenn von dieser API eine unvollständige Liste an DB2 Database for Linux, UNIX and Windows zurückgegeben wird:

- Durch die Anweisung CREATE SCHEMA bereitgestellte alternative Berechtigung: Die Gruppensuchfunktion wird am Parameter für den Berechtigungsnamen (AUTHORIZATION NAME) ausgeführt, wenn verschachtelte Anweisungen CREATE in der Anweisung CREATE SCHEMA enthalten sind.
- Verarbeitung einer JAR-Datei in einer Umgebung mit exklusiver Parallelverarbeitung (MPP): In einer MPP-Umgebung werden die Anforderungen zur JAR-Verarbeitung vom Koordinatorknoten mit der Sitzungsberechtigungs-ID gesendet. Der Katalogknoten empfängt die Anforderungen und verarbeitet die JAR-Dateien auf der Basis der Berechtigung der Sitzungsberechtigungs-ID (d. h. des Benutzers, der die Anforderungen zur JAR-Verarbeitung ausführt).
  - Installieren einer JAR-Datei: Die Sitzungsberechtigungs-ID muss über eine der folgenden Berechtigungen verfügen: DBADM oder CREATEIN (implizit oder explizit für das JAR-Schema). Die Operation schlägt fehl, wenn die zuvor angegebenen Berechtigungen einer Gruppe erteilt wurden, die die Sitzungsberechtigungs-ID enthält, nicht aber explizit der Sitzungsberechtigungs-ID.
  - Entfernen einer JAR-Datei: Die Sitzungsberechtigungs-ID muss über eine der folgenden Berechtigungen verfügen: DBADM oder DROPIN (implizit oder explizit für das JAR-Schema), oder muss der Benutzer sein, der die JAR-Datei definiert hat. Die Operation schlägt fehl, wenn die zuvor angegebenen Berechtigungen einer Gruppe erteilt wurden, die die Sitzungsberechtigungs-ID enthält, nicht aber explizit der Sitzungsberechtigungs-ID, und wenn die Sitzungsberechtigungs-ID sich nicht auf den Benutzer bezieht, der die JAR-Datei definiert hat.
  - Ersetzen einer JAR-Datei: Dies ist dasselbe wie das Entfernen einer JAR-Datei mit anschließendem Installieren einer JAR-Datei. Beide zuvor genannten Szenarios sind zu beachten.
- Absetzen der Anweisung SET SESSION_USER: Nachfolgende DB2-Operationen werden unter dem Kontext der durch diese Anweisung angegebenen Berechti-

gungs-ID ausgeführt. Diese Operationen schlagen fehl, wenn die erforderlichen Berechtigungen einer der Gruppen des Sitzungsbenutzers (SESSION_USER) und nicht explizit der Berechtigungs-ID des Sitzungsbenutzers erteilt sind.

db2secGroupPluginInit (API) - Gruppen-Plug-in initialisieren

Initialisierungs-API für das Plug-in zum Abrufen von Gruppen, das der DB2-Datenbankmanager unmittelbar nach dem Laden des Plug-ins aufruft.

API- und Datenstruktursyntax

```
SQL_API_RC SQL_API_FN db2secGroupPluginInit
                              ( db2int32 version,
                                void *group_fns,
                                db2secLogMessage *logMessage_fn,
                                char     **errormsg,
                                db2int32 *errormsglen );
```

Parameter der API 'db2secGroupPluginInit'

version

Eingabe. Die höchste Version der API, die von der Instanz unterstützt wird, die dieses Plug-in lädt. Der Wert `DB2SEC_API_VERSION` (in `db2secPlugin.h`) enthält die aktuellste Versionsnummer der API, die vom DB2-Datenbankmanager gegenwärtig unterstützt wird.

group_fns

Ausgabe. Ein Zeiger auf die Struktur 'db2secGroupFunctions_<versionsnummer>' (auch als 'group_functions_<versionsnummer>' bezeichnet). Die Struktur 'db2secGroupFunctions_<versionsnummer>' enthält Zeiger auf die APIs, die für das Plug-in zum Abrufen von Gruppen implementiert sind. In Zukunft kann es andere Versionen der APIs geben (z. B. 'db2secGroupFunctions_<versionsnummer>'), sodass der Parameter **group_fns** als Zeiger auf die Struktur 'db2secGroupFunctions_<versionsnummer>' umgesetzt wird, die der Version entspricht, die vom Plug-in implementiert wird. Der erste Parameter der Struktur 'group_functions_<versionsnummer>' teilt DB2 Database for Linux, UNIX and Windows die Version der APIs mit, die vom Plug-in implementiert werden. Hinweis: Die Umsetzung (CAST) erfolgt nur, wenn die DB2-Version höher oder gleich der Version der APIs ist, die vom Plug-in implementiert werden. Die Versionsnummer stellt die Version der APIs dar, die von dem Plug-in implementiert werden. Der Parameter **pluginType** sollte auf den Wert `DB2SEC_PLUGIN_TYPE_GROUP` gesetzt werden.

logMessage_fn

Eingabe. Ein Zeiger auf die API 'db2secLogMessage', die vom DB2-Datenbanksystem implementiert wird. Die API 'db2secGroupPluginInit' kann die API 'db2secLogMessage' aufrufen, um Nachrichten in den **db2diag**-Protokolldateien zu Debug- oder Informationszwecken zu protokollieren. Der erste Parameter (**level**) der API 'db2secLogMessage' gibt den Typ der zu diagnostizierenden Fehler an, die in den **db2diag**-Protokolldateien aufgezeichnet werden. Die beiden letzten Parameter geben die Nachrichtenzeichenfolge und ihre Länge an. Die folgenden Werte sind für den ersten Parameter der API 'db2secLogMessage' (in der Datei `db2secPlugin.h` definiert) gültig:

- `DB2SEC_LOG_NONE`: (0) - Keine Protokollierung
- `DB2SEC_LOG_CRITICAL`: (1) - Schwer wiegender Fehler
- `DB2SEC_LOG_ERROR`: (2) - Fehler

- DB2SEC_LOG_WARNING: (3) - Warnung
- DB2SEC_LOG_INFO: (4) Informativ

Der Nachrichtentext wird in den db2diag-Protokolldateien nur gezeigt, wenn der Wert des Parameters **level** der API 'db2secLogMessage' kleiner oder gleich dem Wert des Konfigurationsparameters **diaglevel** des Datenbankmanagers ist. Wenn Sie zum Beispiel den Wert DB2SEC_LOG_INFO verwenden, wird der Nachrichtentext in den **db2diag**-Protokolldateien nur gezeigt, wenn der Konfigurationsparameter **diaglevel** des Datenbankmanagers auf den Wert 4 gesetzt ist.

errormsg
Ausgabe. Ein Zeiger auf die Adresse einer vom Plug-in zugeordneten ASCII-Fehlernachrichtenzeichenfolge, die in diesem Parameter zurückgegeben werden kann, wenn die API 'db2secGroupPluginInit' nicht erfolgreich ausgeführt wird.

errormsglen
Ausgabe. Ein Zeiger auf einen ganzzahligen Wert (Integer), der die Länge (in Byte) der Fehlernachrichtenzeichenfolge im Parameter **errormsg** angibt.

db2secPluginTerm - Gruppen-Plug-in-Ressourcen bereinigen

Gibt die Ressourcen frei, die vom Plug-in zum Abrufen der Gruppen verwendet wurden.

Diese Anwendungsprogrammierschnittstelle (API) wird vom DB2-Datenbankmanager kurz vor dem Entladen des Plug-ins zum Gruppenabruf aufgerufen. Sie sollte in einer Weise implementiert werden, in der sie eine ordnungsgemäße Bereinigung aller Ressourcen ausführt, die von der Bibliothek des Plug-ins genutzt werden. Sie sollte zum Beispiel den vom Plug-in zugeordneten Speicher freigeben, Dateien schließen, die noch geöffnet sind, und Netzwertverbindungen schließen. Es liegt in der Zuständigkeit des Plug-ins, diese Ressourcen zu verfolgen, um sie freigeben zu können. Diese API wird auf keinem Windows-Betriebssystem aufgerufen.

API- und Datenstruktursyntax

```
SQL_API_RC ( SQL_API_FN *db2secPluginTerm)
                     ( char      **errormsg,
                       db2int32 *errormsglen );
```

Parameter der API 'db2secPluginTerm'

errormsg
Ausgabe. Ein Zeiger auf die Adresse einer vom Plug-in zugeordneten ASCII-Fehlernachrichtenzeichenfolge, die in diesem Parameter zurückgegeben werden kann, wenn die API 'db2secPluginTerm' nicht erfolgreich ausgeführt wird.

errormsglen
Ausgabe. Ein Zeiger auf einen ganzzahligen Wert (Integer), der die Länge (in Byte) der Fehlernachrichtenzeichenfolge im Parameter **errormsg** angibt.

APIs für Plug-ins zur Benutzer-ID/Kennwort-Authentifizierung

Für das Plug-in-Modul zur Benutzer-ID/Kennwort-Authentifizierung müssen Sie die folgenden clientseitigen Anwendungsprogrammierschnittstellen (APIs) implementieren:

- db2secClientAuthPluginInit

Anmerkung: Die API 'db2secClientAuthPluginInit' empfängt als Eingabe den Zeiger `*logMessage_fn` auf eine API mit dem folgenden Prototyp:

```
SQL_API_RC (SQL_API_FN db2secLogMessage)
(
db2int32 level,
void    *data,
db2int32 length
);
```

Die API 'db2secLogMessage' ermöglicht dem Plug-in, Nachrichten in den **db2diag**-Protokolldateien zu Debug- oder Informationszwecken zu protokollieren. Diese API wird vom DB2-Datenbanksystem bereitgestellt, sodass Sie sie nicht zu implementieren brauchen.

- db2secClientAuthPluginTerm
- db2secGenerateInitialCred (nur für 'gssapi' verwendet)
- db2secRemapUserid (optional)
- db2secGetDefaultLoginContext
- db2secValidatePassword
- db2secProcessServerPrincipalName (nur für GSS-API)
- db2secFreeToken (Funktionen zur Freigabe von Speicher, der von der DLL-Datei belegt wird)
- db2secFreeErrormsg
- db2secFreeInitInfo
- Die einzige API, die extern auflösbar sein muss, ist die API db2secClientAuthPluginInit. Diese API empfängt den Parameter `void *`, der auf eine der folgenden Arten umgesetzt werden muss:

```
typedef struct db2secUseridPasswordClientAuthFunctions_1
{
db2int32 version;
db2int32 plugintype;

SQL_API_RC (SQL_API_FN * db2secGetDefaultLoginContext)
(
char        authid[DB2SEC_MAX_AUTHID_LENGTH],
db2int32   *authidlen,
char        userid[DB2SEC_MAX_USERID_LENGTH],
db2int32   *useridlen,
db2int32    useridtype,
char        usernamespace[DB2SEC_MAX_USERNAMESPACE_LENGTH],
db2int32   *usernamespacelen,
db2int32   *usernamespacetype,
const char *dbname,
db2int32    dbnamelen,
void      **token,
char      **errormsg,
db2int32   *errormsglen
);
/* Optional */
SQL_API_RC (SQL_API_FN * db2secRemapUserid)
(
char        userid[DB2SEC_MAX_USERID_LENGTH],
db2int32   *useridlen,
char        usernamespace[DB2SEC_MAX_USERNAMESPACE_LENGTH],
db2int32   *usernamespacelen,
db2int32   *usernamespacetype,
char        password[DB2SEC_MAX_PASSWORD_LENGTH],
db2int32   *passwordlen,
char        newpassword[DB2SEC_MAX_PASSWORD_LENGTH],
db2int32   *newpasswordlen,
```

```
const char *dbname,
db2int32    dbnamelen,
char      **errormsg,
db2int32   *errormsglen
);

SQL_API_RC (SQL_API_FN * db2secValidatePassword)
(
const char *userid,
db2int32    useridlen,
const char *usernamespace,
db2int32    usernamespacelen,
db2int32    usernamespacetype,
const char *password,
db2int32    passwordlen,
const char *newpassword,
db2int32    newpasswordlen,
const char *dbname,
db2int32    dbnamelen,
db2Uint32   connection_details,
void      **token,
char      **errormsg,
db2int32   *errormsglen
);

SQL_API_RC (SQL_API_FN * db2secFreeToken)
(
void     **token,
char     **errormsg,
db2int32  *errormsglen
);

SQL_API_RC (SQL_API_FN * db2secFreeErrormsg)
(
char *errormsg
);

SQL_API_RC (SQL_API_FN * db2secClientAuthPluginTerm)
(
char     **errormsg,
db2int32  *errormsglen
);
}
```

oder

```
typedef struct db2secGssapiClientAuthFunctions_1
{
db2int32 version;
db2int32 plugintype;

SQL_API_RC (SQL_API_FN * db2secGetDefaultLoginContext)
(
char        authid[DB2SEC_MAX_AUTHID_LENGTH],
db2int32   *authidlen,
char        userid[DB2SEC_MAX_USERID_LENGTH],
db2int32   *useridlen,
db2int32    useridtype,
char        usernamespace[DB2SEC_MAX_USERNAMESPACE_LENGTH],
db2int32   *usernamespacelen,
db2int32   *usernamespacetype,
const char *dbname,
db2int32    dbnamelen,
void      **token,
char      **errormsg,
db2int32   *errormsglen
);
```

```
SQL_API_RC (SQL_API_FN * db2secProcessServerPrincipalName)
(
const void *data,
gss_name_t *gssName,
char      **errormsg,
db2int32   *errormsglen
);

SQL_API_RC (SQL_API_FN * db2secGenerateInitialCred)
(
const char    *userid,
db2int32       useridlen,
const char    *usernamespace,
db2int32       usernamespacelen,
db2int32       usernamespacetype,
const char    *password,
db2int32       passwordlen,
const char    *newpassword,
db2int32       newpasswordlen,
const char    *dbname,
db2int32       dbnamelen,
gss_cred_id_t *pGSSCredHandle,
void         **initInfo,
char         **errormsg,
db2int32      *errormsglen
);

SQL_API_RC (SQL_API_FN * db2secFreeToken)
(
void      *token,
char     **errormsg,
db2int32  *errormsglen
);

SQL_API_RC (SQL_API_FN * db2secFreeErrormsg)
(
char *errormsg
);


SQL_API_RC (SQL_API_FN * db2secFreeInitInfo)
(
void      *initInfo,
char     **errormsg,
db2int32  *errormsglen
);

SQL_API_RC (SQL_API_FN * db2secClientAuthPluginTerm)
(
char     **errormsg,
db2int32  *errormsglen
);

/* GSS-API-spezifische Funktionen
   Parameterliste: siehe db2secPlugin.h */

  OM_uint32 (SQL_API_FN * gss_init_sec_context )(<parameterliste>);
  OM_uint32 (SQL_API_FN * gss_delete_sec_context )(<parameterliste>);
  OM_uint32 (SQL_API_FN * gss_display_status )(<parameterliste>);
  OM_uint32 (SQL_API_FN * gss_release_buffer )(<parameterliste>);
  OM_uint32 (SQL_API_FN * gss_release_cred )(<parameterliste>);
  OM_uint32 (SQL_API_FN * gss_release_name )(<parameterliste>);
  }
```

Sie sollten die Struktur db2secUseridPasswordClientAuthFunctions_1 verwenden, wenn Sie ein Plug-in zur Benutzer-ID/Kennwort-Authentifizierung schrei-

ben. Wenn Sie ein GSS-API-Plug-in (einschließlich Kerberos) schreiben, sollten Sie die Struktur db2secGssapiClientAuthFunctions_1 verwenden.

Für die Plug-in-Bibliothek zur Benutzer-ID/Kennwort-Authentifizierung müssen Sie die folgenden serverseitigen APIs implementieren:

- db2secServerAuthPluginInit

  Die API 'db2secServerAuthPluginInit' empfängt als Eingabe den Zeiger *logMessage_fn auf die API 'db2secLogMessage' sowie den Zeiger *getConDetails_fn auf die API 'db2secGetConDetails' mit den folgenden Prototypen:

```
SQL_API_RC (SQL_API_FN db2secLogMessage)
(
db2int32 level,
void    *data,
db2int32 length
);

SQL_API_RC (SQL_API_FN db2secGetConDetails)
(
db2int32    conDetailsVersion,
const void *pConDetails
);
```

  Die API 'db2secLogMessage' ermöglicht dem Plug-in, Nachrichten in den **db2diag**-Protokolldateien zu Debug- oder Informationszwecken zu protokollieren. Die API 'db2secGetConDetails' ermöglicht dem Plug-in das Abrufen von Details zu dem Client, der versucht, eine Datenbankverbindung herzustellen. Sowohl die API 'db2secLogMessage' als auch die API 'db2secGetConDetails' wird vom DB2-Datenbanksystem bereitgestellt, sodass Sie sie nicht zu implementieren brauchen. Die API 'db2secGetConDetails' empfängt als zweiten Parameter (**pConDetails**) wiederum einen Zeiger auf eine der folgenden Strukturen:

  db2sec_con_details_1:

```
typedef struct db2sec_con_details_1
{
  db2int32  clientProtocol;
  db2Uint32 clientIPAddress;
  db2Uint32 connect_info_bitmap;
  db2int32  dbnameLen;
  char      dbname[DB2SEC_MAX_DBNAME_LENGTH + 1];
} db2sec_con_details_1;
```

  db2sec_con_details_2:

```
typedef struct db2sec_con_details_2
{
  db2int32  clientProtocol;     /* Siehe SQL_PROTOCOL_ in sqlenv.h */
  db2Uint32 clientIPAddress;    /* Gesetzt, wenn Protokoll = TCPIP4 */
  db2Uint32 connect_info_bitmap;
  db2int32  dbnameLen;
  char dbname[DB2SEC_MAX_DBNAME_LENGTH + 1];
  db2Uint32 clientIP6Address[4];/* Gesetzt, wenn Protokoll = TCPIP6 */
} db2sec_con_details_2;
```

  db2sec_con_details_3:

```
typedef struct db2sec_con_details_3
{
  db2int32  clientProtocol;     /* Siehe SQL_PROTOCOL_ in sqlenv.h */
  db2Uint32 clientIPAddress; /* Gesetzt, wenn Protokoll = TCPIP4 */
  db2Uint32 connect_info_bitmap;
  db2int32 dbnameLen;
  char dbname[DB2SEC_MAX_DBNAME_LENGTH + 1];
```

```
  db2Uint32 clientIP6Address[4];/* Gesetzt, wenn Protokoll = TCPIP6 */
  db2Uint32 clientPlatform; /* SQLM_PLATFORM_* aus sqlmon.h */
  db2Uint32 _reserved[16];
} db2sec_con_details_3;
```

Die möglichen Werte für **conDetailsVersion** sind DB2SEC_CON_DETAILS_VERSION_1, DB2SEC_CON_DETAILS_VERSION_2 und DB2SEC_CON_DETAILS_VERSION_3, die die Version der API darstellen.

Anmerkung: Wenn Sie 'db2sec_con_details_1', 'db2sec_con_details_2' oder 'db2sec_con_details_3' verwenden, beachten Sie folgende Hinweise:

- Vorhandene Plug-ins, die die Struktur 'db2sec_con_details_1' und den Wert DB2SEC_CON_DETAILS_VERSION_1 verwenden, funktionieren weiterhin wie mit Version 8.2, wenn sie die API 'db2GetConDetails' aufrufen. Wenn diese API auf einer IPv4-Plattform aufgerufen wird, wird die IP-Adresse des Clients im Feld **clientIPAddress** der Struktur zurückgegeben. Wenn diese API auf einer IPv6-Plattform aufgerufen wird, wird im Feld **clientIPAddress** der Wert 0 zurückgegeben. Zum Abrufen der IP-Adresse des Clients auf einer IPv6-Plattform muss der Code des Sicherheits-Plug-ins so geändert werden, dass er entweder die Struktur 'db2sec_con_details_2' und den Wert DB2SEC_CON_DETAILS_VERSION_2 oder die Struktur 'db2sec_con_details_3' und den Wert DB2SEC_CON_DETAILS_VERSION_3 verwendet.
- Neue Plug-ins sollten die Struktur 'db2sec_con_details_3' und den Wert DB2SEC_CON_DETAILS_VERSION_3 verwenden. Wenn die API 'db2secGetConDetails' auf einer IPv4-Plattform aufgerufen wird, wird die IP-Adresse des Clients im Feld **clientIPAddress** der Struktur 'db2sec_con_details_3' zurückgegeben. Wenn die API auf einer IPv6-Plattform aufgerufen wird, wird die IP-Adresse des Clients im Feld **clientIP6Address** der Struktur 'db2sec_con_details_3' zurückgegeben. Das Feld **clientProtocol** der Struktur für die Verbindungsdetails wird auf den Wert SQL_PROTOCOL_TCPIP (für IPv4 bei Version 1 der Struktur), SQL_PROTOCOL_TCPIP4 (für IPv4 bei Version 2 der Struktur) oder SQL_PROTOCOL_TCPIP6 (für IPv6 bei Version 2 oder Version 3 der Struktur) gesetzt.
- Die Struktur 'db2sec_con_details_3' stimmt mit der Struktur 'db2sec_con_details_2' überein, enthält jedoch ein zusätzliches Feld (**clientPlatform**), das den Typ der Clientplattform (wie er durch die Übertragungsschicht gemeldet wird) mithilfe der Plattformtypkonstanten angibt, die in sqlmon.h definiert sind (z. B. SQLM_PLATFORM_AIX).

- db2secServerAuthPluginTerm
- db2secValidatePassword
- db2secGetAuthIDs
- db2secDoesAuthIDExist
- db2secFreeToken
- db2secFreeErrormsg
- Die einzige API, die extern auflösbar sein muss, ist die API db2secServerAuthPluginInit. Diese API empfängt den Parameter void *, der auf eine der folgenden Arten umgesetzt werden muss:

```
typedef struct db2secUseridPasswordServerAuthFunctions_1
{
db2int32 version;
db2int32 plugintype;

  /* Parameterlisten aus Gründen der Lesbarkeit nicht angegeben.
     Parameter: siehe oben. */
```

```
SQL_API_RC (SQL_API_FN * db2secValidatePassword)(<parameterliste>);
SQL_API_RC (SQL_API_FN * db2secGetAuthIDs)(<parameterliste);
SQL_API_RC (SQL_API_FN * db2secDoesAuthIDExist)(<parameterliste>);
SQL_API_RC (SQL_API_FN * db2secFreeToken)(<parameterliste>);
SQL_API_RC (SQL_API_FN * db2secFreeErrormsg)(<parameterliste>);
SQL_API_RC (SQL_API_FN * db2secServerAuthPluginTerm)();
} userid_password_server_auth_functions;
```

oder

```
typedef struct db2secGssapiServerAuthFunctions_1
{
db2int32 version;
db2int32 plugintype;
gss_buffer_desc serverPrincipalName;
gss_cred_id_t ServerCredHandle;
SQL_API_RC (SQL_API_FN * db2secGetAuthIDs)(<parameterliste);
SQL_API_RC (SQL_API_FN * db2secDoesAuthIDExist)(<parameterliste>);
SQL_API_RC (SQL_API_FN * db2secFreeErrormsg)(<parameterliste>);
SQL_API_RC (SQL_API_FN * db2secServerAuthPluginTerm)();

/* GSS-API-spezifische Funktionen:
   Parameterliste: siehe db2secPlugin.h */
OM_uint32 (SQL_API_FN * gss_accept_sec_context )(<parameterliste>);
OM_uint32 (SQL_API_FN * gss_display_name )(<parameterliste>);
OM_uint32 (SQL_API_FN * gss_delete_sec_context )(<parameterliste>);
OM_uint32 (SQL_API_FN * gss_display_status )(<parameterliste>);
OM_uint32 (SQL_API_FN * gss_release_buffer )(<parameterliste>);
OM_uint32 (SQL_API_FN * gss_release_cred )(<parameterliste>);
OM_uint32 (SQL_API_FN * gss_release_name )(<parameterliste>);

} gssapi_server_auth_functions;
```

Sie sollten die Struktur 'db2secUseridPasswordServerAuthFunctions_1' verwenden, wenn Sie ein Plug-in zur Benutzer-ID/Kennwort-Authentifizierung schreiben. Wenn Sie ein GSS-API-Plug-in (einschließlich Kerberos) schreiben, sollten Sie die Struktur db2secGssapiServerAuthFunctions_1 verwenden.

db2secClientAuthPluginInit (API) - Plug-in zur Clientauthentifizierung initialisieren

Initialisierungs-API für das Plug-in zur Clientauthentifizierung, das der DB2-Datenbankmanager unmittelbar nach dem Laden des Plug-ins aufruft.

API- und Datenstruktursyntax

```
SQL_API_RC SQL_API_FN db2secClientAuthPluginInit
                           ( db2int32 version,
                             void *client_fns,
                             db2secLogMessage *logMessage_fn,
                             char    **errormsg,
                             db2int32 *errormsglen );
```

Parameter der API 'db2secClientAuthPluginInit'

version
Eingabe. Die höchste Versionsnummer der API, die vom DB2-Datenbankmanager gegenwärtig unterstützt wird. Der Wert `DB2SEC_API_VERSION` (in `db2secPlugin.h` definiert) enthält die aktuellste Versionsnummer der API, die von DB2 Database for Linux, UNIX and Windows gegenwärtig unterstützt wird.

client_fns
Ausgabe. Ein Zeiger auf Speicher, die vom DB2-Datenbankmanager für eine Struktur 'db2secGssapiClientAuthFunctions_<versionsnummer>' (auch als

'gssapi_client_auth_functions_<versionsnummer>') bekannt, wenn die GSS-API-Authentifizierung verwendet wird, oder für eine Struktur 'db2secUseridPasswordClientAuthFunctions_<versionsnummer>' (auch als 'userid_password_client_auth_functions_<versionsnummer>' bekannt), wenn die Benutzer-ID/Kennwort-Authentifizierung verwendet wird, bereitgestellt wird. Die Struktur 'db2secGssapiClientAuthFunctions_<versionsnummer>' und die Struktur 'db2secUseridPasswordClientAuthFunctions_<versionsnummer>' enthalten wiederum Zeiger auf die APIs, die für die GSS-API-Authentifizierung bzw. die Benutzer-ID/Kennwort-Authentifizierung implementiert sind. In zukünftigen Versionen von DB2 Database for Linux, UNIX and Windows wird es möglicherweise andere Versionen der APIs geben, sodass der Parameter **client_fns** als Zeiger auf die Struktur 'gssapi_client_auth_functions_<versionsnummer>' umgesetzt wird, die der Version entspricht, die vom Plug-in implementiert wird.

Der erste Parameter der Struktur 'gssapi_client_auth_functions_<versionsnummer>' bzw. der Struktur 'userid_password_client_auth_functions_<versionsnummer>' teilt dem DB2-Datenbankmanager die Version der APIs mit, die vom Plug-in implementiert werden.

Anmerkung: Die Umsetzung (CAST) erfolgt nur, wenn die DB2-Version höher oder gleich der Version der APIs ist, die vom Plug-in implementiert werden.

In der Struktur 'gssapi_server_auth_functions_<versionsnummer>' bzw. 'userid_password_server_auth_functions_<versionsnummer>' sollte der Parameter **plugintype** auf einen der Werte DB2SEC_PLUGIN_TYPE_USERID_PASSWORD, DB2SEC_PLUGIN_TYPE_GSSAPI oder **DB2SEC_PLUGIN_TYPE_KERBEROS** gesetzt werden. Andere Werte können in zukünftigen Versionen der API definiert werden.

logMessage_fn
Eingabe. Ein Zeiger auf die API 'db2secLogMessage', die vom DB2-Datenbankmanager implementiert wird. Die API 'db2secClientAuthPluginInit' kann die API 'db2secLogMessage' aufrufen, um Nachrichten in den **db2diag**-Protokolldateien zu Debug- oder Informationszwecken zu protokollieren. Der erste Parameter (**level**) der API 'db2secLogMessage' gibt den Typ der zu diagnostizierenden Fehler an, die in den **db2diag**-Protokolldateien aufgezeichnet werden. Die beiden letzten Parameter geben die Nachrichtenzeichenfolge und ihre Länge an. Die folgenden Werte sind für den ersten Parameter der API 'db2secLogMessage' (in der Datei db2secPlugin.h definiert) gültig:

- DB2SEC_LOG_NONE (0) - Keine Protokollierung
- DB2SEC_LOG_CRITICAL (1) - Schwer wiegender Fehler
- DB2SEC_LOG_ERROR (2) - Fehler
- DB2SEC_LOG_WARNING (3) - Warnung
- DB2SEC_LOG_INFO (4) Informativ

Der Nachrichtentext wird nur in den **db2diag**-Protokolldateien angezeigt, wenn der Wert des Parameters 'level' der API 'db2secLogMessage' kleiner oder gleich dem Wert des Konfigurationsparameters **diaglevel** des Datenbankmanagers ist. Wenn Sie zum Beispiel den Wert DB2SEC_LOG_INFO verwenden, wird der Nachrichtentext in den **db2diag**-Protokolldateien nur gezeigt, wenn der Konfigurationsparameter **diaglevel** des Datenbankmanagers auf den Wert 4 gesetzt ist.

errormsg
Ausgabe. Ein Zeiger auf die Adresse einer vom Plug-in zugeordneten ASCII-

Fehlernachrichtenzeichenfolge, die in diesem Parameter zurückgegeben werden kann, wenn die API 'db2secClientAuthPluginInit' nicht erfolgreich ausgeführt wird.

errormsglen
Ausgabe. Ein Zeiger auf einen ganzzahligen Wert (Integer), der die Länge (in Byte) der Fehlernachrichtenzeichenfolge im Parameter **errormsg** angibt.

db2secClientAuthPluginTerm (API) - Ressourcen für Plug-in zur Clientauthentifizierung bereinigen

Gibt Ressourcen frei, die vom Plug-in zur Clientauthentifizierung verwendet wurden.

Diese Anwendungsprogrammierschnittstelle (API) wird vom DB2-Datenbankmanager kurz vor dem Entladen des Plug-ins zur Clientauthentifizierung aufgerufen. Sie sollte in einer Weise implementiert werden, in der sie eine ordnungsgemäße Bereinigung aller Ressourcen ausführt, die von der Bibliothek des Plug-ins genutzt werden. Sie sollte zum Beispiel den vom Plug-in zugeordneten Speicher freigeben, Dateien schließen, die noch geöffnet sind, und Netzwertverbindungen schließen. Es liegt in der Zuständigkeit des Plug-ins, diese Ressourcen zu verfolgen, um sie freigeben zu können. Diese API wird auf keinem Windows-Betriebssystem aufgerufen.

API- und Datenstruktursyntax

```
SQL_API_RC ( SQL_API_FN *db2secClientAuthPluginTerm)
                      ( char     **errormsg,
                        db2int32 *errormsglen);
```

Parameter der API 'db2secClientAuthPluginTerm'

errormsg
Ausgabe. Ein Zeiger auf die Adresse einer vom Plug-in zugeordneten ASCII-Fehlernachrichtenzeichenfolge, die in diesem Parameter zurückgegeben werden kann, wenn die API 'db2secClientAuthPluginTerm' nicht erfolgreich ausgeführt wird.

errormsglen
Ausgabe. Ein Zeiger auf einen ganzzahligen Wert (Integer), der die Länge (in Byte) der Fehlernachrichtenzeichenfolge im Parameter **errormsg** angibt.

db2secDoesAuthIDExist - Vorhandensein der Berechtigungs-ID überprüfen

Ermittelt, ob **authid** einen einzelnen Benutzer darstellt (z. B. ob die API **authid** einer externen Benutzer-ID zuordnen kann).

Die API sollte den Wert DB2SEC_PLUGIN_OK zurückgeben, wenn sie erfolgreich ist, d. h. wenn die Berechtigungs-ID (**authid**) gültig ist. Ist die Berechtigungs-ID nicht gültig, sollte sie den Wert DB2SEC_PLUGIN_INVALID_USERORGROUP zurückgeben. Lässt sich das Vorhandensein von **authid** nicht feststellen, sollte der Wert DB2SEC_PLUGIN_USERSTATUSNOTKNOWN zurückgegeben werden.

API- und Datenstruktursyntax

```
SQL_API_RC ( SQL_API_FN *db2secDoesAuthIDExist)
                      ( const char *authid,
                        db2int32 authidlen,
                        char       **errormsg,
                        db2int32 *errormsglen );
```

Parameter der API 'db2secDoesAuthIDExist'

authid
Eingabe. Die zu überprüfende Berechtigungs-ID. Dieser Wert liegt in Großbuchstaben und ohne folgende Leerzeichen vor.

authidlen
Eingabe. Die Länge (in Byte) des Werts des Parameters **authid**.

errormsg
Ausgabe. Ein Zeiger auf die Adresse einer vom Plug-in zugeordneten ASCII-Fehlernachrichtenzeichenfolge, die in diesem Parameter zurückgegeben werden kann, wenn die API 'db2secDoesAuthIDExist' nicht erfolgreich ausgeführt wird.

errormsglen
Ausgabe. Ein Zeiger auf einen ganzzahligen Wert (Integer), der die Länge der Fehlernachrichtenzeichenfolge im Parameter **errormsg** angibt.

db2secFreeInitInfo (API) - Von 'db2secGenerateInitialCred' genutzte Ressourcen bereinigen

Gibt alle Ressourcen frei, die durch die API 'db2secGenerateInitialCred' zugeordnet wurden. Dazu können zum Beispiel interne Kennungen (Handles) für Kontexte zugrunde liegender Mechanismen oder ein für die GSS-API erstellter Cache für Berechtigungsnachweise gehören.

API- und Datenstruktursyntax

```
SQL_API_RC ( SQL_API_FN *db2secFreeInitInfo)
                        ( void *initinfo,
                          char     **errormsg,
                          db2int32 *errormsglen);
```

Parameter der API 'db2secFreeInitInfo'

initinfo
Eingabe. Ein Zeiger auf Daten, die dem DB2-Datenbankmanager nicht bekannt sind. Das Plug-in kann diesen Speicher verwenden, um eine Liste von Ressourcen zu verwalten, die im Prozess der Generierung der Berechtigungsnachweiskennung zugeordnet werden. Diese Ressourcen werden durch Aufrufen dieser API freigegeben.

errormsg
Ausgabe. Ein Zeiger auf die Adresse einer vom Plug-in zugeordneten ASCII-Fehlernachrichtenzeichenfolge, die in diesem Parameter zurückgegeben werden kann, wenn die API 'db2secFreeInitInfo' nicht erfolgreich ausgeführt wird.

errormsglen
Ausgabe. Ein Zeiger auf einen ganzzahligen Wert (Integer), der die Länge (in Byte) der Fehlernachrichtenzeichenfolge im Parameter **errormsg** angibt.

db2secFreeToken (API) - Vom Token belegten Speicher freigeben

Gibt den Speicher frei, der von einem Token belegt wird. Diese Anwendungsprogrammierschnittstelle (API) wird vom DB2-Datenbankmanager aufgerufen, wenn er den Speicher, der von dem Parameter 'token' belegt wird, nicht mehr benötigt.

API- und Datenstruktursyntax

```
SQL_API_RC ( SQL_API_FN *db2secFreeToken)
                     ( void *token,
                       char       **errormsg,
                       db2int32 *errormsglen );
```

Parameter der API 'db2secFreeToken'

token
Eingabe. Zeiger auf den freizugebenden Speicher.

errormsg
Ausgabe. Ein Zeiger auf die Adresse einer vom Plug-in zugeordneten ASCII-Fehlernachrichtenzeichenfolge, die in diesem Parameter zurückgegeben werden kann, wenn die API 'db2secFreeToken' nicht erfolgreich ausgeführt wird.

errormsglen
Ausgabe. Ein Zeiger auf einen ganzzahligen Wert (Integer), der die Länge (in Byte) der Fehlernachrichtenzeichenfolge im Parameter **errormsg** angibt.

db2secGenerateInitialCred (API) - Anfangsberechtigungsnachweise generieren

Die API 'db2secGenerateInitialCred' ruft die GSS-API-Anfangsberechtigungsnachweise anhand der übergebenen Kombination aus Benutzer-ID und Kennwort ab.

Für Kerberos ist dies das Ticket-Granting-Ticket (TGT). Die Berechtigungsnachweiskennung, die im Parameter **pGSSCredHandle** zurückgegeben wird, ist die Kennung, die mit der API 'gss_init_sec_context' verwendet wird. Dabei muss es sich entweder um INITIATE-Berechtigungsnachweise oder um BOTH-Berechtigungsnachweise handeln. Die API 'db2secGenerateInitialCred' wird nur aufgerufen, wenn eine Benutzer-ID und möglicherweise ein Kennwort angegeben werden. Ansonsten gibt der DB2-Datenbankmanager den Wert `GSS_C_NO_CREDENTIAL` an, wenn er die API 'gss_init_sec_context' aufruft, um anzuzeigen, dass die aus dem aktuellen Anmeldekontext abgerufenen Standardberechtigungsnachweise zu verwenden sind.

API- und Datenstruktursyntax

```
SQL_API_RC ( SQL_API_FN *db2secGenerateInitialCred)
                     ( const char *userid,
                       db2int32 useridlen,
                       const char *usernamespace,
                       db2int32 usernamespacelen,
                       db2int32 usernamespacetype,
                       const char *password,
                       db2int32 passwordlen,
                       const char *newpassword,
                       db2int32 newpasswordlen,
                       const char *dbname,
                       db2int32 dbnamelen,
                       gss_cred_id_t *pGSSCredHandle,
                       void          **InitInfo,
                       char          **errormsg,
                       db2int32 *errormsglen );
```

Parameter der API 'db2secGenerateInitialCred'

userid
Eingabe. Die Benutzer-ID, deren Kennwort auf dem Datenbankserver zu prüfen ist.

useridlen
Eingabe. Die Länge (in Byte) des Werts des Parameters **userid**.

usernamespace
Eingabe. Der Namensbereich, aus dem die Benutzer-ID abgerufen wurde.

usernamespacelen
Eingabe. Die Länge (in Byte) des Werts des Parameters **usernamespace**.

usernamespacetype
Eingabe. Der Typ des Namensbereichs.

password
Eingabe. Das zu prüfende Kennwort.

passwordlen
Eingabe. Die Länge (in Byte) des Werts des Parameters **password**.

newpassword
Eingabe. Ein neues Kennwort, wenn das Kennwort geändert werden soll. Wenn keine Änderung angefordert wird, ist der Parameter **newpassword** auf den Wert NULL gesetzt. Wenn er nicht NULL ist, sollte die API das alte Kennwort prüfen, bevor sie das Kennwort auf den neuen Wert setzt. Die API muss die Anforderung zum Ändern des Kennworts nicht ausführen. Wenn sie es jedoch nicht tut, sollte sie sofort mit dem Rückgabewert DB2SEC_PLUGIN_CHANGEPASSWORD_NOTSUPPORTED zurückkehren, ohne das alte Kennwort zu prüfen.

newpasswordlen
Eingabe. Die Länge (in Byte) des Werts des Parameters **newpassword**.

dbname
Eingabe. Der Name der Datenbank, zu der die Verbindung hergestellt wird. Die API hat die Freiheit, diesen Parameter zu ignorieren. Alternativ kann die API auch den Wert DB2SEC_PLUGIN_CONNECTION_DISALLOWED zurückgeben, wenn sie der Richtlinie unterliegt, den Zugriff auf bestimmte Datenbanken für Benutzer einzuschränken, die ansonsten gültige Kennwörter besitzen.

dbnamelen
Eingabe. Die Länge (in Byte) des Werts des Parameters **dbname**.

pGSSCredHandle
Ausgabe. Ein Zeiger auf die GSS-API-Berechtigungsnachweiskennung.

InitInfo
Ausgabe. Ein Zeiger auf Daten, die DB2 Database for Linux, UNIX and Windows nicht bekannt sind. Das Plug-in kann diesen Speicher verwenden, um eine Liste von Ressourcen zu verwalten, die im Prozess der Generierung der Berechtigungsnachweiskennung zugeordnet werden. Der DB2-Datenbankmanager ruft die API 'db2secFreeInitInfo' am Ende des Authentifizierungsprozesses auf, sodass diese Ressourcen anschließend freigegeben werden. Wenn die API 'db2secGenerateInitialCred' keine solche Liste zu verwalten braucht, sollte der Wert NULL zurückgegeben werden.

errormsg
Ausgabe. Ein Zeiger auf die Adresse einer vom Plug-in zugeordneten ASCII-Fehlernachrichtenzeichenfolge, die in diesem Parameter zurückgegeben werden kann, wenn die API 'db2secGenerateInitialCred' nicht erfolgreich ausgeführt wird.

Anmerkung: Für diese API sollten keine Fehlernachrichten erstellt werden, wenn der Rückgabewert auf eine falsche Benutzer-ID oder ein falsches Kenn-

wort hinweist. Es sollte nur eine Fehlernachricht zurückgegeben werden, wenn ein interner Fehler in der API aufgetreten ist, der die ordnungsgemäße Ausführung der API verhindert hat.

errormsglen
Ausgabe. Ein Zeiger auf einen ganzzahligen Wert (Integer), der die Länge (in Byte) der Fehlernachrichtenzeichenfolge im Parameter **errormsg** angibt.

db2secGetAuthIDs (API) - Berechtigungs-IDs abrufen

Gibt eine SQL-Berechtigungs-ID (authid) für einen authentifizierten Benutzer zurück. Diese Anwendungsprogrammierschnittstelle (API) wird während der Herstellung von Datenbankverbindungen sowohl beim Verfahren der Benutzer-ID/Kennwort-Authentifizierung als auch beim Verfahren der GSS-API-Authentifizierung aufgerufen.

API- und Datenstruktursyntax

```
SQL_API_RC ( SQL_API_FN *db2secGetAuthIDs)
                      ( const char *userid,
                        db2int32 useridlen,
                        const char *usernamespace,
                        db2int32 usernamespacelen,
                        db2int32 usernamespacetype,
                        const char *dbname,
                        db2int32 dbnamelen,
                        void      **token,
                        char SystemAuthID[DB2SEC_MAX_AUTHID_LENGTH],
                        db2int32 *SystemAuthIDlen,
                        char InitialSessionAuthID[DB2SEC_MAX_AUTHID_LENGTH],
                        db2int32 *InitialSessionAuthIDlen,
                        char username[DB2SEC_MAX_USERID_LENGTH],
                        db2int32 *usernamelen,
                        db2int32 *initsessionidtype,
                        char      **errormsg,
                        db2int32 *errormsglen );
```

Parameter der API 'db2secGetAuthIDs'

userid
Eingabe. Der authentifizierte Benutzer. Dieser Wert wird in der Regel nicht für die GSS-API-Authentifizierung verwendet, es sei denn, es ist ein gesicherter Kontext definiert, der Operationen zum Wechseln von Benutzern ohne Authentifizierung zulässt. In solchen Fällen wird der für die Benutzerwechselanforderung bereitgestellte Benutzername in diesem Parameter übergeben.

useridlen
Eingabe. Die Länge (in Byte) des Werts des Parameters **userid**.

usernamespace
Eingabe. Der Namensbereich, aus dem die Benutzer-ID abgerufen wurde.

usernamespacelen
Eingabe. Die Länge (in Byte) des Werts des Parameters **usernamespace**.

usernamespacetype
Eingabe. Der Wert für den Typ des Namensbereichs. Gegenwärtig wird nur der Wert DB2SEC_NAMESPACE_SAM_COMPATIBLE für den Typ des Namensbereichs unterstützt (entspricht einem Benutzernamen des Formats domäne\meinname).

dbname
Eingabe. Der Name der Datenbank, zu der die Verbindung hergestellt wird. Die API kann diesen Parameter ignorieren oder unterschiedliche Berechti-

gungs-IDs zurückgeben, wenn derselbe Benutzer Verbindungen zu verschiedenen Datenbanken herstellt. Dieser Parameter kann den Wert NULL haben.

dbnamelen
Eingabe. Die Länge (in Byte) des Werts des Parameters **dbname**. Dieser Parameter wird auf den Wert `0` gesetzt, wenn der Parameter **dbname** den Wert NULL hat.

token
Eingabe oder Ausgabe. Daten, die das Plug-in an die API 'db2secGetGroupsForUser' übergeben kann. Für GSS-API ist dies eine Kontextkennung (`gss_ctx_id_t`). Normalerweise ist der Parameter 'token' ein reiner Eingabeparameter, dessen Wert aus der API 'db2secValidatePassword' empfangen wird. Er kann auch ein Ausgabeparameter sein, wenn die Authentifizierung auf dem Client erfolgt und dementsprechend die API 'db2secValidatePassword' nicht aufgerufen wird. In Umgebungen, in denen ein gesicherter Kontext definiert wird, der Operationen zum Benutzerwechsel ohne Authentifizierung zulässt, muss die API 'db2secGetAuthIDs' in der Lage sein, einen NULL-Wert für diesen Parameter 'token' zu empfangen und eine Systemberechtigungs-ID auf der Basis der zuvor beschriebenen Eingabeparameter **userid** und **useridlen** abzuleiten.

SystemAuthID
Ausgabe. Die Systemberechtigungs-ID, die der ID des authentifizierten Benutzers entspricht. Die Größe beträgt 255 Byte, jedoch verwendet der DB2-Datenbankmanager gegenwärtig nur bis zu 30 Byte (einschließlich).

SystemAuthIDlen
Ausgabe. Die Länge (in Byte) des Werts des Parameters **SystemAuthID**.

InitialSessionAuthID
Ausgabe. Die Berechtigungs-ID, die für diese Verbindungssitzung verwendet wird. Dieser Wert stimmt in der Regel mit dem Wert des Parameters **SystemAuthID** überein. Er kann sich in einigen Fällen jedoch unterscheiden, zum Beispiel, wenn eine Anweisung SET SESSION AUTHORIZATION ausgeführt wird. Die Größe beträgt 255 Byte, jedoch verwendet der DB2-Datenbankmanager gegenwärtig nur bis zu 30 Byte (einschließlich).

InitialSessionAuthIDlen
Ausgabe. Die Länge (in Byte) des Werts des Parameters **InitialSessionAuthID**.

username
Ausgabe. Ein Benutzername, der dem authentifizierten Benutzer und der Berechtigungs-ID (authid) entspricht. Dieser Wert wird nur für die Prüffunktion verwendet und im Feld für die Benutzer-ID im Prüfsatz für die CONNECT-Anweisung protokolliert. Wenn die API den Parameter **username** nicht angibt, kopiert der DB2-Datenbankmanager den Wert aus dem Feld 'userid'.

usernamelen
Ausgabe. Die Länge (in Byte) des Werts des Parameters **username**.

initsessionidtype
Ausgabe. Der Typ der Sitzungsberechtigung, der angibt, ob der Parameter **InitialSessionAuthid** eine Rolle oder eine Berechtigungs-ID (authid) angibt. Die API sollte einen der folgenden Werte (in der Datei `db2secPlugin.h` definiert) zurückgeben:

- `DB2SEC_ID_TYPE_AUTHID` (0)
- `DB2SEC_ID_TYPE_ROLE` (1)

errormsg
Ausgabe. Ein Zeiger auf die Adresse einer vom Plug-in zugeordneten ASCII-Fehlernachrichtenzeichenfolge, die in diesem Parameter zurückgegeben werden kann, wenn die API 'db2secGetAuthIDs' nicht erfolgreich ausgeführt wird.

errormsglen
Ausgabe. Ein Zeiger auf einen ganzzahligen Wert (Integer), der die Länge (in Byte) der Fehlernachrichtenzeichenfolge im Parameter **errormsg** angibt.

db2secGetDefaultLoginContext (API) - Standardanmeldekontext abrufen

Diese API ermittelt den Benutzer, der dem Standardanmeldekontext zugeordnet ist. Das heißt, sie ermittelt die DB2-Berechtigungs-ID (authid) des Benutzers, der einen DB2-Befehl aufruft, ohne explizit eine Benutzer-ID anzugeben (entweder durch eine implizite Authentifizierung für eine Datenbank oder durch eine lokale Authentifizierung). Diese API muss sowohl eine Berechtigungs-ID (authid) als auch eine Benutzer-ID zurückgeben.

API- und Datenstruktursyntax

```
SQL_API_RC ( SQL_API_FN *db2secGetDefaultLoginContext)
                      ( char authid[DB2SEC_MAX_AUTHID_LENGTH],
                        db2int32 *authidlen,
                        char userid[DB2SEC_MAX_USERID_LENGTH],
                        db2int32 *useridlen,
                        db2int32 useridtype,
                        char usernamespace[DB2SEC_MAX_USERNAMESPACE_LENGTH],
                        db2int32 *usernamespacelen,
                        db2int32 *usernamespacetype,
                        const char *dbname,
                        db2int32 dbnamelen,
                        void      **token,
                        char      **errormsg,
                        db2int32 *errormsglen );
```

Parameter der API 'db2secGetDefaultLoginContext'

authid
Ausgabe. Der Parameter, in dem die Berechtigungs-ID zurückgegeben werden sollte. Der zurückgegebene Wert muss den DB2-Namensregeln für Berechtigungs-IDs entsprechen. Ansonsten wird der Benutzer nicht zur Ausführung der angeforderten Aktion berechtigt.

authidlen
Ausgabe. Die Länge (in Byte) des Werts des Parameters **authid**.

userid
Ausgabe. Der Parameter, in dem die dem Standardanmeldekontext zugeordnete Benutzer-ID zurückgegeben werden sollte.

useridlen
Ausgabe. Die Länge (in Byte) des Werts des Parameters **userid**.

useridtype
Eingabe. Gibt an, ob die reale oder die effektive Benutzer-ID des Prozesses angegeben wird. Unter Windows ist nur die reale Benutzer-ID vorhanden. Unter UNIX und Linux können sich die reale Benutzer-ID und die effektive Benutzer-ID unterscheiden, wenn die Benutzer-ID `uid` für die Anwendung eine andere ID ist als die ID des Benutzers, der den Prozess ausführt. Folgende Werte für den Parameter **userid** (in der Datei `db2secPlugin.h` definiert) sind gültig:

DB2SEC_PLUGIN_REAL_USER_NAME
Gibt an, dass die reale Benutzer-ID angegeben wird.

DB2SEC_PLUGIN_EFFECTIVE_USER_NAME
Gibt an, dass die effektive (aktuelle) Benutzer-ID angegeben wird.

Anmerkung: Bei einigen Plug-in-Implementierungen wird möglicherweise nicht zwischen der realen und der effektiven Benutzer-ID unterschieden. Insbesondere kann ein Plug-in diesen Unterschied gefahrlos ignorieren, das nicht mit der UNIX- oder Linux-Identität des Benutzers arbeitet, um die DB2-Berechtigungs-ID einzurichten.

usernamespace
Ausgabe. Der Namensbereich der Benutzer-ID.

usernamespacelen
Ausgabe. Die Länge (in Byte) des Werts des Parameters **usernamespace**. Unter der Einschränkung, dass der Parameter **usernamespacetype** auf den Wert DB2SEC_NAMESPACE_SAM_COMPATIBLE (in der Datei db2secPlugin.h definiert) gesetzt sein muss, beträgt die gegenwärtig unterstützte Maximallänge 15 Byte.

usernamespacetype
Ausgabe. Der Wert für den Typ des Namensbereichs. Gegenwärtig wird nur der Wert DB2SEC_NAMESPACE_SAM_COMPATIBLE für den Typ des Namensbereichs unterstützt (entspricht einem Benutzernamen des Formats domäne\meinname).

dbname
Eingabe. Enthält den Namen der Datenbank, zu der die Verbindung hergestellt wird, wenn dieser Aufruf im Kontext einer Datenbankverbindung verwendet wird. Für lokale Berechtigungsaktionen oder Instanzverbindungen (ATTACH) wird dieser Parameter auf NULL gesetzt.

dbnamelen
Eingabe. Die Länge (in Byte) des Werts des Parameters **dbname**.

token
Ausgabe. Dies ist ein Zeiger auf vom Plug-in zugeordnete Daten, die vom Plug-in an nachfolgende Authentifizierungsaufrufe im Plug-in übergeben werden können oder potenziell an das Plug-in zum Abrufen von Gruppen übergeben werden können. Die Struktur dieser Daten wird durch den Autor des Plug-ins festgelegt.

errormsg
Ausgabe. Ein Zeiger auf die Adresse einer vom Plug-in zugeordneten ASCII-Fehlernachrichtenzeichenfolge, die in diesem Parameter zurückgegeben werden kann, wenn die API 'db2secGetDefaultLoginContext' nicht erfolgreich ausgeführt wird.

errormsglen
Ausgabe. Ein Zeiger auf einen ganzzahligen Wert (Integer), der die Länge (in Byte) der Fehlernachrichtenzeichenfolge im Parameter **errormsg** angibt.

db2secProcessServerPrincipalName (API) - Vom Server zurückgegebenen Service-Principal-Namen verarbeiten

Die API 'db2secProcessServerPrincipalName' verarbeitet den Namen des Service-Principals, der vom Server zurückgegeben wurde, und gibt den Namen des Principals im internen Format 'gss_name_t' zur Verwendung mit der API 'gss_init_sec_context' zurück.

Die API 'db2secProcessServerPrincipalName' verarbeitet auch den Namen des Service-Principals, der im Datenbankverzeichnis katalogisiert ist, wenn die Kerberos-Authentifizierung verwendet wird. Normalerweise verwendet diese Konvertierung die API 'gss_import_name'. Wenn der Kontext eingerichtet ist, wird das Objekt 'gss_name_t' durch einen Aufruf der API 'gss_release_name' freigegeben. Die API 'db2secProcessServerPrincipalName' gibt den Wert **DB2SEC_PLUGIN_OK** zurück, wenn der Parameter **gssName** auf einen gültigen GSS-Namen verweist. Ist der Name des Principals ungültig, wird der Fehlercode DB2SEC_PLUGIN_BAD_PRINCIPAL_NAME zurückgegeben.

API- und Datenstruktursyntax

```
SQL_API_RC ( SQL_API_FN *db2secProcessServerPrincipalName)
                       ( const char *name,
                         db2int32 namelen,
                         gss_name_t *gssName,
                         char       **errormsg,
                         db2int32 *errormsglen );
```

Parameter der API 'db2secProcessServerPrincipalName'

name
Eingabe. Der Textname des Service-Principals im Format GSS_C_NT_USER-_NAME, zum Beispiel service/host@REALM.

namelen
Eingabe. Die Länge (in Byte) des Werts des Parameters **name**.

gssName
Ausgabe. Ein Zeiger auf den Ausgabenamen des Service-Principals im internen GSS-API-Format.

errormsg
Ausgabe. Ein Zeiger auf die Adresse einer vom Plug-in zugeordneten ASCII-Fehlernachrichtenzeichenfolge, die in diesem Parameter zurückgegeben werden kann, wenn die API 'db2secProcessServerPrincipalName' nicht erfolgreich ausgeführt wird.

errormsglen
Ausgabe. Ein Zeiger auf einen ganzzahligen Wert (Integer), der die Länge (in Byte) der Fehlernachrichtenzeichenfolge im Parameter **errormsg** angibt.

db2secRemapUserid (API) - Zuordnung von Benutzer-ID und Kennwort ändern

Diese Anwendungsprogrammierschnittstelle (API) wird vom DB2-Datenbankmanager auf der Clientseite aufgerufen, um eine gegebene Benutzer-ID mit zugehörigem Kennwort (und möglicherweise ein neues Kennwort und einen neuen Benutzernamensbereich) anderen als den beim Verbindungsaufbau angegebenen Werten zuzuordnen.

Der DB2-Datenbankmanager ruft diese API nur auf, wenn beim Verbindungsaufbau eine Benutzer-ID und ein Kennwort angegeben werden. Dies verhindert, dass ein Plug-in eine Benutzer-ID selbsttätig einem anderen Paar aus Benutzer-ID und Kennwort zuordnet. Diese API ist optional und wird nicht aufgerufen, wenn dies nicht vorgesehen ist oder sie vom Sicherheits-Plug-in nicht implementiert wird.

API- und Datenstruktursyntax

```
SQL_API_RC ( SQL_API_FN *db2secRemapUserid)
                      ( char userid[DB2SEC_MAX_USERID_LENGTH],
                        db2int32 *useridlen,
                        char usernamespace[DB2SEC_MAX_USERNAMESPACE_LENGTH],
                        db2int32 *usernamespacelen,
                        db2int32 *usernamespacetype,
                        char password[DB2SEC_MAX_PASSWORD_LENGTH],
                        db2int32 *passwordlen,
                        char newpasswd[DB2SEC_MAX_PASSWORD_LENGTH],
                        db2int32 *newpasswdlen,
                        const char *dbname,
                        db2int32 dbnamelen,
                        char      **errormsg,
                        db2int32 *errormsglen);
```

Parameter der API 'db2secRemapUserid'

userid
Eingabe oder Ausgabe. Die Benutzer-ID, deren Zuordnung zu ändern ist. Wenn ein Eingabewert für die Benutzer-ID angegeben ist, muss die API einen Ausgabewert für die Benutzer-ID bereitstellen, der mit dem Eingabewert für die Benutzer-ID identisch oder ein anderer Wert sein kann. Wenn kein Eingabewert für die Benutzer-ID angegeben ist, sollte die API keinen Ausgabewert für die Benutzer-ID zurückgeben.

useridlen
Eingabe oder Ausgabe. Die Länge (in Byte) des Werts des Parameters **userid**.

usernamespace
Eingabe oder Ausgabe. Der Namensbereich der Benutzer-ID. Dieser Wert kann optional anders zugeordnet werden. Wenn kein Eingabeparameterwert angegeben wird, jedoch ein Ausgabewert zurückgegeben wird, wird der Parameter **usernamespace** vom DB2-Datenbankmanager nur für den Authentifizierungstyp CLIENT verwendet und für andere Authentifizierungstypen ignoriert.

usernamespacelen
Eingabe oder Ausgabe. Die Länge (in Byte) des Werts des Parameters **usernamespace**. Unter der Einschränkung, dass der Parameter **usernamespacetype** auf den Wert `DB2SEC_NAMESPACE_SAM_COMPATIBLE` (in der Datei `db2secPlugin.h` definiert) gesetzt sein muss, beträgt die gegenwärtig unterstützte Maximallänge 15 Byte.

usernamespacetype
Eingabe oder Ausgabe. Der alte und der neue Wert für den Typ des Namensbereichs. Gegenwärtig wird nur der Wert `DB2SEC_NAMESPACE_SAM_COMPATIBLE` für den Typ des Namensbereichs unterstützt (entspricht einem Benutzernamen des Formats `domäne\meinname`).

password
Eingabe oder Ausgabe. Als Eingabe ist dieser Wert das Kennwort, das anders zuzuordnen ist. Als Ausgabe ist dieser Wert das anders zugeordnete Kennwort. Wenn ein Eingabewert für diesen Parameter angegeben wird, muss die API einen Ausgabewert zurückgeben können, der sich vom Eingabewert unterscheidet. Wenn kein Eingabeparameter angegeben wird, darf die API keinen Ausgabewert für das Kennwort zurückgeben.

passwordlen
Eingabe oder Ausgabe. Die Länge (in Byte) des Werts des Parameters **password**.

newpasswd
Eingabe oder Ausgabe. Als Eingabe ist dieser Wert das neue Kennwort, das festzulegen ist. Als Ausgabe ist dieser Wert das bestätigte neue Kennwort.

Anmerkung: Dies ist das neue Kennwort, das vom DB2-Datenbankmanager in den Parameter **newpassword** der API 'db2secValidatePassword' auf dem Client oder dem Server (je nach dem Wert des Konfigurationsparameters **authentication** des Datenbankmanagers) eingefügt wird. Wenn ein neues Kennwort als Eingabe übergeben wurde, muss die API einen Ausgabewert zurückgeben können, bei dem es sich um ein anderes neues Kennwort handeln kann. Wenn kein neues Kennwort als Eingabe übergeben wird, sollte die API kein neues Kennwort als Ausgabe zurückgeben.

newpasswdlen
Eingabe oder Ausgabe. Die Länge (in Byte) des Werts des Parameters **newpasswd**.

dbname
Eingabe. Der Name der Datenbank, zu der der Client die Verbindung herstellt.

dbnamelen
Eingabe. Die Länge (in Byte) des Werts des Parameters **dbname**.

errormsg
Ausgabe. Ein Zeiger auf die Adresse einer vom Plug-in zugeordneten ASCII-Fehlernachrichtenzeichenfolge, die in diesem Parameter zurückgegeben werden kann, wenn die API 'db2secRemapUserid' nicht erfolgreich ausgeführt wird.

errormsglen
Ausgabe. Ein Zeiger auf einen ganzzahligen Wert (Integer), der die Länge (in Byte) der Fehlernachrichtenzeichenfolge im Parameter **errormsg** angibt.

db2secServerAuthPluginInit - Plug-in zur Serverauthentifizierung initialisieren

Die API 'db2secServerAuthPluginInit' stellt die Initialisierungs-API für das Plug-in zur Serverauthentifizierung dar, die vom DB2-Datenbankmanager unmittelbar nach dem Laden des Plug-ins aufgerufen wird.

Im Fall der GSS-API ist das Plug-in dafür zuständig, den Namen des Principals im Parameter **serverPrincipalName** in der Struktur 'gssapi_server_auth_functions' bei der Initialisierung auszufüllen und die Berechtigungsnachweiskennung des Servers im Parameter **serverCredHandle** in der Struktur 'gssapi_server_auth_functions' anzugeben. Die Freigabe des Speichers, der zur Aufnahme des Namens des Principals und der Berechtigungsnachweiskennung zugeordnet wird, muss von der API 'db2secServerAuthPluginTerm' durch Aufrufen der APIs 'gss_release_name' und 'gss_release_cred' ausgeführt werden.

API- und Datenstruktursyntax

```
SQL_API_RC SQL_API_FN db2secServerAuthPluginInit
                            ( db2int32 version,
                              void *server_fns,
                              db2secGetConDetails *getConDetails_fn,
                              db2secLogMessage *logMessage_fn,
                              char               **errormsg,
                              db2int32 *errormsglen );
```

Parameter der API 'db2secServerAuthPluginInit'

version
Eingabe. Die höchste Versionsnummer der API, die vom DB2-Datenbankmanager gegenwärtig unterstützt wird. Der Wert DB2SEC_API_VERSION (in db2secPlugin.h) enthält die aktuellste Versionsnummer der API, die vom DB2-Datenbankmanager gegenwärtig unterstützt wird.

server_fns
Ausgabe. Ein Zeiger auf Speicher, der vom DB2-Datenbankmanager für eine Struktur 'db2secGssapiServerAuthFunctions_<versionsnummer>' (auch als 'gssapi_server_auth_functions_<versionsnummer>' bekannt), wenn die GSS-API-Authentifizierung verwendet wird, oder für eine Struktur 'db2secUseridPasswordServerAuthFunctions_<versionsnummer>' (auch als 'userid_password_server_auth_functions_<versionsnummer>' bekannt), wenn die Benutzer-ID/Kennwort-Authentifizierung verwendet wird, bereitgestellt wird. Die Struktur 'db2secGssapiServerAuthFunctions_<versionsnummer>' und die Struktur 'db2secUseridPasswordServerAuthFunctions_<versionsnummer>' enthalten wiederum Zeiger auf die APIs, die für die GSS-API-Authentifizierung bzw. die Benutzer-ID/Kennwort-Authentifizierung implementiert sind.

Der Parameter **server_fns** wird als Zeiger auf die Struktur 'gssapi_server_auth_functions_<versionsnummer>' umgesetzt, die der Version entspricht, die vom Plug-in implementiert wird. Der erste Parameter der Struktur 'gssapi_server_auth_functions_<versionsnummer>' bzw. der Struktur 'userid_password_server_auth_functions_<versionsnummer>' teilt dem DB2-Datenbankmanager die Version der APIs mit, die vom Plug-in implementiert werden.

Anmerkung: Die Umsetzung (CAST) erfolgt nur, wenn die DB2-Version höher oder gleich der Version der APIs ist, die vom Plug-in implementiert werden.

In der Struktur 'gssapi_server_auth_functions_<versionsnummer>' bzw. 'userid_password_server_auth_functions_<versionsnummer>' sollte der Parameter **plugintype** auf einen der Werte DB2SEC_PLUGIN_TYPE_USERID_PASSWORD, DB2SEC_PLUGIN_TYPE_GSSAPI oder DB2SEC_PLUGIN_TYPE_KERBEROS gesetzt werden. Andere Werte können in zukünftigen Versionen der API definiert werden.

getConDetails_fn
Eingabe. Ein Zeiger auf die API 'db2secGetConDetails', die von DB2 implementiert wird. Die API 'db2secServerAuthPluginInit' kann die API 'db2secGetConDetails' in einer beliebigen der anderen Authentifizierungs-APIs aufrufen, um Details zur Datenbankverbindung abzurufen. Zu diesen Details gehören Informationen zu dem Kommunikationsmechanismus, der der Verbindung zugeordnet ist (z.B. die IP-Adresse im Fall von TCP/IP), die der Autor des Plug-ins möglicherweise benötigt, wenn er Entscheidungen in Bezug auf die Authentifizierung trifft. Zum Beispiel könnte das Plug-in eine Verbindung für einen bestimmten Benutzer nicht zulassen, wenn dieser Benutzer die Verbindung nicht über eine bestimmte IP-Adresse herstellt. Die Verwendung der API 'db2secGetConDetails' ist optional.

Wenn die API 'db2secGetConDetails' in einer Situation aufgerufen wird, in der keine Datenbankverbindung besteht, gibt sie den Wert DB2SEC_PLUGIN_NO_CON_DETAILS zurück. Andernfalls gibt sie bei Erfolg den Wert 0 zurück.

Die API 'db2secGetConDetails' empfängt zwei Eingabeparameter: den Parameter **pConDetails**, bei dem es sich um einen Zeiger auf die Struktur 'db2sec_con_details_<versionsnummer>' handelt, und den Parameter **conDe-**

tailsVersion, bei dem es sich um eine Versionsnummer handelt, die angibt, welche Struktur 'db2sec_con_details' zu verwenden ist. Mögliche Werte sind DB2SEC_CON_DETAILS_VERSION_1, wenn 'db2sec_con_details1' verwendet wird, oder DB2SEC_CON_DETAILS_VERSION_2, wenn 'db2sec_con_details2' verwendet wird. Die zur Verwendung empfohlene Versionsnummer ist DB2SEC_CON_DETAILS_VERSION_2.

Bei einer erfolgreichen Rückgabe enthält die Struktur 'db2sec_con_details' (entweder 'db2sec_con_details1' oder 'db2sec_con_details2') die folgenden Informationen:

- Das für die Verbindung zum Server verwendete Protokoll. Die Liste der Protokolldefinitionen (SQL_PROTOCOL_*) befindet sich in der Datei sqlenv.h (im include-Verzeichnis). Diese Information wird in den Parameter **clientProtocol** eingefügt.
- Die TCP/IP-Adresse der eingehenden Verbindung zum Server, wenn der Parameter **clientProtocol** den Wert SQL_PROTOCOL_TCPIP oder SQL_PROTOCOL_TCPIP4 hat. Diese Information wird in den Parameter **clientIPAddress** eingefügt.
- Der Name der Datenbank, zu der der Client eine Verbindung aufbauen will. Dies wird für Instanzverbindungen (mit ATTACH) nicht angegeben. Diese Information wird in die Parameter **dbname** und **dbnameLen** eingefügt.
- Eine Bitzuordnung mit Verbindungsinformationen, die dieselben Details wie die enthält, die im Parameter **connection_details** der API 'db2secValidatePassword' dokumentiert werden. Diese Information wird in den Parameter **connect_info_bitmap** eingefügt.
- Die TCP/IP-Adresse der eingehenden Verbindung zum Server, wenn der Parameter **clientProtocol** den Wert SQL_PROTOCOL_TCPIP6 hat. Diese Information wird in den Parameter **clientIP6Address** eingefügt und ist nur verfügbar, wenn für den Aufruf der API 'db2secGetConDetails' der Wert DB2SEC_CON_DETAILS_VERSION_2 verwendet wird.

logMessage_fn

Eingabe. Ein Zeiger auf die API 'db2secLogMessage', die vom DB2-Datenbankmanager implementiert wird. Die API 'db2secClientAuthPluginInit' kann die API 'db2secLogMessage' aufrufen, um Nachrichten in den **db2diag**-Protokolldateien zu Debug- oder Informationszwecken zu protokollieren. Der erste Parameter (**level**) der API 'db2secLogMessage' gibt den Typ der zu diagnostizierenden Fehler an, die in den **db2diag**-Protokolldateien aufgezeichnet werden. Die beiden letzten Parameter geben die Nachrichtenzeichenfolge und ihre Länge an. Die folgenden Werte sind für den ersten Parameter der API 'db2secLogMessage' (in der Datei db2secPlugin.h definiert) gültig:

- DB2SEC_LOG_NONE (0): Keine Protokollierung
- DB2SEC_LOG_CRITICAL (1): Schwer wiegender Fehler
- DB2SEC_LOG_ERROR (2): Fehler
- DB2SEC_LOG_WARNING (3): Warnung
- DB2SEC_LOG_INFO (4): Informativ

Der Nachrichtentext wird in den **db2diag**-Protokolldateien nur gezeigt, wenn der Wert des Parameters **level** der API 'db2secLogMessage' kleiner oder gleich dem Wert des Konfigurationsparameters **diaglevel** des Datenbankmanagers ist.

Wenn Sie zum Beispiel den Wert DB2SEC_LOG_INFO verwenden, wird der Nachrichtentext in den **db2diag**-Protokolldateien nur gezeigt, wenn der Konfigurationsparameter **diaglevel** des Datenbankmanagers auf den Wert 4 gesetzt ist.

errormsg
Ausgabe. Ein Zeiger auf die Adresse einer vom Plug-in zugeordneten ASCII-Fehlernachrichtenzeichenfolge, die in diesem Parameter zurückgegeben werden kann, wenn die API 'db2secServerAuthPluginInit' nicht erfolgreich ausgeführt wird.

errormsglen
Ausgabe. Ein Zeiger auf einen ganzzahligen Wert (Integer), der die Länge (in Byte) der Fehlernachrichtenzeichenfolge im Parameter **errormsg** angibt.

db2secServerAuthPluginTerm (API) - Ressourcen für Plug-in zur Serverauthentifizierung bereinigen

Die API 'db2secServerAuthPluginTerm' gibt Ressourcen frei, die vom Plug-in zur Serverauthentifizierung verwendet wurden.

Diese Anwendungsprogrammierschnittstelle (API) wird vom DB2-Datenbankmanager kurz vor dem Entladen des Plug-ins zur Serverauthentifizierung aufgerufen. Sie sollte in einer Weise implementiert werden, in der sie eine ordnungsgemäße Bereinigung aller Ressourcen ausführt, die von der Bibliothek des Plug-ins genutzt werden. Sie sollte zum Beispiel den vom Plug-in zugeordneten Speicher freigeben, Dateien schließen, die noch geöffnet sind, und Netzwertverbindungen schließen. Es liegt in der Zuständigkeit des Plug-ins, diese Ressourcen zu verfolgen, um sie freigeben zu können. Diese API wird auf keinem Windows-Betriebssystem aufgerufen.

API- und Datenstruktursyntax

```
SQL_API_RC ( SQL_API_FN *db2secServerAuthPluginTerm)
                        ( char     **errormsg,
                          db2int32 *errormsglen );
```

Parameter der API 'db2secServerAuthPluginTerm'

errormsg
Ausgabe. Ein Zeiger auf die Adresse einer vom Plug-in zugeordneten ASCII-Fehlernachrichtenzeichenfolge, die in diesem Parameter zurückgegeben werden kann, wenn die API 'db2secServerAuthPluginTerm' nicht erfolgreich ausgeführt wird.

errormsglen
Ausgabe. Ein Zeiger auf einen ganzzahligen Wert (Integer), der die Länge (in Byte) der Fehlernachrichtenzeichenfolge im Parameter **errormsg** angibt.

db2secValidatePassword (API) - Kennwort prüfen

Stellt eine Methode zur Ausführung einer Authentifizierung durch Benutzer-ID und Kennwort während einer Operation zur Herstellung einer Datenbankverbindung bereit.

Anmerkung: Wenn die Anwendungsprogrammierschnittstelle (API) auf der Clientseite ausgeführt wird, wird der API-Code mit den Berechtigungen des Benutzers ausgeführt, der die Anweisung CONNECT ausführt. Diese API wird auf der Clientseite nur aufgerufen, wenn der Konfigurationsparameter **authentication** auf den Wert `CLIENT` gesetzt ist.

Wenn die API auf der Serverseite ausgeführt wird, wird der API-Code mit den Berechtigungen des Instanzeigners ausgeführt.

Der Plug-in-Autor sollte die zuvor genannten Szenarios beachten, wenn die Authentifizierung besondere Berechtigungen (z. B. Systemzugriff auf Rootebene unter UNIX) erfordert.

Diese API muss den Wert DB2SEC_PLUGIN_OK (Erfolg) zurückgeben, wenn das Kennwort gültig ist. Wenn das Kennwort ungültig ist, muss sie einen Fehlercode wie zum Beispiel DB2SEC_PLUGIN_BADPWD zurückgeben.

API- und Datenstruktursyntax

```
SQL_API_RC ( SQL_API_FN *db2secValidatePassword)
                       ( const char *userid,
                         db2int32 useridlen,
                         const char *usernamespace,
                         db2int32 usernamespacelen,
                         db2int32 usernamespacetype,
                         const char *password,
                         db2int32 passwordlen,
                         const char *newpasswd,
                         db2int32 newpasswdlen,
                         const char *dbname,
                         db2int32 dbnamelen,
                         db2Uint32 connection_details,
                         void      **token,
                         char      **errormsg,
                         db2int32 *errormsglen );
```

Parameter der API 'db2secValidatePassword'

userid
Eingabe. Die Benutzer-ID, deren Kennwort zu prüfen ist.

useridlen
Eingabe. Die Länge (in Byte) des Werts des Parameters **userid**.

usernamespace
Eingabe. Der Namensbereich, aus dem die Benutzer-ID abgerufen wurde.

usernamespacelen
Eingabe. Die Länge (in Byte) des Werts des Parameters **usernamespace**.

usernamespacetype
Eingabe. Der Typ des Namensbereichs. Folgende Werte für den Parameter **usernamespacetype** (in der Datei db2secPlugin.h definiert) sind gültig:

- DB2SEC_NAMESPACE_SAM_COMPATIBLE: Entspricht einer Benutzernamensdarstellung des Formats 'domäne\meinname'.
- DB2SEC_NAMESPACE_USER_PRINCIPAL: Entspricht einer Benutzernamensdarstellung des Formats 'meinname@domäne.ibm.com'.

Gegenwärtig unterstützt das DB2-Datenbanksystem nur den Wert DB2SEC_NAMESPACE_SAM_COMPATIBLE. Wenn die Benutzer-ID nicht verfügbar ist, wird der Parameter **usernamespacetype** auf den Wert DB2SEC_USER_NAMESPACE_UNDEFINED (in der Datei db2secPlugin.h definiert) gesetzt.

password
Eingabe. Das zu prüfende Kennwort.

passwordlen
Eingabe. Die Länge (in Byte) des Werts des Parameters **password**.

newpasswd
Eingabe. Ein neues Kennwort, wenn das Kennwort geändert werden soll.

Wenn keine Änderung angefordert wird, ist dieser Parameter auf den Wert NULL gesetzt. Wenn dieser Parameter nicht NULL ist, sollte die API das alte Kennwort prüfen, bevor sie es in das neue Kennwort ändert. Die API muss die Anforderung zum Ändern des Kennworts nicht ausführen. Wenn sie es jedoch nicht tut, sollte sie sofort mit dem Rückgabewert DB2SEC_PLUGIN_CHANGEPASSWORD_NOTSUPPORTED zurückkehren, ohne das alte Kennwort zu prüfen.

newpasswdlen
Eingabe. Die Länge (in Byte) des Werts des Parameters **newpasswd**.

dbname
Eingabe. Der Name der Datenbank, zu der die Verbindung hergestellt wird. Die API hat die Freiheit, den Parameter **dbname** zu ignorieren. Alternativ kann die API auch den Wert DB2SEC_PLUGIN_CONNECTIONREFUSED zurückgeben, wenn sie der Richtlinie unterliegt, den Zugriff auf bestimmte Datenbanken für Benutzer einzuschränken, die ansonsten gültige Kennwörter besitzen. Dieser Parameter kann den Wert NULL haben.

dbnamelen
Eingabe. Die Länge (in Byte) des Werts des Parameters **dbname**. Dieser Parameter wird auf den Wert 0 gesetzt, wenn der Parameter **dbname** den Wert NULL hat.

connection_details
Eingabe. Ein 32-Bit-Parameter, in dem 3 Bit gegenwärtig zum Speichern der folgenden Informationen verwendet werden:

- Das äußerste rechte Bit gibt an, ob die Quelle der Benutzer-ID der Standardwert aus der API 'db2secGetDefaultLoginContext' ist oder während des Verbindungsaufbaus explizit angegeben wurde.
- Das zweite Bit von rechts gibt an, ob die Verbindung lokal (durch Interprozesskommunikation (IPC) oder durch eine Verbindung von einem der Knoten in der Datei db2nodes.cfg in der Umgebung mit partitionierten Datenbanken) oder fern (über ein Netz oder durch Loopback) ist. Anhand dieser Information kann die API entscheiden, ob Clients auf demselben System die Verbindung zu dem DB2Server ohne Kennwort herstellen können. Durch das standardmäßig verwendete, betriebssystembasierte Benutzer-ID/Kennwort-Plug-in werden lokale Verbindungen ohne Kennwort von Clients auf demselben System zugelassen (vorausgesetzt, der Benutzer hat Verbindungsberechtigungen).
- Das dritte Bit von rechts gibt an, ob der DB2-Datenbankmanager die API von der Serverseite oder der Clientseite aus aufruft.

Die Bit-Werte sind in der Datei db2secPlugin.h definiert:

- DB2SEC_USERID_FROM_OS (0x00000001): Gibt an, dass die Benutzer-ID aus dem Betriebssystem abgerufen und nicht explizit in der CONNECT-Anweisung angegeben wurde.
- DB2SEC_CONNECTION_ISLOCAL (0x00000002): Gibt eine lokale Verbindung an.
- DB2SEC_VALIDATING_ON_SERVER_SIDE (0x0000004): Gibt an, ob der DB2-Datenbankmanager von der Serverseite oder der Clientseite zur Kennwortprüfung aufruft. Wenn dieser Bit-Wert gesetzt ist, erfolgt der Aufruf des DB2-Datenbankmanagers von der Serverseite aus. Anderenfalls erfolgt er von der Clientseite aus.

Das Standardverhalten des DB2-Datenbanksystems bei einer impliziten Authentifizierung ist, die Verbindung ohne Kennwortprüfung zuzulassen. Plug-in-

Entwickler haben jedoch die Möglichkeit, die implizite Authentifizierung unter Rückgabe des Fehlers DB2SEC_PLUGIN_BADPASSWORD zu verweigern.

token
Eingabe. Ein Zeiger auf Daten, die während der aktuellen Verbindung als Parameter an nachfolgende API-Aufrufe übergeben werden können. Zu möglichen APIs, die aufgerufen werden können, gehören die API 'db2secGetAuthIDs' und die API 'db2secGetGroupsForUser'.

errormsg
Ausgabe. Ein Zeiger auf die Adresse einer vom Plug-in zugeordneten ASCII-Fehlernachrichtenzeichenfolge, die in diesem Parameter zurückgegeben werden kann, wenn die API 'db2secValidatePassword' nicht erfolgreich ausgeführt wird.

errormsglen
Ausgabe. Ein Zeiger auf einen ganzzahligen Wert (Integer), der die Länge (in Byte) der Fehlernachrichtenzeichenfolge im Parameter **errormsg** angibt.

Erforderliche APIs und Definitionen für Plug-ins zur GSS-API-Authentifizierung

Nachfolgend finden Sie eine vollständige Liste der GSS-APIs, die für die DB2-Sicherheits-Plug-in-Schnittstelle erforderlich sind.

Die unterstützten APIs entsprechend den folgenden Spezifikationen: *Generic Security Service Application Program Interface, Version 2* (IETF RFC2743) und *Generic Security Service API Version 2: C-Bindings* (IETF RFC2744). Vor der Implementierung eines GSS-API-basierten Plug-ins sollten Sie sich mit diesen Spezifikationen eingehend vertraut machen.

Tabelle 41. Erforderliche APIs und Definitionen für Plug-ins zur GSS-API-Authentifizierung

API-Typ	API-Name	Beschreibung
Clientseitige APIs	gss_init_sec_context	Initialisiert einen Sicherheitskontext mit einer Peeranwendung.
Serverseitige APIs	gss_accept_sec_context	Akzeptiert einen von einer Peeranwendung initialisierten Sicherheitskontext.
Serverseitige APIs	gss_display_name	Konvertiert einen Namen aus dem internen Format in Text.
Allgemeine APIs	gss_delete_sec_context	Löscht einen eingerichteten Sicherheitskontext.
Allgemeine APIs	gss_display_status	Ruft die einem GSS-API-Statuscode zugeordnete Textfehlernachricht ab.
Allgemeine APIs	gss_release_buffer	Löscht einen Puffer.
Allgemeine APIs	gss_release_cred	Gibt lokale Datenstrukturen frei, die einem GSS-API-Berechtigungsnachweis zugeordnet sind.
Allgemeine APIs	gss_release_name	Löscht den Namen im internen Format.
Erforderliche Definitionen	GSS_C_DELEG_FLAG	Fordert eine Delegierung an.
Erforderliche Definitionen	GSS_C_EMPTY_BUFFER	Signalisiert, dass gss_buffer_desc keine Daten enthält.
Erforderliche Definitionen	GSS_C_GSS_CODE	Gibt einen übergeordneten GSS-Statuscode an.

Tabelle 41. Erforderliche APIs und Definitionen für Plug-ins zur GSS-API-Authentifizierung (Forts.)

API-Typ	API-Name	Beschreibung
Erforderliche Definitionen	GSS_C_INDEFINITE	Gibt an, dass der Mechanismus keinen Gültigkeitsverfall des Kontexts unterstützt.
Erforderliche Definitionen	GSS_C_MECH_CODE	Gibt einen untergeordneten GSS-Statuscode an.
Erforderliche Definitionen	GSS_C_MUTUAL_FLAG	Gegenseitige Authentifizierung angefordert.
Erforderliche Definitionen	GSS_C_NO_BUFFER	Bedeutet, dass die Variable gss_buffer_t nicht auf eine gültige Struktur gss_buffer_desc zeigt.
Erforderliche Definitionen	GSS_C_NO_CHANNEL_BINDINGS	Keine Kommunikationskanalbindungen.
Erforderliche Definitionen	GSS_C_NO_CONTEXT	Bedeutet, dass die Variable gss_ctx_id_t nicht auf einen gültigen Kontext zeigt.
Erforderliche Definitionen	GSS_C_NO_CREDENTIAL	Bedeutet, dass die Variable gss_cred_id_t nicht auf eine gültige Berechtigungsnachweiskennung zeigt.
Erforderliche Definitionen	GSS_C_NO_NAME	Bedeutet, dass die Variable gss_name_t nicht auf einen gültigen internen Namen zeigt.
Erforderliche Definitionen	GSS_C_NO_OID	Standardauthentifizierungsverfahren verwenden.
Erforderliche Definitionen	GSS_C_NULL_OID_SET	Standardverfahren verwenden.
Erforderliche Definitionen	GSS_S_COMPLETE	API erfolgreich abgeschlossen.
Erforderliche Definitionen	GSS_S_CONTINUE_NEEDED	Die Verarbeitung ist nicht abgeschlossen, und die API muss erneut mit dem Antworttoken aus dem Peer aufgerufen werden.

Einschränkungen für Plug-ins zur GSS-API-Authentifizierung

In der nachfolgenden Liste werden Einschränkungen für Plug-ins zur GSS-API-Authentifizierung aufgeführt.

- Es wird immer der Standardsicherheitsmechanismus angenommen. Daher sind keine besonderen OID-Aspekte zu beachten.
- Die einzigen GSS-Services, die in gss_init_sec_context() angefordert werden, sind die Services für gegenseitige Authentifizierung und Delegierung. Der DB2-Datenbankmanager fordert immer ein Ticket für die Delegierung an, verwendet dieses Ticket jedoch nicht zur Generierung eines neuen Tickets.
- Nur die Standardkontextzeit wird angefordert.
- Es werden keine Kontexttoken aus gss_delete_sec_context() vom Client an den Server und umgekehrt gesendet.
- Anonymität wird nicht unterstützt.
- Eine Kanalbindung (Channel Binding) wird nicht unterstützt.
- Wenn die Gültigkeit der Anfangsberechtigungsnachweise abläuft, werden sie vom DB2-Datenbankmanager nicht automatisch erneuert.
- Die GSS-API-Spezifikation legt fest, dass auch wenn gss_init_sec_context() oder gss_accept_sec_context() fehlschlagen, beide Funktionen ein Token an den Peer zurückgeben müssen. Aufgrund der DRDA-Begrenzungen sendet der DB2-Datenbankmanager nur ein Token, wenn gss_init_sec_context() fehlschlägt, und generiert ein Token beim ersten Aufruf.

Kapitel 10. Exitbibliotheken für Kommunikationspuffer

Der Datenbankmanager von DB2 for Linux, UNIX and Windows bietet Kunden und Anbietern die Möglichkeit der Überprüfung von Kommunikationspuffern. Externe, vertrauenswürdige gemeinsam genutzte Bibliotheken werden verwendet, um auf die Kommunikationspuffer, die für den Datenfluss zwischen Clients und Datenbankservern eingesetzt werden, zuzugreifen, bevor diese gesendet und empfangen werden. Diese externen Bibliotheken werden als Exitbibliotheken für Kommunikationspuffer bezeichnet.

Mit Exitbibliotheken für Kommunikationspuffer können Kommunikationspuffer überprüft werden, um Lösungen wie Prüffunktionen oder andere Sicherheitslösungen basierend auf dem Pufferinhalt bereitzustellen. DB2 for Linux, UNIX and Windows bietet Zugriff auf alle von Clients empfangenen Puffer sowie auf alle Puffer, die an Clients gesendet werden sollen. Puffer werden vor der Verschlüsselung entweder mit DATA_ENCRYPT-Authentifizierung oder SSL bereitgestellt. DB2 for Linux, UNIX and Windows verwendet für die Kommunikation zwischen Clients und Server das DRDA-Protokoll. Die an die Exitbibliothek übergebenen Kommunikationspuffer werden gemäß dem DRDA-Protokoll formatiert. Die Exitbibliothek für Kommunikationspuffer muss das für die Kommunikation verwendete DRDA-Protokoll verstehen können.

DB2 for Linux, UNIX and Windows stellt die Puffer unabhängig vom Kommunikationsprotokoll bereit. Exitbibliotheken für Kommunikationspuffer sind mit TCP/IP (IPv4 und IPv6), SSL, IPC (Inter-Process Communication) und benannten Pipes kompatibel.

Zusätzlich zu den Puffern stellt DB2 for Linux, UNIX and Windows Identitätsinformationen bereit, einschließlich des Benutzernamens und der für die Verbindung zur Datenbank erstellte Sitzungsberechtigungs-ID. Diese Informationen sind in Fällen nützlich, in denen GSS-API-Plug-ins wie Kerberos verwendet werden. In diesem Szenario gibt es keinen Standardbenutzernamen; stattdessen werden generische Tickets verwendet, aus denen der Datenbankmanager den Benutzernamen ableitet. Diese Information ist nicht verfügbar, wenn nur der Kommunikationspuffer in Betracht gezogen wird.

Der Datenbankmanager stellt sicher, dass nur vertrauenswürdige Bibliotheken geladen werden. Die Bibliotheken müssen in einer bestimmten Position installiert werden, die nur vom Instanzeigner geändert werden kann. Darüber hinaus kann nur ein Benutzer mit der Berechtigung SYSADM die Bibliothek aktivieren. Die Berechtigungsstufe entspricht der Berechtigungsstufe, die zur Aktivierung der Verschlüsselung erforderlich ist (DATA_ENCRYPT oder SSL).

Die Exitbibliothek für Kommunikationspuffer kann eine Verbindung beenden, wenn ein Puffer Daten enthält, die von der Bibliothek als schädlich betrachtet werden. Dazu gehören Daten, die an den Server gesendet werden, sowie Daten, die an den Client zurückgegeben werden. Die Exitbibliothek für Kommunikationspuffer könnte beispielsweise feststellen, dass die aus einer SELECT-Anweisung zurückgegebenen Daten für den Client nicht geeignet sind. Ein Rückkehrcode aus der Bibliothek weist den Datenbankmanager darauf hin, dass die Verbindung beendet werden muss. Der Datenbankmanager stoppt diesen Vorgang sowie alle weiteren an den Client gerichteten Kommunikationspuffer und beendet die Verbindung.

Anmerkung: Die beschriebenen Exitbibliotheken für Kommunikationspuffer werden normalerweise von anderen Anbietern bereitgestellt. DB2 for Linux, UNIX and Windows enthält jedoch Musterbibliotheken, die Sie im Verzeichnis `sqllib/samples/security/commexit` finden. Diese Musterbibliotheken können Ihnen als Vorlage für die Entwicklung eigener Bibliotheken dienen.

Exitbibliothek für Kommunikationspuffer - Implementierung

Beachten Sie bei der Implementierung einer Exitbibliothek für Kommunikationspuffer die folgenden Hinweise.

In typischen Szenarios werden Exitbibliotheken für Kommunikationspuffer von Anbietern bereitgestellt. In diesen Szenarios erfolgt die Implementierung der Exitbibliotheken für Kommunikationspuffer mithilfe der Installationsscripts dieser Anbieter. Im Folgenden werden die Implementierungsschritte beschrieben, damit Sie bei Bedarf eigene Bibliotheken implementieren können.

Exitbibliothek für Kommunikationspuffer - Position

Exitbibliotheken für Kommunikationspuffer müssen in bestimmten Verzeichnissen abgelegt werden.

Der Datenbankmanager sucht in den folgenden Verzeichnissen nach Exitbibliotheken für Kommunikationspuffer:

Linux und UNIX (32 Bit)
$DB2PATH/security32/plugin/commexit

Linux und UNIX (64 Bit)
$DB2PATH/security64/plugin/commexit

Windows (32 Bit und 64 Bit)
$DB2PATH\security\plugin\instanzname\commexit

Anmerkung: Auf Windows-Plattformen werden die Unterverzeichnisse 'instanzname' und 'commexit' nicht automatisch erstellt. Diese Verzeichnisse müssen vom Instanzeigner manuell erstellt werden.

Exitbibliothek für Kommunikationspuffer - Namenskonventionen und Berechtigungen

Bei Exitbibliotheken für Kommunikationspuffer müssen plattformspezifische Regeln hinsichtlich der Namens- und Berechtigungsvergabe beachtet werden.

Die Länge des Namens einer Exitbibliothek für Kommunikationspuffer beträgt (ohne Dateierweiterung und das Suffix '64') maximal 32 Byte.

Im Folgenden wird die Namenskonvention für die Erweiterung der Bibliotheksdatei auf den einzelnen Plattformen beschrieben:

AIX
Die Erweiterung muss `.a` oder `.so` lauten.

Anmerkung: Wenn beide Erweiterungen (`.a` und `.so`) vorhanden sind, wird `.a` verwendet.

Linux, HP IPF und Solaris
Die Erweiterung muss `.so` lauten.

Windows
Die Erweiterung muss `.dll` lauten.

Im Folgenden wird die Berechtigung für die Bibliotheksdatei auf den einzelnen Plattformen beschrieben:

UNIX und Linux
Die Datei ist Eigentum des Instanzeigners und kann nur vom Instanzeigner gelesen und ausgeführt werden.

Windows
Der Eigner der Datei ist Mitglied der Gruppe DB2AMINS; die Datei kann von einem Mitglied der Gruppe DB2ADMINS gelesen und ausgeführt werden.

Beispiele

Das folgende Beispiel zeigt Erweiterungen für eine Exitbibliothek für Kommunikationspuffer mit dem Namen `mycommexit` für alle Plattformen:

- AIX (64 Bit): `mycommexit.a` oder `mycommexit.so`
- Solaris (64 Bit), Linux (32 Bit oder 64 Bit), HP (64 Bit) auf IPF: `mycommexit.so`
- Windows (32 Bit): `mycommexit.dll`
- Windows (64 Bit): `mycommexit64.dll`

  Anmerkung: Das Dateinamenssuffix '64' ist nur unter Windows (64 Bit) erforderlich.

  Wenn Sie die Datenbankmanagerkonfiguration mit dem Namen einer Exitbibliothek für Kommunikationspuffer aktualisieren, verwenden Sie den vollständigen Namen der Bibliothek ohne das Suffix '64'. Die Dateierweiterung und der qualifizierte Pfad der Datei dürfen bei einer Aktualisierung der Datenbankmanagerkonfiguration ebenfalls nicht angegeben werden.

Das folgende Beispiel zeigt die Aktualisierung der Datenbankmanagerkonfiguration auf einem Windows-System (64 Bit), auf dem die Bibliothek 'mycommexit64.dll' als Exitbibliothek für Kommunikationspuffer definiert wurde.

```
UPDATE DBM CFG USING COMM_EXIT_LIST mycommexit
```

Anmerkung: Der Name **COMM_EXIT_LIST** ist von der Groß-/Kleinschreibung abhängig und muss mit dem Bibliotheksnamen exakt übereinstimmen.

Aktivieren von Exitbibliotheken für Kommunikationspuffer außerhalb von DB2 pureScale-Umgebungen

Die in dieser Task beschriebenen Schritte werden normalerweise von Installationsscripts anderer Anbieter ausgeführt. Diese Schritte sollen Ihnen dabei helfen, eine Exitbibliothek für Kommunikationspuffer zu aktivieren, die Sie selbst entwickelt haben.

Vorbereitende Schritte

Zur Ausführung der Schritte in dieser Task müssen Sie über die Berechtigung SYSADM verfügen.

Einschränkungen

Die Dateien der Exitbibliothek für Kommunikationspuffer unterliegen strengen Dateiberechtigungsrichtlinien. Weitere Informationen zu diesen Richtlinien finden Sie in den zugehörigen Konzepten.

Vorgehensweise

Gehen Sie wie folgt vor, um eine Exitbibliothek für Kommunikationspuffer zu aktivieren:

1. Stoppen Sie den Datenbankmanager. Führen Sie zum Stoppen des Datenbankmanagers den Befehl **db2stop** aus.
2. Kopieren Sie die Datei der Exitbibliothek für Kommunikationspuffer in das richtige Verzeichnis. Weitere Details zur erforderlichen Position für Exitbibliotheken für Kommunikationspuffer finden Sie in den zugehörigen Konzepten. Die Datei kann bei Bedarf einen symbolischen Link zu einer anderen Position aufweisen.
3. Aktualisieren Sie den Konfigurationsparameter **COMM_EXIT_LIST** des Datenbankmanagers mit dem Namen der Bibliothek. Verwenden Sie zum Aktualisieren des Konfigurationsparameters den Befehl **UPDATE DBM CFG**.
4. Starten Sie den Datenbankmanager. Führen Sie zum Starten des Datenbankmanagers den Befehl **db2start** aus.

Ergebnisse

Die Bibliothek wird geladen und initialisiert.

Aktivieren von Exitbibliotheken für Kommunikationspuffer in DB2 pureScale-Umgebungen

Die in dieser Task beschriebenen Schritte werden normalerweise von Installationsscripts anderer Anbieter ausgeführt. Diese Schritte sollen Ihnen dabei helfen, eine Exitbibliothek für Kommunikationspuffer zu aktivieren, die Sie selbst entwickelt haben.

Informationen zu diesem Vorgang

Bei Verwendung einer Exitbibliothek für Kommunikationspuffer, die im Dateinamen eine Versionsnummer enthält, sowie einen symbolischen Link zu dieser Datei für eine Bibliothek ohne Versionsnummer, kann diese Bibliothek memberweise implementiert werden. In einem solchen Szenario muss nicht die gesamte Instanz gestoppt werden, sondern nur einzelne Member.

Einschränkungen

Die Dateien der Exitbibliothek für Kommunikationspuffer unterliegen strengen Dateiberechtigungsrichtlinien. Weitere Informationen zu diesen Richtlinien finden Sie in den zugehörigen Konzepten.

Vorgehensweise

Gehen Sie wie folgt vor, um eine Exitbibliothek für Kommunikationspuffer zu aktivieren:

1. Kopieren Sie die Exitbibliothek für Kommunikationspuffer, die im Dateinamen die Versionsnummer enthält, in das richtige Verzeichnis. Weitere Details zur erforderlichen Position für Exitbibliotheken für Kommunikationspuffer finden Sie in den zugehörigen Konzepten.

2. Erstellen Sie einen symbolischen Link zwischen der Bibliothek ohne Versionsnummer und der Bibliothek mit der Versionsnummer im Dateinamen.
3. Aktualisieren Sie den Konfigurationsparameter **comm_exit_list** des Datenbankmanagers mit dem Namen der Bibliothek. Verwenden Sie zum Aktualisieren des Konfigurationsparameters den Befehl **UPDATE DBM CFG**.
4. Stoppen Sie jedes Member einzeln. Führen Sie dazu den Befehl **db2stop** für jedes Member aus.
5. Starten Sie die gestoppten Member erneut. Führen Sie dazu den Befehl **db2start** aus.

Ergebnisse

Die Bibliothek wird geladen und initialisiert.

Exitbibliothek für Kommunikationspuffer - Fehlerbestimmung

Zur Diagnose von Fehlern bei einer Exitbibliothek für Kommunikationspuffer stehen verschiedene Optionen zur Verfügung.

Die Exitbibliothek für Kommunikationspuffer ist nicht im Lieferumfang von DB2 Database for Linux, UNIX and Windows enthalten. Es handelt sich hierbei um eine Bibliothek, die von Ihnen installiert wird. Sie wird möglicherweise automatisch installiert und von einem verwendeten Tool oder einer verwendeten Anwendung konfiguriert; es besteht auch die Möglichkeit, die Bibliothek selbst zu schreiben.

Der im Konfigurationsparameter **comm_exit_list** des Datenbankmanagers enthaltene Bibliotheksname gibt Rückschlüsse auf die Quelle der Bibliothek.

Treten im Zusammenhang mit der Bibliothek Fehler auf, müssen die Schritte zur Fehlerbestimmung anhand der Dokumentation für das entsprechende Tool bzw. die entsprechende Anwendung festgelegt werden.

Für Exitbibliotheken für Kommunikationspuffer gibt es eine Schnittstelle, über die in die db2diag-Protokolldateien geschrieben werden kann. Anhand der db2diag-Protokolldateien kann festgestellt werden, ob bei der Ausführung der Bibliothek Fehler aufgetreten sind.

Wenn die Leistung der Exitbibliothek für Kommunikationspuffer beeinträchtigt ist, kann die Geschwindigkeit der Bibliothek mithilfe der Überwachungswartezeiten überprüft werden. Weitere Informationen zu diesen Überwachungstools finden Sie in den zugehörigen Referenzen.

Exitbibliothek für Kommunikationspuffer - Entwicklung

Beachten Sie bei der Entwicklung einer Exitbibliothek für Kommunikationspuffer die folgenden Hinweise.

In typischen Szenarios werden Exitbibliotheken für Kommunikationspuffer von Anbietern bereitgestellt. In diesen Szenarios erfolgt die Entwicklung der Exitbibliothek für Kommunikationspuffer durch den Anbieter. Bei Bedarf ist auch die Entwicklung einer eigenen Bibliothek möglich.

Vorgehensweise beim Laden einer Exitbibliothek für Kommunikationspuffer

Beim Start des Datenbankmanagers wird die Exitbibliothek für Kommunikationspuffer dynamisch geladen und initialisiert. Die Bibliothek muss die Initialisierungsfunktion `db2commexitInit` enthalten. Diese Funktion wird als Bibliotheksinitialisierungsfunktion bezeichnet.

Die Bibliotheksinitialisierungsfunktion initialisiert die angegebene Exitbibliothek für Kommunikationspuffer. Die Initialisierung stellt dem Datenbankmanager die Informationen zur Verfügung, die zum Aufrufen der Bibliotheksfunktionen erforderlich sind. Die Bibliotheksinitialisierungsfunktion akzeptiert die folgenden Parameter:

- Die höchste Versionsnummer der Funktionszeigerstruktur, die von der Datenbankinstanz, die die Bibliothek aufruft, unterstützt werden kann.
- Einen Zeiger auf eine Struktur, die Zeiger auf alle APIs enthält, die implementiert werden müssen.
- Einen Zeiger auf eine Funktion, die Protokollnachrichten in die `db2diag`-Protokolldateien einfügt.
- Einen Zeiger auf eine Fehlernachrichtenzeichenfolge.
- Die Länge der Fehlernachricht.

Die Funktionskennung für die Initialisierungsfunktion sieht wie folgt aus:

```
SQL_API_RC SQL_API_FN db2commexitInit
 ( db2int32 version,
   void *commexit_fns,
   db2commexitLogMessage *logMessage_fn,
   char     **errormsg,
   db2int32 *errormsglen );
```

Die Initialisierungsfunktion ist die einzige Funktion in der Bibliothek, die einen vorgeschriebenen Funktionsnamen verwendet. Die anderen Bibliotheksfunktionsfunktionen werden über Funktionszeiger angegeben, die aus der Initialisierungsfunktion zurückgegeben werden.

Diese Funktion hat die folgenden speziellen Aufgaben:

- Sie setzt den Funktionszeiger in einen Zeiger einer entsprechenden Funktionsstruktur um.
- Sie ordnet die Zeiger den anderen Funktionen in der Bibliothek zu.
- Sie ordnet die Versionsnummer der Funktionszeigerstruktur zu, die zurückgegeben wird.

Die Funktion `db2commexitInit` muss mit `extern "C"` deklariert werden, wenn die Bibliothek als C++ kompiliert wird.

APIs der Exitbibliothek für Kommunikationspuffer

APIs werden in der Exitbibliothek für Kommunikationspuffer implementiert.

API 'db2commexitInit' - Initialisierung

Beim Start des Datenbankmanagers mit dem Befehl **db2start** wird die Exitbibliothek für Kommunikationspuffer geladen. Unmittelbar nach dem Laden der Bibliothek wird diese Funktion aufgerufen. Diese Funktion ist für die Initialisierung der

Exitbibliothek für Kommunikationspuffer zuständig. Ferner sorgt sie dafür, dass alle implementierten Funktionen an den Datenbankmanager zurückgegeben werden.

Diese Funktion muss mit `extern "C"` deklariert werden, wenn die Bibliothek als C++ kompiliert wird.

Diese Funktion muss nicht unbedingt threadsicher sein, da sie nur einmal aufgerufen wird.

API-Headerdatei

`db2commexit.h`

API- und Datenstruktursyntax

```
SQL_API_RC ( SQL_API_FN * db2commexitInit )
(
      db2int32 version,
      void *commexit_fns,
      db2commexitLogMessage *logMessage_fn,
      char     **errormsg,
      db2int32 *errormsglen
);
```

Parameter der API 'db2commexitInit'

`version`
Eingabe. Die höchste Version der API, die von der Instanz unterstützt wird, die diese Bibliothek lädt. Der Wert `DB2COMMEXIT_API_VERSION` in `db2commexit.h` enthält die aktuellste Versionsnummer der API, die vom Datenbankmanager gegenwärtig unterstützt wird.

`commexit_fns`
Ausgabe. Ein Zeiger auf die Struktur 'db2commexitFunctions_<versionsnummer>', die Zeiger auf die APIs enthält, die für die Exitbibliothek für Kommunikationspuffer implementiert werden. Möglicherweise gibt es unterschiedliche Versionen der APIs, sodass der Parameter **`commexit_fns`** auf die Struktur 'db2commexitFunctions_<versionsnummer>' umgesetzt wird, die der Version entspricht, die von der Bibliothek implementiert wird. Der erste Parameter der Struktur 'db2commexitFunctions_<versionsnummer>' gibt die Version der APIs an, die vom Plug-in implementiert werden.

`logMessage_fn`
Eingabe. Ein Zeiger auf die API 'db2commexitLogMessage', die vom DB2-Datenbanksystem implementiert wird. Die API 'db2commexitInit' kann die API 'db2commexitLogMessage' aufrufen, um Nachrichten in den `db2diag`-Protokolldateien zu Debug- oder Informationszwecken zu protokollieren. Der erste Parameter der API 'db2commexitLogMessage' gibt den Typ der zu diagnostizierenden Fehler an, die in den `db2diag`-Protokolldateien aufgezeichnet werden. Die beiden letzten Parameter geben die Nachrichtenzeichenfolge und ihre Länge an. Die folgenden Werte sind für den ersten Parameter der API 'db2commexitLogMessage' (in der Datei `db2commexit.h` definiert) gültig:

- `DB2COMMEXIT_LOG_NONE`: (0) - Keine Protokollierung
- `DB2COMMEXIT_LOG_CRITICAL`: (1) - Schwer wiegender Fehler
- `DB2COMMEXIT_LOG_ERROR`: (2) - Fehler
- `DB2COMMEXIT_LOG_WARNING`: (3) - Warnung

- DB2COMMEXIT_LOG_INFO: (4) - Informativ

  Der Nachrichtentext wird nur dann in den db2diag-Protokolldateien angezeigt, wenn der Wert des Parameters 'level' der API 'db2commexitLogMessage' kleiner oder gleich dem Wert des Konfigurationsparameters **diaglevel** des Datenbankmanagers ist. Wenn Sie zum Beispiel den Wert DB2SEC_LOG_INFO verwenden, wird der Nachrichtentext nur dann protokolliert, wenn der Konfigurationsparameter **diaglevel** des Datenbankmanagers auf den Wert 4 gesetzt ist.

errormsg
Ausgabe. Ein Zeiger auf die Adresse einer vom Plug-in zugeordneten ASCII-Fehlernachrichtenzeichenfolge, die in diesem Parameter zurückgegeben werden kann, wenn die Funktionsausführung nicht erfolgreich ist. Dieser Speicher wird durch Aufruf von 'db2commexitFreeErrormsg' nicht freigegeben.

errormsglen
Ausgabe. Ein Zeiger auf einen ganzzahligen Wert (Integer), der die Länge (in Byte) der Fehlernachrichtenzeichenfolge im Parameter **errormsg** angibt.

API 'db2commexitTerm' - Beendigung

Diese Funktion gibt von der Exitbibliothek für Kommunikationspuffer verwendete Ressourcen frei.

Diese API wird vom Datenbankmanager während der Verarbeitung von **db2stop** kurz vor dem Entladen der Exitbibliothek für Kommunikationspuffer aufgerufen. Die API muss in einer Weise implementiert werden, dass sie eine ordnungsgemäße Bereinigung aller Ressourcen vornimmt, die von der Bibliothek belegt werden. Die API muss beispielsweise den von der Bibliothek zugeordneten Speicher freigeben, Dateien schließen, die noch geöffnet sind, und Netzwertverbindungen schließen. Es liegt in der Zuständigkeit der Bibliothek, diese Ressourcen zu verfolgen, um sie freigeben zu können.

Diese Funktion muss nicht unbedingt threadsicher sein, da sie nur einmal aufgerufen wird.

API-Headerdatei

```
db2commexit.h
```

API- und Datenstruktursyntax

```
SQL_API_RC ( SQL_API_FN * db2commexitTerm )
(
   char     **errormsg,
   db2int32 *errormsglen
);
```

Parameter der API 'db2commexitTerm'

errormsg
Ausgabe. Ein Zeiger auf die Adresse einer von der von der Exitbibliothek für Kommunikationspuffer zugeordneten ASCII-Fehlernachrichtenzeichenfolge. Diese Fehlernachrichtenzeichenfolge kann zurückgegeben werden kann, wenn die Funktionsausführung nicht erfolgreich ist. Dieser Speicher wird durch Aufruf von 'db2commexitFreeErrormsg' nicht freigegeben.

errormsglen
Ausgabe. Ein Zeiger auf einen ganzzahligen Wert (Integer), der die Länge (in Byte) der Fehlernachrichtenzeichenfolge im Parameter **errormsg** angibt.

API 'db2commexitRegister' - Registrierung

Diese Funktion registriert den Agenten für die Verbindung.

Diese Funktion wird vom Datenbankmanager immer dann aufgerufen, wenn ein Agent ein Socket akzeptiert und dann an diesem Socket mit dem Senden und Empfangen von Daten beginnt. Diese Aktivität wird normalerweise einer neuen SQL-Verbindung zur Datenbank oder Instanzverbindung zugeordnet.

Ferner wird diese Funktion aufgerufen, wenn einem Agenten eine inaktive Verbindung zugeteilt wird, um eine neue Anforderung des Clients zu verarbeiten.

Diese Funktion ist den SQL-Verbindungen zur Datenbank nicht direkt zugeordnet. Ein Eingabeparameter für die Funktion unterscheidet zwischen einem neuen Socket und einem vorhandenen Socket, das einem neuen Agenten zugeteilt wird.

API-Headerdatei

```
db2commexit.h
```

API- und Datenstruktursyntax

```
SQL_API_RC ( SQL_API_FN * db2commexitRegister )
(
   void                          ** pConnectionContext,
   const db2commexitCommInfo_v1   * pCommInfo,
   db2int32                         state,
   db2int64                       * pReservedFlags,
   char                          ** errormsg,
   db2int32                       * errormsglen
);
```

Parameter der API 'db2commexitRegister'

pConnectionContext
Ein-/Ausgabe. Ein Zeiger auf Daten, die sich auf die Exitbibliothek für Kommunikationspuffer beziehen. Dieser Zeiger bezieht sich auf die eingehende Verbindung. Der Parameter wird als Eingabe für jeden Funktionsaufruf für diese Verbindung übergeben. Die Bibliothek kann verbindungsspezifische Informationen zuordnen und speichern und diese in jedem Funktionsaufruf zur Verfügung stellen. Der Speicher für den Parameter muss beim Aufrufen von db2commexitDeregister freigegeben werden. Der Datenbankmanager kann nicht auf den Speicher zugreifen, auf den von diesem Parameter verwiesen wird.

pCommInfo
Eingabe. Ein Zeiger auf eine Struktur, die Informationen enthält, die den Datenbankserver und protokollspezifische Informationen für die eingehende Verbindung identifizieren. Einige der Felder in der Struktur werden erst dann definiert, wenn mit dem Client mehrere Puffer ausgetauscht werden. Die Felder stehen bei späteren Aufrufen von 'db2commexitRecv' und 'db2commexitSend' zur Verfügung. Dieses Szenario bezieht sich speziell auf **inbound_appl_id**, **outbound_appl_id** und **connection_type**. Sobald diese Werte bekannt sind, gibt der Parameter **connection_type** an, ob die Verbindung für eine lokale Datenbank oder eine Gateway-Verbindung vorgesehen ist.

State
Eingabe. Gibt an, unter welchen Bedingungen die Funktion aufgerufen wird. Mögliche Werte:

- NEW_CONNECTION - gibt eine neue physische eingehende Clientverbindung an.

- AGENT_ASSOCIATION - gibt eine vorhandene inaktive Clientverbindung an, die erneut aktiviert wird und zur Verarbeitung der Anforderung einem Agenten zugeordnet wird.

pReservedFlags
Ein-/Ausgabe. Zur zukünftigen Verwendung reserviert. Der Wert muss bei der Ausgabe auf 0 gesetzt werden.

errormsg
Ausgabe. Ein Zeiger auf die Adresse einer von der von der Exitbibliothek für Kommunikationspuffer zugeordneten ASCII-Fehlernachrichtenzeichenfolge. Diese Fehlernachrichtenzeichenfolge kann zurückgegeben werden kann, wenn die Funktionsausführung nicht erfolgreich ist. Dieser Speicher wird durch Aufruf von 'db2commexitFreeErrormsg' nicht freigegeben.

errormsglen
Ausgabe. Ein Zeiger auf einen ganzzahligen Wert (Integer), der die Länge (in Byte) der Fehlernachrichtenzeichenfolge im Parameter **errormsg** angibt.

API 'db2commexitDeregister' - Zurücknehmen der Registrierung

Diese Funktion hebt die Zuordnung des Agenten zu einer Verbindung auf.

Diese Funktion wird vom Datenbankmanager immer dann aufgerufen, wenn der Agent keine Anforderungen mehr über die Verbindung verarbeitet. Diese Situation tritt auf, wenn die physische Verbindung mit dem Client beendet wird oder der Client inaktiv ist und die Zuordnung des Agenten zu diesem Client aufgehoben wird.

API-Headerdatei

```
db2commexit.h
```

API- und Datenstruktursyntax

```
SQL_API_RC ( SQL_API_FN * db2commexitDeregister )
(
   void                          * pConnectionContext,
   const db2commexitCommInfo_v1  * pCommInfo,
   db2int32                        state,
   db2int64                      * pReservedFlags,
   char                         ** errormsg,
   db2int32                      * errormsglen
);
```

Parameter der API 'db2commexitDeregister'

pConnectionContext
Eingabe. Ein Zeiger auf Daten, die sich auf die Exitbibliothek für Kommunikationspuffer beziehen. Dieser Zeiger bezieht sich auf die eingehende Verbindung. Der Parameter wird als Eingabe für jeden Funktionsaufruf für diese Verbindung übergeben. Der Datenbankmanager kann nicht auf den Speicher zugreifen, auf den von diesem Parameter verwiesen wird. Der Speicher muss durch diese Funktion freigegeben werden.

pCommInfo
Eingabe. Ein Zeiger auf eine Struktur, die Informationen enthält, die den Datenbankserver und protokollspezifische Informationen für die eingehende Verbindung identifizieren.

State
Eingabe. Gibt an, unter welchen Bedingungen die Funktion aufgerufen wird. Mögliche Werte:

- `CONNECTION_TERM` - gibt an, dass die physische Verbindung mit dem Client beendet wird.
- `AGENT_DISASSOCIATION` - gibt an, dass die Clientverbindung inaktiv ist und die Zuordnung des Agenten zu dieser Verbindung aufgehoben wird.

pReservedFlags
Ein-/Ausgabe. Zur zukünftigen Verwendung reserviert. Der Wert muss bei der Ausgabe auf `0` gesetzt werden.

errormsg
Ausgabe. Ein Zeiger auf die Adresse einer von der von der Exitbibliothek für Kommunikationspuffer zugeordneten ASCII-Fehlernachrichtenzeichenfolge. Diese Fehlernachrichtenzeichenfolge kann zurückgegeben werden kann, wenn die Funktionsausführung nicht erfolgreich ist. Dieser Speicher wird durch Aufruf von 'db2commexitFreeErrormsg' nicht freigegeben.

errormsglen
Ausgabe. Ein Zeiger auf einen ganzzahligen Wert (Integer), der die Länge (in Byte) der Fehlernachrichtenzeichenfolge im Parameter **errormsg** angibt.

API 'db2commexitRecv' - Empfangen

Diese Funktion wird für jeden Puffer aufgerufen, den der Datenbankmanager von einem Client empfängt.

Diese Funktion wird vom Datenbankmanager unmittelbar nach dem Empfang eines Kommunikationspuffers vom Client aufgerufen. Die Funktion wird nach der Entschlüsselung des Puffers aufgerufen, sodass die Exitbibliothek für Kommunikationspuffer auf den unverschlüsselten Puffer zugreifen kann.

API-Headerdatei

```
db2commexit.h
```

API- und Datenstruktursyntax

```
SQL_API_RC ( SQL_API_FN * db2commexitRecv )
(
   void                          * pConnectionContext,
   const db2commexitCommInfo_v1  * pCommInfo,
   const db2commexitBuffer       * pBuffer,
   db2int64                      * pReservedFlags,
   char                         ** errormsg,
   db2int32                      * errormsglen
);
```

Parameter der API 'db2commexitRecv'

pConnectionContext
Eingabe. Ein Zeiger auf Daten, die sich auf die Exitbibliothek für Kommunikationspuffer beziehen. Dieser Zeiger bezieht sich auf die eingehende Verbindung. Der Parameter wird als Eingabe für jeden Funktionsaufruf für diese Verbindung übergeben. Der Datenbankmanager kann nicht auf den Speicher zugreifen, auf den von diesem Parameter verwiesen wird. Der Speicher muss durch diese Funktion freigegeben werden.

pCommInfo
Eingabe. Ein Zeiger auf eine Struktur, die Informationen enthält, die den Da-

tenbankserver und protokollspezifische Informationen für die eingehende Verbindung identifizieren. Einige der Felder in der Struktur werden erst dann definiert, wenn mit dem Client mehrere Puffer ausgetauscht werden. Die Felder stehen bei späteren Aufrufen von 'db2commexitRecv' und 'db2commexitSend' zur Verfügung. Dieses Szenario bezieht sich speziell auf **inbound_appl_id**, **outbound_appl_id** und **connection_type**.

pBuffer
Eingabe. Ein Zeiger auf eine Struktur, in der die Länge des vom Datenbankmanager empfangenen Puffers sowie ein Zeiger auf den Puffer enthalten ist. Wenn der Puffer verschlüsselt ist, wird er vor dem Aufruf dieser Funktion entschlüsselt.

pReservedFlags
Ein-/Ausgabe. Das Bit DB2COMMEXIT_RECV_IN_FLAG_END_DECRYPT wird gesetzt, um anzugeben, dass dies der letzte Aufruf dieser Funktion für eine verschlüsselte DSS (Data Stream Structure) ist. Die Länge der DSS, die als Eingabe übergeben wird, gibt die Länge der verschlüsselten DSS an. Die DSS wird dann jedoch entschlüsselt und die Auffüllung entfernt. Da immer eine Auffüllung vorhanden ist, ist die DSS nicht so lang wie angegeben. Die in der Struktur 'pBuffer' angegebene Länge sind die letzten Daten für die DSS. Es ist möglich, dass der Wert null ist, wenn die Auffüllung eine volle Blockgröße beträgt.

Dieser Wert ist für die Ausgabe zur zukünftigen Verwendung reserviert. Der Wert muss bei der Ausgabe auf 0 gesetzt werden.

errormsg
Ausgabe. Ein Zeiger auf die Adresse einer von der von der Exitbibliothek für Kommunikationspuffer zugeordneten ASCII-Fehlernachrichtenzeichenfolge. Diese Fehlernachrichtenzeichenfolge kann zurückgegeben werden kann, wenn die Funktionsausführung nicht erfolgreich ist. Dieser Speicher wird durch Aufruf von 'db2commexitFreeErrormsg' nicht freigegeben.

errormsglen
Ausgabe. Ein Zeiger auf einen ganzzahligen Wert (Integer), der die Länge (in Byte) der Fehlernachrichtenzeichenfolge im Parameter **errormsg** angibt.

API 'db2commexitSend' - Senden

Diese Funktion wird für jeden Puffer aufgerufen, den der Datenbankmanager an einen Client sendet.

Diese Funktion wird vom Datenbankmanager unmittelbar nach dem Senden eines Kommunikationspuffers an den Client aufgerufen. Die Funktion wird vor der Verschlüsselung des Puffers aufgerufen, sodass die Exitbibliothek für Kommunikationspuffer auf den unverschlüsselten Puffer zugreifen kann.

API-Headerdatei

```
db2commexit.h
```

API- und Datenstruktursyntax

```
SQL_API_RC ( SQL_API_FN * db2commexitSend )
(
   void                          * pConnectionContext,
   const db2commexitCommInfo_v1  * pCommInfo,
   const db2commexitBuffer       * pBuffer,
   db2int64                      * pReservedFlags,
   char                         ** errormsg,
   db2int32                      * errormsglen
);
```

Parameter der API 'db2commexitSend'

pConnectionContext
Eingabe. Ein Zeiger auf Daten, die sich auf die Exitbibliothek für Kommunikationspuffer beziehen. Dieser Zeiger bezieht sich auf die eingehende Verbindung. Der Parameter wird als Eingabe für jeden Funktionsaufruf für diese Verbindung übergeben. Der Datenbankmanager kann nicht auf den Speicher zugreifen, auf den von diesem Parameter verwiesen wird.

pCommInfo
Eingabe. Ein Zeiger auf eine Struktur, die Informationen enthält, die den Datenbankserver und protokollspezifische Informationen für die eingehende Verbindung identifizieren. Einige der Felder in der Struktur werden erst dann definiert, wenn mit dem Client mehrere Puffer ausgetauscht werden. Die Felder stehen bei späteren Aufrufen von 'db2commexitRecv' und 'db2commexitSend' zur Verfügung. Dieses Szenario bezieht sich speziell auf **inbound_appl_id**, **outbound_appl_id** und **connection_type**.

pBuffer
Eingabe. Ein Zeiger auf eine Struktur, in der die Länge des an den Client gesendeten Puffers sowie ein Zeiger auf den Puffer enthalten ist. Wenn der Puffer verschlüsselt ist, wird er vor dem Aufruf dieser Funktion entschlüsselt.

pReservedFlags
Ein-/Ausgabe. Das Bit DB2COMMEXIT_SEND_IN_FLAG_PURGE wird definiert, wenn der Datenbankmanager einen Fehler feststellt und einige Puffer löschen muss, die an den Client gesendet werden sollten. Einige dieser Puffer wurden möglicherweise als Eingabe an die Exitbibliothek für Kommunikationspuffer übergeben.

Dieser Wert ist für die Ausgabe zur zukünftigen Verwendung reserviert. Der Wert muss bei der Ausgabe auf 0 gesetzt werden.

errormsg
Ausgabe. Ein Zeiger auf die Adresse einer von der von der Exitbibliothek für Kommunikationspuffer zugeordneten ASCII-Fehlernachrichtenzeichenfolge. Diese Fehlernachrichtenzeichenfolge kann zurückgegeben werden kann, wenn die Funktionsausführung nicht erfolgreich ist. Dieser Speicher wird durch Aufruf von 'db2commexitFreeErrormsg' nicht freigegeben.

errormsglen
Ausgabe. Ein Zeiger auf einen ganzzahligen Wert (Integer), der die Länge (in Byte) der Fehlernachrichtenzeichenfolge im Parameter **errormsg** angibt.

API 'db2commexitUserIdentity' - Benutzeridentität

Diese Funktion wird aufgerufen, um die Identität des Benutzers für das aktuelle Socket anzugeben.

Diese Funktion wird aufgerufen, um der Exitbibliothek für Kommunikationspuffer den Benutzernamen und die Sitzungsberechtigungs-ID zur Herstellung der Verbindung mitzuteilen. Außerdem wird die Funktion aufgerufen, wenn sich diese Parameter aufgrund eines Wechsels des Benutzers in einem gesicherten Kontext oder der Anweisung SET SESSION AUTHORIZATION ändern. Der Benutzername und die Sitzungsberechtigungs-ID werden erst dann festgelegt, wenn der Datenbankmanager den Benutzer authentifiziert hat. Diese Funktion wird erst dann aufgerufen, wenn db2commexitRegister und mehrere db2commexitSend- und db2commexitRecv-Funktionen aufgerufen werden, um die Authentifizierung durchzuführen.

API-Headerdatei

db2commexit.h

API- und Datenstruktursyntax

```
SQL_API_RC ( SQL_API_FN * db2commexitUserIdentity )
(
   void                          * pConnectionContext,
   const db2commexitCommInfo_v1  * pCommInfo,
   db2int32                        state,
   db2int32                        usernameLen,
   const char                    * pUserame,
   db2int32                        sessionAuthidLen,
   const char                    * pSessionAuthid,
   db2int64                      * pReservedFlags,
   char                         ** errormsg,
   db2int32                      * errormsglen
);
```

Parameter der API 'db2commexitUserIdentity'

pConnectionContext
Eingabe. Ein Zeiger auf Daten, die sich auf die Exitbibliothek für Kommunikationspuffer beziehen. Dieser Zeiger bezieht sich auf die eingehende Verbindung. Der Parameter wird als Eingabe für jeden Funktionsaufruf für diese Verbindung übergeben. Der Datenbankmanager kann nicht auf den Speicher zugreifen, auf den von diesem Parameter verwiesen wird.

pCommInfo
Eingabe. Ein Zeiger auf eine Struktur, die Informationen enthält, die den Datenbankserver und protokollspezifische Informationen für die eingehende Verbindung identifizieren. Einige der Felder in der Struktur werden erst dann definiert, wenn mit dem Client mehrere Puffer ausgetauscht werden. Die Felder stehen bei späteren Aufrufen von 'db2commexitRecv' und 'db2commexitSend' zur Verfügung. Dieses Szenario bezieht sich speziell auf **inbound_appl_id**, **outbound_appl_id** und **connection_type**.

State
Eingabe. Gibt an, unter welchen Bedingungen die Funktion aufgerufen wird. Mögliche Werte:

- DB2COMMEXIT_USERIDENT_NEW_CONNECTION - gibt eine neue Verbindung an.
- DB2COMMEXIT_USERIDENT_TC_SWITCH_USER - gibt einen Benutzerwechsel in einem gesicherten Kontext an.
- DB2COMMEXIT_USERIDENT_SET_SESSION_USER - Die SQL-Anweisung SET SESSION AUTHORIZATION wird ausgegeben, um die Sitzungsberechtigungs-ID zu ändern.

usernameLen
Eingabe. Die Länge von **pUsername**.

pUsername
Eingabe. Der Benutzername, der zur Herstellung der Verbindung verwendet wird.

sessionAuthidLen
Eingabe. Die Länge von **pSessionAuthid**.

pSessionAuthid
Eingabe. Die für diese Verbindung erstellte Sitzungsberechtigungs-ID.

pReservedFlags
Ein-/Ausgabe. Zur zukünftigen Verwendung reserviert. Der Wert muss bei der Ausgabe auf `0` gesetzt werden.

errormsg
Ausgabe. Ein Zeiger auf die Adresse einer von der von der Exitbibliothek für Kommunikationspuffer zugeordneten ASCII-Fehlernachrichtenzeichenfolge. Diese Fehlernachrichtenzeichenfolge kann zurückgegeben werden kann, wenn die Funktionsausführung nicht erfolgreich ist. Dieser Speicher wird durch Aufruf von 'db2commexitFreeErrormsg' nicht freigegeben.

errormsglen
Ausgabe. Ein Zeiger auf einen ganzzahligen Wert (Integer), der die Länge (in Byte) der Fehlernachrichtenzeichenfolge im Parameter **errormsg** angibt.

API 'db2commexitFreeErrormsg' - Speicher für Fehlernachricht freigeben

Diese Funktion gibt den Speicher frei, der zur Aufnahme einer Fehlernachricht aus einem früheren API-Aufruf verwendet wurde.

API-Headerdatei

`db2commexit.h`

API- und Datenstruktursyntax

```
SQL_API_RC ( SQL_API_FN * db2commexitFreeErrormsg )
 ( char * errormsg );
```

db2commexitFreeErrormsg API Parameters

errormsg
Eingabe. Ein Zeiger auf die Fehlernachricht, die aus einem früheren API-Aufruf zurückgegeben wurde.

Exitbibliothek für Kommunikationspuffer - Funktionsstruktur

Die Funktion `db2commexitInit` verwendet den Parameter `void * commexit_fns`. Dieser Parameter wird auf die versionsspezifische Struktur umgesetzt, die alle von der Exitbibliothek für Kommunikationspuffer implementierten Funktionen enthält. Die Funktion `db2commexitInit` muss die Funktionszeiger zuordnen, damit der Datenbankmanager diese Funktionen aufrufen kann.

Die folgende Struktur muss ausgeführt werden, einschließlich eines Funktionszeigers für jede API.

```
struct db2commexitFunctions_v1
{
    db2int32 version;

    SQL_API_RC ( SQL_API_FN * db2commexitTerm )
    (
       char     **errormsg,
       db2int32 *errormsglen
    );

    SQL_API_RC ( SQL_API_FN * db2commexitRegister )
    (
       void                          ** ppConnectionContext,
       const db2commexitCommInfo_v1  * pCommInfo,
       db2int32                        state,
       db2int64                      * pReservedFlags,
       char                          ** errormsg,
```

```
    db2int32                     * errormsglen
  );

  SQL_API_RC ( SQL_API_FN * db2commexitDeregister )
  (
    void                         * pConnectionContext,
    const db2commexitCommInfo_v1 * pCommInfo,
    db2int32                       state,
    db2int64                     * pReservedFlags,
    char                        ** errormsg,
    db2int32                     * errormsglen
  );

  SQL_API_RC ( SQL_API_FN * db2commexitRecv )
  (
    void                         * pConnectionContext,
    const db2commexitCommInfo_v1 * pCommInfo,
    const db2commexitBuffer      * pBuffer,
    db2int64                     * pReservedFlags,
    char                        ** errormsg,
    db2int32                     * errormsglen
  );

  SQL_API_RC ( SQL_API_FN * db2commexitSend )
  (
    void                         * pConnectionContext,
    const db2commexitCommInfo_v1 * pCommInfo,
    const db2commexitBuffer      * pBuffer,
    db2int64                     * pReservedFlags,
    char                        ** errormsg,
    db2int32                     * errormsglen
  );

  SQL_API_RC ( SQL_API_FN * db2commexitUserIdentity )
  (
    void                         * pConnectionContext,
    const db2commexitCommInfo_v1 * pCommInfo,
    db2int32                       state,
    db2int32                       usernameLen,
    const char                   * pUserame,
    db2int32                       sessionAuthidLen,
    const char                   * pSessionAuthid,
    db2int64                     * pReservedFlags,
    char                        ** errormsg,
    db2int32                     * errormsglen
  );

  SQL_API_RC ( SQL_API_FN * db2commexitFreeErrormsg )
  (
    char * errormsg
  );

};
```

Exitbibliothek für Kommunikationspuffer - Informationsstruktur

Die Informationsstruktur gibt die Kommunikationsprotokollinformationen für die aktuelle physische Verbindung an.

Die folgende Struktur `db2commexitCommInfo_v1` wird an jede Funktion der Exitbibliothek für Kommunikationspuffer übergeben. Diese Struktur ist in der Datei `db2commexit.h` enthalten.

```
struct db2commexitIPV4Info
{
   sockaddr_in client_sockaddr;
   sockaddr_in server_sockaddr;
};

struct db2commexitIPV6Info
{
   sockaddr_in6 client_sockaddr;
   sockaddr_in6 server_sockaddr;
};

struct db2commexitIPCInfo
{
   void * pSharedMemSegmentHandle;
};

struct db2commexitNamedPipeInfo
{
   void * handle;
};


struct db2commexitCommInfo_v1
{
   db2int32 clientProtocol;   // SQL_PROTOCOL_ ...
   db2int32 connectionType; // unknown, local or gateway

   db2int32 hostnameLen;
   db2int32 instanceLen;
   db2int32 dbnameLen;
   db2int32 dbaliasLen;
   db2int32 inbound_appl_id_len;
   db2int32 outbound_appl_id_len;

   db2int32 reserved1;
   db2int32 reserved2;

   db2NodeType member;

   char hostname[SQL_HOSTNAME_SZ+1];
   char instance[DB2COMMEXIT_INSTANCE_SZ + 1];
   char dbname[DB2COMMEXIT_DBNAME_SZ + 1];
   char dbalias[DB2COMMEXIT_DBNAME_SZ + 1];
   char inbound_appl_id[SQLM_APPLID_SZ + 1];
   char outbound_appl_id[SQLM_APPLID_SZ + 1];

   char reservedChar1[128];

   union
   {
      db2commexitIPV4Info ipv4Info;
      db2commexitIPV6Info ipv6Info;
      db2commexitIPCInfo  ipcInfo;
      db2commexitNamedPipeInfo namedPipeInfo;
   }
};
```

Pufferstruktur der Exitbibliothek für Kommunikationspuffer

Die Pufferstruktur ist die Struktur, die als Eingabe an die Funktionen db2commexitSend und db2commexitRecv übergeben wird.

Die Pufferstruktur sieht wie folgt aus:

```
struct db2commexitBuffer
{
   const unsigned char * pBuffer;
   db2int64 buffer_len;

   db2int32 reserved1;
   db2int32 reserved2;
};
```

Exitbibliothek für Kommunikationspuffer - Verbindungssteuerung

Die Exitbibliothek für Kommunikationspuffer kann jederzeit erzwingen, dass die Verbindung zum Client getrennt wird.

Wenn die Exitbibliothek für Kommunikationspuffer beim Aufrufen von `db2commexitUserIdentity`, `db2commexitRegister`, `db2commexitDeregister`, `db2commexitRecv` oder `db2commexitSend` den entsprechenden Fehlercode zurückgibt, wird die Verbindung zum Client vom Datenbankmanager umgehend geschlossen.

Mit dieser Funktionalität kann die Exitbibliothek für Kommunikationspuffer auf Grundlage der überprüften Puffer feststellen, ob eine nicht ordnungsgemäße Aktivität vorliegt. Sollte dies der Fall sein, können weitere Aktionen des Datenbankmanagers für diese Verbindung verhindert werden.

Exitbibliothek für Kommunikationspuffer - API-Versionen

Das DB2-Datenbanksystem unterstützt die Nummerierung der API-Versionen der Exitbibliothek für Kommunikationspuffer. Bei diesen Versionsnummern handelt es sich um ganzzahlige Werte, die mit 1 beginnen.

Die Versionsnummer, die der Datenbankmanager an die Initialisierungsfunktion der Sicherheitsbibliothek übergibt, ist die höchste unterstützte Versionsnummer der API. Wenn die Bibliothek höhere API-Versionen unterstützt, muss sie Funktionszeiger für die Version zurückgeben, die der Datenbankmanager anfordert. Wenn die Bibliothek nur eine niedrigere Version der API unterstützt, muss die Bibliothek Funktionszeiger für die unterstützte Version definieren. In beiden Fällen muss die Initialisierungsfunktion der Bibliothek die unterstützte Versionsnummer für die API im Versionsfeld der Funktionsstruktur zurückgeben.

Die Versionsnummern der APIs der Exitbibliothek für Kommunikationspuffer werden nur bei Bedarf geändert. Dies ist beispielsweise der Fall, wenn Parameter der APIs geändert werden. Die Versionsnummern werden nicht automatisch mit den Releasenummern des Datenbankmanagers geändert.

Die Versionsnummern ermöglichen die Einführung neuer oder geänderter APIs. Die Unterstützung der Bibliothek für ältere Versionen wird beibehalten.

Exitbibliothek für Kommunikationspuffer - Fehlerbehandlung und Rückkehrcodes

Wenn in einer API der Exitbibliothek für Kommunikationspuffer ein Fehler auftritt, kann die API eine ASCII-Textzeichenfolge im Feld **`errormsg`** zurückgeben. Diese ASCII-Textzeichenfolge enthält eine genauere Beschreibung des Problems als der Rückkehrcode. Der Datenbankmanager schreibt die gesamte Zeichenfolge in die `db2diag`-Protokolldateien.

Der Speicher für diese Fehlernachrichten muss von der Exitbibliothek für Kommunikationspuffer zugeordnet werden. Daher muss die Bibliothek auch eine API zur Freigabe dieses Speichers bereitstellen: db2commexitFreeErrormsg.

Zusätzlich zum Feld **errormsg** wird bei der Initialisierung ein Zeiger (logMessage_fn) der Nachrichtenprotokollierungsfunktion an die Exitbibliothek für Kommunikationspuffer übergeben. Die Bibliothek kann die Funktion zum Protokollieren beliebiger Debugging-Informationen in den db2diag-Protokolldateien verwenden. Beispiel:

```
// // Protokollieren einer Nachricht 'Init erfolgreich'
  (*(logMessage_fn))(DB2COMMEXIT_LOG_CRITICAL,
                     "comm exit-Initialisierung erfolgreich",
                      strlen("comm. exit-Initialisierung erfolgreich"));
```

Weitere detaillierte Informationen zu den Parametern für die Funktion db2secLogMessage finden Sie in den Beschreibungen der Initialisierungs-API db2commexitInit in den zugehörigen Referenzen.

Rückkehrcodes

Tabelle 42. Rückkehrcodes, die eine Exitbibliothek für Kommunikationspuffer an den Datenbankmanager zurückgeben kann

Rückkehrcode	DEFINE-Wert	Details
0	DB2COMMEXIT_SUCCESS	Erfolgreiche Ausführung
-1	DB2COMMEXIT_ ERR_UNKNOWN	Die Bibliothek hat einen unerwarteten Fehler festgestellt.
-2	DB2COMMEXIT_ERR_ DROP_CONNECTION	Die Bibliothek hat festgestellt, dass die Verbindung, für die sie aufgerufen wurde, beendet werden muss.

Exitbibliothek für Kommunikationspuffer - Einschränkungen bei der Entwicklung

Bei der Entwicklung einer Exitbibliothek für Kommunikationspuffer gibt es einige Einschränkungen und Hinweise, die beachtet werden müssen.

Einschränkungen

C-Verlinkung
Die Exitbibliothek für Kommunikationspuffer muss in C/C++ geschrieben und mit der C-Verlinkung (C-linkage) verlinkt werden. Headerdateien, die Prototypen definieren, Datenstrukturen, die die Bibliothek implementieren, sowie Fehlercodedefinitionen werden nur für C/C++ zur Verfügung gestellt. Die Funktion db2commexitInit muss mit extern "C" deklariert werden, wenn die Bibliothek als C++ kompiliert wird.

Signalroutinen
Die Exitbibliothek für Kommunikationspuffer darf keine Signalroutinen (Signal Handler) installieren oder die Signalmaske ändern, da dies zu einer Kollision mit den Signalroutinen des Datenbankmanagers führt. Eine Kollision mit den Signalroutinen des Datenbankmanagers könnte die Fähigkeit, Fehler zu melden und zu beheben, ernsthaft beeinträchtigen.

Ausnahmebedingungen
Die APIs der Exitbibliothek für Kommunikationspuffer dürfen keine C++-Aus-

nahmebedingungen auslösen. Solche Ausnahmebedingungen können die Fehlerbehandlung des Datenbankmanagers stören.

Threadsicherheit
Die Funktionen der Exitbibliothek für Kommunikationspuffer müssen threadsicher sein. Die Funktionen `db2commexitInit` und `db2commexitTerm` sind die einzigen APIs, auf die diese Anforderung nicht zutrifft.

Exitsteuerroutinen
Die Exitbibliothek für Kommunikationspuffer darf keine Exitsteuerroutinen (Exit Handler) oder `pthread_atfork`-Steuerroutinen installieren. Die Verwendung von Exitsteuerroutinen wird nicht unterstützt, weil die Exitbibliothek für Kommunikationspuffer aus dem Speicher entladen wird, bevor der Datenbankmanagerprozess beendet wird.

Verzweigung (Fork)/Threads
Die Exitbibliothek für Kommunikationspuffer darf keine Verzweigung aufrufen oder neue Threads erstellen, da dies zu einem nicht definierten Verhalten wie beispielsweise Traps im Datenbankmanager führen kann.

Bibliotheksabhängigkeiten
Unter Linux und UNIX wird die Exitbibliothek für Kommunikationspuffer aus einem Prozess ('setuid' oder 'setgid') geladen. Dabei kann nicht sichergestellt werden, dass die Umgebungsvariable **LD_LIBRARY_PATH**, **SHLIB_PATH** oder **LIBPATH** abhängige Bibliotheken findet. Daher darf die Bibliothek nur dann von weiteren Bibliotheken abhängig sein, wenn auf diese über andere Methoden zugegriffen werden kann. Beispiele:

- Die Bibliotheken befinden sich im Verzeichnis '/lib' oder '/usr/lib'.
- Die Verzeichnisse, in denen sich die Bibliotheken befinden, sind betriebssystemweit bekannt (z. B. durch Angabe in der Datei 'ld.so.conf' unter Linux).
- Die Bibliotheken sind in RPATH in der Bibliothek selbst angegeben.

Symbolkollisionen
Die Exitbibliothek für Kommunikationspuffer sollte nach Möglichkeit mit allen verfügbaren Optionen kompiliert und verlinkt werden, die die Wahrscheinlichkeit von Symbolkollisionen verringern, wie zum Beispiel mit den Optionen, die nicht gebundene externe symbolische Referenzen reduzieren. Zum Beispiel kann die Verwendung der Linkeroption "-Bsymbolic" unter HP, Solaris und Linux dabei helfen, Probleme im Zusammenhang mit Symbolkollisionen zu vermeiden. Allerdings darf für Bibliotheken, die unter AIX geschrieben werden, die Linkeroption "-brtl" weder explizit noch implizit verwendet werden.

32-Bit- und 64-Bit-Version
Der Datenbankmanager verfügt je nach Plattform über eine 32-Bit- und eine 64-Bit-Version. Eine 32-Bit-Exitbibliothek für Kommunikationspuffer muss auf einem 32-Bit-Datenbankmanager aktiviert werden, und eine 64-Bit-Exitbibliothek für Kommunikationspuffer muss auf einem 64-Bit-Datenbankmanager aktiviert werden. Die beiden Versionen können nicht miteinander kombiniert werden.

Gespeicherte Prozeduren, Trigger und andere interne SQL
Interaktionen von gespeicherten Prozeduren mit dem Server werden an die Exitbibliothek für Kommunikationspuffer übergeben. Viele Interaktionen erfolgen nicht über die Standardkommunikationskanäle und passen nicht in das für die Exitbibliothek verwendete Modell. Gleichermaßen werden Trigger und andere Quellen interner SQL nicht über die Standardkommunikationskanäle übertragen und nicht an die Exitbibliothek für Kommunikationspuffer übergeben.

Kommunikationspuffer dürfen nicht bearbeitet werden
Es wird davon ausgegangen, dass die übergebenen Puffer von der Exitbibliothek nicht bearbeitet oder geändert werden.

Unterstützung für schrittweise Aktualisierungen
DB2 for Linux, UNIX and Windows unterstützt die Aktualisierung der Fixpackstufe einzelner Member in DB2 pureScale-Umgebungen, ohne andere Member stoppen zu müssen. Dieses Verhalten wird als schrittweise Aktualisierung bezeichnet. Ebenso kann die für einzelne Member verwendete Bibliotheksversion aktualisiert werden, wie im Abschnitt zur Implementierung der Exitbibliothek für Kommunikationspuffer beschrieben. Es besteht die Möglichkeit, dass auf zwei unterschiedlichen Membern gleichzeitig zwei unterschiedliche Versionen der Exitbibliothek für Kommunikationspuffer ausgeführt werden, wobei jedes Member eine andere Fixpackstufe aufweisen kann. Die Exitbibliothek für Kommunikationspuffer muss solche Bedingungen fehlerfrei tolerieren können.

Exitbibliothek für Kommunikationspuffer - API-Aufrufreihenfolgen

Die API-Aufrufreihenfolgen können je nach Szenario unterschiedlich sein.

In den folgenden Abschnitten werden verschiedene Szenarios beschrieben, die bei der Entwicklung von Exitbibliotheken für Kommunikationspuffer beachtet werden müssen. Anhand dieser Informationen können Sie die für Ihre Umgebung am bestem geeignete Aufrufreihenfolge bestimmen.

API-Aufrufreihenfolge - Normaler Verbindungsaufbau in einem einzelnen Agenten

Das häufigste Szenario ist ein Client, der eine Verbindung zum Datenbankmanager herstellt, einige SQL-Anweisungen absetzt und dann die Verbindung trennt.

In einem solchen Fall verarbeitet ein einzelner Agent oder Thread die Verbindung und die folgenden Aufrufe werden ausgeführt:

1. `db2commexitRegister` für eine neue Socketverbindung
2. `db2commexitRecv` und `db2commexitSend` zur Authentifizierung (möglicherweise mehrmals)
3. `db2commexitUserIdentity` für eine neue Verbindung
4. `db2commexitRecv` und `db2commexitSend` zur Verarbeitung von SQL-Anforderungen der Clients (möglicherweise mehrmals)
5. `db2commexitDeregister` zum Beenden der Socketverbindung

API-Aufrufreihenfolge - Verbindung ohne CONNECT RESET

In diesem Szenario wird eine Verbindung über ein vorhandenes Socket beschrieben. Der Client versucht möglicherweise, eine weitere SQL-Verbindung herzustellen, ohne zuvor eine CONNECT RESET-Anweisung abzusetzen.

Nachdem der Datenbankmanager die SQL-CONNECT-Anweisung vom Client empfangen hat, veranlasst dieser implizit ein internes CONNECT RESET, bevor der Verbindungsaufbau fortgesetzt wird. Der Socketstatus ändert sich nicht, da es sich um normale Anforderungen und Antworten handelt. In diesem Fall werden alle Anforderungen von einem einzelnen Agenten verarbeitet. Da die Puffer, in denen die Verbindungsanforderung des Clients enthalten ist, über `db2commexitRecv` bereitgestellt werden, kann die Exitbibliothek für Kommunikationspuffer beim Parsing des Puffers eine neue Verbindungsanforderung feststellen. Die folgenden Aufrufe werden ausgeführt:

1. db2commexitRegister für eine neue Socketverbindung
2. db2commexitRecv und db2commexitSend zur Authentifizierung (möglicherweise mehrmals)
3. db2commexitUserIdentity für eine neue Verbindung
4. db2commexitRecv und db2commexitSend zur Verarbeitung von SQL-Anforderungen des Clients (möglicherweise mehrmals)
5. db2commexitRecv und db2commexitSend zur Authentifizierung (möglicherweise mehrmals)
6. db2commexitUserIdentity für eine neue Verbindung
7. db2commexitRecv und db2commexitSend zur Verarbeitung von SQL-Anforderungen des Clients (möglicherweise mehrmals)
8. db2commexitDeregister zum Beenden der Socketverbindung

Anmerkung:
db2commexitRegister und db2commexitDeregister werden jeweils nur einmal aufgerufen, auch wenn der Datenbankmanager zwei SQL-Verbindungen verarbeitet.

API-Aufrufreihenfolge - Gesicherter Kontext und Benutzerwechsel

Dieses Szenario ist mit einem Verbindungsaufbau ohne CONNECT RESET vergleichbar. Der Unterschied besteht darin, dass der Client einen Benutzerwechsel in einem gesicherten Kontext anfordert, anstatt eine neue SQL-CONNECT-Anforderung zu senden.

Die folgenden Aufrufe werden ausgeführt:

1. db2commexitRegister für eine neue Socketverbindung
2. db2commexitRecv und db2commexitSend zur Authentifizierung (möglicherweise mehrmals)
3. db2commexitUserIdentity für eine neue Verbindung
4. db2commexitRecv und db2commexitSend zur Verarbeitung von SQL-Anforderungen der Clients (möglicherweise mehrmals)
5. db2commexitRecv und db2commexitSend zur Authentifizierung (möglicherweise mehrmals)

   Zu einem späteren Zeitpunkt sendet der Client an den Server eine Anforderung für einen Benutzerwechsel in einem gesicherten Kontext, um den Benutzer für die Verbindung zu ändern.
6. db2commexitUserIdentity für einen Benutzerwechsel in einem gesicherten Kontext.
7. db2commexitRecv und db2commexitSend zur Verarbeitung von SQL-Anforderungen der Clients (möglicherweise mehrmals)
8. db2commexitDeregister zum Beenden der Socketverbindung

API-Aufrufreihenfolge - Verbindungskonzentrator

Dieses Szenario beschreibt die API-Aufrufreihenfolge bei Verwendung eines Verbindungskonzentrators. Mit der Funktion des Verbindungskonzentrators kann der Datenbankmanager erheblich mehr Clients verarbeiten, als koordinierende Agenten oder Threads zur Verfügung stehen.

Wenn ein Client eine UOW-Grenze erreicht und nicht sofort eine weitere Anforderung sendet, werden die Client-Sockets in einen inaktiven Pool versetzt. Der Agent, der zuvor die Clientanforderungen bearbeitet hat, fährt mit einem anderen Client fort. Sobald das inaktive Socket Daten lesen muss, sucht ein Dispatcher dafür einen

inaktiven Agenten. Während des Bestehens einer SQL-Verbindung wird die Clientanforderung möglicherweise von mehreren Agenten bearbeitet. Jedes Mal, wenn das Socket in den inaktiven Pool versetzt oder aus dem inaktiven Pool herausgenommen wird, werden `db2commexitDeregister` und `db2commexitRegister` aufgerufen. Die folgenden Aufrufe werden ausgeführt:

1. `db2commexitRegister` für eine neue Socketverbindung
2. `db2commexitRecv` und `db2commexitSend` zur Authentifizierung (möglicherweise mehrmals)
3. `db2commexitUserIdentity` für eine neue Verbindung
4. `db2commexitRecv` und `db2commexitSend` zur Verarbeitung von SQL-Anforderungen des Clients (möglicherweise mehrmals)

   Der Client sendet nicht umgehend eine weitere Anforderung und das Socket wird in einen inaktiven Pool versetzt.
5. `db2commexitDeregister` zum Aufheben der Zuordnung zum Agenten

   Zu einem späteren Zeitpunkt sendet der Client eine weitere Anforderung, woraufhin der Dispatcher einen inaktiven Agenten auswählt, der wahrscheinlich nicht der zuvor verwendete Agent ist:
6. `db2commexitRegister` für die Zuordnung eines Agenten
7. `db2commexitRecv` und `db2commexitSend` zur Verarbeitung von SQL-Anforderungen des Clients (möglicherweise mehrmals)
8. `db2commexitDeregister` zum Beenden der Socketverbindung

Anmerkung: Für eine einzelne SQL-Verbindung werden `db2commexitRegister` und `db2commexitDeregister` mehrmals aufgerufen.

API-Aufrufreihenfolge - Anweisung SET SESSION AUTHORIZATION

In diesem Szenario wird die API-Aufrufreihenfolge bei Verwendung der Anweisung SET SESSION AUTHORIZATION beschrieben.

Mit der Anweisung SET SESSION AUTHORIZATION wird die Sitzungsberechtigungs-ID für die aktuelle Verbindung geändert. 'Db2commexitUserIdentity' wird aufgerufen, um die Exitbibliothek für Kommunikationspuffer darüber zu informieren, dass die Identitätsinformationen für die aktuelle Verbindung geändert wurden. Die folgenden Aufrufe werden ausgeführt:

1. db2commexitRegister für eine neue Socketverbindung
2. db2commexitRecv und db2commexitSend zur Authentifizierung (möglicherweise mehrmals)
3. db2commexitUserIdentity für eine neue Verbindung
4. db2commexitRecv und db2commexitSend zur Verarbeitung von SQL-Anforderungen der Clients (möglicherweise mehrmals)

   Der Benutzer setzt die Anweisung SET SESSION AUTHORIZATION ab. Diese Anforderung wird an 'db2commexitRecv' übergeben. Diese Anweisung unterscheidet sich nicht von anderen SQL-Anweisungen.
5. db2commexitUserIdentity für die Anweisung SET SESSION AUTHORIZATION.
6. db2commexitRecv und db2commexitSend zur Verarbeitung von SQL-Anforderungen des Clients (möglicherweise mehrmals)
7. db2commexitDeregister zum Beenden der Socketverbindung

Hinweise zum Definieren des logischen Zielknotens

Beachten Sie beim Definieren des logischen Zielknotens mit der Variablen *DB2NODE* oder dem Befehl **SET CLIENT** die folgenden Hinweise.

Wenn der Client in einer Umgebung mit partitionierten Datenbanken ein Member über die Variable *DB2NODE* angibt, das nicht das für die Verbindung konfigurierte Member ist, wechselt der Datenbankmanager bei der Verbindung zu dem in der Variablen angegebenen neuen Member. Die Clientverbindung wird über das verbundene Member an das ferne Member weitergeleitet. In diesem Fall wird die Exitbibliothek für Kommunikationspuffer bei beiden Membern aufgerufen. Dabei sind einige Punkte zu beachten:

- Beim verbundenen Member gibt die Clientadresse den tatsächlichen Client an.
- Beim fernen Member gibt die Clientadresse das verbundene Member an.
- Die ID der Anwendung für abgehende Daten entspricht beim verbundenen Member der ID der Anwendung für ankommende Daten beim fernen Member.

Nach der Erstellung der Anwendungs-IDs wird **connectionType** in der Struktur `db2commexitCommInfo_v1` auf `GATEWAY` gesetzt.

Hinweise zum Connect-Gateway

Beachten Sie die folgenden Hinweise, wenn der Datenbankmanager als Connect-Gateway für einen anderen DRDA-Datenbankserver fungiert.

Wenn DB2 for Linux, UNIX and Windows als Connect-Gateway fungiert, wird die Exitbibliothek für Kommunikationspuffer auf dieselbe Weise aufgerufen wie eine Standardverbindung. Nach erfolgter Authentifizierung und Erstellung der Anwendungs-IDs wird **connectionType** in der Struktur `db2commexitCommInfo_v1` auf `GATEWAY` gesetzt. `outbound_application_id` stimmt mit der Anwendungs-ID für die Verbindung beim DRDA-Datenbankserver überein.

Hinweise zu DATA_ENCRYPT

Beachten Sie bei Verwendung des Authentifizierungstyps `DATA_ENCRYPT` die folgenden Hinweise.

Besonders hervorzuheben sind Kommunikationsaktivitäten, die durch den Authentifizierungstyp `DATA_ENCRYPT` geschützt sind. Im Gegensatz SSL erfolgt die erforderliche Verschlüsselung und Entschlüsselung zur Unterstützung von `DATA_ENCRYPT` durch den Datenbankmanager, nachdem Daten vom Client empfangen wurden und bevor eine Antwort an den Client gesendet wird.

Empfangen und DATA_ENCRYPT

Beim Empfang einer verschlüsselten DSS (Data Stream Structure) vom Client wird der Puffer gemäß den Erfordernissen des Datenbankmanagers entschlüsselt. Das heißt, dass nicht sofort der gesamte Puffer entschlüsselt wird. Die Exitbibliothek für Kommunikationspuffer wird während der Entschlüsselung mit den entschlüsselten Daten aufgerufen.

Die Länge der DSS bzw. die Länge der DSS-Fortsetzung, wenn die DSS länger als ein logischer Datensatz ist, enthält die Länge der verschlüsselten DSS und nicht die Länge des entschlüsselten Puffers. Da bei der Verschlüsselung immer eine Auffüllung hinzugefügt wird, übersteigt diese Länge immer die Länge des unverschlüsselten Texts. Die Länge der Auffüllung für die DSS beträgt maximal 8 Byte.

Bei Ausführung des letzten Aufrufs von db2CommexitRecv wird die Markierung *DB2COMMEXIT_RECV_IN_FLAG_END_DECRYPT* als Eingabe übergeben, um das Ende der verschlüsselten DSS anzugeben.

Anmerkung: Möglicherweise ist die Länge in diesem Fall 0, was darauf hinweist, dass die Auffüllung eine volle Blockgröße beträgt.

Senden und DATA_ENCRYPT

Bei der Verschlüsselung einer DSS-Antwort an den Client kann der Puffer, der an den Client gesendet wird, aus mehreren unverschlüsselten und verschlüsselten DSS bestehen. Bei der Vorbereitung dieser DSS werden diese als Eingabe an die Routine db2commexitSend übergeben. Diese Übergaben erfolgen jeweils nacheinander, da die unverschlüsselten Daten vor der Verschlüsselung als Eingabe verwendet werden müssen. Der Datenbankmanager stellt möglicherweise eine Fehlerbedingung fest, die erforderlich macht, dass zuvor vorbereitete, aber nicht gesendete DSS gelöscht werden müssen. Die Exitbibliothek für Kommunikationspuffer hat möglicherweise bereits Kenntnis über diese Bibliotheken. Die Funktion db2CommexitSend wird mit einer Länge von 0 und der Markierung DB2COMMEXIT_SEND_IN_FLAG_PURGE aufgerufen, um anzugeben, dass ein Löschvorgang erfolgt ist.

Kapitel 11. Aufbau der Datensätze der Prüffunktion

Jeder Prüfsatz, der aus dem Prüfprotokoll extrahiert wird, hat eines der in den folgenden Tabellen dargestellten Formate. Jeder Tabelle ist ein Beispieldatensatz vorangestellt.

Die einzelnen Einträge in den Prüfsätzen werden in der dazugehörigen Tabelle Zeile für Zeile beschrieben. Jeder Eintrag wird in der Tabelle in derselben Reihenfolge dargestellt, in der er auch in der Datei mit begrenzter Satzlänge nach der Extrahierungsoperation ausgegeben wird.

Anmerkung:

1. Abhängig vom Prüfereignis sind nicht in allen Feldern Prüfsätze enthalten. Wenn in einem Feld keine Werte enthalten sind, erscheint es nicht in der Prüfausgabe.
2. Einige Felder, wie z. B. „Access Attempted", werden als Bitmaps in begrenztem ASCII-Format gespeichert. In der hier gezeigten, unstrukturierten Berichtsdatei werden diese Felder jedoch durch Zeichenfolgen dargestellt, die die Bitmapwerte darstellen.

Prüfsatzobjekttypen

In der folgenden Tabelle wird für jeden Prüfsatzobjekttyp angegeben, ob er CHECKING-, OBJMAINT- und SECMAINT-Ereignisse generieren kann.

Tabelle 43. Prüfsatzobjekttypen auf der Basis der Prüfereignisse

Objekttyp	CHECKING-Ereignisse	OBJMAINT-Ereignisse	SECMAINT-Ereignisse
ACCESS_RULE			X
ALIAS	X	X	
ALL	X		
AUDIT_POLICY	X	X	
BUFFERPOOL	X	X	
CHECK_CONSTRAINT		X	
DATABASE	X		X
DATA TYPE		X	
EVENT_MONITOR	X	X	
FOREIGN_KEY		X	
FUNCTION	X	X	X
FUNCTION MAPPING	X	X	
GLOBAL_VARIABLE	X	X	X
HISTOGRAM TEMPLATE	X	X	
INDEX	X	X	X
INDEX EXTENSION		X	
INSTANCE	X		
JAR_FILE		X	

Tabelle 43. Prüfsatzobjekttypen auf der Basis der Prüfereignisse (Forts.)

Objekttyp	CHECKING-Ereignisse	OBJMAINT-Ereignisse	SECMAINT-Ereignisse
METHOD_BODY	X	X	X
MODULE	X	X	X
NICKNAME	X	X	X
NODEGROUP	X	X	
NONE	X	X	X
OPTIMIZATION PROFILE	X		
PACKAGE	X	X	X
PACKAGE CACHE	X		
PRIMARY_KEY		X	
REOPT_VALUES	X		
ROLE	X	X	X
SCHEMA	X	X	X
SECURITY LABEL		X	X
SECURITY LABEL COMPONENT		X	
SECURITY POLICY		X	X
SEQUENCE	X	X	
SERVER	X	X	X
SERVER OPTION	X	X	
SERVICE CLASS	X	X	
STORED_PROCEDURE	X	X	X
SUMMARY TABLES	X	X	X
TABLE	X	X	X
TABLESPACE	X	X	X
THRESHOLD	X	X	
TRIGGER		X	
TRUSTED CONTEXT	X	X	X
TYPE MAPPING	X	X	
TYPE&TRANSFORM	X	X	
UNIQUE_CONSTRAINT		X	
USER MAPPING	X	X	
USER_TEMPORARY_TABLE	X	X	X
VIEW	X	X	X
WORK ACTION SET	X	X	
WORK CLASS SET	X	X	
WORKLOAD	X	X	X
WRAPPER	X	X	
XSR-Objekt	X	X	X

Prüfsatzaufbau für AUDIT-Ereignisse

Die folgende Tabelle zeigt den Prüfsatzaufbau für AUDIT-Ereignisse.

Beispiel für einen Prüfsatz:

```
timestamp=2007-04-10-08.29.52.000001;
category=AUDIT;
audit event=START;
event correlator=0;
event status=0;
userid=newton;
authid=NEWTON;
application id=*LOCAL_APPLICATION;
application name=db2audit.exe;
```

Tabelle 44. Prüfsatzaufbau für AUDIT-Ereignisse

NAME	FORMAT	BESCHREIBUNG
Timestamp (Zeitmarke)	CHAR(26)	Datum und Uhrzeit des Prüfereignisses
Category (Kategorie)	CHAR(8)	Die Kategorie des Prüfereignisses. Mögliche Werte: AUDIT
Audit Event (Prüfereignis)	VARCHAR(32)	Spezifisches Prüfereignis. Eine Liste gültiger Werte finden Sie im Abschnitt über die Kategorie AUDIT unter „Prüfereignisse" auf Seite 376.
Event Correlator (Ereigniskorrelationswert)	INTEGER	Korrelationskennung für die geprüfte Operation. Kann zur Erkennung verwendet werden, welche Prüfsätze einem einzelnen Ereignis zugeordnet sind.
Event Status (Ereignisstatus)	INTEGER	Status des Prüfereignisses, dargestellt durch einen SQLCODE, für den Folgendes gilt: Erfolgreiches Ereignis > = 0 Fehlgeschlagenes Ereignis < 0
User ID (Benutzer-ID)	VARCHAR(1024)	Die Benutzer-ID zum Zeitpunkt des Prüfereignisses.
Authorization ID (Berechtigungs-ID)	VARCHAR(128)	Die Berechtigungs-ID zum Zeitpunkt des Prüfereignisses.
Original User ID (ursprüngliche Benutzer-ID)	VARCHAR(1024)	Der Wert der globalen Variablen CLIENT_ORIGUSERID zu dem Zeitpunkt, zu dem das Prüfereignis auftrat.
Database Name (Datenbankname)	CHAR(8)	Name der Datenbank, für die das Ereignis generiert wurde. Nicht belegt, wenn es sich um ein Prüfereignis auf Instanzebene handelt.
Origin Node Number (Nummer des Ursprungsknotens)	SMALLINT	Nummer des Members, für das das Prüfereignis aufgetreten ist.
Coordinator Node Number (Nummer des Koordinatorknotens)	SMALLINT	Nummer des Members, das als Koordinatormember definiert ist.
Application ID (Anwendungs-ID)	VARCHAR(255)	Die ID der Anwendung, die beim Auftreten des Prüfereignisses verwendet wurde.
Application Name (Anwendungsname)	VARCHAR(1024)	Der Name der Anwendung, die beim Auftreten des Prüfereignisses verwendet wurde.

Tabelle 44. Prüfsatzaufbau für AUDIT-Ereignisse (Forts.)

NAME	FORMAT	BESCHREIBUNG
Package Schema (Paketschema)	VARCHAR(128)	Das Schema des Pakets, das beim Auftreten des Prüfereignisses verwendet wurde.
Package Name (Paketname)	VARCHAR(128)	Der Name des Pakets, das beim Auftreten des Prüfereignisses verwendet wurde.
Package Section (Paketabschnitt)	SMALLINT	Die Abschnittsnummer innerhalb des Pakets, das beim Auftreten des Prüfereignisses verwendet wurde.
Package Version (Paketversion)	VARCHAR(64)	Die Version des Pakets, das beim Auftreten des Prüfereignisses verwendet wurde.
Local Transaction ID (Lokale Transaktions-ID)	VARCHAR(10) FOR BIT DATA	Die lokale Transaktions-ID, die beim Auftreten des Prüfereignisses verwendet wurde. Dies ist die Struktur SQLU_TID, die Teil der Transaktionsprotokolle ist.
Global Transaction ID (Globale Transaktions-ID)	VARCHAR(30) FOR BIT DATA	Die globale Transaktions-ID, die beim Auftreten des Prüfereignisses verwendet wurde. Dies ist das Datenfeld in der Struktur SQLP_GXID, die Teil der Transaktionsprotokolle ist.
Client User ID (Clientbenutzer-ID)	VARCHAR(255)	Der Wert des Sonderregisters CURRENT CLIENT USERID zu dem Zeitpunkt, als das Prüfereignis auftrat.
Client Workstation Name (Name der Client-Workstation)	VARCHAR(255)	Der Wert des Sonderregisters CURRENT CLIENT_WRKSTNNAME zu dem Zeitpunkt, als das Prüfereignis auftrat.
Client Application Name (Name der Clientanwendung)	VARCHAR(255)	Der Wert des Sonderregisters CURRENT CLIENT_APPLNAME zu dem Zeitpunkt, als das Prüfereignis auftrat.
Client Accounting String (Abrechnungs-zeichenfolge für Client)	VARCHAR(255)	Der Wert des Sonderregisters CURRENT CLIENT_ACCTNG zu dem Zeitpunkt, als das Prüfereignis auftrat.
Trusted Context Name (Name des gesicherten Kontexts)	VARCHAR(128)	Der Name des gesicherten Kontexts, der der gesicherten Verbindung zugeordnet ist.
Connection Trust Type (Typ der gesicherten Verbindung)	INTEGER	Mögliche Werte: IMPLICIT_TRUSTED_CONNECTION EXPLICIT_TRUSTED_CONNECTION
Role Inherited (Übernommene Rolle)	VARCHAR(128)	Die Rolle, die durch eine gesicherte Verbindung übernommen wurde.
Policy Name (Name der Richtlinie)	VARCHAR(128)	Der Name der Prüfrichtlinie.
Policy Association Object Type (Objekttyp der Richtlinien-zuordnung)	CHAR(1)	Der Typ des Objekts, dem die Prüfrichtlinie zugeordnet ist. Mögliche Werte: • N = Kurzname (Nickname) • S = MQT (Materialized Query Table) • T = Tabelle (nicht typisiert) • i = Berechtigungs-ID • g = Berechtigung • x = Gesicherter Kontext • leer = Datenbank

Tabelle 44. Prüfsatzaufbau für AUDIT-Ereignisse (Forts.)

NAME	FORMAT	BESCHREIBUNG
Policy Association Subobject Type (Subobjekttyp der Richtlinienzuordnung)	CHAR(1)	Der Typ des Subobjekts, dem die Prüfrichtlinie zugeordnet ist. Wenn der Objekttyp ? (Berechtigungs-ID) ist, sind folgende Werte möglich: • U = Benutzer (User) • G = Gruppe (Group) • R = Rolle (Role)
Policy Association Object Name (Objektname der Richtlinienzuordnung)	VARCHAR(128)	Der Name des Objekts, dem die Prüfrichtlinie zugeordnet ist.
Policy Association Object Schema (Objektschema der Richtlinienzuordnung)	VARCHAR(128)	Der Schemaname des Objekts, dem die Prüfrichtlinie zugeordnet ist. Dieser Wert ist NULL, wenn der Objekttyp der Richtlinienzuordnung ein Objekt angibt, für das kein Schema verwendet wird.
Audit Status (AUDIT-Status)	CHAR(1)	Der Status der Kategorie AUDIT in einer Prüfrichtlinie. Mögliche Werte: • B-Beide (Both) • F-Fehler (Failure) • N-Kein (None) • S-Erfolg (Success)
Checking Status (CHECKING-Status)	CHAR(1)	Der Status der Kategorie CHECKING in einer Prüfrichtlinie. Mögliche Werte: • B-Beide (Both) • F-Fehler (Failure) • N-Kein (None) • S-Erfolg (Success)
Context Status (CONTEXT-Status)	CHAR(1)	Der Status der Kategorie CONTEXT in einer Prüfrichtlinie. Mögliche Werte: • B-Beide (Both) • F-Fehler (Failure) • N-Kein (None) • S-Erfolg (Success)
Execute Status (EXECUTE-Status)	CHAR(1)	Der Status der Kategorie EXECUTE in einer Prüfrichtlinie. Mögliche Werte: • B-Beide (Both) • F-Fehler (Failure) • N-Kein (None) • S-Erfolg (Success)
Execute With Data (Ausführen mit Daten)	CHAR(1)	Die Option WITH DATA der Kategorie EXECUTE in der Prüfrichtlinie. Mögliche Werte: • Y-WITH DATA (mit Daten) • N-WITHOUT DATA (ohne Daten)

Tabelle 44. Prüfsatzaufbau für AUDIT-Ereignisse (Forts.)

NAME	FORMAT	BESCHREIBUNG
Objmaint Status (OBJMAINT-Status)	CHAR(1)	Der Status der Kategorie OBJMAINT in einer Prüfrichtlinie. Mögliche Werte: • B-Beide (Both) • F-Fehler (Failure) • N-Kein (None) • S-Erfolg (Success)
Secmaint Status (SECMAINT-Status)	CHAR(1)	Der Status der Kategorie SECMAINT in einer Prüfrichtlinie. Mögliche Werte: siehe Feld 'AUDIT-Status'.
Sysadmin Status (SYSADMIN-Status)	CHAR(1)	Der Status der Kategorie SYSADMIN in einer Prüfrichtlinie. Mögliche Werte: • B-Beide (Both) • F-Fehler (Failure) • N-Kein (None) • S-Erfolg (Success)
Validate Status (VALIDATE-Status)	CHAR(1)	Der Status der Kategorie VALIDATE in einer Prüfrichtlinie. Mögliche Werte: • B-Beide (Both) • F-Fehler (Failure) • N-Kein (None) • S-Erfolg (Success)
Error Type (Fehlertyp)	CHAR(8)	Der Fehlertyp in einer Prüfrichtlinie. Mögliche Werte: AUDIT und NORMAL.
Data Path (Datenpfad)	VARCHAR(1024)	Der Pfad zu den aktiven Prüfprotokollen, der im Befehl **db2audit configure** angegeben wurde.
Archive Path (Archivpfad)	VARCHAR(1024)	Der Pfad zu den archivierten Prüfprotokollen, der im Befehl **db2audit configure** angegeben wurde.

Prüfsatzaufbau für CHECKING-Ereignisse

Das Format des Prüfsatzes für CHECKING-Ereignisse wird in der folgenden Tabelle gezeigt.

Beispiel für einen Prüfsatz:

```
timestamp=1998-06-24-08.42.11.622984;
category=CHECKING;
audit event=CHECKING_OBJECT;
event correlator=2;
event status=0;
database=FOO;
userid=boss;
authid=BOSS;
application id=*LOCAL.newton.980624124210;
application name=testapp;
package schema=NULLID;
package name=SYSSH200;
package section=0;
object schema=GSTAGER;
object name=NONE;
object type=REOPT_VALUES;
access approval reason=DBADM;
access attempted=STORE;
```

Tabelle 45. Prüfsatzaufbau für CHECKING-Ereignisse

NAME	FORMAT	BESCHREIBUNG
Timestamp (Zeitmarke)	CHAR(26)	Datum und Uhrzeit des Prüfereignisses
Category (Kategorie)	CHAR(8)	Die Kategorie des Prüfereignisses. Mögliche Werte: CHECKING
Audit Event (Prüfereignis)	VARCHAR(32)	Spezifisches Prüfereignis. Eine Liste gültiger Werte finden Sie im Abschnitt über die Kategorie CHECKING unter „Prüfereignisse" auf Seite 376.
Event Correlator (Ereigniskorrelationswert)	INTEGER	Korrelationskennung für die geprüfte Operation. Kann zur Erkennung verwendet werden, welche Prüfsätze einem einzelnen Ereignis zugeordnet sind.
Event Status (Ereignisstatus)	INTEGER	Status des Prüfereignisses, dargestellt durch einen SQLCODE, für den Folgendes gilt: Erfolgreiches Ereignis > = 0 Fehlgeschlagenes Ereignis < 0
Database Name (Datenbankname)	CHAR(8)	Name der Datenbank, für die das Ereignis generiert wurde. Nicht belegt, wenn es sich um ein Prüfereignis auf Instanzebene handelt.
User ID (Benutzer-ID)	VARCHAR(1024)	Die Benutzer-ID zum Zeitpunkt des Prüfereignisses.
Authorization ID (Berechtigungs-ID)	VARCHAR(128)	Die Berechtigungs-ID zum Zeitpunkt des Prüfereignisses.
Original User ID (ursprüngliche Benutzer-ID)	VARCHAR(1024)	Der Wert der globalen Variablen CLIENT_ORIGUSERID zu dem Zeitpunkt, zu dem das Prüfereignis auftrat.
Origin Node Number (Nummer des Ursprungsknotens)	SMALLINT	Nummer des Members, für das das Prüfereignis aufgetreten ist.
Coordinator Node Number (Nummer des Koordinatorknotens)	SMALLINT	Nummer des Members, das als Koordinatormember definiert ist.
Application ID (Anwendungs-ID)	VARCHAR(255)	Die ID der Anwendung, die beim Auftreten des Prüfereignisses verwendet wurde.
Application Name (Anwendungsname)	VARCHAR(1024)	Der Name der Anwendung, die beim Auftreten des Prüfereignisses verwendet wurde.
Package Schema (Paketschema)	VARCHAR(128)	Das Schema des Pakets, das beim Auftreten des Prüfereignisses verwendet wurde.
Package Name (Paketname)	VARCHAR(128)	Der Name des Pakets, das beim Auftreten des Prüfereignisses verwendet wurde.
Package Section Number (Paketabschnittsnummer)	SMALLINT	Die Abschnittsnummer innerhalb des Pakets, das beim Auftreten des Prüfereignisses verwendet wurde.
Object Schema (Objektschema)	VARCHAR(128)	Das Schema des Objekts, für das das Prüfereignis generiert wurde.
Object Name (Objektname)	VARCHAR(128)	Der Name des Objekts, für das das Prüfereignis generiert wurde.
Object Type (Objekttyp)	VARCHAR(32)	Der Typ des Objekts, für das das Prüfereignis generiert wurde. Mögliche Werte: siehe Abschnitt „Prüfsatzobjekttypen".

Tabelle 45. Prüfsatzaufbau für CHECKING-Ereignisse (Forts.)

NAME	FORMAT	BESCHREIBUNG
Access Approval Reason (Grund für Zugriffgewährung)	CHAR(18)	Gibt an, warum der Zugriff für dieses Prüfereignis gewährt wurde. Mögliche Werte: siehe Abschnitt „Gründe für die Zugriffsgewährung bei CHECKING-Ereignissen".
Access Attempted (Zugriffsversuch)	CHAR(18)	Gibt an, welche Art von Zugriff versucht wurde. Gültige Werte: siehe Abschnitt „Typen von Zugriffsversuchen bei CHECKING-Ereignissen".
Package Version (Paketversion)	VARCHAR (64)	Die Version des Pakets, das beim Auftreten des Prüfereignisses verwendet wurde.
Checked Authorization ID (Überprüfte Berechtigungs-ID)	VARCHAR(128)	Die Berechtigungs-ID wird überprüft, wenn sie sich von der Berechtigungs-ID beim Prüfereignis unterscheidet. Dabei kann es sich zum Beispiel um den Zieleigner in der Anweisung TRANSFER OWNERSHIP handeln. Wenn das Prüfereignis SWITCH_USER ist, stellt dieses Feld die Berechtigungs-ID dar, zu der gewechselt wird.
Local Transaction ID (Lokale Transaktions-ID)	VARCHAR(10) FOR BIT DATA	Die lokale Transaktions-ID, die beim Auftreten des Prüfereignisses verwendet wurde. Dies ist die Struktur SQLU_TID, die Teil der Transaktionsprotokolle ist.
Global Transaction ID (Globale Transaktions-ID)	VARCHAR(30) FOR BIT DATA	Die globale Transaktions-ID, die beim Auftreten des Prüfereignisses verwendet wurde. Dies ist das Datenfeld in der Struktur SQLP_GXID, die Teil der Transaktionsprotokolle ist.
Client User ID (Clientbenutzer-ID)	VARCHAR(255)	Der Wert des Sonderregisters CURRENT CLIENT USERID zu dem Zeitpunkt, als das Prüfereignis auftrat.
Client Workstation Name (Name der Client-Workstation)	VARCHAR(255)	Der Wert des Sonderregisters CURRENT CLIENT_WRKSTNNAME zu dem Zeitpunkt, als das Prüfereignis auftrat.
Client Application Name (Name der Clientanwendung)	VARCHAR(255)	Der Wert des Sonderregisters CURRENT CLIENT_APPLNAME zu dem Zeitpunkt, als das Prüfereignis auftrat.
Client Accounting String (Abrechnungs-zeichenfolge für Client)	VARCHAR(255)	Der Wert des Sonderregisters CURRENT CLIENT_ACCTNG zu dem Zeitpunkt, als das Prüfereignis auftrat.
Trusted Context Name (Name des gesicherten Kontexts)	VARCHAR(128)	Der Name des gesicherten Kontexts, der der gesicherten Verbindung zugeordnet ist.
Connection Trust Type (Typ der gesicherten Verbindung)	INTEGER	Mögliche Werte: IMPLICIT_TRUSTED_CONNECTION EXPLICIT_TRUSTED_CONNECTION
Role Inherited (Übernommene Rolle)	VARCHAR(128)	Die Rolle, die durch eine gesicherte Verbindung übernommen wurde.

Gründe für die Zugriffsgewährung bei CHECKING-Ereignissen

Die folgende Liste zeigt die möglichen Gründe der Zugriffsgewährung für CHECKING-Ereignisse.

Es ist zu beachten, dass Prüfsätze teilweise vorgeben, dass mehrere Gründe für die Zugriffsgewährung (access approval reason) erfüllt sein müssen, z. B.: `access approval reason=DATAACCESS,ACCESSCTRL;`. Sind mehrere Gründe für die Zugriffsgewährung angegeben, muss der Benutzer über alle angegebenen Berechtigungen und Zugriffsrechte verfügen, um die Berechtigungsprüfung für den versuchten Zugriff bestehen zu können.

0x0000000000000001 ACCESS DENIED
Der Zugriff wird nicht gewährt, sondern verweigert.

0x0000000000000002 SYSADM
Der Zugriff wird gewährt. Die Anwendung oder der Benutzer verfügt über die Berechtigung SYSADM.

0x0000000000000004 SYSCTRL
Der Zugriff wird gewährt. Die Anwendung oder der Benutzer verfügt über die Berechtigung SYSCTRL.

0x0000000000000008 SYSMAINT
Der Zugriff wird gewährt. Die Anwendung oder der Benutzer verfügt über die Berechtigung SYSMAINT.

0x0000000000000010 DBADM
Der Zugriff wird gewährt. Die Anwendung oder der Benutzer verfügt über die Berechtigung DBADM.

0x0000000000000020 DATABASE
Der Zugriff wird gewährt. Die Anwendung oder der Benutzer verfügt über ein explizites Zugriffsrecht für die Datenbank.

0x0000000000000040 OBJECT
Der Zugriff wird gewährt. Die Anwendung oder der Benutzer verfügt über ein Zugriffsrecht für das Objekt oder die Funktion.

0x0000000000000080 DEFINER
Der Zugriff wird gewährt. Die Anwendung oder der Benutzer ist der definierende Benutzer bzw. die definierende Anwendung für das Objekt oder die Funktion.

0x0000000000000100 OWNER
Der Zugriff wird gewährt. Die Anwendung oder der Benutzer ist der Eigner des Objekts oder der Funktion.

0x0000000000000200 CONTROL
Der Zugriff wird gewährt. Die Anwendung oder der Benutzer verfügt über das Zugriffsrecht CONTROL für das Objekt oder die Funktion.

0x0000000000000400 BIND
Der Zugriff wird gewährt. Die Anwendung oder der Benutzer verfügt über das Zugriffsrecht BIND für das Paket.

0x0000000000000800 SYSQUIESCE
Der Zugriff wird gewährt. Wenn sich die Instanz oder die Datenbank im Quiescemodus befindet, kann die Anwendung bzw. der Benutzer eine Verbindung (CONNECT oder ATTACH) herstellen.

0x0000000000001000 SYSMON
Der Zugriff wird gewährt. Die Anwendung oder der Benutzer verfügt über die Berechtigung SYSMON.

0x0000000000002000 SECADM
Der Zugriff wird gewährt. Die Anwendung oder der Benutzer verfügt über die Berechtigung SECADM.

0x0000000000004000 SETSESSIONUSER
Der Zugriff wird gewährt. Die Anwendung oder der Benutzer verfügt über die Berechtigung SETSESSIONUSER.

0x0000000000008000 TRUSTED_CONTEXT_MATCH
Verbindungsattribute stimmten mit den Attributen eines eindeutigen gesicherten Kontexts auf dem DB2-Server überein.

0x0000000000010000 TRUSTED_CONTEXT_USE
Zugriff wird zur Verwendung des gesicherten Kontexts gewährt.

0x0000000000020000 SQLADM
Der Zugriff wird gewährt. Die Anwendung oder der Benutzer verfügt über die Berechtigung SQLADM.

0x0000000000040000 WLMADM
Der Zugriff wird gewährt. Die Anwendung oder der Benutzer verfügt über die Berechtigung WLMADM.

0x0000000000080000 EXPLAIN
Der Zugriff wird gewährt. Die Anwendung oder der Benutzer verfügt über die Berechtigung EXPLAIN.

0x0000000000100000 DATAACCESS
Der Zugriff wird gewährt. Die Anwendung oder der Benutzer verfügt über die Berechtigung DATAACCESS.

0x0000000000200000 ACCESSCTRL
Der Zugriff wird gewährt. Die Anwendung oder der Benutzer verfügt über die Berechtigung ACCESSCTRL.

Typen von Zugriffsversuchen bei CHECKING-Ereignissen

Die folgende Liste zeigt die möglichen Typen von Zugriffsversuchen für CHECKING-Ereignisse.

Wenn das Prüfereignis CHECKING_TRANSFER ist, gibt der Prüfeintrag an, ob ein Zugriffsrecht enthalten ist oder nicht.

0x0000000000000001 CONTROL
Es wurde versucht, zu prüfen, ob das Zugriffsrecht CONTROL erteilt wurde.

0x0000000000000002 ALTER
Es wurde versucht, ein Objekt zu ändern bzw. zu prüfen, ob das Zugriffsrecht ALTER erteilt wurde, wenn das Prüfereignis CHECKING_TRANSFER lautet.

0x0000000000000004 DELETE
Es wurde versucht, ein Objekt zu löschen bzw. zu prüfen, ob das Zugriffsrecht DELETE erteilt wurde, wenn das Prüfereignis CHECKING_TRANSFER lautet.

0x0000000000000008 INDEX
Es wurde versucht, einen Index zu verwenden bzw. zu prüfen, ob das Zugriffsrecht INDEX erteilt wurde, wenn das Prüfereignis CHECKING_TRANSFER lautet.

0x0000000000000010 INSERT
Es wurde ein Objekteinfügevorgang versucht bzw. es wurde versucht, zu prüfen, ob das Zugriffsrecht INSERT erteilt wurde, wenn das Prüfereignis CHECKING_TRANSFER lautet.

0x0000000000000020 SELECT
Es wurde versucht, eine Tabelle oder eine Sicht abzufragen bzw. zu prüfen, ob das Zugriffsrecht SELECT erteilt wurde, wenn das Prüfereignis CHECKING_TRANSFER lautet.

0x0000000000000040 UPDATE
Es wurde versucht, Daten in einem Objekt zu aktualisieren bzw. zu prüfen, ob das Zugriffsrecht UPDATE erteilt wurde, wenn das Prüfereignis CHECKING_TRANSFER lautet.

0x0000000000000080 REFERENCE
Es wurde versucht, referenzielle Integritätsbedingungen zwischen Objekten zu erstellen bzw. zu prüfen, ob das Zugriffsrecht REFERENCE erteilt wurde, wenn das Prüfereignis CHECKING_TRANSFER lautet.

0x0000000000000100 CREATE
Es wurde versucht, ein Objekt zu erstellen.

0x0000000000000200 DROP
Es wurde versucht, ein Objekt zu löschen.

0x0000000000000400 CREATEIN
Es wurde versucht, ein Objekt in einem anderen Schema zu erstellen.

0x0000000000000800 DROPIN
Es wurde versucht, ein in einem anderen Schema gefundenes Objekt zu löschen.

0x0000000000001000 ALTERIN
Es wurde versucht, ein in einem anderen Schema gefundenes Objekt zu ändern.

0x0000000000002000 EXECUTE
Es wurde versucht, eine Anwendung auszuführen oder eine Routine aufzurufen, eine Funktion aus der Routine als Quelle (gilt nur für Funktionen) zu erstellen oder auf eine Routine in einer DDL-Anweisung zu verweisen bzw. zu prüfen, ob das Zugriffsrecht EXECUTE erteilt wurde, wenn das Prüfereignis CHECKING_TRANSFER lautet.

0x0000000000004000 BIND
Es wurde versucht, eine Anwendung zu binden oder vorzubereiten.

0x0000000000008000 SET EVENT MONITOR
Es wurde versucht, Ereignismonitorschalter zu setzen.

0x0000000000010000 SET CONSTRAINTS
Es wurde versucht, Integritätsbedingungen für ein Objekt zu definieren.

0x0000000000020000 COMMENT ON
Es wurde versucht, Kommentare zu einem Objekt zu erstellen.

0x0000000000040000 GRANT
Es wurde versucht, einer anderen Berechtigungs-ID Zugriffsrechte oder Rollen für ein Objekt zu erteilen.

0x0000000000080000 REVOKE
Es wurde versucht, einer Berechtigungs-ID Zugriffsrechte oder Rollen für ein Objekt zu entziehen.

0x0000000000100000 LOCK
Es wurde versucht, ein Objekt zu sperren.

0x0000000000200000 RENAME
Es wurde versucht, ein Objekt umzubenennen.

0x0000000000400000 CONNECT
Es wurde versucht, eine Verbindung zu einer Datenbank herzustellen.

0x0000000000800000 Member of SYS Group
Es wurde versucht, auf ein Mitglied der SYS-Gruppe zuzugreifen oder ein Mitglied der SYS-Gruppe zu verwenden.

0x0000000001000000 Access All
Es wurde versucht, eine Anweisung mit allen erforderlichen Zugriffsrechten für gespeicherte Objekte auszuführen (wird nur für DBADM/SYSADM verwendet).

0x0000000002000000 Drop All
Es wurde versucht, mehrere Objekte zu löschen.

0x0000000004000000 LOAD
Es wurde versucht, eine Tabelle in einen Tabellenbereich zu laden.

0x0000000008000000 USE
Es wurde versucht, eine Tabelle in einem Tabellenbereich zu erstellen bzw. zu prüfen, ob das Zugriffsrecht USE erteilt wurde, wenn das Prüfereignis CHECKING_TRANSFER lautet.

0x0000000010000000 SET SESSION_USER
Es wurde versucht, eine Anweisung SET SESSION_USER auszuführen.

0x0000000020000000 FLUSH
Es wurde versucht, eine Anweisung FLUSH auszuführen.

0x0000000040000000 STORE
Es wurde versucht, die Werte einer reoptimierten Anweisung in der Tabelle EXPLAIN_PREDICATE anzuzeigen.

0x0000000400000000 TRANSFER
Es wurde versucht, ein Objekt zu übertragen.

0x0000000800000000 ALTER_WITH_GRANT
Es wurde versucht, zu prüfen, ob das Zugriffsrecht ALTER with GRANT erteilt wurde.

0x0000001000000000 DELETE_WITH_GRANT
Es wurde versucht, zu prüfen, ob das Zugriffsrecht DELETE with GRANT erteilt wurde.

0x0000002000000000 INDEX_WITH_GRANT
Es wurde versucht, zu prüfen, ob das Zugriffsrecht INDEX with GRANT erteilt wurde.

0x0000004000000000 INSERT_WITH_GRANT
Es wurde versucht, zu prüfen, ob das Zugriffsrecht INSERT with GRANT erteilt wurde.

0x0000008000000000 SELECT_WITH_GRANT
Es wurde versucht, zu prüfen, ob das Zugriffsrecht SELECT with GRANT erteilt wurde.

0x0000010000000000 UPDATE_WITH_GRANT
Es wurde versucht, zu prüfen, ob das Zugriffsrecht UPDATE with GRANT erteilt wurde.

0x0000020000000000 REFERENCE_WITH_GRANT
Es wurde versucht, zu prüfen, ob das Zugriffsrecht REFERENCE with GRANT erteilt wurde.

0x0000040000000000 USAGE
Es wurde versucht, eine Reihenfolge oder ein XSR-Objekt zu verwenden bzw. zu prüfen, ob das Zugriffsrecht USAGE erteilt wurde, wenn das Prüfereignis CHECKING_TRANSFER ist.

0x0000080000000000 SET ROLE
Es wurde versucht, eine Rolle festzulegen.

0x0000100000000000 EXPLICIT_TRUSTED_CONNECTION
Es wurde versucht, eine explizite gesicherte Verbindung herzustellen.

0x0000200000000000 IMPLICIT_TRUSTED_CONNECTION
Es wurde versucht, eine implizite gesicherte Verbindung herzustellen.

0x0000400000000000 READ
Es wurde versucht, eine globale Variable zu lesen.

0x0000800000000000 WRITE
Es wurde versucht, eine globale Variable zu schreiben.

0x0001000000000000 SWITCH_USER
Es wurde versucht, eine Benutzer-ID über eine explizite gesicherte Verbindung zu wechseln.

0x0002000000000000 AUDIT_USING
Es wurde versucht, einem Objekt eine Prüfrichtlinie zuzuordnen.

0x0004000000000000 AUDIT_REPLACE
Es wurde versucht, eine Prüfrichtlinienzuordnung eines Objekts zu ersetzen.

0x0008000000000000 AUDIT_REMOVE
Es wurde versucht, eine Prüfrichtlinienzuordnung eines Objekts zu entfernen.

0x0010000000000000 AUDIT_ARCHIVE
Es wurde versucht, das Prüfprotokoll zu archivieren.

0x0020000000000000 AUDIT_EXTRACT
Es wurde versucht, das Prüfprotokoll zu extrahieren.

0x0040000000000000 AUDIT_LIST_LOGS
Es wurde versucht, die Prüfprotokolle aufzulisten.

0x0080000000000000 IGNORE_TRIGGERS
Es wurde versucht, die einem Datenbankobjekt zugeordneten Trigger zu ignorieren.

0x0100000000000000 PREPARE
Es wurde versucht, eine SQL-Anweisung vorzubereiten, obwohl der Benutzer nicht über das erforderliche Zugriffsrecht der Objektebene oder die Berechtigung DATAACCESS verfügt.

0x0200000000000000 DESCRIBE
Es wurde versucht, eine Anweisung zu beschreiben, obwohl der Benutzer nicht über das erforderliche Zugriffsrecht der Objektebene oder die Berechtigung DATAACCESS verfügt.

Prüfsatzaufbau für OBJMAINT-Ereignisse

Das Format des Prüfsatzes für OBJMAINT-Ereignisse wird in der folgenden Tabelle gezeigt.

Beispiel für einen Prüfsatz:

```
timestamp=1998-06-24-08.42.41.957524;
category=OBJMAINT;
audit event=CREATE_OBJECT;
event correlator=3;
event status=0;
database=FOO;
userid=boss;
authid=BOSS;
application id=*LOCAL.newton.980624124210;
application name=testapp;
package schema=NULLID;
package name=SQLC28A1;
package section=0;
object schema=BOSS;
object name=AUDIT;
object type=TABLE;
```

Tabelle 46. Prüfsatzaufbau für OBJMAINT-Ereignisse

NAME	FORMAT	BESCHREIBUNG
Timestamp (Zeitmarke)	CHAR(26)	Datum und Uhrzeit des Prüfereignisses
Category (Kategorie)	CHAR(8)	Die Kategorie des Prüfereignisses. Mögliche Werte: OBJMAINT
Audit Event (Prüfereignis)	VARCHAR(32)	Spezifisches Prüfereignis. Eine Liste gültiger Werte finden Sie im Abschnitt über die Kategorie OBJMAINT unter „Prüfereignisse” auf Seite 376.
Event Correlator (Ereigniskorrelationswert)	INTEGER	Korrelationskennung für die geprüfte Operation. Kann zur Erkennung verwendet werden, welche Prüfsätze einem einzelnen Ereignis zugeordnet sind.
Event Status (Ereignisstatus)	INTEGER	Status des Prüfereignisses, dargestellt durch einen SQLCODE, für den Folgendes gilt: Erfolgreiches Ereignis > = 0 Fehlgeschlagenes Ereignis < 0
Database Name (Datenbankname)	CHAR(8)	Name der Datenbank, für die das Ereignis generiert wurde. Nicht belegt, wenn es sich um ein Prüfereignis auf Instanzebene handelt.
User ID (Benutzer-ID)	VARCHAR(1024)	Die Benutzer-ID zum Zeitpunkt des Prüfereignisses.
Authorization ID (Berechtigungs-ID)	VARCHAR(128)	Die Berechtigungs-ID zum Zeitpunkt des Prüfereignisses.
Original User ID (ursprüngliche Benutzer-ID)	VARCHAR(1024)	Der Wert der globalen Variablen CLIENT_ORIGUSERID zu dem Zeitpunkt, zu dem das Prüfereignis auftrat.
Origin Node Number (Nummer des Ursprungsknotens)	SMALLINT	Nummer des Members, für das das Prüfereignis aufgetreten ist.

Tabelle 46. Prüfsatzaufbau für OBJMAINT-Ereignisse (Forts.)

NAME	FORMAT	BESCHREIBUNG
Coordinator Node Number (Nummer des Koordinatorknotens)	SMALLINT	Nummer des Members, das als Koordinatormember definiert ist.
Application ID (Anwendungs-ID)	VARCHAR(255)	Die ID der Anwendung, die beim Auftreten des Prüfereignisses verwendet wurde.
Application Name (Anwendungsname)	VARCHAR(1024)	Der Name der Anwendung, die beim Auftreten des Prüfereignisses verwendet wurde.
Package Schema (Paketschema)	VARCHAR(128)	Das Schema des Pakets, das beim Auftreten des Prüfereignisses verwendet wurde.
Package Name (Paketname)	VARCHAR(256)	Der Name des Pakets, das beim Auftreten des Prüfereignisses verwendet wurde.
Package Section Number (Paketabschnittsnummer)	SMALLINT	Die Abschnittsnummer innerhalb des Pakets, das beim Auftreten des Prüfereignisses verwendet wurde.
Object Schema (Objektschema)	VARCHAR(128)	Das Schema des Objekts, für das das Prüfereignis generiert wurde.
Object Name (Objektname)	VARCHAR(128)	Der Name des Objekts, für das das Prüfereignis generiert wurde.
Object Type (Objekttyp)	VARCHAR(32)	Der Typ des Objekts, für das das Prüfereignis generiert wurde. Mögliche Werte: siehe Abschnitt „Prüfsatzobjekttypen".
Package Version (Paketversion)	VARCHAR(64)	Die Version des Pakets, das beim Auftreten des Prüfereignisses verwendet wurde.
Security Policy Name (Name der Sicherheitsrichtlinie)	VARCHAR(128)	Der Name der Sicherheitsrichtlinie, wenn der Objekttyp TABLE ist und dieser Tabelle eine Sicherheitsrichtlinie zugeordnet ist.
Alter Action (ALTER-Aktion)	VARCHAR(32)	Die spezielle ALTER-Operation. Mögliche Werte: • ADD_PROTECTED_COLUMN (Geschützte Spalte hinzufügen) • ADD_COLUMN_PROTECTION (Spaltenschutz hinzufügen) • DROP_COLUMN_PROTECTION (Spaltenschutz löschen) • ADD_ROW_PROTECTION (Zeilenschutz hinzufügen) • ADD_SECURITY_POLICY (Sicherheitsrichtlinie hinzufügen) • ADD_ELEMENT (Element hinzufügen) • ADD COMPONENT (Komponente hinzufügen) • USE GROUP AUTHORIZATIONS (Gruppenberechtigungen verwenden) • IGNORE GROUP AUTHORIZATIONS (Gruppenberechtigungen ignorieren) • USE ROLE AUTHORIZATIONS (Rollenberechtigungen verwenden) • IGNORE ROLE AUTHORIZATIONS (Rollenberechtigungen ignorieren) • OVERRIDE NOT AUTHORIZED WRITE SECURITY LABEL (Nicht berechtigten Sicherheitskennsatz überschreiben) • RESTRICT NOT AUTHORIZED WRITE SECURITY LABEL (Nicht berechtigten Sicherheitskennsatz einschränken)

Tabelle 46. Prüfsatzaufbau für OBJMAINT-Ereignisse (Forts.)

NAME	FORMAT	BESCHREIBUNG
Protected Column Name (Name der geschützten Spalte)	VARCHAR(128)	Wenn 'Alter Action' den Wert ADD_COLUMN_PROTECTION oder DROP_COLUMN_PROTECTION hat, ist dies der Name der betroffenen Spalte.
Column Security Label (Sicherheitskennsatz der Spalte)	VARCHAR(128)	Der Sicherheitskennsatz, der die Spalte schützt, die im Feld 'Column Name' (Spaltenname) angegeben ist.
Security Label Column Name (Name der Spalte mit dem Sicherheitskennsatz)	VARCHAR(128)	Der Name der Spalte, die den Sicherheitskennsatz enthält, der die Zeile schützt.
Local Transaction ID (Lokale Transaktions-ID)	VARCHAR(10) FOR BIT DATA	Die lokale Transaktions-ID, die beim Auftreten des Prüfereignisses verwendet wurde. Dies ist die Struktur SQLU_TID, die Teil der Transaktionsprotokolle ist.
Global Transaction ID (Globale Transaktions-ID)	VARCHAR(30) FOR BIT DATA	Die globale Transaktions-ID, die beim Auftreten des Prüfereignisses verwendet wurde. Dies ist das Datenfeld in der Struktur SQLP_GXID, die Teil der Transaktionsprotokolle ist.
Client User ID (Clientbenutzer-ID)	VARCHAR(255)	Der Wert des Sonderregisters CURRENT CLIENT USERID zu dem Zeitpunkt, als das Prüfereignis auftrat.
Client Workstation Name (Name der Client-Workstation)	VARCHAR(255)	Der Wert des Sonderregisters CURRENT CLIENT_WRKSTNNAME zu dem Zeitpunkt, als das Prüfereignis auftrat.
Client Application Name (Name der Clientanwendung)	VARCHAR(255)	Der Wert des Sonderregisters CURRENT CLIENT_APPLNAME zu dem Zeitpunkt, als das Prüfereignis auftrat.
Client Accounting String (Abrechnungszeichenfolge für Client)	VARCHAR(255)	Der Wert des Sonderregisters CURRENT CLIENT_ACCTNG zu dem Zeitpunkt, als das Prüfereignis auftrat.
Trusted Context Name (Name des gesicherten Kontexts)	VARCHAR(128)	Der Name des gesicherten Kontexts, der der gesicherten Verbindung zugeordnet ist.
Connection Trust Type (Typ der gesicherten Verbindung)	INTEGER	Mögliche Werte: IMPLICIT_TRUSTED_CONNECTION EXPLICIT_TRUSTED_CONNECTION
Role Inherited (Übernommene Rolle)	VARCHAR(128)	Die Rolle, die durch eine gesicherte Verbindung übernommen wurde.
Object Module (Objektmodul)	VARCHAR(128)	Der Name des Moduls, zu dem das Objekt gehört.

Prüfsatzaufbau für SECMAINT-Ereignisse

Das Format des Prüfsatzes für SECMAINT-Ereignisse wird in der folgenden Tabelle gezeigt.

Beispiel für einen Prüfsatz:

```
timestamp=1998-06-24-11.57.45.188101;
category=SECMAINT;
audit event=GRANT;
event correlator=4;
event status=0;
database=FOO;
userid=boss;
authid=BOSS;
application id=*LOCAL.boss.980624155728;
application name=db2bp;
package schema=NULLID;
package name=SQLC28A1;
package section=0;
object schema=BOSS;
object name=T1;
object type=TABLE;
grantor=BOSS;
grantee=WORKER;
grantee type=USER;
privilege=SELECT;
```

Tabelle 47. Prüfsatzaufbau für SECMAINT-Ereignisse

NAME	FORMAT	BESCHREIBUNG
Timestamp (Zeitmarke)	CHAR(26)	Datum und Uhrzeit des Prüfereignisses
Category (Kategorie)	CHAR(8)	Die Kategorie des Prüfereignisses. Mögliche Werte: SECMAINT
Audit Event (Prüfereignis)	VARCHAR(32)	Spezifisches Prüfereignis. Eine Liste gültiger Werte finden Sie im Abschnitt über die Kategorie SECMAINT unter „Prüfereignisse" auf Seite 376.
Event Correlator (Ereigniskorrelationswert)	INTEGER	Korrelationskennung für die geprüfte Operation. Kann zur Erkennung verwendet werden, welche Prüfsätze einem einzelnen Ereignis zugeordnet sind.
Event Status (Ereignisstatus)	INTEGER	Status des Prüfereignisses, dargestellt durch einen SQLCODE, für den Folgendes gilt: Erfolgreiches Ereignis > = 0 Fehlgeschlagenes Ereignis < 0
Database Name (Datenbankname)	CHAR(8)	Name der Datenbank, für die das Ereignis generiert wurde. Nicht belegt, wenn es sich um ein Prüfereignis auf Instanzebene handelt.
User ID (Benutzer-ID)	VARCHAR(1024)	Die Benutzer-ID zum Zeitpunkt des Prüfereignisses.
Authorization ID (Berechtigungs-ID)	VARCHAR(128)	Die Berechtigungs-ID zum Zeitpunkt des Prüfereignisses.
Original User ID (ursprüngliche Benutzer-ID)	VARCHAR(1024)	Der Wert der globalen Variablen CLIENT_ORIGUSERID zu dem Zeitpunkt, zu dem das Prüfereignis auftrat.
Origin Node Number (Nummer des Ursprungsknotens)	SMALLINT	Nummer des Members, für das das Prüfereignis aufgetreten ist.

Tabelle 47. Prüfsatzaufbau für SECMAINT-Ereignisse (Forts.)

NAME	FORMAT	BESCHREIBUNG
Coordinator Node Number (Nummer des Koordinatorknotens)	SMALLINT	Nummer des Members, das als Koordinatormember definiert ist.
Application ID (Anwendungs-ID)	VARCHAR(255)	Die ID der Anwendung, die beim Auftreten des Prüfereignisses verwendet wurde.
Application Name (Anwendungsname)	VARCHAR(1024)	Der Name der Anwendung, die beim Auftreten des Prüfereignisses verwendet wurde.
Package Schema (Paketschema)	VARCHAR(128)	Das Schema des Pakets, das beim Auftreten des Prüfereignisses verwendet wurde.
Package Name (Paketname)	VARCHAR(128)	Der Name des Pakets, das beim Auftreten des Prüfereignisses verwendet wurde.
Package Section Number (Paketabschnittsnummer)	SMALLINT	Die Abschnittsnummer innerhalb des Pakets, das beim Auftreten des Prüfereignisses verwendet wurde.
Object Schema (Objektschema)	VARCHAR(128)	Das Schema des Objekts, für das das Prüfereignis generiert wurde. Wenn das Objekttypfeld den Wert ACCESS_RULE hat, enthält dieses Feld den Namen der Sicherheitsrichtlinie, die der Regel zugeordnet ist. Der Name der Regel wird im Feld für den Objektnamen gespeichert. Wenn das Objekttypfeld den Wert SECURITY_LABEL hat, enthält dieses Feld den Namen der Sicherheitsrichtlinie, zu der der Sicherheitskennsatz gehört. Der Name des Sicherheitskennsatzes wird im Feld für den Objektnamen gespeichert.
Object Name (Objektname)	VARCHAR(128)	Der Name des Objekts, für das das Prüfereignis generiert wurde. Dabei handelt es sich um einen Rollennamen, wenn es sich bei dem Prüfereignis um eines der folgenden Ereignisse handelt: • ADD_DEFAULT_ROLE • DROP_DEFAULT_ROLE • ALTER_DEFAULT_ROLE • ADD_USER • DROP_USER • ALTER_USER_ADD_ROLE • ALTER_USER_DROP_ROLE • ALTER_USER_AUTHENTICATION Wenn das Objekttypfeld den Wert ACCESS_RULE hat, enthält dieses Feld den Namen der Regel. Der Name der Sicherheitsrichtlinie, die der Regel zugeordnet ist, wird im Feld für das Objektschema gespeichert. Wenn das Objekttypfeld den Wert SECURITY_LABEL hat, enthält dieses Feld den Namen des Sicherheitskennsatzes. Der Name der Sicherheitsrichtlinie, der er zugeordnet ist, wird im Feld für das Objektschema gespeichert.

Tabelle 47. Prüfsatzaufbau für SECMAINT-Ereignisse (Forts.)

NAME	FORMAT	BESCHREIBUNG
Object Type (Objekttyp)	VARCHAR(32)	Der Typ des Objekts, für das das Prüfereignis generiert wurde. Mögliche Werte: siehe Abschnitt „Prüfsatzobjekttypen". Es ergibt sich der Wert ROLE, wenn es sich bei dem Prüfereignis um eines der folgenden Ereignisse handelt: • ADD_DEFAULT_ROLE • DROP_DEFAULT_ROLE • ALTER_DEFAULT_ROLE • ADD_USER • DROP_USER • ALTER_USER_ADD_ROLE • ALTER_USER_DROP_ROLE • ALTER_USER_AUTHENTICATION
Grantor (Berechtigungsgeber)	VARCHAR(128)	Die ID des Benutzers, der das Zugriffsrecht bzw. die Berechtigung erteilt oder entzogen hat.
Grantee (Berechtigter)	VARCHAR(128)	Die ID des Berechtigten, dem ein Zugriffsrecht oder eine Berechtigung erteilt oder entzogen wurde. Dabei handelt es sich um ein Objekt für gesicherte Kontexte, wenn es sich bei dem Prüfereignis um eines der folgenden Ereignisse handelt: • ADD_DEFAULT_ROLE • DROP_DEFAULT_ROLE • ALTER_DEFAULT_ROLE • ADD_USER, DROP_USER • ALTER_USER_ADD_ROLE • ALTER_USER_DROP_ROLE • ALTER_USER_AUTHENTICATION
Grantee Type (Typ des Berechtigten)	VARCHAR(32)	Der Typ des Berechtigten, dem eine Berechtigung erteilt oder entzogen wurde. Mögliche Werte: USER, GROUP, ROLE, AMBIGUOUS oder TRUSTED_CONTEXT, wenn es sich bei dem Prüfereignis um eines der folgenden Ereignisse handelt: • ADD_DEFAULT_ROLE • DROP_DEFAULT_ROLE • ALTER_DEFAULT_ROLE • ADD_USER • DROP_USER • ALTER_USER_ADD_ROLE • ALTER_USER_DROP_ROLE • ALTER_USER_AUTHENTICATION

Tabelle 47. Prüfsatzaufbau für SECMAINT-Ereignisse (Forts.)

NAME	FORMAT	BESCHREIBUNG
Privilege or Authority (Zugriffsrecht oder Berechtigung)	CHAR(34)	Gibt an, welcher Typ von Zugriffsrecht oder Berechtigung erteilt oder entzogen wurde. Mögliche Werte: siehe Abschnitt „SECMAINT-Zugriffsrechte bzw. -Berechtigungen". Es ergibt sich der Wert ROLE MEMBERSHIP, wenn es sich bei dem Prüfereignis um eines der folgenden Ereignisse handelt: • ADD_DEFAULT_ROLE, DROP_DEFAULT_ROLE • ALTER_DEFAULT_ROLE • ADD_USER • DROP_USER • ALTER_USER_ADD_ROLE • ALTER_USER_DROP_ROLE • ALTER_USER_AUTHENTICATION
Package Version (Paketversion)	VARCHAR(64)	Die Version des Pakets, das beim Auftreten des Prüfereignisses verwendet wurde.
Access Type (Zugriffstyp)	VARCHAR(32)	Der Zugriffstyp, dem ein Sicherheitskennsatz erteilt wurde. Gültige Werte: • READ • WRITE • ALL Der Zugriffstyp, für den eine Sicherheitsrichtlinie geändert wurde. Gültige Werte: • USE GROUP AUTHORIZATIONS • IGNORE GROUP AUTHORIZATIONS • USE ROLE AUTHORIZATIONS • IGNORE ROLE AUTHORIZATIONS • OVERRIDE NOT AUTHORIZED WRITE SECURITY LABEL • RESTRICT NOT AUTHORIZED WRITE SECURITY LABEL
Assumable Authid (Anzunehmende Berechtigungs-ID)	VARCHAR(128)	Wenn es sich bei dem erteilten Zugriffsrecht um das Zugriffsrecht SETSESSIONUSER handelt, ist dies die Berechtigungs-ID, die der Berechtigte als Sitzungsbenutzer definieren darf.
Local Transaction ID (Lokale Transaktions-ID)	VARCHAR(10) FOR BIT DATA	Die lokale Transaktions-ID, die beim Auftreten des Prüfereignisses verwendet wurde. Dies ist die Struktur SQLU_TID, die Teil der Transaktionsprotokolle ist.
Global Transaction ID (Globale Transaktions-ID)	VARCHAR(30) FOR BIT DATA	Die globale Transaktions-ID, die beim Auftreten des Prüfereignisses verwendet wurde. Dies ist das Datenfeld in der Struktur SQLP_GXID, die Teil der Transaktionsprotokolle ist.
Grantor Type (Typ des Berechtigungsgebers)	VARCHAR(32)	Der Typ des Berechtigungsgebers. Mögliche Werte: USER.
Client User ID (Clientbenutzer-ID)	VARCHAR(255)	Der Wert des Sonderregisters CURRENT CLIENT USERID zu dem Zeitpunkt, als das Prüfereignis auftrat.
Client Workstation Name (Name der Client-Workstation)	VARCHAR(255)	Der Wert des Sonderregisters CURRENT CLIENT_WRKSTNNAME zu dem Zeitpunkt, als das Prüfereignis auftrat.
Client Application Name (Name der Clientanwendung)	VARCHAR(255)	Der Wert des Sonderregisters CURRENT CLIENT_APPLNAME zu dem Zeitpunkt, als das Prüfereignis auftrat.

Tabelle 47. Prüfsatzaufbau für SECMAINT-Ereignisse (Forts.)

NAME	FORMAT	BESCHREIBUNG
Client Accounting String (Abrechnungszeichenfolge für Client)	VARCHAR(255)	Der Wert des Sonderregisters CURRENT CLIENT_ACCTNG zu dem Zeitpunkt, als das Prüfereignis auftrat.
Trusted Context User (Benutzer des gesicherten Kontexts)	VARCHAR(128)	Gibt einen Benutzer des gesicherten Kontexts an, wenn das Prüfereignis ADD_USER oder DROP_USER ist.
Trusted Context User Authentication (Benutzerauthentifizierung für gesicherten Kontext)	INTEGER	Gibt die Authentifizierungseinstellung für einen Benutzer des gesicherten Kontexts an, wenn das Prüfereignis ADD_USER, DROP_USER oder ALTER_USER_AUTHENTICATION ist. 1 : Authentifizierung erforderlich 0 : Authentifizierung nicht erforderlich
Trusted Context Name (Name des gesicherten Kontexts)	VARCHAR(128)	Der Name des gesicherten Kontexts, der der gesicherten Verbindung zugeordnet ist.
Connection Trust Type (Typ der gesicherten Verbindung)	INTEGER	Mögliche Werte: IMPLICIT_TRUSTED_CONNECTION EXPLICIT_TRUSTED_CONNECTION
Role Inherited (Übernommene Rolle)	VARCHAR(128)	Die Rolle, die durch eine gesicherte Verbindung übernommen wurde.

SECMAINT-Zugriffsrechte bzw. -Berechtigungen

Die folgende Liste zeigt die möglichen SECMAINT-Zugriffsrechte bzw. -Berechtigungen.

0x00000000000000000000000000000001 Control Table
Zugriffsrecht CONTROL einer Tabelle oder Sicht erteilt oder entzogen.

0x00000000000000000000000000000002 ALTER
Zugriffsrecht zum Ändern einer Tabelle oder Sequenz erteilt oder entzogen.

0x00000000000000000000000000000004 ALTER with GRANT
Zugriffsrecht zum Ändern einer Tabelle oder Sequenz mit der Möglichkeit zum Erteilen (GRANT) von Zugriffsrechten erteilt oder entzogen.

0x00000000000000000000000000000008 DELETE TABLE
Zugriffsrecht DELETE zum Löschen einer Tabelle oder Sicht erteilt oder entzogen.

0x00000000000000000000000000000010 DELETE TABLE with GRANT
Zugriffsrecht DELETE zum Löschen einer Tabelle mit der Möglichkeit zum Erteilen (GRANT) von Zugriffsrechten erteilt oder entzogen.

0x00000000000000000000000000000020 Table Index
Zugriffsrecht einem Index erteilt oder entzogen.

0x00000000000000000000000000000040 Table Index with GRANT
Zugriffsrecht mit der Möglichkeit zum Erteilen (GRANT) von Zugriffsrechten einem Index erteilt oder entzogen.

0x00000000000000000000000000000080 Table INSERT
Zugriffsrecht INSERT zum Einfügen für eine Tabelle oder Sicht erteilt oder entzogen.

0x00000000000000000000000000000100 Table INSERT with GRANT
Zugriffsrecht INSERT zum Einfügen für eine Tabelle oder Sicht mit der Möglichkeit zum Erteilen (GRANT) von Zugriffsrechten erteilt oder entzogen.

0x00000000000000000000000000000200 Table SELECT
Zugriffsrecht SELECT zum Auswählen in einer Tabelle erteilt oder entzogen.

0x00000000000000000000000000000400 Table SELECT with GRANT
Zugriffsreicht SELECT zum Auswählen in einer Tabelle mit der Möglichkeit zum Erteilen (GRANT) von Zugriffsrechten erteilt oder entzogen.

0x00000000000000000000000000000800 Table UPDATE
Zugriffsrecht UPDATE zum Aktualisieren für eine Tabelle oder Sicht erteilt oder entzogen.

0x00000000000000000000000000001000 Table UPDATE with GRANT
Zugriffsrecht UPDATE zum Aktualisieren für eine Tabelle oder Sicht mit der Möglichkeit zum Erteilen (GRANT) von Zugriffsrechten erteilt oder entzogen.

0x00000000000000000000000000002000 Table REFERENCE
Zugriffsrecht REFERENCE zum Verweisen auf eine Tabelle erteilt oder entzogen.

0x00000000000000000000000000004000 Table REFERENCE with GRANT
Zugriffsrecht REFERENCE zum Verweisen auf eine Tabelle mit der Möglichkeit zum Erteilen (GRANT) von Zugriffsrechten erteilt oder entzogen.

0x00000000000000000000000000020000 CREATEIN Schema
Zugriffsrecht CREATEIN zum Erstellen in einem Schema erteilt oder entzogen.

0x00000000000000000000000000040000 CREATEIN Schema with GRANT
Zugriffsrecht CREATEIN zum Erstellen in einem Schema mit der Möglichkeit zum Erteilen (GRANT) von Zugriffsrechten erteilt oder entzogen.

0x00000000000000000000000000080000 DROPIN Schema
Zugriffsrecht DROPIN zum Löschen in einem Schema erteilt oder entzogen.

0x00000000000000000000000000100000 DROPIN Schema with GRANT
Zugriffsrecht DROPIN zum Löschen in einem Schema mit der Möglichkeit zum Erteilen (GRANT) von Zugriffsrechten erteilt oder entzogen.

0x00000000000000000000000000200000 ALTERIN Schema
Zugriffsrecht ALTERIN zum Ändern in einem Schema erteilt oder entzogen.

0x00000000000000000000000000400000 ALTERIN Schema with GRANT
Zugriffsrecht ALTERIN zum Ändern in einem Schema mit der Möglichkeit zum Erteilen (GRANT) von Zugriffsrechten erteilt oder entzogen.

0x00000000000000000000000000800000 DBADM Authority
Berechtigung DBADM erteilt oder entzogen.

0x00000000000000000000000001000000 CREATETAB Authority
CREATETAB-Berechtigung erteilt oder entzogen.

0x00000000000000000000000002000000 BINDADD Authority
Berechtigung BINDADD erteilt oder entzogen.

0x00000000000000000000000004000000 CONNECT Authority
Berechtigung CONNECT erteilt oder entzogen.

0x00000000000000000000000008000000 Create not fenced Authority
Berechtigung 'Create not fenced' (nicht abgeschirmte erstellen) erteilt oder entzogen.

0x00000000000000000000000010000000 Implicit Schema Authority
Berechtigung 'Implicit schema' erteilt oder entzogen.

0x00000000000000000000000020000000 Server PASSTHRU
Zugriffsrecht für die Verwendung der Durchgriffsfunktion für diesen Server (Datenquelle föderierter Datenbanken) erteilt oder entzogen.

0x00000000000000000000000040000000 ESTABLISH TRUSTED CONNECTION
Gesicherte Verbindung wurde erstellt.

0x00000000000000000000000100000000 Table Space USE
Zugriffsrecht USE zum Erstellen einer Tabelle in einem Tabellenbereich erteilt oder entzogen.

0x00000000000000000000000200000000 Table Space USE with GRANT
Zugriffsrecht USE zum Erstellen einer Tabelle in einem Tabellenbereich mit der Möglichkeit zum Erteilen (GRANT) von Zugriffsrechten erteilt oder entzogen.

0x00000000000000000000000400000000 Column UPDATE
Zugriffsrecht UPDATE zum Aktualisieren einer oder mehrerer bestimmter Spalten einer Tabelle erteilt oder entzogen.

0x00000000000000000000000800000000 Column UPDATE with GRANT
Zugriffsrecht UPDATE zum Aktualisieren einer oder mehrerer bestimmter Spalten einer Tabelle mit der Möglichkeit zum Erteilen (GRANT) von Zugriffsrechten erteilt oder entzogen.

0x00000000000000000000001000000000 Column REFERENCE
Zugriffsrecht REFERENCE zum Verweisen auf eine oder mehrere bestimmte Spalten einer Tabelle erteilt oder entzogen.

0x00000000000000000000002000000000 Column REFERENCE with GRANT
Zugriffsrecht REFERENCE zum Verweisen auf eine oder mehrere bestimmte Spalten einer Tabelle mit der Möglichkeit zum Erteilen von Zugriffsrechten (GRANT) erteilt oder entzogen.

0x00000000000000000000004000000000 LOAD Authority
Berechtigung LOAD erteilt oder entzogen.

0x00000000000000000000008000000000 Package BIND
Zugriffsrecht BIND für ein Paket erteilt oder entzogen.

0x00000000000000000000010000000000 Package BIND with GRANT
Zugriffsrecht BIND für ein Paket mit der Möglichkeit zum Erteilen (GRANT) von Zugriffsrechten erteilt oder entzogen.

0x00000000000000000000020000000000 EXECUTE
Zugriffsrecht EXECUTE für ein Paket oder eine Routine erteilt oder entzogen.

0x00000000000000000000040000000000 EXECUTE with GRANT
Zugriffsrecht EXECUTE für ein Paket oder eine Routine mit der Möglichkeit zum Erteilen (GRANT) von Zugriffsrechten erteilt oder entzogen.

0x00000000000000000000080000000000 EXECUTE IN SCHEMA
Zugriffsrecht EXECUTE für alle Routinen in einem Schema erteilt oder entzogen.

0x00000000000000000000100000000000 EXECUTE IN SCHEMA with GRANT
Zugriffsrecht EXECUTE für alle Routinen in einem Schema mit der Möglichkeit zum Erteilen (GRANT) von Zugriffsrechten erteilt oder entzogen.

0x00000000000000000000200000000000 EXECUTE IN TYPE
Zugriffsrecht EXECUTE für alle Routinen in einem Typ erteilt oder entzogen.

0x00000000000000000000400000000000 EXECUTE IN TYPE with GRANT
Zugriffsrecht EXECUTE für alle Routinen in einem Typ mit der Möglichkeit zum Erteilen (GRANT) von Zugriffsrechten erteilt oder entzogen.

0x00000000000000000000800000000000 CREATE EXTERNAL ROUTINE
Zugriffsrecht CREATE EXTERNAL ROUTINE erteilt oder entzogen.

0x00000000000000000001000000000000 QUIESCE_CONNECT
Zugriffsrecht QUIESCE_CONNECT erteilt oder entzogen.

0x00000000000000000004000000000000 SECADM Authority
Berechtigung SECADM erteilt oder entzogen.

0x00000000000000000008000000000000 USAGE Authority
Zugriffsrecht USAGE für eine Sequenz erteilt oder entzogen.

0x00000000000000000010000000000000 USAGE with GRANT Authority
Zugriffsrecht USAGE für eine Sequenz mit der Möglichkeit zum Erteilen (GRANT) von Zugriffsrechten erteilt oder entzogen.

0x00000000000000000020000000000000 WITH ADMIN Option
Zugriffsrecht WITH ADMIN OPTION für eine Rolle erteilt oder entzogen.

0x00000000000000000040000000000000 SETSESSIONUSER Privilege
Zugriffsrecht SETSESSIONUSER erteilt oder entzogen.

0x00000000000000000080000000000000 Exemption
Freistellung erteilt oder entzogen.

0x00000000000000000100000000000000 Security label
Sicherheitskennsatz erteilt oder entzogen.

0x00000000000000000200000000000000 WRITE with GRANT
Zugriffsrecht WRITE zum Schreiben einer globalen Variablen mit der Möglichkeit zum Erteilen (GRANT) von Zugriffsrechten erteilt oder entzogen.

0x00000000000000000400000000000000 Role Membership
Rollenzugehörigkeit, die erteilt oder entzogen wurde.

0x00000000000000000800000000000000 Role Membership with ADMIN Option
Rollenzugehörigkeit mit dem Zugriffsrecht WITH ADMIN OPTION, die erteilt oder entzogen wurde.

0x00000000000000001000000000000000 READ
Zugriffsrecht READ zum Lesen einer globalen Variablen erteilt oder entzogen.

0x00000000000000002000000000000000 READ with GRANT
Zugriffsrecht READ zum Lesen einer globalen Variablen mit der Möglichkeit zum Erteilen (GRANT) von Zugriffsrechten erteilt oder entzogen.

0x00000000000000004000000000000000 WRITE
Zugriffsrecht WRITE zum Schreiben einer globalen Variablen erteilt oder entzogen.

0x00000000000000010000000000000000 SQLADM
Berechtigung SQLADM erteilt oder entzogen.

0x00000000000000020000000000000000 WLMADM
Berechtigung WLMADM erteilt oder entzogen.

0x00000000000000040000000000000000 EXPLAIN
Berechtigung EXPLAIN erteilt oder entzogen.

0x00000000000000080000000000000000 DATAACCESS
Berechtigung DATAACCESS erteilt oder entzogen.

0x00000000000000100000000000000000 ACCESSCTRL
Berechtigung ACCESSCTRL erteilt oder entzogen.

Prüfsatzaufbau für SYSADMIN-Ereignisse

Die folgende Tabelle zeigt den Prüfsatzaufbau für SYSADMIN-Ereignisse.

Beispiel für einen Prüfsatz:

```
timestamp=1998-06-24-11.54.04.129923;
category=SYSADMIN;
audit event=DB2AUDIT;
event correlator=1;
event status=0;
userid=boss;authid=BOSS;
application id=*LOCAL.boss.980624155404;
application name=db2audit;
```

Tabelle 48. Prüfsatzaufbau für SYSADMIN-Ereignisse

NAME	FORMAT	BESCHREIBUNG
Timestamp (Zeitmarke)	CHAR(26)	Datum und Uhrzeit des Prüfereignisses
Category (Kategorie)	CHAR(8)	Die Kategorie des Prüfereignisses. Mögliche Werte: SYSADMIN
Audit Event (Prüfereignis)	VARCHAR(32)	Spezifisches Prüfereignis. Eine Liste gültiger Werte finden Sie im Abschnitt über die Kategorie SYSADMIN unter „Prüfereignisse” auf Seite 376.
Event Correlator (Ereigniskorrelationswert)	INTEGER	Korrelationskennung für die geprüfte Operation. Kann zur Erkennung verwendet werden, welche Prüfsätze einem einzelnen Ereignis zugeordnet sind.
Event Status (Ereignisstatus)	INTEGER	Status des Prüfereignisses, dargestellt durch einen SQLCODE, für den Folgendes gilt: Erfolgreiches Ereignis > = 0 Fehlgeschlagenes Ereignis < 0
Database Name (Datenbankname)	CHAR(8)	Name der Datenbank, für die das Ereignis generiert wurde. Nicht belegt, wenn es sich um ein Prüfereignis auf Instanzebene handelt.
User ID (Benutzer-ID)	VARCHAR(1024)	Die Benutzer-ID zum Zeitpunkt des Prüfereignisses.
Authorization ID (Berechtigungs-ID)	VARCHAR(128)	Die Berechtigungs-ID zum Zeitpunkt des Prüfereignisses.

Tabelle 48. Prüfsatzaufbau für SYSADMIN-Ereignisse (Forts.)

NAME	FORMAT	BESCHREIBUNG
Original User ID (ursprüngliche Benutzer-ID)	VARCHAR(1024)	Der Wert der globalen Variablen CLIENT_ORIGUSERID zu dem Zeitpunkt, zu dem das Prüfereignis auftrat.
Origin Node Number (Nummer des Ursprungsknotens)	SMALLINT	Nummer des Members, für das das Prüfereignis aufgetreten ist.
Coordinator Node Number (Nummer des Koordinatorknotens)	SMALLINT	Nummer des Members, das als Koordinatormember definiert ist.
Application ID (Anwendungs-ID)	VARCHAR(255)	Die ID der Anwendung, die beim Auftreten des Prüfereignisses verwendet wurde.
Application Name (Anwendungsname)	VARCHAR(1024)	Der Name der Anwendung, die beim Auftreten des Prüfereignisses verwendet wurde.
Package Schema (Paketschema)	VARCHAR(128)	Das Schema des Pakets, das beim Auftreten des Prüfereignisses verwendet wurde.
Package Name (Paketname)	VARCHAR(128)	Der Name des Pakets, das beim Auftreten des Prüfereignisses verwendet wurde.
Package Section Number (Paketabschnittsnummer)	SMALLINT	Die Abschnittsnummer innerhalb des Pakets, das beim Auftreten des Prüfereignisses verwendet wurde.
Package Version (Paketversion)	VARCHAR(64)	Die Version des Pakets, das beim Auftreten des Prüfereignisses verwendet wurde.
Local Transaction ID (Lokale Transaktions-ID)	VARCHAR(10) FOR BIT DATA	Die lokale Transaktions-ID, die beim Auftreten des Prüfereignisses verwendet wurde. Dies ist die Struktur SQLU_TID, die Teil der Transaktionsprotokolle ist.
Global Transaction ID (Globale Transaktions-ID)	VARCHAR(30) FOR BIT DATA	Die globale Transaktions-ID, die beim Auftreten des Prüfereignisses verwendet wurde. Dies ist das Datenfeld in der Struktur SQLP_GXID, die Teil der Transaktionsprotokolle ist.
Client User ID (Clientbenutzer-ID)	VARCHAR(255)	Der Wert des Sonderregisters CURRENT CLIENT USERID zu dem Zeitpunkt, als das Prüfereignis auftrat.
Client Workstation Name (Name der Client-Workstation)	VARCHAR(255)	Der Wert des Sonderregisters CURRENT CLIENT_WRKSTNNAME zu dem Zeitpunkt, als das Prüfereignis auftrat.
Client Application Name (Name der Clientanwendung)	VARCHAR(255)	Der Wert des Sonderregisters CURRENT CLIENT_APPLNAME zu dem Zeitpunkt, als das Prüfereignis auftrat.
Client Accounting String (Abrechnungszeichenfolge für Client)	VARCHAR(255)	Der Wert des Sonderregisters CURRENT CLIENT_ACCTNG zu dem Zeitpunkt, als das Prüfereignis auftrat.
Trusted Context Name (Name des gesicherten Kontexts)	VARCHAR(128)	Der Name des gesicherten Kontexts, der der gesicherten Verbindung zugeordnet ist.
Connection Trust Type (Typ der gesicherten Verbindung)	INTEGER	Mögliche Werte: IMPLICIT_TRUSTED_CONNECTION EXPLICIT_TRUSTED_CONNECTION
Role Inherited (Übernommene Rolle)	VARCHAR(128)	Die Rolle, die durch eine gesicherte Verbindung übernommen wurde.

Prüfsatzaufbau für VALIDATE-Ereignisse

Das Format des Prüfsatzes für VALIDATE-Ereignisse wird in der folgenden Tabelle gezeigt.

Beispiel für einen Prüfsatz:

```
timestamp=2007-05-07-10.30.51.585626;
category=VALIDATE;
audit event=AUTHENTICATION;
event correlator=1;
event status=0;
userid=newton;
authid=NEWTON;
execution id=gstager;
application id=*LOCAL.gstager.070507143051;
application name=db2bp;
auth type=SERVER;
plugin name=IBMOSauthserver;
```

Tabelle 49. Prüfsatzaufbau für VALIDATE-Ereignisse

NAME	FORMAT	BESCHREIBUNG
Timestamp (Zeitmarke)	CHAR(26)	Datum und Uhrzeit des Prüfereignisses
Category (Kategorie)	CHAR(8)	Die Kategorie des Prüfereignisses. Mögliche Werte: VALIDATE
Audit Event (Prüfereignis)	VARCHAR(32)	Spezifisches Prüfereignis. Mögliche Werte: `GET_GROUPS`, `GET_USERID`, `AUTHENTICATE_PASSWORD`, `VALIDATE_USER`, `AUTHENTICATION` und `GET_USERMAPPING_FROM_PLUGIN`.
Event Correlator (Ereigniskorrelationswert)	INTEGER	Korrelationskennung für die geprüfte Operation. Kann zur Erkennung verwendet werden, welche Prüfsätze einem einzelnen Ereignis zugeordnet sind.
Event Status (Ereignisstatus)	INTEGER	Status des Prüfereignisses, dargestellt durch einen SQLCODE, für den Folgendes gilt: Erfolgreiches Ereignis > = 0 Fehlgeschlagenes Ereignis < 0
Database Name (Datenbankname)	CHAR(8)	Name der Datenbank, für die das Ereignis generiert wurde. Nicht belegt, wenn es sich um ein Prüfereignis auf Instanzebene handelt.
User ID (Benutzer-ID)	VARCHAR(1024)	Die Benutzer-ID zum Zeitpunkt des Prüfereignisses.
Authorization ID (Berechtigungs-ID)	VARCHAR(128)	Die Berechtigungs-ID zum Zeitpunkt des Prüfereignisses.
Execution ID (Ausführungs-ID)	VARCHAR(1024)	Die Ausführungs-ID, die beim Auftreten des Prüfereignisses verwendet wurde.
Original User ID (ursprüngliche Benutzer-ID)	VARCHAR(1024)	Der Wert der globalen Variablen CLIENT_ORIGUSERID zu dem Zeitpunkt, zu dem das Prüfereignis auftrat.
Origin Node Number (Nummer des Ursprungsknotens)	SMALLINT	Nummer des Members, für das das Prüfereignis aufgetreten ist.

Tabelle 49. Prüfsatzaufbau für VALIDATE-Ereignisse (Forts.)

NAME	FORMAT	BESCHREIBUNG
Coordinator Node Number (Nummer des Koordinatorknotens)	SMALLINT	Nummer des Members, das als Koordinatormember definiert ist.
Application ID (Anwendungs-ID)	VARCHAR(255)	Die ID der Anwendung, die beim Auftreten des Prüfereignisses verwendet wurde.
Application Name (Anwendungsname)	VARCHAR(1024)	Der Name der Anwendung, die beim Auftreten des Prüfereignisses verwendet wurde.
Authentication Type (Authentifizierungstyp)	VARCHAR(32)	Der Authentifizierungstyp beim Auftreten des Prüfereignisses.
Package Schema (Paketschema)	VARCHAR(128)	Das Schema des Pakets, das beim Auftreten des Prüfereignisses verwendet wurde.
Package Name (Paketname)	VARCHAR(128)	Der Name des Pakets, das beim Auftreten des Prüfereignisses verwendet wurde.
Package Section Number (Paketabschnittsnummer)	SMALLINT	Die Abschnittsnummer innerhalb des Pakets, das beim Auftreten des Prüfereignisses verwendet wurde.
Package Version (Paketversion)	VARCHAR(64)	Die Version des Pakets, das beim Auftreten des Prüfereignisses verwendet wurde.
Plug-in Name (Plug-in-Name)	VARCHAR(32)	Der Name des Plug-ins, das beim Auftreten des Prüfereignisses verwendet wurde.
Local Transaction ID (Lokale Transaktions-ID)	VARCHAR(10) FOR BIT DATA	Die lokale Transaktions-ID, die beim Auftreten des Prüfereignisses verwendet wurde. Dies ist die Struktur SQLU_TID, die Teil der Transaktionsprotokolle ist.
Global Transaction ID (Globale Transaktions-ID)	VARCHAR(30) FOR BIT DATA	Die globale Transaktions-ID, die beim Auftreten des Prüfereignisses verwendet wurde. Dies ist das Datenfeld in der Struktur SQLP_GXID, die Teil der Transaktionsprotokolle ist.
Client User ID (Clientbenutzer-ID)	VARCHAR(255)	Der Wert des Sonderregisters CURRENT CLIENT USERID zu dem Zeitpunkt, als das Prüfereignis auftrat.
Client Workstation Name (Name der Client-Workstation)	VARCHAR(255)	Der Wert des Sonderregisters CURRENT CLIENT_WRKSTNNAME zu dem Zeitpunkt, als das Prüfereignis auftrat.
Client Application Name (Name der Clientanwendung)	VARCHAR(255)	Der Wert des Sonderregisters CURRENT CLIENT_APPLNAME zu dem Zeitpunkt, als das Prüfereignis auftrat.
Client Accounting String (Abrechnungszeichenfolge für Client)	VARCHAR(255)	Der Wert des Sonderregisters CURRENT CLIENT_ACCTNG zu dem Zeitpunkt, als das Prüfereignis auftrat.
Trusted Context Name (Name des gesicherten Kontexts)	VARCHAR(128)	Der Name des gesicherten Kontexts, der der gesicherten Verbindung zugeordnet ist.
Connection Trust Type (Typ der gesicherten Verbindung)	INTEGER	Mögliche Werte: IMPLICIT_TRUSTED_CONNECTION EXPLICIT_TRUSTED_CONNECTION
Role Inherited (Übernommene Rolle)	VARCHAR(128)	Der Name der Rolle, die durch den gesicherten Kontext übernommen wurde.

Prüfsatzaufbau für CONTEXT-Ereignisse

Die folgende Tabelle zeigt den Prüfsatzaufbau für CONTEXT-Ereignisse.

Beispiel für einen Prüfsatz:

```
timestamp=1998-06-24-08.42.41.476840;
category=CONTEXT;
audit event=EXECUTE_IMMEDIATE;
event correlator=3;
database=FOO;
userid=boss;
authid=BOSS;
application id=*LOCAL.newton.980624124210;
application name=testapp;
package schema=NULLID;
package name=SQLC28A1;
package section=203;
text=create table audit(c1 char(10), c2 integer);
```

Tabelle 50. Prüfsatzaufbau für CONTEXT-Ereignisse

NAME	FORMAT	BESCHREIBUNG
Timestamp (Zeitmarke)	CHAR(26)	Datum und Uhrzeit des Prüfereignisses
Category (Kategorie)	CHAR(8)	Die Kategorie des Prüfereignisses. Mögliche Werte: CONTEXT
Audit Event (Prüfereignis)	VARCHAR(32)	Spezifisches Prüfereignis. Eine Liste gültiger Werte finden Sie im Abschnitt über die Kategorie CONTEXT unter „Prüfereignisse" auf Seite 376.
Event Correlator (Ereigniskorrelationswert)	INTEGER	Korrelationskennung für die geprüfte Operation. Kann zur Erkennung verwendet werden, welche Prüfsätze einem einzelnen Ereignis zugeordnet sind.
Database Name (Datenbankname)	CHAR(8)	Name der Datenbank, für die das Ereignis generiert wurde. Nicht belegt, wenn es sich um ein Prüfereignis auf Instanzebene handelt.
User ID (Benutzer-ID)	VARCHAR(1024)	Die Benutzer-ID zum Zeitpunkt des Prüfereignisses. Wenn das Prüfereignis SWITCH_USER ist, stellt dieses Feld die Benutzer-ID dar, zu der gewechselt wird.
Authorization ID (Berechtigungs-ID)	VARCHAR(128)	Die Berechtigungs-ID zum Zeitpunkt des Prüfereignisses. Wenn das Prüfereignis SWITCH_USER ist, stellt dieses Feld die Berechtigungs-ID dar, zu der gewechselt wird.
Original User ID (ursprüngliche Benutzer-ID)	VARCHAR(1024)	Der Wert der globalen Variablen CLIENT_ORIGUSERID zu dem Zeitpunkt, zu dem das Prüfereignis auftrat.
Origin Node Number (Nummer des Ursprungsknotens)	SMALLINT	Nummer des Members, für das das Prüfereignis aufgetreten ist.
Coordinator Node Number (Nummer des Koordinatorknotens)	SMALLINT	Nummer des Members, das als Koordinatormember definiert ist.
Application ID (Anwendungs-ID)	VARCHAR(255)	Die ID der Anwendung, die beim Auftreten des Prüfereignisses verwendet wurde.

Tabelle 50. Prüfsatzaufbau für CONTEXT-Ereignisse (Forts.)

NAME	FORMAT	BESCHREIBUNG
Application Name (Anwendungsname)	VARCHAR(1024)	Der Name der Anwendung, die beim Auftreten des Prüfereignisses verwendet wurde.
Package Schema (Paketschema)	VARCHAR(128)	Das Schema des Pakets, das beim Auftreten des Prüfereignisses verwendet wurde.
Package Name (Paketname)	VARCHAR(128)	Der Name des Pakets, das beim Auftreten des Prüfereignisses verwendet wurde.
Package Section Number (Paketabschnittsnummer)	SMALLINT	Die Abschnittsnummer innerhalb des Pakets, das beim Auftreten des Prüfereignisses verwendet wurde.
Statement Text (Anweisungstext)	CLOB(8M)	Text der SQL- oder XQuery-Anweisung, falls vorhanden. Null, wenn kein Text für die SQL- oder XQuery-Anweisung verfügbar ist.
Package Version (Paketversion)	VARCHAR(64)	Die Version des Pakets, das beim Auftreten des Prüfereignisses verwendet wurde.
Local Transaction ID (Lokale Transaktions-ID)	VARCHAR(10) FOR BIT DATA	Die lokale Transaktions-ID, die beim Auftreten des Prüfereignisses verwendet wurde. Dies ist die Struktur SQLU_TID, die Teil der Transaktionsprotokolle ist.
Global Transaction ID (Globale Transaktions-ID)	VARCHAR(30) FOR BIT DATA	Die globale Transaktions-ID, die beim Auftreten des Prüfereignisses verwendet wurde. Dies ist das Datenfeld in der Struktur SQLP_GXID, die Teil der Transaktionsprotokolle ist.
Client User ID (Clientbenutzer-ID)	VARCHAR(255)	Der Wert des Sonderregisters CURRENT CLIENT USERID zu dem Zeitpunkt, als das Prüfereignis auftrat.
Client Workstation Name (Name der Client-Workstation)	VARCHAR(255)	Der Wert des Sonderregisters CURRENT CLIENT_WRKSTNNAME zu dem Zeitpunkt, als das Prüfereignis auftrat.
Client Application Name (Name der Clientanwendung)	VARCHAR(255)	Der Wert des Sonderregisters CURRENT CLIENT_APPLNAME zu dem Zeitpunkt, als das Prüfereignis auftrat.
Client Accounting String (Abrechnungszeichenfolge für Client)	VARCHAR(255)	Der Wert des Sonderregisters CURRENT CLIENT_ACCTNG zu dem Zeitpunkt, als das Prüfereignis auftrat.
Trusted Context Name (Name des gesicherten Kontexts)	VARCHAR(128)	Der Name des gesicherten Kontexts, der der gesicherten Verbindung zugeordnet ist.
Connection Trust Type (Typ der gesicherten Verbindung)	INTEGER	Mögliche Werte: IMPLICIT_TRUSTED_CONNECTION EXPLICIT_TRUSTED_CONNECTION
Role Inherited (Übernommene Rolle)	VARCHAR(128)	Die Rolle, die durch eine gesicherte Verbindung übernommen wurde.

Prüfsatzaufbau für EXECUTE-Ereignisse

Die folgende Tabelle beschreibt alle Felder, die von der Kategorie EXECUTE der Prüffunktion geprüft werden.

Beispiel für einen Prüfsatz:

Anmerkung: Im Unterschied zu anderen Prüfkategorien können bei der Kategorie EXECUTE, wenn das Prüfprotokoll in einem Tabellenformat angezeigt wird, mehrere Zeilen für die Beschreibung ein und desselben Ereignisses angezeigt werden. Der erste Prüfsatz beschreibt das Hauptereignis, und die Ereignisspalte enthält das Schlüsselwort STATEMENT. Die übrigen Zeilen beschreiben die Parametermarken oder Hostvariablen (jeweils in einer Zeile), und die entsprechende Ereignisspalte enthält das Schlüsselwort DATA. Wenn das Prüfprotokoll im Berichtsformat angezeigt wird, ist jeweils ein Prüfsatz vorhanden, jedoch mit mehreren Einträgen für den Anweisungswert ('Statement Value'). Das Schlüsselwort DATA wird nur im Tabellenformat verwendet.

```
timestamp=2006-04-10-13.20.51.029203;
  category=EXECUTE;
  audit event=STATEMENT;
  event correlator=1;
  event status=0;
  database=SAMPLE;
  userid=smith;
  authid=SMITH;
  session authid=SMITH;
  application id=*LOCAL.prodrig.060410172044;
  application name=myapp;
  package schema=NULLID;
  package name=SQLC2F0A;
  package section=201;
  uow id=2;
  activity id=3;
  statement invocation id=0;
  statement nesting level=0;
  statement text=SELECT * FROM DEPARTMENT WHERE DEPTNO = ? AND DEPTNAME = ?;
  statement isolation level=CS;
  compilation environment=
    isolation level=CS
    query optimization=5
    min_dec_div_3=NO
    degree=1
    sqlrules=DB2
    refresh age=+00000000000000.000000
    schema=SMITH
    maintained table type=SYSTEM
    resolution timestamp=2006-04-10-13.20.51.000000
    federated asynchrony=0;
  value index=0;
  value type=CHAR;
  value data=C01;
  value index=1;
  value type=VARCHAR;
  value extended indicator=-1;
  value index=INFORMATION CENTER;  local_start_time=2006-04-10-13.20.51.021507
```

Tabelle 51. Prüfsatzaufbau für EXECUTE-Ereignisse

NAME	FORMAT	BESCHREIBUNG
Timestamp (Zeitmarke)	CHAR(26)	Datum und Uhrzeit des Prüfereignisses
Category (Kategorie)	CHAR(8)	Die Kategorie des Prüfereignisses. Mögliche Werte: EXECUTE
Audit Event (Prüfereignis)	VARCHAR(32)	Spezifisches Prüfereignis. Eine Liste gültiger Werte finden Sie im Abschnitt über die Kategorie EXECUTE unter „Prüfereignisse" auf Seite 376.
Event Correlator (Ereigniskorrelationswert)	INTEGER	Korrelationskennung für die geprüfte Operation. Kann zur Erkennung verwendet werden, welche Prüfsätze einem einzelnen Ereignis zugeordnet sind.
Event Status (Ereignisstatus)	INTEGER	Status des Prüfereignisses, dargestellt durch einen SQLCODE, für den Folgendes gilt: Erfolgreiches Ereignis > = 0 und Fehlgeschlagenes Ereignis < 0
Database Name (Datenbankname)	CHAR(8)	Name der Datenbank, für die das Ereignis generiert wurde. Nicht belegt, wenn es sich um ein Prüfereignis auf Instanzebene handelt.
User ID (Benutzer-ID)	VARCHAR(1024)	Die Benutzer-ID zum Zeitpunkt des Prüfereignisses.
Authorization ID (Berechtigungs-ID)	VARCHAR(128)	Die Berechtigungs-ID der Anweisung zum Zeitpunkt des Prüfereignisses.
Session Authorization ID (Sitzungsberechtigungs-ID)	VARCHAR(128)	Die Berechtigungs-ID der Sitzung zum Zeitpunkt des Prüfereignisses.
Origin Node Number (Nummer des Ursprungsknotens)	SMALLINT	Nummer des Members, für das das Prüfereignis aufgetreten ist.
Coordinator Node Number (Nummer des Koordinatorknotens)	SMALLINT	Nummer des Members, das als Koordinatormember definiert ist.
Application ID (Anwendungs-ID)	VARCHAR(255)	Die ID der Anwendung, die beim Auftreten des Prüfereignisses verwendet wurde.
Application Name (Anwendungsname)	VARCHAR(1024)	Der Name der Anwendung, die beim Auftreten des Prüfereignisses verwendet wurde.

Tabelle 51. Prüfsatzaufbau für EXECUTE-Ereignisse (Forts.)

NAME	FORMAT	BESCHREIBUNG
Client User ID (Clientbenutzer-ID)	VARCHAR(255)	Der Wert des Sonderregisters CURRENT CLIENT USERID zu dem Zeitpunkt, als das Prüfereignis auftrat.
Client Accounting String (Abrechnungszeichenfolge für Client)	VARCHAR(255)	Der Wert des Sonderregisters CURRENT CLIENT_ACCTNG zu dem Zeitpunkt, als das Prüfereignis auftrat.
Client Workstation Name (Name der Client-Workstation)	VARCHAR(255)	Der Wert des Sonderregisters CURRENT CLIENT_WRKSTNNAME zu dem Zeitpunkt, als das Prüfereignis auftrat.
Client Application Name (Name der Clientanwendung)	VARCHAR(255)	Der Wert des Sonderregisters CURRENT CLIENT_APPLNAME zu dem Zeitpunkt, als das Prüfereignis auftrat.
Trusted Context Name (Name des gesicherten Kontexts)	VARCHAR(128)	Der Name des gesicherten Kontexts, der der gesicherten Verbindung zugeordnet ist.
Connection Trust type (Typ der gesicherten Verbindung)	INTEGER	Mögliche Werte: IMPLICIT_TRUSTED_ CONNECTION und EXPLICIT_TRUSTED_ CONNECTION.
Role Inherited (Übernommene Rolle)	VARCHAR(128)	Die Rolle, die durch eine gesicherte Verbindung übernommen wurde.
Package Schema (Paketschema)	VARCHAR(128)	Das Schema des Pakets, das beim Auftreten des Prüfereignisses verwendet wurde.
Package Name (Paketname)	VARCHAR(128)	Der Name des Pakets, das beim Auftreten des Prüfereignisses verwendet wurde.
Package Section (Paketabschnitt)	SMALLINT	Die Abschnittsnummer innerhalb des Pakets, das beim Auftreten des Prüfereignisses verwendet wurde.
Package Version (Paketversion)	VARCHAR(164)	Die Version des Pakets, das beim Auftreten des Prüfereignisses verwendet wurde.
Local Transaction ID (Lokale Transaktions-ID)	VARCHAR(10) FOR BIT DATA	Die lokale Transaktions-ID, die beim Auftreten des Prüfereignisses verwendet wurde. Dies ist die Struktur SQLU_TID, die Teil der Transaktionsprotokolle ist.

Tabelle 51. Prüfsatzaufbau für EXECUTE-Ereignisse (Forts.)

NAME	FORMAT	BESCHREIBUNG
Global Transaction ID (Globale Transaktions-ID)	VARCHAR(30) FOR BIT DATA	Die globale Transaktions-ID, die beim Auftreten des Prüfereignisses verwendet wurde. Dies ist das Datenfeld in der Struktur SQLP_GXID, die Teil der Transaktionsprotokolle ist.
UOW ID (Kennung der UOW)	BIGINT	Die Kennung der UOW (Unit of Work, Arbeitseinheit), aus der eine Aktivität stammt. Dieser Wert ist innerhalb einer Anwendungs-ID für jede UOW eindeutig.
Activity ID (Aktivitäts-ID)	BIGINT	Die eindeutige Aktivitäts-ID innerhalb der UOW.
Statement Invocation ID (Aufruf-ID der Anweisung)	BIGINT	Eine Kennung (ID), die einen bestimmten Aufruf einer Routine von anderen auf derselben Verschachtelungsebene innerhalb einer UOW unterscheidet. Sie ist innerhalb einer UOW für eine bestimmte Verschachtelungsebene eindeutig.
Statement Nesting Level (Verschachtelungsebene der Anweisung)	BIGINT	Die aktive Verschachtelungs- bzw. Rekursionsebene, als die Anweisung ausgeführt wurde. Jede Verschachtelungsebene entspricht dem verschachtelten oder rekursivem Aufruf einer gespeicherten Prozedur oder benutzerdefinierten Funktion (UDF).
Activity Type (Aktivitätstyp)	VARCHAR(32)	Der Typ der Aktivität. Mögliche Werte: • READ_DML • WRITE_DML • DDL • CALL • NONE
Statement Text (Anweisungstext)	CLOB(8M)	Text der SQL- oder XQuery-Anweisung, falls vorhanden.

Tabelle 51. Prüfsatzaufbau für EXECUTE-Ereignisse (Forts.)

NAME	FORMAT	BESCHREIBUNG
Statement Isolation Level (Isolationsstufe der Anweisung)	CHAR(8)	Der aktive Isolationswert für die Anweisung während der Ausführung. Mögliche Werte: • NONE (keine Isolation angegeben) • UR (Nicht festgeschriebener Lesevorgang) • CS (Cursorstabilität) • RS (Lesestabilität) • RR (Wiederholtes Lesen)
Compilation Environment Description (Beschreibung der Kompilierungsumgebung)	BLOB(8K)	Die Kompilierungsumgebung, die beim Kompilieren der SQL-Anweisung verwendet wurde. Sie können dieses Element als Eingabe für die Tabellenfunktion COMPILATION_ENV oder als Angabe in der SQL-Anweisung SET COMPILATION ENVIRONMENT verwenden.
Rows Modified (Geänderte Zeilen)	INTEGER	Enthält die Gesamtanzahl der gelöschten, eingefügten oder aktualisierten Zeilen, die das Ergebnis folgender Aktivitäten sind: • Umsetzung von Integritätsbedingungen nach einer erfolgreichen DELETE-Operation • Verarbeitung von ausgelösten SQL-Anweisungen aus aktivierten integrierten Triggern Wenn Compound-SQL-Anweisungen aufgerufen werden, enthält dieses Feld die summierte Anzahl solcher Zeilen für alle Unteranweisungen. Wenn in einigen Fällen ein Fehler festgestellt wird, enthält dieses Feld einen negativen Wert, der einen Zeiger für einen internen Fehler darstellt. Dieser Wert ist äquivalent mit dem Feld 'sqlerrd(5)' des SQL-Kommunikationsbereichs (SQLCA).

Tabelle 51. Prüfsatzaufbau für EXECUTE-Ereignisse (Forts.)

NAME	FORMAT	BESCHREIBUNG
Rows Returned (Zurückgegebene Zeilen)	BIGINT	Enthält die Gesamtanzahl der Zeilen, die durch die Anweisung zurückgegeben wurden.
Savepoint ID (Sicherungspunkt-ID)	BIGINT	Die aktive Sicherungspunkt-ID für die Anweisung während der Ausführung. Wenn das Prüfereignis SAVEPOINT, RELEASE_SAVEPOINT oder ROLLBACK_SAVEPOINT ist, stellt die Sicherungspunkt-ID den Sicherungspunkt dar, der gesetzt, freigegeben oder rückgängig gemacht wird.
Statement Value Index (Anweisungswertindex)	INTEGER	Die Position der in der SQL-Anweisung verwendeten Eingabeparametermarke bzw. Hostvariablen.
Statement Value Type (Anweisungswerttyp)	CHAR(16)	Eine Zeichenfolgedarstellung des Typs eines Datenwerts, der der SQL-Anweisung zugeordnet ist. Beispiele für mögliche Werte sind INTEGER oder CHAR.
Statement Value Data (Anweisungswertdaten)	CLOB(128K)	Eine Zeichenfolgedarstellung eines Datenwerts für die SQL-Anweisung. Parameter für LOB-, LONG- und XML-Datentypen sowie strukturierte Datentypen sind nicht enthalten. Felder für Datum, Zeit und Zeitmarken werden im ISO-Format erfasst.
Statement Value Extended Indicator (Erweiterungsanzeiger für Anweisungswert)	INTEGER	Der Wert des für diesen Anweisungswert angegebenen Erweiterungsanzeigers. Die möglichen Werte sind: • 0, wenn der Anweisungswert wie durch den Anzeigerwert zugeordnet angegeben wurde • -1, wenn vom Anzeigerwert NULL angegeben wurde • -5, wenn vom Anzeigerwert DEFAULT angegeben wurde • -7, wenn vom Anzeigerwert UNASSIGNED angegeben wurde

Tabelle 51. Prüfsatzaufbau für EXECUTE-Ereignisse (Forts.)

NAME	FORMAT	BESCHREIBUNG
Local Start Time (lokale Startzeit)	CHAR(26)	Der Zeitpunkt, an dem die Verarbeitung dieser Aktivität für die Partition begann. Dieses Feld kann eine leere Zeichenfolge enthalten, wenn für die Aktivität kein Paket erforderlich ist (z. B. für CONNECT, CONNECT RESET, COMMIT und ROLLBACK). Der Wert wird in Ortszeit (lokaler Zeit) protokolliert.

Prüfereignisse

Für jede Prüfkategorie können bestimmte Ereignistypen Prüfsätze erstellen.

Ereignisse für die Kategorie AUDIT

- ALTER_AUDIT_POLICY
- ARCHIVE
- AUDIT_REMOVE
- AUDIT_REPLACE
- AUDIT_USING
- CONFIGURE
- CREATE_AUDIT_POLICY
- DB2AUD
- DROP_AUDIT_POLICY
- EXTRACT
- FLUSH
- LIST_LOGS
- PRUNE (ab Version 9.5 nicht mehr generiert)
- START
- STOP
- UPDATE_DBM_CFG

Ereignisse für die Kategorie CHECKING

- CHECKING_FUNCTION
- CHECKING_MEMBERSHIP_IN_ROLES
- CHECKING_OBJECT
- CHECKING_TRANSFER

Ereignisse für die Kategorie CONTEXT

- ADD_NODE
- ATTACH
- BACKUP_DB
- BIND
- CLOSE_CONTAINER_QUERY

- CLOSE_CURSOR
- CLOSE_HISTORY_FILE
- CLOSE_TABLESPACE_QUERY
- COMMIT
- CONNECT
- CONNECT_RESET
- CREATE_DATABASE
- DARI_START
- DARI_STOP
- DBM_CFG_OPERATION
- DESCRIBE
- DESCRIBE_DATABASE
- DETACH
- DISCOVER
- DROP_DATABASE
- ENABLE_MULTIPAGE
- ESTIMATE_SNAPSHOT_SIZE
- EXECUTE
- EXECUTE_IMMEDIATE
- EXTERNAL_CANCEL
- FETCH_CONTAINER_QUERY
- FETCH_CURSOR
- FETCH_HISTORY_FILE
- FETCH_TABLESPACE
- FORCE_APPLICATION
- GET_DB_CFG
- GET_DFLT_CFG
- GET_SNAPSHOT
- GET_TABLESPACE_STATISTIC
- IMPLICIT_REBIND
- LOAD_MSG_FILE
- LOAD_TABLE
- OPEN_CONTAINER_QUERY
- OPEN_CURSOR
- OPEN_HISTORY_FILE
- OPEN_TABLESPACE_QUERY
- PREPARE
- PRUNE_RECOVERY_HISTORY
- QUIESCE_TABLESPACE
- READ_ASYNC_LOG_RECORD
- REBIND
- REDISTRIBUTE
- REORG
- REQUEST_ROLLBACK
- RESET_DB_CFG

- RESET_MONITOR
- RESTORE_DB
- ROLLBACK
- ROLLFORWARD_DB
- RUNSTATS
- SET_APPL_PRIORITY
- SET_MONITOR
- SET_RUNTIME_DEGREE
- SET_TABLESPACE_CONTAINERS
- SINGLE_TABLESPACE_QUERY
- SWITCH_USER
- UNLOAD_TABLE
- UNQUIESCE_TABLESPACE
- UPDATE_AUDIT
- UPDATE_DBM_CFG
- UPDATE_RECOVERY_HISTORY

Ereignisse für die Kategorie EXECUTE

- COMMIT: Ausführung einer Anweisung COMMIT
- CONNECT: Herstellung einer Datenbankverbindung
- CONNECT RESET: Beendigung einer Datenbankverbindung
- DATA: Werte von Hostvariablen oder Parametermarken für die Anweisung

  Dieses Ereignis wird für jede Hostvariable oder Parametermarke in der Anweisung wiederholt. Es ist nur in den Daten enthalten, die aus dem Prüfprotokoll in Dateien mit begrenzter Satzlänge extrahiert wurden.
- GLOBAL COMMIT: Ausführung einer Anweisung COMMIT innerhalb einer globalen Transaktion
- GLOBAL ROLLBACK: Ausführung einer Anweisung ROLLBACK innerhalb einer globalen Transaktion
- RELEASE SAVEPOINT: Ausführung einer Anweisung RELEASE SAVEPOINT
- ROLLBACK: Ausführung einer Anweisung ROLLBACK
- SAVEPOINT: Ausführung einer Anweisung SAVEPOINT
- STATEMENT: Ausführung einer SQL-Anweisung
- SWITCH USER: Wechseln eines Benutzers innerhalb einer gesicherten Verbindung

Ereignisse für die Kategorie OBJMAINT

- ALTER_OBJECT (wird beim Ändern von geschützten Tabellen oder von Modulen generiert)
- CREATE_OBJECT
- DROP_OBJECT
- RENAME_OBJECT

Ereignisse für die Kategorie SECMAINT

- ADD_DEFAULT_ROLE
- ADD_USER
- ALTER_DEFAULT_ROLE

- ALTER SECURITY POLICY
- ALTER_USER_ADD_ROLE
- ALTER_USER_AUTHENTICATION
- ALTER_USER_DROP_ROLE
- DROP_DEFAULT_ROLE
- DROP_USER
- GRANT
- IMPLICIT_GRANT
- IMPLICIT_REVOKE
- REVOKE
- SET_SESSION_USER
- TRANSFER_OWNERSHIP
- UPDATE_DBM_CFG

Ereignisse für die Kategorie SYSADMIN

- ACTIVATE_DB
- ADD_NODE
- ALTER_BUFFERPOOL
- ALTER_DATABASE
- ALTER_NODEGROUP
- ALTER_TABLESPACE
- BACKUP_DB
- CATALOG_DB
- CATALOG_DCS_DB
- CATALOG_NODE
- CHANGE_DB_COMMENT
- CLOSE_CONTAINER_QUERY
- CLOSE_TABLESPACE_QUERY
- CREATE_BUFFERPOOL
- CREATE_DATABASE
- CREATE_DB_AT_NODE
- CREATE_EVENT_MONITOR
- CREATE_INSTANCE
- CREATE_NODEGROUP
- CREATE_TABLESPACE
- DB2AUD
- DB2AUDIT
- DB2REMOT
- DB2SET
- DB2TRC
- DEACTIVATE_DB
- DELETE_INSTANCE
- DESCRIBE_DATABASE
- DROP_BUFFERPOOL
- DROP_DATABASE

- DROP_EVENT_MONITOR
- DROP_NODEGROUP
- DROP_NODE_VERIFY
- DROP_TABLESPACE
- ENABLE_MULTIPAGE
- ESTIMATE_SNAPSHOT_SIZE
- FETCH_CONTAINER_QUERY
- FETCH_TABLESPACE
- FORCE_APPLICATION
- GET_SNAPSHOT
- GET_TABLESPACE_STATISTICS
- GRANT_DBADM (in Version 9.7 nicht mehr generiert)
- GRANT_DB_AUTH (in Version 9.7 nicht mehr generiert)
- KILLDBM
- LIST_DRDA_INDOUBT_TRANSACTIONS
- LOAD_TABLE
- MERGE_DBM_CONFIG_FILE
- MIGRATE_DB
- MIGRATE_DB_DIR
- MIGRATE_SYSTEM_DIRECTORY
- OPEN_CONTAINER_QUERY
- OPEN_TABLESPACE_QUERY
- PRUNE_RECOVERY_HISTORY
- QUIESCE_TABLESPACE
- READ_ASYNC_LOG_RECORD
- REDISTRIBUTE_NODEGROUP
- RENAME_TABLESPACE
- RESET_ADMIN_CFG
- RESET_DBM_CFG
- RESET_DB_CFG
- RESET_MONITOR
- RESTORE_DB
- REVOKE_DBADM (in Version 9.7 nicht mehr generiert)
- REVOKE_DB_AUTH (in Version 9.7 nicht mehr generiert)
- ROLLFORWARD_DB
- SET_APPL_PRIORITY
- SET_EVENT_MONITOR
- SET_RUNTIME_DEGREE
- SET_TABLESPACE_CONTAINERS
- SINGLE_TABLESPACE_QUERY
- START_DB2
- STOP_DB2
- UNCATALOG_DB
- UNCATALOG_DCS_DB
- UNCATALOG_NODE

- UNLOAD_TABLE
- UPDATE_ADMIN_CFG
- UPDATE_CLI_CONFIGURATION
- UPDATE_DATABASE_VERSION
- UPDATE_DBM_CFG
- UPDATE_DB_CFG
- UPDATE_MON_SWITCHES
- UPDATE_RECOVERY_HISTORY

Ereignisse für die Kategorie VALIDATE

- AUTHENTICATE
- CHECK_GROUP_MEMBERSHIP (ab Version 9.5 nicht mehr generiert)
- GET_USERMAPPING_FROM_PLUGIN
- GET_GROUPS (ab Version 9.5 nicht mehr generiert)
- GET_USERID (ab Version 9.5 nicht mehr generiert)

Kapitel 12. Arbeiten mit der Betriebssystemsicherheit

Betriebssysteme stellen Sicherheitseinrichtungen bereit, die Sie zur Unterstützung der Sicherheit für Ihre Datenbankinstallation verwenden können.

DB2- und Windows-Sicherheit

Bei einer Windows-Domäne handelt es sich um eine Zusammenstellung von Client- und Servercomputern, auf die mit einem bestimmten, eindeutigen Namen verwiesen wird und die eine Benutzerkontendatenbank gemeinsam nutzen, die als Security Access Manager (SAM) bezeichnet wird. Einer der Computer in der Domäne ist der Domänencontroller. Der Domänencontroller verwaltet alle Aspekte der Interaktionen zwischen den Benutzern und der Domäne.

Mithilfe der Informationen in der Benutzerkontendatenbank der Domäne authentifiziert der Domänencontroller Benutzer, die sich über Domänenkonten anmelden. Für jede Domäne ist ein Domänencontroller der primäre Domänencontroller (PDC - Primary Domain Controller). Innerhalb der Domäne kann es außerdem Backup-Domänen-Controller (BDC) geben, welche die Authentifizierung von Benutzerkonten durchführen, wenn kein primärer Domänencontroller vorhanden oder der primäre Domänencontroller nicht verfügbar ist. Backup-Domänen-Controller enthalten eine Kopie der Windows Security Account Manager-Datenbank (SAM), die regelmäßig mit der Masterkopie auf dem primären Domänencontroller synchronisiert wird.

Benutzerkonten, Benutzer-IDs und Kennwörter müssen nur auf dem primären Domänencontroller definiert werden, um auf die Ressourcen der Domäne zugreifen zu können.

Anmerkung: Zweiteilige Benutzer-IDs werden von der Anweisung CONNECT und dem Befehl ATTACH unterstützt. Das Qualifikationsmerkmal der SAM-kompatiblen Benutzer-ID ist ein Name der Form 'Domäne\Benutzer', der eine Länge von maximal 15 Zeichen besitzen kann.

Beim Setup wird bei Installation eines Windows-Servers das Erstellen folgender Einheiten zur Auswahl gestellt:

- Primärer Domänencontroller in einer neuen Domäne
- Backup-Domänen-Controller in einer bekannten Domäne
- Alleinstehender Server in einer bekannten Domäne

Durch die Auswahl „Controller" in einer neuen Domäne wird dieser Server als primärer Domänencontroller eingerichtet.

Der Benutzer kann sich an der lokalen Maschine oder, wenn die betreffende Maschine in einer Windows-Domäne installiert ist, bei der Domäne anmelden. Zur Authentifizierung des Benutzers überprüft DB2 zunächst die lokale Maschine, dann den Domänencontroller der aktuellen Domäne und zuletzt sonstige vertrauenswürdige Domänen, die dem Domänencontroller bekannt sind.

Zur Veranschaulichung dazu ein Beispiel: Angenommen, die DB2-Instanz erfordert eine Serverauthentifizierung. Die Konfiguration sieht folgendermaßen aus:

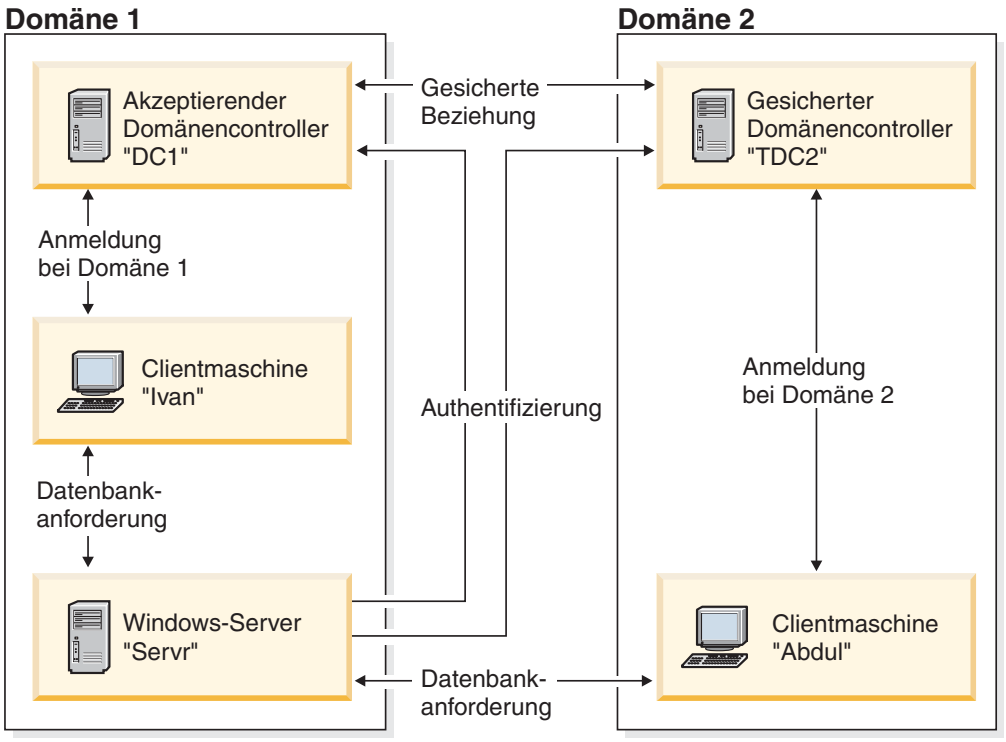


Abbildung 7. Authentifizierung in Windows-Domänen

Jede Maschine verfügt über eine SAM-Sicherheitsdatenbank (Security Access Management). 'DC1' ist der Domänencontroller, auf dem die Clientmaschine 'Ivan' und der DB2-Server 'Servr' registriert sind. 'TDC2' ist für 'DC1' eine sichere Domäne, und die Clientmaschine 'Abdul' ist ein Mitglied der Domäne von 'TDC2'.

Authentifizierungsszenarios

Szenario mit Serverauthentifizierung (Windows)

Das nachfolgende Beispiel dient zur Veranschaulichung der Authentifizierung eines Benutzers über einen Server.

1. Adbul meldet sich an der Domäne TDC2 an (d. h. dieser Client ist in der SAM-Datenbank von TDC2 registriert).
2. Abdul stellt dann die Verbindung zu einer DB2-Datenbank her, die laut Katalog auf SRV3 gespeichert ist:

   ```
   db2 connect to remotedb user Abdul using fredpw
   ```

3. SRV3 stellt fest, wo Abdul bekannt ist. Die API, die zum Abrufen dieser Informationen eingesetzt wird, durchsucht zunächst die lokale Maschine (SRV3) und anschließend den Domänencontroller (DC1), bevor sie mit anderen gesicherten Domänen fortfährt. Der Benutzername Abdul wird auf TDC2 gefunden. Diese Suchreihenfolge setzt einen einzigen Namensbereich (Namespace) für Benutzer und Gruppen voraus.
4. SRV3 unternimmt dann folgende Schritte:
   a. Er überprüft den Benutzernamen und das Kennwort mithilfe von TDC2.
   b. Er ermittelt, ob Abdul ein Administrator ist, indem er TDC2 befragt.
   c. Er zählt alle Gruppen von Abdul auf, indem er TDC2 befragt.

Szenario mit Clientauthentifizierung und einer Windows-Clientmaschine

Das nachfolgende Beispiel dient zur Veranschaulichung der Authentifizierung eines Benutzers über einen Client-Computer.

1. Dale, der Administrator, meldet sich an SRV3 an und ändert den Authentifizierungstyp für die Datenbankinstanz in CLIENT:

   ```
   db2 update dbm cfg using authentication client
   db2stop
   db2start
   ```

2. Ivan meldet sich von der Windows-Clientmaschine aus an der Domäne DC1 an (d. h., er ist in der SAM-Datenbank von DC1 registriert).
3. Ivan stellt dann die Verbindung zu einer DB2-Datenbank her, die laut Katalog auf SRV3 gespeichert ist:

   ```
   DB2 CONNECT to remotedb user Ivan using johnpw
   ```

4. Ivans Maschine überprüft den Benutzernamen und das Kennwort. Die API, die zum Abrufen dieser Informationen eingesetzt wird, durchsucht zunächst die lokale Maschine (Ivan) und anschließend die Domänensteuereinheit (DC1), bevor sie mit anderen gesicherten Domänen fortfährt. Der Benutzername Ivan wird auf DC1 gefunden.
5. Ivans Maschine überprüft dann den Benutzernamen und das Kennwort mithilfe von DC1.
6. SRV3 unternimmt dann folgende Schritte:
   a. Er stellt fest, wo Ivan registriert ist.
   b. Er ermittelt, ob Ivan ein Administrator ist, indem er DC1 befragt.
   c. Er zählt alle Gruppen von Ivan auf, indem er DC1 befragt.

Anmerkung: Stellen Sie sicher, dass der DB2-Sicherheitsservice gestartet wurde, bevor Sie versuchen, eine Verbindung zur DB2-Datenbank herzustellen. Der Sicherheitsservice wird im Rahmen der Windows-Installation installiert. DB2 wird anschließend installiert und als Windows-Dienst (Service) „registriert", jedoch nicht automatisch gestartet. Geben Sie den Befehl `NET START DB2NTSECSERVER` ein, um den DB2-Sicherheitsservice zu starten.

Unterstützung globaler Gruppen (Windows)

Das DB2-Datenbanksystem unterstützt globale Gruppen.

Zur Nutzung globaler Gruppen müssen Sie die gewünschten globalen Gruppen als Mitglieder in eine lokale Gruppe einfügen. Wenn der DB2-Datenbankmanager alle Gruppen aufzählt, deren Mitglied eine Person ist, führt er auch die lokalen Gruppen auf, deren Mitglied der Benutzer indirekt ist (dadurch, dass er in einer globalen Gruppe ist, die selbst wiederum zu einer oder mehreren lokalen Gruppen gehört).

Globale Gruppen können auf zwei Arten verwendet werden:

- Als Mitglieder in einer lokalen Gruppe. Berechtigungen müssen dieser lokalen Gruppe erteilt werden.
- Als eigenständige Gruppe auf einem Domänencontroller. Berechtigungen müssen der globalen Gruppe erteilt werden.

Benutzerauthentifizierung und Gruppeninformationen mit DB2 unter Windows

Einschränkungen für Benutzer- und Gruppennamen (Windows)

Es gibt einige wenige Einschränkungen, die speziell für die Windows-Umgebung zu berücksichtigen sind. Beachten Sie, dass die allgemeinen Namensregeln für DB2-Objekte ebenfalls gelten.

- Bei unter Windows verwendeten Benutzernamen muss die Groß-/Kleinschreibung nicht beachtet werden, jedoch bei den zugehörigen Kennwörtern.
- Benutzernamen und Gruppennamen können aus einer Kombination von Groß- und Kleinbuchstaben eingegeben werden. Allerdings werden sie für die Verwendung innerhalb des DB2-Datenbanksystems normalerweise in Großbuchstaben umgesetzt. Wenn Sie z. B. eine Verbindung zu einer Datenbank herstellen und die Tabelle schema1.tabelle1 erstellen, wird diese Tabelle unter dem Namen SCHEMA1.TABELLE1 in der Datenbank gespeichert. (Wenn Sie Objektnamen in Kleinbuchstaben verwenden wollen, müssen Sie über den Befehlszeilenprozessor entsprechende Befehle absetzen. Dabei müssen die Objektnamen in Anführungszeichen eingeschlossen werden. Alternativ können Sie auch ODBC-Front-End-Tools anderer Hersteller verwenden.)
- Der DB2-Datenbankmanager unterstützt nur einen Namensbereich. Das heißt für eine Umgebung mit gesicherten (vertrauten) Domänen, dass Sie kein Benutzerkonto eines Namens haben sollten, der in mehreren Domänen vorhanden ist oder der in der lokalen SAM-Datenbank der Servermaschine und in einer anderen Domäne vorhanden ist.
- Ein Benutzername sollte nicht mit einem Gruppennamen übereinstimmen.
- Eine lokale Gruppe sollte nicht mit demselben Namen bezeichnet werden wie eine Gruppe auf Domänenebene.

Gruppen- und Benutzerauthentifizierung unter Windows

Benutzer werden unter Windows definiert, indem Benutzerkonten über ein Windows-Verwaltungstool mit dem Namen „Benutzer-Manager" erstellt werden. Eine Gruppe ist ein Konto, das andere Konten enthält, die in diesem Fall als Mitglieder bezeichnet werden.

Gruppen bieten Windows-Administratoren die Möglichkeit, Benutzern einer Gruppe gleichzeitig Zugriffsrechte und Berechtigungen zu erteilen, ohne jeden Benutzer einzeln verwalten zu müssen. Gruppen werden ebenso wie Benutzerkonten in der SAM-Datenbank (SAM - Security Access Manager) definiert und gepflegt.

Es gibt zwei Arten von Gruppen:

- Lokale Gruppen. Eine lokale Gruppe kann Benutzerkonten enthalten, die in der lokalen Kontendatenbank erstellt wurden. Wenn sich die lokale Gruppe auf einer Maschine befindet, die zu einer Domäne gehört, kann die lokale Gruppe außerdem Domänenkonten und Gruppen aus der Windows-Domäne enthalten. Wenn die lokale Gruppe auf einer Workstation erstellt wird, ist sie für diese Workstation spezifisch.
- Globale Gruppen. Eine globale Gruppe ist nur auf einem Domänencontroller vorhanden und enthält Benutzerkonten der SAM-Datenbank der Domäne. Das heißt, eine globale Gruppe kann nur Benutzerkonten aus der Domäne enthalten, in der sie erstellt wurde. Sie kann keine anderen Gruppen als Mitglieder enthalten. Eine globale Gruppe kann in Servern und Workstations der eigenen Domäne und auch in akzeptierenden (vertrauenden) Domänen verwendet werden.

Vertrauensstellungen zwischen Domänen unter Windows

Vertrauensstellungen stellen eine Verwaltungs- und Kommunikationsverbindung zwischen zwei Domänen dar. Eine Vertrauensstellung zwischen zwei Domänen stellt die Möglichkeit bereit, Benutzerkonten und globale Gruppen in einer anderen Domäne zu verwenden als der, in der die Konten definiert sind.

Konteninformationen werden gemeinsam verwendet, um die Gültigkeit von Zugriffsrechten und Berechtigungen von Benutzerkonten und globalen Gruppen ohne erneute Authentifizierung zu bestätigen. Vertrauensstellungen vereinfachen die Benutzerverwaltung, indem sie zwei oder mehr Domänen zu einer administrativen Einheit verbinden.

In einer Vertrauensstellung befinden sich zwei Domänen:

- Die vertrauende Domäne. Diese Domäne vertraut der anderen Domäne in Bezug auf die Authentifizierung von Benutzern für beide Domänen.
- Die vertraute (gesicherte) Domäne. Diese Domäne authentifiziert Benutzer im Auftrag (im Vertrauen) einer anderen Domäne.

Vertrauensstellungen sind nicht transitiv. Dies bedeutet, dass in beide Richtungen zwischen den Domänen explizite Vertrauensstellungen eingerichtet werden müssen. Zum Beispiel muss die vertrauende Domäne nicht unbedingt auch eine vertraute Domäne sein.

Authentifizierung mit Gruppen und Domänensicherheit (Windows)

Das DB2-Datenbanksystem gibt Ihnen die Möglichkeit, beim Erteilen von Zugriffsrechten oder Definieren von Berechtigungsstufen eine lokale Gruppe oder eine globale Gruppe anzugeben.

Informationen zu diesem Vorgang

Ein Benutzer wird als Mitglied einer Gruppe erkannt, wenn das Konto des Benutzers explizit in der lokalen bzw. globalen Gruppe oder implizit durch seine Mitgliedschaft in einer globalen Gruppe definiert ist, die wiederum als Mitglied einer lokalen Gruppe definiert ist.

Der DB2-Datenbankmanager unterstützt die folgenden Gruppenarten:

- Lokale Gruppen
- Globale Gruppen
- Globale Gruppen als Mitglieder einer lokalen Gruppe

  Der DB2-Datenbankmanager zählt die lokalen und globalen Gruppen auf, zu denen der Benutzer gehört, und verwendet dazu die Sicherheitsdatenbank, in der der Benutzer lokalisiert wurde. Das DB2-Datenbanksystem stellt eine Überschreibungsfunktion zur Verfügung, durch die die Ausführung der Gruppenaufzählung auf dem lokalen Windows-Server erzwungen wird, auf dem die DB2-Datenbank installiert ist, unabhängig davon, wo das Benutzerkonto lokalisiert wurde. Diese Überschreibung kann mit den folgenden Befehlen veranlasst werden:

  – Für globale Einstellungen:

    ```
    db2set -g DB2_GRP_LOOKUP=local
    ```

  – Für Instanzeinstellungen:

    ```
    db2set -i instanzname DB2_GRP_LOOKUP=local
    ```

Nach der Ausführung dieses Befehls müssen Sie die DB2-Datenbankinstanz stoppen und erneut starten, um die Änderung in Kraft zu setzen. Erstellen Sie anschließend lokale Gruppen und fügen Sie Domänenkonten oder globale Gruppen in die lokale Gruppe als Mitglieder ein.

Geben Sie Folgendes ein, um alle definierten Variablen der DB2-Profilregistrierdatenbank anzuzeigen:

```
db2set -all
```

Wenn die Variable **DB2_GRP_LOOKUP** der Profilregistrierdatenbank auf den Wert 'local' gesetzt ist, versucht der DB2-Datenbankmanager, nur die Gruppen des Benutzers auf der lokalen Maschine aufzuzählen. Wenn der Benutzer nicht als Mitglied einer lokalen Gruppe oder einer globalen Gruppe definiert ist, die in eine lokale Gruppe verschachtelt ist, schlägt die Aufzählung der Gruppen fehl. Der DB2-Datenbankmanager versucht **nicht**, die Gruppen des Benutzers auf einer anderen Maschine in der Domäne oder auf dem Domänencontroller aufzuzählen.

Wenn der DB2-Datenbankmanager auf einer Maschine ausgeführt wird, die als primärer Domänencontroller (PDC) oder Backup-Domänen-Controller (BDC) in der Ressourcendomäne eingesetzt ist, kann er jeden Domänencontroller in jeder beliebigen sicheren Domäne lokalisieren. Der Grund dafür ist der, dass die Namen der Domänen von Backup-Domänen-Controllern in gesicherten Domänen einer Maschine nur bekannt sind, wenn sie ein Domänencontroller ist.

Anfordern von Benutzergruppeninformationen mithilfe eines Zugriffstokens (Windows)

Ein Zugriffstoken ist ein Objekt, das den Sicherheitskontext eines Prozesses oder Threads beschreibt. Die Informationen in einem Zugriffstoken enthalten die Identität und die Zugriffsrechte des Benutzerkontos, das dem Prozess oder Thread zugeordnet ist.

Wenn Sie sich anmelden, überprüft das System Ihr Kennwort, indem es dieses mit den in einer Sicherheitsdatenbank gespeicherten Informationen vergleicht. Wenn das Kennwort authentifiziert wird, erstellt das System ein Zugriffstoken. Jeder Prozess, der in Ihrem Namen ausgeführt wird, verwendet eine Kopie dieses Zugriffstokens.

Ein Zugriffstoken kann außerdem auf der Grundlage eines im Cache gespeicherten Berechtigungsnachweises empfangen werden. Wenn Sie für das System authentifiziert wurden, werden Ihre Berechtigungsnachweise vom Betriebssystem im Cache gespeichert. Auf das Zugriffstoken der letzten Anmeldung kann im Cache zurückgegriffen werden, wenn es nicht möglich ist, den Domänencontroller zu kontaktieren.

Das Zugriffstoken enthält Informationen über alle Gruppen, denen Sie angehören: lokale Gruppen und verschiedene Domänengruppen (globale Gruppen, lokale Domänengruppen, universelle Gruppen).

Anmerkung: Eine Gruppensuche (Lookup) unter Verwendung der Clientauthentifizierung wird über eine Remoteverbindung nicht unterstützt, selbst wenn die Unterstützung für Zugriffstoken aktiviert ist.

Zur Aktivierung der Unterstützung für Zugriffstoken müssen Sie die Registrierdatenbankvariable **DB2_GRP_LOOKUP** mithilfe des Befehls **db2set** aktualisieren. **DB2_GRP_LOOKUP** kann über bis zu zwei durch Komma getrennte Parameter verfügen:

- Der erste Parameter dient zur konventionellen Gruppensuche und kann folgende Werte annehmen: " ", "LOCAL" oder "DOMAIN".
- Der zweite Parameter dient zur Gruppensuche über Token und kann die folgenden Werte annehmen: "TOKEN", "TOKENDOMAIN" oder "TOKENLOCAL".

Wird der zweite Parameter (TOKEN, TOKENDOMAIN oder TOKENLOCAL) angegeben, hat dieser Parameter Vorrang vor der konventionellen Gruppenaufzählung. Schlägt die tokenbezogene Gruppenaufzählung fehl, erfolgt eine konventionelle Gruppensuche, wenn der erste Parameter für **DB2_GRP_LOOKUP** angegeben wurde.

Die Werte TOKEN, TOKENDOMAIN und TOKENLOCAL haben folgende Bedeutung:

- TOKENLOCAL

  Das Token wird für die Gruppenaufzählung auf dem lokalen System verwendet. Dies entspricht der konventionellen Gruppensuche mit "LOCAL".
- TOKENDOMAIN

  Das Token wird für die Gruppenaufzählung an der Position verwendet, an der der Benutzer definiert ist (auf dem lokalen System bei einem lokalen Benutzer und in der Domäne bei einem Domänenbenutzer). Dies entspricht der konventionellen Gruppensuche mit " " oder "DOMAIN".
- TOKEN

  Das Token wird für die Gruppenaufzählung in der Domäne und auf dem lokalen System verwendet. Bei einem lokalen Benutzer handelt es sich bei den zurückgegebenen Gruppen um lokale Gruppen. Bei einem Domänenbenutzer sind unter den zurückgegebenen Gruppen sowohl Domänengruppen als auch lokale Gruppen enthalten. Bei der konventionellen Gruppensuche gibt es keine Entsprechung für TOKEN.

Die folgende Einstellung von **DB2_GRP_LOOKUP** aktiviert beispielsweise die Zugriffstokenunterstützung für die Aufzählung lokaler Gruppen:

```
db2set DB2_GRP_LOOKUP=LOCAL,TOKENLOCAL
```

Der folgende Beispielbefehl aktiviert die Unterstützung für Zugriffstoken zur Aufzählung von Gruppen sowohl auf dem lokalen System als auch an der Position, an der das Benutzerkonto definiert ist (wenn das Konto auf der Domänenebene definiert ist):

```
db2set DB2_GRP_LOOKUP=,TOKEN
```

Der nachfolgende Beispielbefehl schließlich aktiviert die Unterstützung für Zugriffstoken zur Aufzählung von Domänengruppen an der Position, an der die Benutzer-ID definiert ist:

```
db2set DB2_GRP_LOOKUP=DOMAIN,TOKENDOMAIN
```

Anmerkung: Die Unterstützung für Zugriffstoken kann bei allen Authentifizierungstypen außer der Clientauthentifizierung (CLIENT) aktiviert werden.

Umgebungsvariable DB2_GRP_LOOKUP und DB2-Gruppenaufzählung (Windows)

Unter Windows kann ein Benutzer Gruppen angehören, die auf Domänenebene und/oder auf dem lokalen System definiert sind.

Die Umgebungsvariable **DB2_GRP_LOOKUP** steuert, ob Gruppen auf dem lokalen System aufgezählt werden und an welcher Position Benutzer definiert werden (auf dem lokalen System, wenn es sich um lokale Benutzer handelt, oder auf Domänenebene, wenn es sich um Domänenbenutzer handelt). Beim Erteilen von Berechti-

gungen und Zugriffsrechten muss der Sicherheitsadministator darauf achten, dass die Variable **DB2_GRP_LOOKUP** wie beabsichtigt definiert ist und die Benutzer die für sie vorgesehene Berechtigung erhalten.

Ist die Variable **DB2_GRP_LOOKUP** der Profilregistrierdatenbank nicht definiert, gilt Folgendes:

1. Das DB2-Datenbanksystem versucht zunächst, den Benutzer auf derselben Maschine zu finden.
2. Ist der Benutzername lokal definiert, wird auch die Authentifizierung lokal ausgeführt.
3. Wird der Benutzer auf dem lokalen System nicht gefunden, versucht das DB2-Datenbanksystem, den Benutzernamen in der zugehörigen Domäne und anschließend in sicheren (vertrauenswürdigen) Domänen zu lokalisieren.

Die folgende Beispielsituation, in der **DB2_GRP_LOOKUP** nicht definiert ist, dient zur Veranschaulichung:

1. Der Domänenbenutzer DBENUTZER1 gehört der lokalen Gruppe GRUPPE1 an.
2. Der Sicherheitsadministrator (mit der Berechtigung SECADM) erteilt GRUPPE1 die Berechtigung DBADM.

   ```
   GRANT DBADM ON datenbank TO GROUP GRUPPE1
   ```

3. Da **DB2_GRP_LOOKUP** nicht definiert ist, werden Gruppen an der Position aufgezählt, an der Benutzer definiert sind. Die Gruppen für DBENUTZER1 werden somit auf Domänenebene aufgezählt. Da der Benutzer DBENUTZER1 der Gruppe GRUPPE1 auf Domänenebene nicht angehört, erhält DBENUTZER1 die Berechtigung DBADM nicht.

Dazu ein weiteres komplexeres Szenario zum Befehl **UPGRADE DATABASE**, bei dem **DB2_GRP_LOOKUP** nicht definiert ist:

1. Der Domänenbenutzer DBENUTZER2 gehört der lokalen Administratorgruppe an.
2. Der Konfigurationsparameter **sysadm_group** ist nicht definiert. Die Mitglieder der lokalen Administratorgruppe verfügen deshalb automatisch über die Berechtigung SYSADM.
3. Der Benutzer DBENUTZER2 kann den Befehl **UPGRADE DATABASE** ausgeben, da DBENUTZER2 über die Berechtigung SYSADM verfügt. Mit dem Befehl **UPGRADE DATABASE** wird der Gruppe SYSADM (in diesem Fall die Administratorgruppe) die Berechtigung DBADM für die Datenbank erteilt, die aufgerüstet wird.
4. Da **DB2_GRP_LOOKUP** nicht definiert ist, werden Gruppen an der Position aufgezählt, an der Benutzer definiert sind. Die Gruppen für DBENUTZER2 werden somit auf Domänenebene aufgezählt. Da DBENUTZER2 der Administratorgruppe auf Domänenebene nicht angehört, erhält DBENUTZER2 die Berechtigung DBADM nicht.

Für dieses Szenario gibt es verschiedene Lösungen. Nehmen Sie z. B. eine der folgenden Änderungen vor:

- Definieren Sie Folgendes: **DB2_GRP_LOOKUP** = `local`
- Fügen Sie die Benutzer, die über die Berechtigung DBADM verfügen sollen, der Administratorgruppe oder der Gruppe GRUPPE1 auf dem Domänencontroller hinzu.

Mit der Tabellenfunktion SYSPROC.AUTH_LIST_AUTHORITIES_FOR_AUTHID können Sie die Berechtigungen überprüfen, über die die einzelnen Benutzer verfügen. Dazu folgendes Beispiel für DBENUTZER1:

```
SELECT AUTHORITY, D_USER, D_GROUP, D_PUBLIC, ROLE_USER, ROLE_GROUP, ROLE_PUBLIC, D_ROLE
   FROM TABLE (SYSPROC.AUTH_LIST_AUTHORITIES_FOR_AUTHID ('DBENUTZER1', 'U') ) AS T
   ORDER BY AUTHORITY
```

Mit der Tabellenfunktion SYSPROC.AUTH_LIST_AUTHORITIES_FOR_AUTHID können Sie die Gruppen überprüfen, denen ein Benutzer laut DB2-Datenbankmanager zugeordnet ist. Dazu folgendes Beispiel für DBENUTZER1:

```
SELECT * FROM TABLE (SYSPROC.AUTH_LIST_GROUPS_FOR_AUTHID ('DBENUTZER1')) AS T
```

Anmerkung: Wenn Sie einen Gruppennamen sowohl auf Domänenebene als auch auf dem lokalen System verwenden, kann dies zu Verwechslungen führen, da der DB2-Datenbankmanager Gruppennamen nicht vollständig qualifiziert.

Authentifizierung mit einer geordneten Domänenliste

Benutzer-IDs können mehr als einmal in einer gesicherten (vertrauten) Domänengesamtstruktur definiert sein. Eine gesicherte Domänengesamtstruktur ist eine Gruppe von Domänen, die durch ein Netz miteinander verbunden sind.

Informationen zu diesem Vorgang

Es ist möglich, dass ein Benutzer in einer Domäne die gleiche Benutzer-ID wie ein anderer Benutzer in einer anderen Domäne besitzt. Dies kann beim Versuch einer der folgenden Aktionen zu Schwierigkeiten führen:

- Authentifizierung mehrerer Benutzer, die die gleiche Benutzer-ID, jedoch in verschiedenen Domänen haben
- Gruppensuche zum Zweck des Erteilens und Widerrufens von Zugriffsrechten für Gruppen
- Überprüfung von Kennwörtern
- Steuerung des Datenaustauschs im Netz

Zur Vermeidung von Schwierigkeiten aufgrund der möglichen Existenz mehrerer Benutzer mit der gleichen Benutzer-ID in einer Domänengesamtstruktur sollten Sie eine geordnete Domänenliste verwenden, wie sie in der Registrierdatenbankvariablen **DB2DOMAINLIST** mithilfe des Befehls **db2set** definiert werden kann. Bei der Definition der Reihenfolge müssen die Domänen, die in die Liste aufgenommen werden sollen, durch Kommas voneinander getrennt werden. Sie müssen eine bewusste Entscheidung im Hinblick auf die Reihenfolge treffen, in der die Domänen bei der Authentifizierung von Benutzern durchsucht werden.

Die Benutzer-IDs, die sich in Domänen weiter hinten in der Liste befinden, müssen von Ihnen umbenannt werden, wenn Sie für den Zugriff authentifiziert werden sollen.

Eine Steuerung des Zugriffs kann über die Domänenliste erfolgen. Wenn zum Beispiel die Domäne eines Benutzers nicht in der Liste enthalten ist, erhält der Benutzer keine Berechtigung zur Herstellung einer Verbindung.

Anmerkung: Die Registrierdatenbankvariable **DB2DOMAINLIST** ist nur wirksam, wenn in der Konfiguration des Datenbankmanagers eine CLIENT-Authentifizierung definiert ist, und wird benötigt, wenn eine Einzelanmeldung (Single Sign-on) über einen Windows-Desktop in einer Windows-Domänenumgebung erforderlich ist. Die Registrierdatenbankvariable **DB2DOMAINLIST** wird von einigen Versionen von

DB2-Servern unterstützt. Allerdings wird die Registrierdatenbankvariable **DB2DOMAINLIST** nicht umgesetzt, wenn sich weder der Client noch der Server in einer Windows-Umgebung befinden.

Unterstützung der Domänensicherheit (Windows)

Das folgende Beispiel erläutert, wie das DB2-Datenbankmanagementsystem die Windows-Domänensicherheit unterstützten kann. Die Verbindung funktioniert, weil sich der Benutzername und die lokale Gruppe in derselben Domäne befinden.

Die Verbindung funktioniert im folgenden Szenario, weil sich der Benutzername und die lokale oder globale Gruppe in derselben Domäne befinden.

Beachten Sie hierbei, dass der Benutzername und die lokale oder globale Gruppe nicht in der Domäne definiert sein müssen, in der der Datenbankserver ausgeführt wird. Sie müssen sich jedoch in derselben Domäne befinden.

Tabelle 52. Erfolgreiche Verbindung mit einem Domänencontroller

Domäne1	**Domäne2**
Zu Domäne2 besteht eine Vertrauensstellung.	• Zu Domäne1 besteht eine Vertrauensstellung. • Die lokale oder globale Gruppe `grp2` wurde definiert. • Der Benutzername `id2` wurde definiert. • Der Benutzername `id2` gehört zu `grp2`.

Der DB2-Server wird in dieser Domäne ausgeführt. Die folgenden DB2-Befehle werden über dieses System abgesetzt:
`REVOKE CONNECT ON db FROM public`
`GRANT CONNECT ON db TO GROUP grp2`
`CONNECT TO db USER id2`	

Die lokale oder globale Domäne wird durchsucht, `id2` kann jedoch nicht gefunden werden. Die Einrichtung für die Domänensicherheit wird durchsucht.	
	Der Benutzername `id2` wird in dieser Domäne gefunden. DB2 ruft zusätzliche Informationen zu diesem Benutzernamen (der zu der Gruppe `grp2` gehört) ab.
Die Verbindung funktioniert, weil sich der Benutzername und die lokale oder globale Gruppe in derselben Domäne befinden.	

Definieren von Benutzern mit der Berechtigung SYSADM (Windows)

Bestimmte Benutzer verfügen über die Berechtigung SYSADM, wenn der Konfigurationsparameter **sysadm_group** des Datenbankmanagers nicht definiert ist (d. h. gleich null ist).

Dabei handelt es sich um folgende Benutzer:

- Mitglieder der lokalen Administratorgruppe
- Mitglieder der Gruppe 'Administratoren' auf dem Domänencontroller, wenn der DB2-Datenbankmanager so konfiguriert ist, dass er Gruppen für Benutzer an der Position aufzählt, an der die Benutzer definiert sind (die Gruppenaufzählung können Sie über die Umgebungsvariable **DB2_GRP_LOOKUP** konfigurieren).

- Mitglieder der Gruppe DB2ADMNS, wenn die erweiterte Windows-Sicherheit aktiviert ist. Die Position der Gruppe DB2ADMNS wird bei der Installation festgelegt.
- Das lokale Systemkonto (LocalSystem)

In manchen Fällen ist das zuvor beschriebene Standardverhalten nicht wünschenswert. Mithilfe des Konfigurationsparameters **sysadm_group** des Datenbankmanagers können Sie dieses Verhalten überschreiben, indem Sie eine der folgenden Methoden verwenden:

- Erstellen Sie eine lokale Gruppe auf dem DB2-Serversystem und fügen Sie dieser Benutzer (Domänenbenutzer oder lokale Benutzer) hinzu, denen Sie die Berechtigung SYSADM erteilen wollen. Der DB2-Datenbankmanager sollte so konfiguriert werden, dass er Gruppen für den Benutzer auf dem lokalen System aufzählt.
- Erstellen Sie eine Domänengruppe und fügen Sie dieser Benutzer hinzu, denen Sie die Berechtigung SYSADM erteilen wollen. Der DB2-Datenbankmanager sollte so konfiguriert werden, dass er Gruppen für Benutzer an der Position aufzählt, an der die Benutzer definiert sind.

Aktualisieren Sie anschließend den Konfigurationsparameter **sysadm_group** des Datenbankmanagers mit dieser Gruppe, indem Sie die folgenden Befehle verwenden:

```
DB2 UPDATE DBM CFG USING SYSADM_GROUP gruppenname
DB2STOP
DB2START
```

Unterstützung für das lokale Systemkonto (LocalSystem) unter Windows

Auf Windows-Plattformen unterstützt das DB2-Datenbanksystem Anwendungen, die unter dem Kontext des lokalen Systemkontos (LSA) mit impliziter lokaler Verbindung ausgeführt werden. Die Berechtigungs-ID für das lokale Systemkonto ist SYSTEM.

Das lokale Systemkonto wird als Systemadministratorkonto (mit der Berechtigung SYSADM) betrachtet, wenn der Konfigurationsparameter **sysadm_group** des Datenbankmanagers auf den Wert NULL gesetzt ist.

Wenn über die Anwendungen, die unter dem Kontext des lokalen Systemkontos ausgeführt, Datenbankaktionen vorgenommen werden sollen, die nicht in den Geltungsbereich der Berechtigung SYSADM fallen, müssen Sie das lokale Systemkonto mit den erforderlichen Datenbankzugriffsrechten oder -berechtigungen ausstatten. Erteilen Sie dem lokalen Systemkonto z. B. mit der Anweisung GRANT (Datenbankberechtigungen) die Berechtigung DBADM, wenn eine Anwendung Datenbankadministratorberechtigungen erfordert.

Entwickler, die Anwendungen zur Ausführung unter diesem Konto schreiben, müssen sich über die Einschränkungen im DB2-Datenbanksystem für Objekte mit Schemanamen, die mit „SYS" beginnen, im Klaren sein. Wenn Ihre Anwendungen DDL-Anweisungen enthalten, durch die DB2-Datenbankobjekte erstellt werden, sollten diese daher so geschrieben sein, dass sie folgende Voraussetzungen erfüllen:

- Für statische Abfragen sollten sie mit einem anderen Wert für die QUALIFIER-Optionen als dem Standardwert (SYSTEM) gebunden werden.
- Für dynamische Abfragen sollten die zu erstellenden Objekte explizit mit einem vom DB2-Datenbankmanager unterstützten Schemanamen qualifiziert werden

oder das Register CURRENT SCHEMA muss auf einen Schemanamen gesetzt werden, der vom DB2-Datenbankmanager unterstützt wird.

Gruppeninformationen für das lokale Systemkonto (LocalSystem) werden bei der ersten Suchanforderung für Gruppen (Lookup) nach dem Start der DB2-Datenbankinstanz erfasst und erst wieder aktualisiert, wenn die Instanz erneut gestartet wird.

Erweiterte Windows-Sicherheit mithilfe der Gruppen DB2ADMNS und DB2USERS

Die erweiterte Sicherheit ist standardmäßig in allen DB2-Datenbankprodukten unter Windows-Betriebssystemen mit Ausnahme von IBM Data Server Runtime Client und DB2-Treibern aktiviert. IBM Data Server Runtime Client und DB2-Treiber unterstützen die erweiterte Sicherheit auf Windows-Plattformen nicht.

Bei der Installation von DB2-Datenbankprodukten erscheint in der Anzeige **Betriebssystemsicherheit für DB2-Objekte** das Kontrollkästchen **Betriebssystemsicherheit aktivieren**. Wenn Sie diese Option nicht inaktivieren, erstellt das Installationsprogramm die beiden neuen Gruppen DB2ADMNS und DB2USERS. Die Namen DB2ADMNS und DB2USERS sind die Standardnamen der Gruppen. Bei der Installation können Sie optional andere Namen für diese Gruppen angeben (wenn Sie eine nicht überwachte Installation auswählen, können Sie diese Namen in der Antwortdatei für die Installation ändern). Wenn Sie sich entscheiden, Gruppen zu verwenden, die in Ihrem System bereits vorhanden sind, müssen Sie beachten, dass die Zugriffsrechte für diese Gruppen modifiziert werden. Ihnen werden nach Bedarf die Zugriffsrechte erteilt, die in der nachfolgenden Tabelle aufgeführt sind. Es wichtig zu beachten, dass diese Gruppen für den Schutz auf der *Betriebssystemebene* verwendet werden und in keiner Weise den DB2-Berechtigungsstufen (z. B. SYSADM, SYSMAINT und SYSCTRL) zugeordnet sind. Allerdings kann Ihr Datenbankadministrator anstelle der Standardgruppe für Administratoren die Gruppe DB2ADMNS für eine oder auch alle DB2-Berechtigungsstufen verwenden. Dies liegt im Ermessen des Installationsverantwortlichen oder des Administrators. Es empfiehlt sich, bei Angabe einer SYSADM-Gruppe die Gruppe DB2ADMNS anzugeben. Dies kann während der Installation oder nachfolgend durch einen Administrator eingerichtet werden.

Anmerkung: Sie können Ihre DB2-Administratorengruppe (DB2ADMNS oder den Namen, den Sie bei der Installation auswählen) und Ihre DB2-Benutzergruppe (DB2USERS oder den Namen, den Sie bei der Installation auswählen) entweder als lokale Gruppen oder Domänengruppen angeben. Beide Gruppen müssen vom selben Typ sein, das heißt, beide müssen entweder lokale Gruppen oder Domänengruppen sein.

Wenn Sie den Computernamen ändern und die Computergruppen DB2ADMNS und DB2USERS lokale Computergruppen sind, müssen Sie die globalen Registrierdatenbanken DB2_ADMINGROUP und DB2_USERSGROUP aktualisieren. Zum Aktualisieren der Registrierdatenbankvariablen nach dem Umbenennen und erneuten Starten des Computers führen Sie den folgenden Befehl aus:

1. Öffnen Sie eine Eingabeaufforderung.
2. Führen Sie den Befehl **db2extsec** zum Aktualisieren der Sicherheitseinstellungen:

   ```
   db2extsec -a neuer computername\DB2ADMNS -u neuer computername\DB2USERS
   ```

Anmerkung: Wenn die erweiterte Sicherheit für DB2-Datenbankprodukte unter Windows Vista aktiviert wird, können nur Benutzer, die Mitglied der Gruppe DB2ADMNS sind, die grafischen DB2-Verwaltungstools ausführen. Darüber hinaus müssen die Mitglieder der Gruppe DB2ADMNS die Tools mit den vollständigen Administratorrechten starten. Hierzu müssen Sie mit der rechten Maustaste auf den Direktaufruf klicken und dann die Option zur Ausführung als Administrator auswählen.

Durch die Gruppen DB2ADMNS und DB2USERS erteilte Berechtigungen

Die Gruppen DB2ADMNS und DB2USERS statten Mitglieder mit den folgenden Berechtigungen aus:

- DB2ADMNS

  Vollständige Steuerung über alle DB2-Objekte (siehe folgende Liste der geschützten Objekte)
- DB2USERS

  Lese- und Ausführungszugriff auf alle DB2-Objekte, die sich im Installationsverzeichnis und im Instanzverzeichnis befinden, jedoch keinen Zugriff auf Objekte unter dem Datenbanksystemverzeichnis sowie begrenzten Zugriff auf IPC-Ressourcen

  Für bestimmte Objekte können je nach Bedarf zusätzliche Zugriffsrechte verfügbar sein (z. B. Zugriffsrechte zum Schreiben, Hinzufügen und Aktualisieren usw.). Mitglieder dieser Gruppe haben keinen Zugriff auf Objekte unter dem Datenbanksystemverzeichnis.

  Anmerkung: Die Bedeutung des Ausführungszugriffs hängt von der Art des Objekts ab. Bei Dateien mit den Erweiterungen **.dll** oder **.exe** bedeutet Ausführungszugriff, dass Sie berechtigt sind, die jeweilige Datei auszuführen. Bei einem Verzeichnis bedeutet dieser Zugriff, dass Sie berechtigt sind, das Verzeichnis zu durchqueren.

Im Idealfall sollten alle DB2-Administratoren Mitglieder der Gruppe DB2ADMNS (sowie Mitglieder der lokalen Administratorengruppe) sein, jedoch ist dies keine strikte Voraussetzung. Alle anderen Personen, die Zugriff auf das DB2-Datenbanksystem benötigen, *müssen* Mitglied der Gruppe DB2USERS sein. Gehen Sie wie folgt vor, um einer dieser Gruppen einen Benutzer hinzuzufügen:

1. Starten Sie das Verwaltungstool für Benutzer und Kennwörter.
2. Wählen Sie den hinzuzufügenden Benutzernamen in der Liste aus.
3. Klicken Sie **Eigenschaften** an. Klicken Sie im Eigenschaftenfenster die Registerkarte für Gruppenmitgliedschaft an.
4. Wählen Sie das Optionsfeld **Andere** aus.
5. Wählen Sie die gewünschte Gruppe in der Dropdown-Liste aus.

Hinzufügen der erweiterten Sicherheit nach der Installation (Befehl 'db2extsec')

Wenn das DB2-Datenbanksystem ohne Aktivierung der erweiterten Sicherheit (Extended Security) installiert wurde, können Sie diese aktivieren, indem Sie den Befehl **db2extsec** ausführen. . Zur Ausführung des Befehls **db2extsec** müssen Sie Mitglied der lokalen Administratorgruppe sein, sodass Sie berechtigt sind, die Zugriffssteuerungsliste (ACL) der geschützten Objekte zu modifizieren.

Sie können den Befehl **db2extsec** mehrfach ausführen, falls erforderlich. Wenn Sie dies jedoch tun, können Sie die erweiterte Sicherheit nur dann wieder inaktivieren, wenn Sie den Befehl **db2extsec –r** direkt im Anschluss an *jede* Ausführung des Befehls **db2extsec** ausführen.

Entfernen der erweiterten Sicherheit

Vorsicht:
Entfernen Sie die erweiterte Sicherheit nicht, nachdem sie aktiviert wurde, sofern dies nicht absolut notwendig ist.

Sie können die erweiterte Sicherheit entfernen, indem Sie den Befehl **db2extsec -r** ausführen. Dies ist jedoch nur dann erfolgreich, wenn keine anderen Datenbankoperationen (z. B. Erstellen einer Datenbank, Erstellen einer neuen Instanz, Hinzufügen von Tabellenbereichen usw.) nach der Aktivierung der erweiterten Sicherheit ausgeführt wurden. Die sicherste Methode zum Entfernen der Option für die erweiterte Sicherheit besteht darin, das DB2-Datenbanksystem zu deinstallieren, alle relevanten DB2-Verzeichnisse (einschließlich der Datenbankverzeichnisse) zu löschen und das DB2-Datenbanksystem anschließend erneut zu installieren, ohne die erweiterte Sicherheit zu aktivieren.

Geschützte Objekte

Die folgenden *statischen* Objekte können mithilfe der Gruppen DB2ADMNS und DB2USERS geschützt werden:

- Dateisystem
  - Datei
  - Verzeichnis
- Services (Dienste)
- Registrierungsschlüssel

Die folgenden *dynamischen* Objekte können mithilfe der Gruppen DB2ADMNS und DB2USERS geschützt werden:

- Ressourcen der Interprozesskommunikation (IPC-Ressourcen):
  - Pipes
  - Semaphore
  - Ereignisse
- Gemeinsam genutzter Speicher

Zugriffsrechte der Gruppen DB2ADMNS und DB2USERS

In der folgenden Tabelle sind die Zugriffsrichte aufgelistet, die den Gruppen DB2ADMNS und DB2USERS zugewiesen sind:

Tabelle 53. Zugriffsrechte für die Gruppen DB2ADMNS und DB2USERS

Zugriffsrecht	DB2ADMNS	DB2USERS	Erläuterung
Tokenobjekt erstellen (SeCreateTokenPrivilege)	J	N	Tokenbearbeitung (für bestimmte Operationen zur Tokenbearbeitung erforderlich und bei der Authentifizierung und Autorisierung verwendet)
Token auf Prozessebene ersetzen (SeAssignPrimaryTokenPrivilege)	J	N	Prozesserstellung als anderer Benutzer

Tabelle 53. Zugriffsrechte für die Gruppen DB2ADMNS und DB2USERS (Forts.)

Zugriffsrecht	DB2ADMNS	DB2USERS	Erläuterung
Quoten anheben (SeIncreaseQuotaPrivilege)	J	N	Prozesserstellung als anderer Benutzer
Als Teil des Betriebssystems handeln (SeTcbPrivilege)	J	N	LogonUser (vor Windows XP zur Ausführung der API 'LogonUser' zu Authentifizierungszwecken erforderlich)
Sicherheitsprotokolle generieren (SeSecurityPrivilege)	J	N	Bearbeitung von Prüf- und Sicherheitsprotokollen
Eigentumsrecht an Dateien und anderen Objekten übernehmen (SeTakeOwnershipPrivilege)	J	N	Modifizieren von Zugriffssteuerlisten (ACLs) von Objekten
Ausführungspriorität anheben (SeIncreaseBasePriorityPrivilege)	J	N	Modifizieren des Prozessarbeitsbereichs
Backup von Dateien und Verzeichnissen durchführen (SeBackupPrivilege)	J	N	Bearbeitung von Profilen/ Registrierdatenbank (zur Ausführung bestimmter Bearbeitungsroutinen für Benutzerprofile und die Registrierung erforderlich: LoadUserProfile, RegSaveKey(Ex), RegRestoreKey, RegReplaceKey, RegLoadKey(Ex))
Restore von Dateien und Verzeichnissen ausführen (SeRestorePrivilege)	J	N	Bearbeitung von Profilen/ Registrierdatenbank (zur Ausführung bestimmter Bearbeitungsroutinen für Benutzerprofile und die Registrierung erforderlich: LoadUserProfile, RegSaveKey(Ex), RegRestoreKey, RegReplaceKey, RegLoadKey(Ex))
Programme debuggen (SeDebugPrivilege)	J	N	Tokenbearbeitung (für bestimmte Operationen zur Tokenbearbeitung erforderlich und bei der Authentifizierung und Autorisierung verwendet)
Prüf- und Sicherheitsprotokoll verwalten (SeAuditPrivilege)	J	N	Generierung von Prüfprotokolleinträgen
Als Service anmelden (SeServiceLogonRight)	J	N	Ausführung von DB2 als Service (Dienst)
Auf diesen Computer über das Netz zugreifen (SeNetworkLogonRight)	J	J	Zulassen von Netzberechtigungsnachweisen (ermöglicht dem DB2-Datenbankmanager die Verwendung der Option LOGON32_LOGON_NETWORK zur Authentifizierung, die Auswirkungen auf die Leistung hat)
Identität eines Clients nach Authentifizierung vortäuschen (SeImpersonatePrivilege)	J	N	Vortäuschen der Identität eines Clients (für Windows erforderlich, damit bestimmte APIs die Identität von DB2-Clients vortäuschen können: ImpersonateLoggedOnUser, ImpersonateSelf, RevertToSelf usw.)
Seiten im Speicher sperren (SeLockMemoryPrivilege)	J	N	Unterstützung großer Seiten
Globale Objekte erstellen (SeCreateGlobalPrivilege)	J	J	Terminal-Server-Unterstützung (unter Windows erforderlich)

Aspekte zu Windows 2008 und Windows Vista oder höher: Benutzerzugriffssteuerungsfunktion (User Access Control, UAC)

Die Benutzerzugriffssteuerungsfunktion (User Access Control, UAC) von Windows 2008, Windows Vista und Windows 7 hat die nachfolgend erläuterten Auswirkungen auf das DB2-Datenbanksystem.

Starten von Anwendungen mit vollständigen Administratorrechten

Unter Windows 2008, Windows Vista und Windows 7 werden Anwendungen standardmäßig nur mit den Standardbenutzerrechten gestartet, auch wenn der Benutzer ein lokaler Administrator ist. Wenn Sie eine Anwendung mit weitreichenderen Zugriffsrechten starten wollen, müssen Sie den Befehl über ein Befehlsfenster ausführen, das mit den vollständigen verwaltungsbezogenen Zugriffsrechten ausgeführt wird. Der DB2-Installationsprozess erstellt eine Verknüpfung mit dem Namen "Befehlsfenster - Administrator" speziell für Windows 2008-, Windows Vista- und Windows 7-Benutzer. Es wird empfohlen, das Fenster über diese Verknüpfung zu starten, wenn Sie Administratorbefehle ausführen möchten.

Wenn Sie nicht über die vollständigen verwaltungsbezogenen Zugriffsrechte verfügen und versuchen, DB2-Verwaltungstasks über eine Eingabeaufforderung oder ein grafisch orientiertes Tool unter Windows 2008, Windows Vista und Windows 7 auszuführen, können verschiedene Fehlernachrichten ausgegeben werden, die implizieren, dass Ihnen der Zugriff verweigert wurde und die Tasks nicht erfolgreich ausgeführt werden können.

Um festzustellen, ob die Aktion, die Sie ausführen wollen, als Verwaltungstask betrachtet wird, überprüfen Sie, ob Folgendes auf sie zutrifft:

- Sie erfordert die Berechtigung SYSADM, SYSCTRL oder SYSMAINT.
- Sie ändert Registrierungsschlüssel unter dem Zweig HKLM in der Registrierdatenbank.
- Sie schreibt in die Verzeichnisse unter dem Verzeichnis für Programmdateien (`Program Files`).

Zum Beispiel werden die folgenden Aktionen alle als Verwaltungstasks betrachtet:

- Erstellen und Löschen von DB2-Instanzen
- Starten und Stoppen von DB2-Instanzen
- Erstellen von Datenbanken
- Aktualisieren von Konfigurationsparametern des Datenbankmanagers oder Konfigurationsparametern des DB2-Verwaltungsservers (DAS)
- Aktualisieren von CLI-Konfigurationsparametern und Konfigurieren von Systemdatenquellennamen (DSN)
- Starten der DB2-Tracefunktion
- Ausführen des Dienstprogramms **db2pd**
- Ändern von Variablen der DB2-Profilregistrierdatenbank

Zur Lösung des Problems müssen Sie DB2-Verwaltungstasks über eine Eingabeaufforderung bzw. ein grafisch orientiertes Tool ausführen, die bzw. das mit den vollständigen Administratorrechten ausgeführt wird. Zum Starten einer Eingabeaufforderung oder eines grafisch orientierten Tools mit vollständigen Administratorrechten klicken Sie die Verknüpfung mit der rechten Maustaste an und wählen die Option **Als Administrator ausführen** aus.

Anmerkung: Wenn die erweiterte Sicherheit aktiviert ist, müssen Sie darüber hinaus Mitglied der Gruppe DB2ADMNS sein, um grafische Verwaltungstools (z. B. das IBM Data Studio) starten zu können.

Speicherposition von Benutzerdaten

Benutzerdaten (z. B. Dateien unter den Instanzverzeichnissen) werden in `ProgramData\IBM\DB2\`*`name_der_kopie`* gespeichert. Dabei ist *name_der_kopie* der Name der DB2-Kopie. (Standardmäßig hat die erste installierte Kopie den Namen DB2COPY1.) Unter anderen Windows-Versionen als Windows 2008, Windows Vista und Windows 7 werden Benutzerdaten im Verzeichnis `Dokumente und Einstellungen\All Users\Application Data\IBM\DB2\`*`Name_der_Kopie`* gespeichert.

DB2- und UNIX-Sicherheit

Bei UNIX-Plattformen sind bestimmte Sicherheitsaspekte zu beachten.

Die DB2-Datenbank unterstützt keine direkte Funktion der Rootberechtigung als Datenbankadministrator. Sie sollten den Befehl **su - <instanzeigner>** als Datenbankadministrator verwenden.

Aus Sicherheitsgründen sollten Sie den Instanznamen nicht als Namen der Gruppe für abgeschirmte Funktionen (Fenced ID) verwenden. Wenn Sie jedoch nicht beabsichtigen, abgeschirmte UDFs oder gespeicherte Prozeduren zu verwenden, können Sie die Fenced ID auf den Instanznamen setzen, anstatt eine weitere Benutzer-ID zu erstellen.

Es wird empfohlen, eine Benutzer-ID zu erstellen, die als dieser Gruppe zugeordnet erkannt wird. Der Benutzer für abgeschirmte UDFs und gespeicherte Prozeduren wird als Parameter der Prozedur zur Instanzerstellung angegeben (**db2icrt ... -u <AbgeschirmtID>**). Dies ist nicht erforderlich, wenn Sie die DB2-Clients oder das DB2 Software Developer's Kit installieren.

DB2- und Linux-Sicherheit

Bei Linux-Plattformen sind bestimmte Sicherheitsaspekte zu beachten.

Unterstützung für Kennwortänderung (Linux)

DB2-Datenbankprodukte stellen eine Unterstützung für die Änderung von Kennwörtern unter Linux-Betriebssystemen bereit.

Diese Unterstützung wird durch die Verwendung der Sicherheits-Plug-in-Bibliotheken mit den Namen `IBMOSchgpwdclient.so` und `IBMOSchgpwdserver.so` implementiert.

Zur Aktivierung der Unterstützung für die Kennwortänderung unter Linux setzen Sie den Konfigurationsparameter **clnt_pw_plugin** des Datenbankmanagers auf den Wert `IBMOSchgpwdclient` und den Konfigurationsparameter **srvcon_pw_plugin** des Datenbankmanagers auf den Wert `IBMOSchgpwdserver`.

Darüber hinaus müssen Sie eine PAM-Konfigurationsdatei mit dem Namen "db2" im Verzeichnis `/etc/pam.d` erstellen.

Implementieren eines Plug-ins zur Kennwortänderung (Linux)

Zur Aktivierung der Unterstützung für das Ändern von Kennwörtern in DB2-Datenbankprodukten unter Linux müssen Sie die DB2-Instanz zur Verwendung der Sicherheits-Plug-ins IBMOSchgpwdclient und IBMOSchgpwdserver konfigurieren.

Vorbereitende Schritte

Die Plug-in-Bibliotheken befinden sich in den folgenden Verzeichnissen:

- *INSTHOME*/sqllib/security*XX*/plugin/IBM/client/IBMOSchgpwdclient.so
- *INSTHOME*/sqllib/security*XX*/plugin/IBM/server/IBMOSchgpwdserver.so

Dabei ist *INSTHOME* das Ausgangsverzeichnis des Instanzeigners, und *securityXX* ist abhängig von der Bit-Breite der Instanz entweder security32 oder security64.

Vorgehensweise

Führen Sie die folgenden Schritte aus, um die Sicherheits-Plug-ins in einer DB2-Instanz zu implementieren:

1. Melden Sie sich als Benutzer mit Rootberechtigung an.
2. Erstellen Sie eine PAM-Konfigurationsdatei: /etc/pam.d/db2

   Stellen Sie sicher, dass die Datei die richtige Gruppe von Regeln enthält, wie dies durch Ihren Systemadministrator definiert wurde. Zum Beispiel können unter SLES 9 folgende Angaben verwendet werden:

   ```
   auth     required pam_unix2.so    nullok
   account  required pam_unix2.so
   password required pam_pwcheck.so  nullok tries=1
   password required pam_unix2.so    nullok use_authtok use_first_pass
   session  required pam_unix2.so
   ```

   Unter RHEL können folgende Angaben verwendet werden:

   ```
   #%PAM-1.0
   auth     required   /lib/security/$ISA/pam_env.so
   auth     sufficient /lib/security/$ISA/pam_unix.so likeauth nullok
   auth     required   /lib/security/$ISA/pam_deny.so

   account  required   /lib/security/$ISA/pam_unix.so
   account  sufficient /lib/security/$ISA/pam_succeed_if.so uid < 100 quiet
   account  required   /lib/security/$ISA/pam_permit.so

   password requisite  /lib/security/$ISA/pam_cracklib.so retry=3 dcredit=-1 ucredit=-1
   password sufficient /lib/security/$ISA/pam_unix.so nullok use_authtok md5 shadow remember=3
   password required   /lib/security/$ISA/pam_deny.so

   session  required   /lib/security/$ISA/pam_limits.so
   session  required   /lib/security/$ISA/pam_unix.so
   ```

3. Aktivieren Sie die Sicherheits-Plug-ins in der DB2-Instanz:
   a. Aktualisieren Sie den Konfigurationsparameter **SRVCON_PW_PLUGIN** des Datenbankmanagers mit dem Wert IBMOSchgpwdserver:

      ```
      db2 update dbm cfg using srvcon_pw_plugin IBMOSchgpwdserver
      ```

   b. Aktualisieren Sie den Konfigurationsparameter **CLNT_PW_PLUGIN** des Datenbankmanagers mit dem Wert IBMOSchgpwdclient:

      ```
      db2 update dbm cfg using CLNT_PW_PLUGIN IBMOSchgpwdclient
      ```

   c. Stellen Sie sicher, dass entweder der Konfigurationsparameter **SRVCON_AUTH** des Datenbankmanagers auf den Wert CLIENT, SERVER, SERVER_ENCRYPT, DATA_ENCRYPT oder DATA_ENCRYPT_CMP gesetzt wird oder der Konfigurationsparameter **SRVCON_AUTH** des Datenbankmanagers auf den Wert

NOT_SPECIFIED und der Parameter **AUTHENTICATION** auf den Wert CLIENT, SERVER, SERVER_ENCRYPT, DATA_ENCRYPT oder DATA_ENCRYPT_CMP gesetzt werden.

Anhang A. Übersicht über technische Informationen zu DB2

Technische Informationen zu DB2 liegen in verschiedenen Formaten vor, die auf unterschiedliche Weise abgerufen werden können.

Die technischen Informationen zu DB2 stehen über die folgenden Tools und Methoden zur Verfügung:

- DB2 Information Center
  - Themen (zu Tasks, Konzepten und Referenzinformationen)
  - Beispielprogramme
  - Lernprogramme
- DB2-Bücher
  - PDF-Dateien (für den Download verfügbar)
  - PDF-Dateien (auf der DB2-PDF-DVD)
  - Gedruckte Bücher
- Hilfe für Befehlszeile
  - Hilfe für Befehle
  - Hilfe für Nachrichten

Anmerkung: Die Themen des DB2 Information Center werden häufiger aktualisiert als die PDF- und Hardcopybücher. Um stets die neuesten Informationen zur Verfügung zu haben, sollten Sie die Dokumentationsaktualisierungen installieren, sobald diese verfügbar sind, oder das DB2 Information Center unter ibm.com aufrufen.

Darüber hinaus können Sie auf zusätzliche technische Informationen zu DB2, wie beispielsweise technische Hinweise (Technotes), White Papers und IBM Redbooks, online über ibm.com zugreifen. Rufen Sie dazu die Website 'DB2 Information Management - Software - Library' unter http://www.ibm.com/software/data/sw-library/ auf.

Feedback zur Dokumentation

Senden Sie uns Ihr Feedback zur DB2-Dokumentation! Wenn Sie Anregungen zur Verbesserung der DB2-Dokumentation haben, senden Sie eine E-Mail an db2docs@ca.ibm.com. Das DB2-Dokumentationsteam bearbeitet das gesamte Feedback, kann jedoch nicht im Einzelnen auf Ihre E-Mails antworten. Nennen Sie uns, wenn möglich, konkrete Beispiele, sodass wir die Problemstellung besser beurteilen können. Wenn Sie uns Feedback zu einem bestimmten Thema oder einer bestimmten Hilfedatei senden, geben Sie den entsprechenden Titel sowie die URL an.

Verwenden Sie diese E-Mail-Adresse nicht, wenn Sie sich an den DB2-Kundendienst wenden möchten. Wenn ein technisches Problem bei DB2 vorliegt, das Sie mithilfe der Dokumentation nicht beheben können, fordern Sie beim zuständigen IBM Service-Center Unterstützung an.

Bibliothek mit technischen Informationen zu DB2 im Hardcopy- oder PDF-Format

Die folgenden Tabellen enthalten eine Beschreibung der DB2-Bibliothek, die im IBM Publications Center unter www.ibm.com/e-business/linkweb/publications/servlet/pbi.wss zur Verfügung steht. Über die folgende Adresse können Sie englische Handbücher im PDF-Format sowie übersetzte Versionen zu DB2 Version 10.1 herunterladen: www.ibm.com/support/docview.wss?rs=71&uid=swg2700947.

In den Tabellen sind die Bücher, die in gedruckter Form zur Verfügung stehen, gekennzeichnet; möglicherweise sind diese in Ihrem Land oder Ihrer Region jedoch nicht verfügbar.

Die Formnummer wird bei jeder Aktualisierung eines Handbuchs erhöht. Anhand der nachfolgenden Liste können Sie sicherstellen, dass Sie die jeweils neueste Version des Handbuchs lesen.

Anmerkung: Das *DB2 Information Center* wird häufiger aktualisiert als die PDF- und Hardcopybücher.

Tabelle 54. Technische Informationen zu DB2

Name	IBM Form	In gedruckter Form verfügbar	Letzte Aktualisierung
Administrative API Reference	SC27-3864-00	Ja	April 2012
Administrative Routines and Views	SC27-3865-00	Nein	April 2012
Call Level Interface Guide and Reference Volume 1	SC27-3866-00	Ja	April 2012
Call Level Interface Guide and Reference Volume 2	SC27-3867-00	Ja	April 2012
Command Reference	SC27-3868-00	Ja	April 2012
Datenbankverwaltung - Konzepte und Konfiguration - Referenzinformationen	SC12-4673-00	Ja	April 2012
Dienstprogramme für das Versetzen von Daten - Handbuch und Referenz	SC12-4691-00	Ja	April 2012
Datenbanküberwachung - Handbuch und Referenz	SC12-4674-00	Ja	April 2012
Datenrecovery und hohe Verfügbarkeit - Handbuch und Referenz	SC12-4692-00	Ja	April 2012
Datenbanksicherheit	SC12-4693-00	Ja	April 2012
DB2 Workload Management - Handbuch und Referenz	SC12-4683-00	Ja	April 2012

Tabelle 54. Technische Informationen zu DB2 (Forts.)

Name	IBM Form	In gedruckter Form verfügbar	Letzte Aktualisierung
Developing ADO.NET and OLE DB Applications	SC27-3873-00	Ja	April 2012
Developing Embedded SQL Applications	SC27-3874-00	Ja	April 2012
Developing Java Applications	SC27-3875-00	Ja	April 2012
Developing Perl, PHP, Python, and Ruby on Rails Applications	SC27-3876-00	Nein	April 2012
Developing User-defined Routines (SQL and External)	SC27-3877-00	Ja	April 2012
Getting Started with Database Application Development	GI13-2046-00	Ja	April 2012
Installation und Verwaltung von DB2 unter Linux und Windows - Erste Schritte	GI11-3285-00	Ja	April 2012
Globalisierung	SC12-4694-00	Ja	April 2012
DB2-Server - Installation	SC12-4677-00	Ja	April 2012
IBM Data Server-Clients - Installation	SC12-4678-00	Nein	April 2012
Fehlernachrichten, Band 1	SC12-4686-00	Nein	April 2012
Fehlernachrichten, Band 2	SC12-4687-00	Nein	April 2012
Net Search Extender - Verwaltung und Benutzerhandbuch	SC12-4689-00	Nein	April 2012
Partitionierung und Clustering	SC12-4695-00	Ja	April 2012
pureXML - Handbuch	SC12-4684-00	Ja	April 2012
Spatial Extender - Benutzer- und Referenzhandbuch	SC12-4688-00	Nein	April 2012
SQL Procedural Languages: Application Enablement and Support	SC27-3896-00	Ja	April 2012
SQL Reference Volume 1	SC27-3885-00	Ja	April 2012
SQL Reference Volume 2	SC27-3886-00	Ja	April 2012
Text Search	SC12-4674-00	Ja	April 2012
Fehlerbehebung und Optimieren der Datenbankleistung	SC12-4675-00	Ja	April 2012

Tabelle 54. Technische Informationen zu DB2 (Forts.)

Name	IBM Form	In gedruckter Form verfügbar	Letzte Aktualisierung
Upgrade auf DB2 Version 10.1	SC12-4676-00	Ja	April 2012
Neuerungen in DB2 Version 10.1	SC12-4682-00	Ja	April 2012
XQuery - Referenz	SC12-4685-00	Nein	April 2012

Tabelle 55. Technische Informationen zu DB2 Connect

Name	IBM Form	In gedruckter Form verfügbar	Letzte Aktualisierung
DB2 Connect - Installation und Konfiguration von DB2 Connect Personal Edition	SC12-4679-00	Ja	April 2012
DB2 Connect - Installation und Konfiguration von DB2 Connect-Servern	SC12-4680-00	Ja	April 2012
DB2 Connect - Benutzerhandbuch	SC12-4681-00	Ja	April 2012

Aufrufen der Hilfe für den SQL-Status über den Befehlszeilenprozessor

DB2-Produkte geben für Bedingungen, die aufgrund einer SQL-Anweisung generiert werden können, einen SQLSTATE-Wert zurück. Die SQLSTATE-Hilfe erläutert die Bedeutung der SQL-Statuswerte und der SQL-Statusklassencodes.

Vorgehensweise

Zum Starten der Hilfe für SQL-Statuswerte müssen Sie den Befehlszeilenprozessor öffnen und Folgendes eingeben:

```
? SQL-Status oder ? Klassencode
```

Hierbei steht *SQL-Status* für einen gültigen fünfstelligen SQL-Statuswert und *Klassencode* für die ersten beiden Ziffern dieses Statuswerts.
So kann beispielsweise durch die Eingabe von `? 08003` Hilfe für den SQL-Statuswert 08003 angezeigt werden, durch die Eingabe von `? 08` Hilfe für den Klassencode 08.

Zugriff auf verschiedene Versionen des DB2 Information Center

Die Dokumentation für andere Versionen der DB2-Produkte finden Sie in den jeweiligen Information Centers unter ibm.com.

Informationen zu diesem Vorgang

Für Themen aus DB2 Version 10.1 lautet die URL für das *DB2 Information Center* http://publib.boulder.ibm.com/infocenter/db2luw/v10r1.

Für Themen aus DB2 Version 9.8 lautet die URL des *DB2 Information Center* http://publib.boulder.ibm.com/infocenter/db2luw/v9r8/.

Für Themen aus DB2 Version 9.7 lautet die URL des *DB2 Information Center* http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/.

Für Themen aus DB2 Version 9.5 lautet die URL des *DB2 Information Center* http://publib.boulder.ibm.com/infocenter/db2luw/v9r5.

Für Themen aus DB2 Version 9.1 lautet die URL des *DB2 Information Center* http://publib.boulder.ibm.com/infocenter/db2luw/v9/.

Für Themen aus DB2 Version 8 lautet die URL des *DB2 Information Center* http://publib.boulder.ibm.com/infocenter/db2luw/v8/.

Aktualisieren des auf Ihrem Computer oder Intranet-Server installierten DB2 Information Center

Ein lokal installiertes DB2 Information Center muss regelmäßig aktualisiert werden.

Vorbereitende Schritte

Ein DB2 Version 10.1 Information Center muss bereits installiert sein. Einzelheiten hierzu finden Sie unter „Installation des DB2 Information Center mit dem DB2-Installationsassistenten" in *DB2-Server - Installation*. Alle für die Installation des Information Center geltenden Voraussetzungen und Einschränkungen gelten auch für die Aktualisierung des Information Center.

Informationen zu diesem Vorgang

Ein vorhandenes DB2 Information Center kann automatisch oder manuell aktualisiert werden:

- Mit automatischen Aktualisierungen werden vorhandene Komponenten und Sprachen des Information Center aktualisiert. Ein Vorteil von automatischen Aktualisierungen ist, dass das Information Center im Vergleich zu einer manuellen Aktualisierung nur für einen kurzenZeitraum nicht verfügbar ist. Darüber hinaus können automatische Aktualisierungen so konfiguriert werden, dass sie als Teil anderer, regelmäßig ausgeführter Stapeljobs ausgeführt werden.
- Mit manuellen Aktualisierungen können Sie vorhandene Komponenten und Sprachen des Information Center aktualisieren. Automatische Aktualisierungen reduzieren die Ausfallzeiten während des Aktualisierungsprozesses, Sie müssen jedoch den manuellen Prozess verwenden, wenn Sie Komponenten oder Sprachen hinzufügen möchten. Beispiel: Ein lokales Information Center wurde ursprünglich sowohl mit englischer als auch mit französischer Sprachunterstützung installiert; nun soll auch die deutsche Sprachunterstützung installiert werden. Bei einer manuellen Aktualisierung werden sowohl eine Installation der deutschen Sprachunterstützung als auch eine Aktualisierung der vorhandenen Komponenten und Sprachen des Information Center durchgeführt. Sie müssen jedoch bei einer manuellen Aktualisierung das Information Center manuell stoppen, aktualisieren und erneut starten. Das Information Center ist während des gesamten Aktualisierungsprozesses nicht verfügbar. Während des automatischen Aktualisierungsprozesses kommt es zu einem Ausfall des Information Center, und es wird erst wieder nach der Aktualisierung erneut gestartet.

Dieser Abschnitt enthält Details zum Prozess der automatischen Aktualisierung. Anweisungen zur manuellen Aktualisierung finden Sie im Abschnitt „Manuelles Aktualisieren des auf Ihrem Computer oder Intranet-Server installierten DB2 Information Center".

Vorgehensweise

Gehen Sie wie folgt vor, um das auf Ihrem Computer bzw. Intranet-Server installierte DB2 Information Center automatisch zu aktualisieren:

1. Unter Linux:
   a. Navigieren Sie zu dem Pfad, in dem das Information Center installiert ist. Standardmäßig ist das DB2 Information Center im Verzeichnis `/opt/ibm/db2ic/V10.1` installiert.
   b. Navigieren Sie vom Installationsverzeichnis in das Verzeichnis `doc/bin`.
   c. Führen Sie das Script `update-ic` aus:
      ```
      update-ic
      ```
2. Unter Windows:
   a. Öffnen Sie ein Befehlsfenster.
   b. Navigieren Sie zu dem Pfad, in dem das Information Center installiert ist. Standardmäßig ist das DB2 Information Center im Verzeichnis `<Programme>\IBM\DB2 Information Center\Version 10.1` installiert, wobei `<Programme>` das Verzeichnis der Programmdateien angibt.
   c. Navigieren Sie vom Installationsverzeichnis in das Verzeichnis `doc\bin`.
   d. Führen Sie die Datei `update-ic.bat` aus:
      ```
      update-ic.bat
      ```

Ergebnisse

Das DB2 Information Center wird automatisch erneut gestartet. Standen Aktualisierungen zur Verfügung, zeigt das Information Center die neuen und aktualisierten Abschnitte an. Waren keine Aktualisierungen für das Information Center verfügbar, wird eine entsprechende Nachricht zum Protokoll hinzugefügt. Die Protokolldatei befindet sich im Verzeichnis `doc\eclipse\configuration`. Der Name der Protokolldatei ist eine Zufallszahl. Beispiel: `1239053440785.log`.

Manuelles Aktualisieren des auf Ihrem Computer oder Intranet-Server installierten DB2 Information Center

Wenn Sie das DB2 Information Center lokal installiert haben, können Sie Dokumentationsaktualisierungen von IBM abrufen und installieren.

Informationen zu diesem Vorgang

Zur manuellen Aktualisierung des lokal installierten *DB2 Information Center* sind die folgenden Schritte erforderlich:

1. Stoppen Sie das *DB2 Information Center* auf Ihrem Computer und starten Sie das Information Center im Standalone-Modus erneut. Die Ausführung des Information Center im Standalone-Modus verhindert, dass andere Benutzer in Ihrem Netz auf das Information Center zugreifen, und ermöglicht das Anwenden von Aktualisierungen. Die Workstationversion des DB2 Information Center wird stets im Standalone-Modus ausgeführt.

2. Verwenden Sie die Aktualisierungsfunktion, um zu prüfen, welche Aktualisierungen verfügbar sind. Falls Aktualisierungen verfügbar sind, die Sie installieren müssen, können Sie die Aktualisierungsfunktion verwenden, um diese abzurufen und zu installieren.

   Anmerkung: Wenn es in der verwendeten Umgebung erforderlich ist, die Aktualisierungen für das *DB2 Information Center* auf einer Maschine zu installieren, die nicht über ein Verbindung zum Internet verfügt, spiegeln Sie die Aktualisierungssite auf ein lokales Dateisystem und verwenden Sie dabei eine Maschine, die mit dem Internet verbunden ist und auf der das *DB2 Information Center* installiert ist. Wenn viele Benutzer Ihres Netzes die Dokumentationsaktualisierungen installieren sollen, können Sie die Zeit, die jeder einzelne Benutzer für die Aktualisierungen benötigt, reduzieren, indem Sie die Aktualisierungssite lokal spiegeln und ein Proxy dafür erstellen.
   Ist dies der Fall, verwenden Sie die Aktualisierungsfunktion, um die Pakete abzurufen. Die Aktualisierungsfunktion ist jedoch nur im Standalone-Modus verfügbar.
3. Stoppen Sie das im Standalone-Modus gestartete Information Center und starten Sie das *DB2 Information Center* auf Ihrem Computer erneut.

Anmerkung: Unter Windows 2008 und Windows Vista (und neueren Versionen) müssen die in diesem Abschnitt aufgeführten Befehle mit Administratorberechtigung ausgeführt werden. Zum Öffnen einer Eingabeaufforderung oder eines Grafiktools mit vollen Administratorberechtigungen klicken Sie mit der rechten Maustaste die Verknüpfung an und wählen Sie **Als Administrator ausführen** aus.

Vorgehensweise

Gehen Sie wie folgt vor, um das auf Ihrem Computer bzw. Intranet-Server installierte *DB2 Information Center* zu aktualisieren:

1. Stoppen Sie das *DB2 Information Center*.
   - Unter Windows: Klicken Sie **Start** > **Systemsteuerung** > **Verwaltung** > **Dienste** an. Klicken Sie mit der rechten Maustaste das **DB2 Information Center** an und wählen Sie **Beenden** aus.
   - Unter Linux: Geben Sie den folgenden Befehl ein:
     ```
     /etc/init.d/db2icdv10 stop
     ```
2. Starten Sie das Information Center im Standalone-Modus.
   - Unter Windows:
     a. Öffnen Sie ein Befehlsfenster.
     b. Navigieren Sie zu dem Pfad, in dem das Information Center installiert ist. Standardmäßig ist das *DB2 Information Center* im Verzeichnis *Programme*\IBM\DB2 Information Center\Version 10.1 installiert, wobei *Programme* das Verzeichnis der Programmdateien angibt.
     c. Navigieren Sie vom Installationsverzeichnis in das Verzeichnis doc\bin.
     d. Führen Sie die Datei help_start.bat aus:
        ```
        help_start.bat
        ```
   - Unter Linux:
     a. Navigieren Sie zu dem Pfad, in dem das Information Center installiert ist. Standardmäßig ist das *DB2 Information Center* im Verzeichnis /opt/ibm/db2ic/V10.1 installiert.
     b. Navigieren Sie vom Installationsverzeichnis in das Verzeichnis doc/bin.
     c. Führen Sie das Script help_start aus:

```
help_start
```

Der standardmäßig auf dem System verwendete Web-Browser wird geöffnet und zeigt die Standalone-Version des Information Center an.

3. Klicken Sie die Aktualisierungsschaltfläche ( ) an. (JavaScript muss im verwendeten Browser aktiviert sein.) Klicken Sie im rechten Fenster des Information Center die Schaltfläche für die Suche nach Aktualisierungen an. Eine Liste der Aktualisierungen für die vorhandene Dokumentation wird angezeigt.
4. Wählen Sie zum Initiieren des Installationsprozesses die gewünschten Aktualisierungen aus und klicken Sie anschließend die Schaltfläche für die Installation der Aktualisierungen an.
5. Klicken Sie nach Abschluss des Installationsprozesses **Fertigstellen** an.
6. Stoppen Sie das im Standalone-Modus gestartete Information Center:
   - Unter Windows: Navigieren Sie innerhalb des Installationsverzeichnisses zum Verzeichnis `doc\bin`, und führen Sie die Datei `help_end.bat` aus:
     ```
     help_end.bat
     ```
     Anmerkung: Die Stapeldatei `help_end` enthält die Befehle, die erforderlich sind, um die Prozesse, die mit der Stapeldatei `help_start` gestartet wurden, ordnungsgemäß zu stoppen. Verwenden Sie nicht die Tastenkombination `Strg+C` oder eine andere Methode, um `help_start.bat` zu stoppen.
   - Unter Linux: Navigieren Sie innerhalb des Installationsverzeichnisses zum Verzeichnis `doc/bin`, und führen Sie das Script `help_end` aus:
     ```
     help_end
     ```
     Anmerkung: Das Script `help_end` enthält die Befehle, die erforderlich sind, um die Prozesse, die mit dem Script `help_start` gestartet wurden, ordnungsgemäß zu stoppen. Verwenden Sie keine andere Methode, um das Script `help_start` zu stoppen.
7. Starten Sie das *DB2 Information Center* erneut.
   - Unter Windows: Klicken Sie **Start** > **Systemsteuerung** > **Verwaltung** > **Dienste** an. Klicken Sie mit der rechten Maustaste das **DB2 Information Center** an und wählen Sie **Start** aus.
   - Unter Linux: Geben Sie den folgenden Befehl ein:
     ```
     /etc/init.d/db2icdv10 start
     ```

Ergebnisse

Im aktualisierten *DB2 Information Center* werden die neuen und aktualisierten Themen angezeigt.

DB2-Lernprogramme

Die DB2-Lernprogramme unterstützen Sie dabei, sich mit den unterschiedlichen Aspekten der DB2-Produkte vertraut zu machen. Die Lerneinheiten bieten eine in einzelne Schritte unterteilte Anleitung.

Vorbereitungen

Die XHTML-Version des Lernprogramms kann über das Information Center unter http://publib.boulder.ibm.com/infocenter/db2luw/v10r1/ angezeigt werden.

In einigen der Lerneinheiten werden Beispieldaten und Codebeispiele verwendet. Informationen zu bestimmten Voraussetzungen für die Ausführung der Tasks finden Sie in der Beschreibung des Lernprogramms.

DB2-Lernprogramme

Klicken Sie zum Anzeigen des Lernprogramms den Titel an.

„pureXML" in *pureXML - Handbuch*
Einrichten einer DB2-Datenbank, um XML-Daten zu speichern und Basisoperationen mit dem nativen XML-Datenspeicher auszuführen.

Informationen zur Fehlerbehebung in DB2

Es steht eine breite Palette verschiedener Informationen zur Fehlerbestimmung und Fehlerbehebung zur Verfügung, um Sie bei der Verwendung von DB2-Datenbankprodukten zu unterstützen.

DB2-Dokumentation
Informationen zur Fehlerbehebung stehen im Handbuch *Fehlerbehebung und Optimieren der Datenbankleistung* oder im Abschnitt mit grundlegenden Informationen zu Datenbanken im *DB2 Information Center* zur Verfügung, darunter:

- Informationen zum Eingrenzen und Aufdecken von Problemen mithilfe der Diagnosetools und -dienstprogramme von DB2.
- Lösungsvorschläge zu den am häufigsten auftretenden Problemen.
- Ratschläge zum Lösen anderer Probleme, die bei Verwendung der DB2-Datenbankprodukte auftreten können.

IBM Support Portal
Im IBM Support Portal finden Sie Informationen zu Problemen und den möglichen Ursachen und Fehlerbehebungsmaßnahmen. Die Website mit technischer Unterstützung enthält Links zu den neuesten DB2-Veröffentlichungen, technischen Hinweisen (TechNotes), APARs (Authorized Program Analysis Reports) und Fehlerkorrekturen, Fixpacks sowie weiteren Ressourcen. Sie können diese Wissensbasis nach möglichen Lösungen für aufgetretene Probleme durchsuchen.

Sie können auf das IBM Support Portal über die folgende Website zugreifen: http://www.ibm.com/support/entry/portal/Overview/Software/Information_Management/DB2_for_Linux,_UNIX_and_Windows.

Bedingungen

Die Berechtigungen zur Nutzung dieser Veröffentlichungen werden Ihnen auf der Basis der folgenden Bedingungen gewährt.

Anwendbarkeit: Diese Bedingungen gelten zusätzlich zu den Nutzungsbedingungen für die IBM Website.

Persönliche Nutzung: Sie dürfen diese Veröffentlichungen für Ihre persönliche, nicht kommerzielle Nutzung unter der Voraussetzung vervielfältigen, dass alle Eigentumsvermerke erhalten bleiben. Sie dürfen diese Veröffentlichungen oder Teile dieser Veröffentlichungen ohne ausdrückliche Genehmigung von IBM nicht weitergeben, anzeigen oder abgeleitete Werke davon erstellen.

Kommerzielle Nutzung: Sie dürfen diese Veröffentlichungen nur innerhalb Ihres Unternehmens und unter der Voraussetzung, dass alle Eigentumsvermerke erhalten bleiben, vervielfältigen, weitergeben und anzeigen. Sie dürfen diese Veröffentlichungen oder Teile dieser Veröffentlichungen ohne ausdrückliche Genehmigung von IBM außerhalb Ihres Unternehmens nicht vervielfältigen, weitergeben, anzeigen oder abgeleitete Werke davon erstellen.

Rechte: Abgesehen von den hier gewährten Berechtigungen erhalten Sie keine weiteren Berechtigungen, Lizenzen oder Rechte (veröffentlicht oder stillschweigend) in Bezug auf die Veröffentlichungen oder darin enthaltene Informationen, Daten, Software oder geistiges Eigentum.

IBM behält sich das Recht vor, die in diesem Dokument gewährten Berechtigungen nach eigenem Ermessen zurückzuziehen, wenn sich die Nutzung der Veröffentlichungen für IBM als nachteilig erweist oder wenn die obigen Nutzungsbestimmungen nicht genau befolgt werden.

Sie dürfen diese Informationen nur in Übereinstimmung mit allen anwendbaren Gesetzen und Vorschriften, einschließlich aller US-amerikanischen Exportgesetze und Verordnungen, herunterladen und exportieren.

IBM übernimmt keine Gewährleistung für den Inhalt dieser Informationen. Diese Veröffentlichungen werden auf der Grundlage des gegenwärtigen Zustands (auf "as-is"-Basis) und ohne eine ausdrückliche oder stillschweigende Gewährleistung für die Handelsüblichkeit, die Verwendungsfähigkeit oder die Freiheit der Rechte Dritter zur Verfügung gestellt.

IBM Marken: IBM, das IBM Logo und ibm.com sind Marken oder eingetragene Marken der International Business Machines Corporation. Weitere Produkt- oder Servicenamen können Marken von oder anderen Herstellern sein. IBM oder anderen Herstellern sein. Eine aktuelle Liste der IBM Marken finden Sie auf der Webseite www.ibm.com/legal/copytrade.shtml.

Anhang B. Bemerkungen

Die vorliegenden Informationen wurden für Produkte und Services entwickelt, die auf dem deutschen Markt angeboten werden. Die Informationen über Produkte anderer Hersteller als IBM basieren auf den zum Zeitpunkt der ersten Veröffentlichung dieses Dokuments verfügbaren Informationen und können geändert werden.

Möglicherweise bietet IBM die in dieser Dokumentation beschriebenen Produkte, Services oder Funktionen in anderen Ländern nicht an. Informationen über die gegenwärtig im jeweiligen Land verfügbaren Produkte und Services sind beim zuständigen IBM Ansprechpartner erhältlich. Hinweise auf IBM Lizenzprogramme oder andere IBM Produkte bedeuten nicht, dass nur Programme, Produkte oder Services von IBM verwendet werden können. Anstelle der IBM Produkte, Programme oder Services können auch andere, ihnen äquivalente Produkte, Programme oder Services verwendet werden, solange diese keine gewerblichen oder anderen Schutzrechte von IBM verletzen. Die Verantwortung für den Betrieb von Produkten, Programmen und Services anderer Anbieter liegt beim Kunden.

Für in diesem Handbuch beschriebene Erzeugnisse und Verfahren kann es IBM Patente oder Patentanmeldungen geben. Mit der Auslieferung dieses Handbuchs ist keine Lizenzierung dieser Patente verbunden. Lizenzanforderungen sind schriftlich an folgende Adresse zu richten (Anfragen an diese Adresse müssen auf Englisch formuliert werden):

IBM Director of Licensing
IBM Europe, Middle East & Africa
Tour Descartes
2, avenue Gambetta
92066 Paris La Defense
France

Trotz sorgfältiger Bearbeitung können technische Ungenauigkeiten oder Druckfehler in dieser Veröffentlichung nicht ausgeschlossen werden. Die hier enthaltenen Informationen werden in regelmäßigen Zeitabständen aktualisiert und als Neuausgabe veröffentlicht. IBM kann ohne weitere Mitteilung jederzeit Verbesserungen und/oder Änderungen an den in dieser Veröffentlichung beschriebenen Produkten und/oder Programmen vornehmen.

Verweise in diesen Informationen auf Websites anderer Anbieter werden lediglich als Service für den Kunden bereitgestellt und stellen keinerlei Billigung des Inhalts dieser Websites dar. Das über diese Websites verfügbare Material ist nicht Bestandteil des Materials für dieses IBM Produkt. Die Verwendung dieser Websites geschieht auf eigene Verantwortung.

Werden an IBM Informationen eingesandt, können diese beliebig verwendet werden, ohne dass eine Verpflichtung gegenüber dem Einsender entsteht.

Lizenznehmer des Programms, die Informationen zu diesem Produkt wünschen mit der Zielsetzung: (i) den Austausch von Informationen zwischen unabhängig voneinander erstellten Programmen und anderen Programmen (einschließlich des vorliegenden Programms) sowie (ii) die gemeinsame Nutzung der ausgetauschten Informationen zu ermöglichen, wenden sich an folgende Adresse:

IBM Canada Limited
U59/3600
3600 Steeles Avenue East
Markham, Ontario L3R 9Z7
CANADA

Die Bereitstellung dieser Informationen kann unter Umständen von bestimmten Bedingungen - in einigen Fällen auch von der Zahlung einer Gebühr - abhängig sein.

Die Lieferung des im Dokument aufgeführten Lizenzprogramms sowie des zugehörigen Lizenzmaterials erfolgt auf der Basis der IBM Rahmenvereinbarung bzw. der Allgemeinen Geschäftsbedingungen von IBM, der IBM Internationalen Nutzungsbedingungen für Programmpakete oder einer äquivalenten Vereinbarung.

Alle in diesem Dokument enthaltenen Leistungsdaten stammen aus einer kontrollierten Umgebung. Die Ergebnisse, die in anderen Betriebsumgebungen erzielt werden, können daher erheblich von den hier erzielten Ergebnissen abweichen. Einige Daten stammen möglicherweise von Systemen, deren Entwicklung noch nicht abgeschlossen ist. Eine Gewährleistung, dass diese Daten auch in allgemein verfügbaren Systemen erzielt werden, kann nicht gegeben werden. Darüber hinaus wurden einige Daten unter Umständen durch Extrapolation berechnet. Die tatsächlichen Ergebnisse können davon abweichen. Benutzer dieses Dokuments sollten die entsprechenden Daten in ihrer spezifischen Umgebung prüfen.

Alle Informationen zu Produkten anderer Anbieter stammen von den Anbietern der aufgeführten Produkte, deren veröffentlichen Ankündigungen oder anderen allgemein verfügbaren Quellen. IBM hat diese Produkte nicht getestet und kann daher keine Aussagen zu Leistung, Kompatibilität oder anderen Merkmalen machen. Fragen zu den Leistungsmerkmalen von Produkten anderer Anbieter sind an den jeweiligen Anbieter zu richten.

Aussagen über Pläne und Absichten von IBM unterliegen Änderungen oder können zurückgenommen werden und repräsentieren nur die Ziele von IBM.

Diese Veröffentlichung kann Beispiele für Daten und Berichte des alltäglichen Geschäftsablaufes enthalten. Sie sollen nur die Funktionen des Lizenzprogramms illustrieren; sie können Namen von Personen, Firmen, Marken oder Produkten enthalten. Alle diese Namen sind frei erfunden; Ähnlichkeiten mit tatsächlichen Namen und Adressen sind rein zufällig.

COPYRIGHTLIZENZ:

Diese Veröffentlichung enthält Musteranwendungsprogramme, die in Quellensprache geschrieben sind und Programmiertechniken in verschiedenen Betriebsumgebungen veranschaulichen. Sie dürfen diese Musterprogramme kostenlos kopieren, ändern und verteilen, wenn dies zu dem Zweck geschieht, Anwendungsprogramme zu entwickeln, zu verwenden, zu vermarkten oder zu verteilen, die mit der Anwendungsprogrammierschnittstelle für die Betriebsumgebung konform sind, für die diese Musterprogramme geschrieben werden. Diese Beispiele wurden nicht unter allen denkbaren Bedingungen getestet. Daher kann IBM die Zuverlässigkeit, Wartungsfreundlichkeit oder Funktion dieser Programme weder zusagen noch gewährleisten. Die Musterprogramme werden ohne Wartung (auf "as-is"-Basis) und ohne jegliche Gewährleistung zur Verfügung gestellt. IBM haftet nicht für Schäden, die durch Verwendung der Musterprogramme entstehen.

Kopien oder Teile der Musterprogramme bzw. daraus abgeleiteter Code müssen folgenden Copyrightvermerk beinhalten:

© (*Name Ihrer Firma*) (*Jahr*). Teile des vorliegenden Codes wurden aus Musterprogrammen der IBM Corp. abgeleitet. © Copyright IBM Corp. *_Jahr/Jahre angeben_*. Alle Rechte vorbehalten.

Marken

IBM, das IBM Logo und ibm.com sind Marken oder eingetragene Marken der IBM Corporation in den USA und/oder anderen Ländern. Weitere Produkt- oder Servicenamen können Marken von oder anderen Herstellern sein. IBM oder anderen Herstellern sein. Eine aktuelle Liste der IBM Marken finden Sie auf der Webseite „Copyright and trademark information" unter www.ibm.com/legal/copytrade.shtml.

Die folgenden Namen sind Marken oder eingetragene Marken anderer Unternehmen.

- Linux ist eine eingetragene Marke von Linus Torvalds in den USA und/oder anderen Ländern.
- Java und alle auf Java basierenden Marken und Logos sind Marken oder eingetragene Marken von Oracle und/oder ihren verbundenen Unternehmen.
- UNIX ist eine eingetragene Marke von The Open Group in den USA und anderen Ländern.
- Intel, das Intel-Logo, Intel Inside, Intel Inside logo, Celeron, Intel SpeedStep, Itanium und Pentium sind Marken oder eingetragene Marken der Intel Corporation oder deren Tochtergesellschaften in den USA und anderen Ländern.
- Microsoft, Windows, Windows NT und das Windows-Logo sind Marken der Microsoft Corporation in den USA und/oder anderen Ländern.

Weitere Unternehmens-, Produkt- oder Servicenamen können Marken anderer Hersteller sein.

Index

Sonderzeichen

.NET
  .Net Data Provider-Clients 77
  GSKit 77
  SSL 77

A

ACCESSCTRL (Zugriffssteuerung), Berechtigung
  Details 42
  Übersicht 37
AIX 250, 251
AIX EFS 103
Aktualisierungen
  Auswirkungen von LBAC 211
  DB2 Information Center 407, 408
ALTER, Zugriffsrecht
  Details 49, 52
alternate_auth_enc, Konfigurationsparameter
  Verschlüsselung mit AES 256–Bit-Algorithmus 8
Anweisungswertdaten, Feld 122
Anweisungswertindex, Feld 122
Anweisungswerttyp, Feld 122
APIs
  Benutzer-ID/Kennwort-Plug-in 286
  Exitbibliothek für Kommunikationspuffer 318
    db2commexitDeregister 322
    db2commexitFreeErrormsg 327
    db2commexitInit 319
    db2commexitRecv 323
    db2commexitRegister 321
    db2commexitSend 324
    db2commexitTerm 320
    db2commexitUserIdentity 325
  Plug-in zum Abrufen von Gruppen 278
  Sicherheits-Plug-in
    db2secClientAuthPluginInit 292
    db2secClientAuthPluginTerm 294
    db2secDoesAuthIDExist 294
    db2secDoesGroupExist 280
    db2secFreeErrormsg 280
    db2secFreeGroupListMemory 281
    db2secFreeInitInfo 295
    db2secFreeToken 296
    db2secGenerateInitialCred 296
    db2secGetAuthIDs 298
    db2secGetDefaultLoginContext 300
    db2secGetGroupsForUser 281
    db2secGroupPluginInit 285
    db2secPluginTerm 286
    db2secProcessServerPrincipalName 302
    db2secRemapUserid 302
    db2secServerAuthPluginInit 304
    db2secServerAuthPluginTerm 307
    db2secValidatePassword 307
    Übersicht 277
archivepath, Parameter 113
Archivieren
  Prüfprotokolldateien 113
audit_buf_sz, Konfigurationsparameter
  zeitlichen Ablauf beim Schreiben von Prüfsätzen bestimmen 130
AUDIT-Ereignisse 376
Aufzählung von Gruppen 389
Authentifizierung
  Details 2
  Domänensicherheit 387
  ferne Clients 14
  Geordnete Domänenliste 391
  Gruppen 387
  GSS-API 231
  ID und Kennwort 231
  Kerberos 16, 231
  LDAP-Benutzer 263
  Methoden 8
  partitionierte Datenbanken 15
  Plug-ins 247
    API zum Abrufen von Berechtigungs-IDs 298
    API zum Bereinigen der Plug-in-Ressourcen zur Clientauthentifizierung 294
    API zum Bereinigen der Plug-in-Ressourcen zur Serverauthentifizierung 307
    API zum Bereinigen von Ressourcen, die von API db2secGenerateInitialCred genutzt werden 295
    API zum Initialisieren des Serverauthentifizierungs-Plug-ins 304
    API zum Initialisieren eines Plug-ins zur Clientauthentifizierung 292
    API zum Prüfen von Kennwörtern 307
    API zum Überprüfen, ob Berechtigungs-ID vorhanden ist 294
    APIs für Plug-ins zur Benutzer-ID/Kennwort-Authentifizierung 286
    implementieren 242, 243, 246, 400
    Sicherheit 231
    Speicherpositionen für Bibliotheken 236
  Sicherheits-Plug-ins 231
  Suche
    konfigurieren 250, 253, 254, 256
  Typen
    CLIENT 8
    KERBEROS 8
    KRB_SERVER_ENCRYPT 8
    SERVER 8
    SERVER_ENCRYPT 8
  Übersicht 1
  zweiteilige Benutzer-IDs 238
AUTHID_ATTRIBUTE 257

B

Backups
  Sicherheitsrisiken 66, 100
  Verschlüsselung 100
Bedingungen
  Veröffentlichungen 412
Bemerkungen 413
Benutzer-IDs
  auswählen 5
  LDAP 261

Benutzer-IDs *(Forts.)*
wechseln 154
zweiteilig 238
Benutzerdefinierte Funktionen (UDFs)
nicht abgeschirmt 37
Benutzernamen
Windows-Einschränkungen 386
Berechtigung LOAD
Details 45
Berechtigungen
Autorisierung, Übersicht 3
Datenbankverwaltung (DBADM) 40, 46
Datenzugriff (DATAACCESS) 43
DBADM durch Benutzer mit SYSCTRL entziehen 35
Implizites Schema (IMPLICIT_SCHEMA) 46
LOAD 45
Prüfrichtlinie 108
Sicherheitsadministrator (SECADM) 39
spaltenspezifischer Schutz 183
SQL-Verwaltung (SQLADM) 43
Systemmonitor (SYSMON) 36
Systempflege (SYSMAINT) 36
Systemsteuerung (SYSCTRL) 35
Systemverwaltung (SYSADM) 34
Übersicht 23, 30
Verwaltungsberechtigung EXPLAIN 45
Verzeichnisse 7
Workloadverwaltung (WLMADM) 45
zeilenspezifischer Schutz 183
Zugriffssteuerung (ACCESSCTRL) 42
Berechtigungen der Datenbankebene
Übersicht 30
Berechtigungs-IDs
Details 3
gesicherter Client 8
implizit 59
LDAP 261
SETSESSIONUSER, Zugriffsrecht 47
Typen 53
Übersicht über das Sicherheitsmodell 1
Berechtigungsnamen
abrufen
erteilte Zugriffsrechte 224
Namen mit Berechtigung DBADM 223
Namen mit erteilten Zugriffsrechten 222
Namen mit Tabellenzugriffsberechtigung 223
Sichten für Informationen zu Zugriffsrechten erstellen 224
Bibliotheken
Sicherheits-Plug-ins
Einschränkungen 265
in DB2 laden 263
BIND, Befehl
OWNER, Option 60
BIND, Zugriffsrecht
Details 51
BINDADD, Berechtigung
Details 37
Binden
Rebind ungültiger Pakete 57

C

CHECKING-Ereignisse 376
Cipher Suites 84
CLIENT, Authentifizierungstyp
Details 8
Clientauthentifizierung, Plug-ins 247
CONNECT, Berechtigung 37
CONTEXT-Ereignisse 376
CONTROL, Zugriffsrecht
Details 49
implizit 59
Pakete 51
Sichten 49
CREATE DATABASE, Befehl
RESTRICTIVE, Option 224
CREATE_EXTERNAL_ROUTINE, Berechtigung 37
CREATE_NOT_FENCED_ROUTINE, Berechtigung 37
CREATE ROLE, Anweisung
Rollen erstellen 138
Zugehörigkeit zu Rollen erteilen 138
CREATE TRUSTED CONTEXT, Anweisung
Beispiel 153
CREATETAB, Berechtigung 37

D

DATAACCESS (Datenzugriff), Berechtigung
Details 43
Übersicht 37
Database Encryption Expert 100
datapath, Parameter 113
Dateinamen
Prüfprotokolle 116
Datenbankberechtigungen
erteilen
Übersicht 37
Übersicht 37
widerrufen 37
Datenbanken
Kennsatzbasierte Zugriffssteuerung (LBAC) 183
Zugriff
implizite Zugriffsrechte durch Pakete 60
Standardberechtigungen 55
Datenbankobjekte
Rollen 137
Datenbankverzeichnisse
Berechtigungen 7
Datendarstellung
indirekter Zugriff 66
kennsatzbasierte Zugriffssteuerung (Label-Based Access Control, LBAC)
Schutz hinzufügen 204
Übersicht 204
Kennsatzbasierte Zugriffssteuerung (LBAC)
aktualisieren 211
einfügen 208
lesen 205
Schutz entfernen 219
Prüfdaten
in Tabellen laden 118
Tabellen erstellen 117
Sicherheit
Systemkatalog 224
Übersicht 1
Verschlüsselung 68
Datensätze
prüfen 106
DB2_GRP_LOOKUP, Registrierdatenbankvariable 388
DB2_GRP_LOOKUP, Umgebungsvariable 389, 392
DB2 Information Center
Aktualisierung 407, 408
Versionen 406

DB2ADMNS, Gruppe
Benutzer mit der Berechtigung SYSADM definieren 392
Details 394
db2audit.log, Datei 106
db2cluster, Befehl
DB2-Cluster-Services, Administrator 135
Sicherheitsmodell 135
DB2COMM, Registrierdatenbankvariable
SSL-Unterstützung konfigurieren 70
DB2LBACRULES LBAC, Regelsatz 197
DB2LDAPSecurityConfig, Umgebungsvariable
Übersicht 257
DB2SECURITYLABEL, Datentyp
als Zeichenfolge anzeigen 202
explizite Werte angeben 202
DB2USERS, Benutzergruppe
Details 394
DBADM (Datenbankadministrator), Berechtigung
Details 40
Namen abrufen 223
Übersicht 37
Zugriffssteuerung 64
Debugging
Sicherheits-Plug-ins 241
Definierter Name (DN) 261
DELETE, Zugriffsrecht 49
Differenzierte Zugriffssteuerung
siehe RCAC 159
Digitale Zertifikate
Übersicht 83
verwalten 70
Dokumentation
gedruckt 404
Nutzungsbedingungen 412
PDF-Dateien 404
Übersicht 403
Domänen
Geordnete Domänenliste 391
Sicherheit
Authentifizierung 387
Vertrauensstellungen 387
Windows 392
Domänencontroller
Übersicht 383
Dynamisches SQL
EXECUTE, Zugriffsrecht 60

E

EFS 103
efsenable, Befehl 103
efskeymgr, Befehl 103
efsmgr, Befehl 103
Eigentumsrecht
Datenbankobjekte 23, 221
Einfügen von Daten
LBAC-geschützte 208
ENABLE_SSL, Parameter 257
Encryption Expert 100
Erweiterte Windows-Sicherheit 394
ExampleBANK, RCAC-Szenario
Datenabfragen 181
Datenbankbenutzer und -rollen 177
Datenbanktabellen 178
Einführung 176
Sicherheitsrichtlinie 177
Spaltenmasken 180
ExampleBANK, RCAC-Szenario *(Forts.)*
Zeilenberechtigungen 179
ExampleHMO, RCAC-Szenario
Ansichtserstellung 172
Datenabfragen 170
Datenaktualisierungen 169
Datenbankbenutzer und -rollen 162
Datenbanktabellen 163
Einfügen von Daten 169
Einführung 161
Entziehen von Berechtigungen 176
sichere Funktionen 173
sichere Trigger 175
Sicherheitsrichtlinie 161
Spaltenmasken 167
Verwaltung von Sicherheitsfunktionen 166
Zeilenberechtigungen 166
EXECUTE, Kategorie
Prüfdaten 126
Prüfsätze 370
Übersicht 122
wiederholen 128
EXECUTE, Zugriffsrecht
Datenbankzugriff 60
Pakete 51
Routinen 52
EXECUTE-Ereignisse 376
Exitbibliothek für Kommunikationspuffer
aktivieren 315, 316
APIs
db2commexitDeregister 322
db2commexitFreeErrormsg 327
db2commexitInit 319
db2commexitRecv 323
db2commexitRegister 321
db2commexitSend 324
db2commexitTerm 320
db2commexitUserIdentity 325
Bibliothek laden 318
Entwicklung 317
API-Aufrufreihenfolge, SET SESSION AUTHORIZATION 335
API-Aufrufreihenfolgen 333
API-Aufrufreihenfolgen, gesicherter Kontext 334
API-Aufrufreihenfolgen, normaler Verbindungsaufbau 333
API-Aufrufreihenfolgen, ohne CONNECT RESET 333
API-Aufrufreihenfolgen, Verbindungskonzentrator 334
API-Versionen 330
Connect-Gateway 336
DATA_ENCRYPT 336
Einschränkungen 331
Fehlerbehandlung, Rückkehrcodes 331
Funktionsstruktur 327
Informationsstruktur 328
logischer Zielknoten 336
Pufferstruktur 329
Verbindungssteuerung 330
Fehlerbestimmung 317
Implementierung 314
Namenskonvention, Berechtigung 314
Position 314
Übersicht 313
EXPLAIN, Berechtigung
Details 45
Übersicht 37

Explizite gesicherte Verbindungen
Benutzer-ID wechseln 147, 154
herstellen 147

F

Fehler
Benutzer wechseln 156
gesicherte Kontexte 156
Fehlerbehebung
Lernprogramme 411
Onlineinformationen 411
Sicherheits-Plug-ins 241
Fehlerbestimmung
Lernprogramme 411
Sicherheits-Plug-ins 241
verfügbare Informationen 411
Fehlernachrichten
Sicherheit-Plug-ins 272
Feld 'Savepoint ID' 122
FGAC
siehe RCAC 159
Firewalls
Anwendungsproxy 229
Circuit-Level 230
Details 229
Screening-Router 229
Stateful Multi-Layer Inspection (SMLI) 230
Funktionen
Client-Plug-in
API db2secClientAuthPluginInit 292
API db2secClientAuthPluginTerm 294
API db2secDoesAuthIDExist 294
API db2secFreeInitInfo 295
API db2secFreeToken 296
API db2secGenerateInitialCred 296
API db2secGetAuthIDs 298
API db2secGetDefaultLoginContext 300
API db2secProcessServerPrincipalName 302
API db2secRemapUserid 302
API db2secServerAuthPluginInit 304
API db2secServerAuthPluginTerm 307
API db2secValidatePassword 307
Gruppen-Plug-in
API db2secDoesGroupExist 280
API db2secFreeErrormsg 280
API db2secFreeGroupListMemory 281
API db2secGetGroupsForUser 281
API db2secGroupPluginInit 285
API db2secPluginTerm 286
Skalar
DECRYPT_BIN 68
DECRYPT_CHAR 68
ENCRYPT 68
GETHINT 68
Zugriffsrechte 52

G

Geordnete Domänenlisten 391
Gesicherte Clients
Sicherheit auf CLIENT-Ebene 8
Gesicherte Kontexte
Fehlerbestimmung 156
Prüfrichtlinien 108
Übernahme der Rollenzugehörigkeit 153
Gesicherte Kontexte *(Forts.)*
Übersicht 149
Gesicherte Verbindungen
explizite gesicherte Verbindungen herstellen 147
Übersicht 149
Globale Gruppen, Unterstützung 385
GRANT, Anweisung
Beispiel 57
implizite Berechtigungen 59
Übersicht 57
GROUP_BASEDN, Parameter 257
GROUP_LOOKUP_ATTRIBUTE, Attribut 262
GROUP_LOOKUP_METHOD, Parameter
LDAP-Plug-in-Module konfigurieren 257, 262
GROUP_OBJECTCLASS, Parameter 257
GROUPNAME_ATTRIBUTE, Parameter 257
Gruppen
auswählen 5
Benutzerauthentifizierung 386
enumeration (Windows) 389
Namenseinschränkungen (Windows) 386
Rollenvergleich 143
Zugriffstoken 388
Gruppensuche, Unterstützung
Details 247, 262
GSKCapiCmd, Tool
SSL-Unterstützung konfigurieren 70, 77
GSKit
Regeln für Bibliotheken und Prozesse 85
Rückkehrcodes 87
SSL-Unterstützung konfigurieren 70, 77
GSS-APIs
Authentifizierungs-Plug-ins 310

H

Handshakes
Übersicht 82
Hilfe
SQL-Anweisungen 406
HP-UX 254

I

IBM Database Encryption Expert 100
IBMLDAPSecurity.ini, Datei 257
IKEYCMD, Tool 70, 77
iKeyman, Tool 70, 77
IMPLICIT_SCHEMA (implizites Schema), Berechtigung
Details 46
Übersicht 37
Implizite Berechtigung
verwalten 59
INDEX, Zugriffsrecht
Details 52
Indizes
Zugriffsrechte
Übersicht 52
INSERT, Zugriffsrecht 49
Instanzen
Berechtigungen 30
konfigurieren
SSL-Kommunikation 70
Instanzverzeichnisse 7

Integrierte Sichten
AUTHORIZATIONIDS
Beispiel 222
Zugriff beschränken 224
OBJECTOWNERS
Zugriff beschränken 224
PRIVILEGES
Beispiel 222
Zugriff beschränken 224

K

Kennsatzbasierte Zugriffssteuerung
siehe LBAC 183
Kennwörter
ändern
Linux 399
auf Servern verwalten 23
Kerberos-Authentifizierungsprotokoll
Aktivierung 20
Benennung 19
einrichten 17
IBM i-Kompatibilität 22
Plug-ins
erstellen 21
implementieren 246
Principals 19
Server 8
System z-Kompatibilität 22
Übersicht 16
Windows-Kompatibilität 22
Zuordnung 19
Konfiguration
LDAP
Plug-ins 257
KRB_SERVER_ENCRYPT, Authentifizierungstyp 8
Kurznamen
Zugriffsrechte
indirekt durch Pakete 61

L

LBAC
Aktualisierung von Daten 211
Ausnahmen
Auswirkung auf den Sicherheitskennsatzvergleich 195
Details 201
Berechtigungsnachweise 183
Daten einfügen 208
Daten lesen 205
geschützte Tabellen 183
Regelsätze
DB2LBACRULES 197
Details 196
Sicherheitskennsätze vergleichen 195
Schutz entfernen 219
Sicherheitsadministratoren 183
Sicherheitskennsätze
ARRAY, Komponententyp 188
Details 192
Erstellen 192
Erteilen (mit GRANT) 192
kompatible Datentypen 192
Komponenten 187
SET, Komponententyp 188
TREE, Komponententyp 189
LBAC *(Forts.)*
Sicherheitskennsätze *(Forts.)*
Übersicht 183
Vergleiche 195
widerrufen 192
Zeichenfolgeformat 194
Sicherheitsrichtlinien
Details 185
Übersicht 183
zu Tabelle hinzufügen 204
Übersicht 23, 183
LDAP
Plug-ins 257, 260
Sicherheits-Plug-ins 247
transparentes LDAP
LDAP 250, 253, 254, 256
LDAP, Kerberos 251
LDAP_HOST, Parameter 257
Lernprogramme
Fehlerbehebung 411
Fehlerbestimmung 411
Liste 411
pureXML 411
Linux 253
Sicherheit 399
LOAD, Berechtigung
Übersicht 37
LocalSystem, Konto
Berechtigung 34
SYSADM, Berechtigung 392
Unterstützung 393
Löschen
LBAC-Sicherheitskennsätze 192
Spalten (LBAC-geschützt) 216

M

Methoden
Zugriffsrechte 52
Migration
Rollen 145

N

Namenskonventionen
Windows-Einschränkungen 386
NESTED_GROUPS, Parameter 257

O

Objekte
Eigentumsrecht 23
OBJMAINT-Ereignisse 376
Option WITH DATA
Details 122

P

Pakete
Berechtigungs-IDs
Ableitung 53
verwenden 53
Eigentumsrecht 60
Zugriffsrechte
entziehen (Übersicht) 57

Pakete *(Forts.)*
Zugriffsrechte *(Forts.)*
Übersicht 51
Zugriffsrechte bei Abfragen 60
Plug-ins
Gruppen abrufen 278
GSS-API-Authentifizierung 310
ID-Authentifizierung 286
Kennwortauthentifizierung 286
LDAP 247
Sicherheit
APIs 273, 277
Einschränkungen (GSS-API-Authentifizierung) 311
Einschränkungen (Plug-in-Bibliotheken) 265
Einschränkungen (Zusammenfassung) 268
Fehlernachrichten 272
implementieren 242, 243, 244, 246, 400
Namenskonventionen 237
Rückkehrcodes 269
Versionen 240
Pluggable Authentication Module
PAM 250, 253, 254, 256
PRECOMPILE, Befehl
OWNER, Option 60
Protokolle
prüfen 106
Prozeduren
Zugriffsrechte 52
Prüffunktion
Aktionen 106
Archiv 120
asynchrones Schreiben von Datensätzen 130
Berechtigungen 106
CHECKING, Ereignistabelle 344
CHECKING-Ereignis, Gründe für Zugriffsgewährung 347
CHECKING-Ereignis, Typen von Zugriffsversuchen 349
CONTEXT, Ereignistabelle 368
Ereignisse 106
ERRORTYPE, Parameter 130
EXECUTE-Ereignisse 122, 370
EXECUTE-Zeitmarke 126
Fehlerbehandlung 130
Funktionsweise 130
Objektsatztypen 339
OBJMAINT, Ereignistabelle 353
Prüfdaten in Tabellen
Tabellen erstellen 117
Tabellen laden 118
Prüfereignistabelle 341
Prüfsätze für EXECUTE-Ereignisse 370
Richtlinien 108
Satzaufbau 339
Satzobjekttypen 339
SECMAINT, Ereignistabelle 356
SECMAINT-Berechtigungen 360
SECMAINT-Zugriffsrechte 360
synchrones Schreiben von Datensätzen 130
SYSADMIN, Ereignistabelle 364
Tipps und Verfahrensweisen 132
VALIDATE, Ereignistabelle 366
Zugriffsrechte 106
Prüfprotokolle
archivieren 113, 120
Dateinamen 116
Position 113
PUBLIC
Datenbankberechtigungen, automatisch erteilt 37

Q

QUIESCE_CONNECT, Berechtigung 37

R

RCAC
Bedingungen in Berechtigungen, Bedingungen in Regelmanagement für Masken
Skalarfunktionen, Bedingen in Berechtigungen, Bedingungen in Masken 161
ExampleBANK
siehe ExampleBANK RCAC-Szenario 176
ExampleHMO
siehe ExampleHMO, RCAC-Szenario 161
Regelmanagement
SQL-Anweisungen 160
Regeln 160
Szenario
siehe ExampleBANK RCAC-Szenario 176
siehe ExampleHMO, RCAC-Szenario 161
Übersicht 159
REFERENCES, Zugriffsrecht 49
Regelsätze (LBAC)
Ausnahmen 201
Details 196
Registrierdatenbankvariable
DB2COMM 70
RESTRICTIVE-Option des Befehls CREATE DATABASE
Zugriffsrechte an PUBLIC verweigern 224
REVOKE, Anweisung
Beispiel 57
implizit erteilen 59
Übersicht 57
Rollen
Details 137
erstellen 138
Hierarchien 140
im Vergleich zu Gruppen 143
von IBM Informix Dynamic Server migrieren 145
WITH ADMIN OPTION, Klausel 142
Zugriffsrechte entziehen 141
Routinenaufrufer-Berechtigungs-IDs 53
Rückkehrcodes
GSKit 87
Ruhende Daten (Data At Rest, DAR) 100

S

Schemata
Zugriffsrechte 47
SEARCH_DN, Parameter 257
SEARCH_PW, Parameter 257
SECADM (Sicherheitsadministrator), Berechtigung
Details 39
Übersicht 37
SECLABEL, Skalarfunktion
Übersicht 202
SECLABEL_BY_NAME, Skalarfunktion
Übersicht 202
SECLABEL_TO_CHAR, Skalarfunktion
Übersicht 202
SECMAINT-Ereignisse 376
SELECT, Zugriffsrecht 49
Sequenzen
Zugriffsrechte 52

SERVER, Authentifizierungstyp
Übersicht 8
SERVER_ENCRYPT, Authentifizierungstyp
Übersicht 8
Serverauthentifizierung, Plug-ins 247
SET ENCRYPTION PASSWORD, Anweisung
Kennwörter verschlüsseln 68
SETSESSIONUSER, Zugriffsrecht
Details 47
Sicherheit
APIs (Exitbibliothek für Kommunikationspuffer) 318
Authentifizierung 2
CLIENT-Ebene 8
Daten 1
db2extsec, Befehl 394
erweiterte Sicherheit 394
erweiterte Sicherheit aktivieren 394
erweiterte Sicherheit inaktivieren 394
Exitbibliothek für Kommunikationspuffer
aktivieren 315, 316
API-Aufrufreihenfolge, SET SESSION AUTHORIZATION 335
API-Aufrufreihenfolgen 333
API-Aufrufreihenfolgen, gesicherter Kontext 334
API-Aufrufreihenfolgen, normaler Verbindungsaufbau 333
API-Aufrufreihenfolgen, ohne CONNECT RESET 333
API-Aufrufreihenfolgen, Verbindungskonzentrator 334
API-Versionen 330
Bibliothek laden 318
Connect-Gateway 336
DATA_ENCRYPT 336
Einschränkungen 331
Entwicklung 317
Fehlerbehandlung, Rückkehrcodes 331
Fehlerbestimmung 317
Funktionsstruktur 327
Implementierung 314
Informationsstruktur 328
logischer Zielknoten 336
Namenskonvention, Berechtigung 314
Position 314
Pufferstruktur 329
Übersicht 313
Verbindungssteuerung 330
explizite gesicherte Verbindungen herstellen 147
Gesicherte Kontexte 149
kennsatzbasierte Zugriffssteuerung (Label-Based Access Control, LBAC) 183
Kennwortverwaltung auf Servern 23
Plug-ins
aktivieren 231
API zum Prüfen von Kennwörtern 307
APIs 277, 280, 281, 285, 286, 292, 294, 295, 296, 298, 300, 302, 304, 307
APIs (Benutzer-ID/Kennwort) 286
APIs (Gruppenabruf) 278
APIs (GSS-API) 310
APIs (Versionen) 240
Aufrufreihenfolge 273
Benennung 237
Bibliotheken 236
Einschränkungen für Bibliotheken 265
entwickeln 231
Fehlerbestimmung 241
Fehlernachrichten 272
GSS-API (Einschränkungen) 311
Sicherheit *(Forts.)*
Plug-ins *(Forts.)*
GSS-API (Implementierung) 244
Hinweise zu 32-Sicherheits-Plug-ins 241
Hinweise zu 64-Bit-Sicherheits-Plug-ins 241
implementieren 231, 242, 243, 244, 246, 268, 400
Initialisierung 263
laden 231, 263
LDAP (Lightweight Directory Access Protocol) 247
Rückkehrcodes 269
SQLCODE-Werte 241
SQLSTATE-Werte 241
Übersicht 231
zweiteilige Benutzer-IDs, Unterstützung 238
Risiken 66
spaltenspezifisch 183
UNIX 399
Windows
Benutzer 392
Domänensicherheit 392
erweitert 394
Übersicht 383
zeilenspezifisch 183
Zugriffssteuerung für Zeilen und Spalten, differenzierte Zugriffssteuerung
siehe RCAC 159
Sicherheitskennsätze (LBAC)
ARRAY, Komponententyp 188
kompatible Datentypen 192
Komponenten 187
Richtlinien
Details 185
SET, Komponententyp 188
TREE, Komponententyp 189
verwenden 192
Zeichenfolgeformat 194
Sichten
Berechtigungsinformationen 224
Spaltenzugriff 62
Tabellenzugriffssteuerung 62
Zugriff auf Zeilen 62
Zugriffsrechte, Beispiele 62
Sitzungsberechtigungs-IDs
Übersicht 53
Solaris-Betriebssystem 256
Spalten
LBAC-geschützte
aktualisieren 211
einfügen 208
lesen 205
löschen 216
LBAC-Schutz
entfernen 219
hinzufügen 204
SQL-Anweisungen
Berechtigungs-IDs 53
Hilfe
anzeigen 406
SQLADM (SQL-Verwaltung)
Details 43
SQLADM (SQL-Verwaltung), Berechtigung
Übersicht 37
SSL
CATALOG TCPIP NODE, Befehl 77
Cipher Suites 84
CLI-Clients 77
CLP-Clients 77

SSL *(Forts.)*
DB2 Connect 70
digitale Zertifikate 83
eingebettete SQL-Clients 77
Handshake 82
Konfiguration
DB2-Clients 77
konfigurieren
DB2-Instanzen 70
Protokoll 82
Zertifizierungsstellen (CAs) 83
ssl_cipherspecs, Konfigurationsparameter
Cipher Suites angeben 70, 84
ssl_client_keystash, Verbindungsparameter
SSL konfigurieren 77
ssl_client_keystoredb, Verbindungsparameter
SSL konfigurieren 77
ssl_clnt_keydb, Konfigurationsparameter
SSL konfigurieren 77
ssl_clnt_stash, Konfigurationsparameter
SSL konfigurieren 77
SSL_KEYFILE 257
SSL_PW 257
SSL_RSA_FIPS_WITH_3DES_EDE_CBC_SHA, Cipher Suite 70
ssl_svcename, Konfigurationsparameter
SSL konfigurieren 70
ssl_svr_keydb, Konfigurationsparameter
SSL konfigurieren 70
ssl_svr_stash, Konfigurationsparameter
SSL konfigurieren 70
ssl_versions, Konfigurationsparameter
SSL konfigurieren 70
SSLClientKeystash, Verbindungsparameter
SSL konfigurieren 77
SSLClientKeystoredb, Verbindungsparameter
SSL konfigurieren 77
Standardzugriffsrechte 55
Statisches SQL
EXECUTE, Zugriffsrecht 60
SYSADM (Systemverwaltung)
Details 34
Windows 392
sysadm_group, Konfigurationsparameter
Windows 392
SYSADMIN-Ereignisse 376
SYSCAT-Sichten
Sicherheitsaspekte 221
SYSCTRL (Systemsteuerung), Berechtigung
Details 35
SYSDEFAULTADMWORKLOAD, Auslastung 52
SYSDEFAULTUSERWORKLOAD, Auslastung 52
SYSMAINT (Systempflege), Berechtigung
Details 36
SYSMON (Systemmonitor)
Details 36
SYSPROC.AUDIT_ARCHIVE, gespeicherte Prozedur 113, 120
SYSPROC.AUDIT_DELIM_EXTRACT, gespeicherte Prozedur 113, 120
SYSPROC.AUDIT_LIST_LOGS, gespeicherte Prozedur 120
Systemberechtigungs-IDs 53
Systemkataloge
abrufen
Berechtigungsnamen mit Zugriffsrechten 222
Namen erteilte Zugriffsrechte 224
Namen mit Berechtigung DBADM 223
Namen mit Tabellenzugriffsberechtigung 223
Systemkataloge *(Forts.)*
Sicherheit 224
Zugriffsrechte auflisten 221

T

Tabellen
Auswirkung von LBAC auf das Lesen 205
in LBAC-geschützte einfügen 208
Informationen abrufen
Namen mit Zugriff 223
LBAC-Schutz entfernen 219
mit LBAC schützen 183, 204
Prüfrichtlinie 108
Zugriffsrechte 57
Zugriffsrechte entziehen 57
Zugriffssteuerung 62
Tabellenbereiche
Zugriffsrechte 49
TLS (Transport Layer Security) 82
TLS_RSA_WITH_3DES_EDE_CBC_SHA, Cipher Suite 70, 84
TLS_RSA_WITH_AES_128_CBC_SHA, Cipher Suite 70, 84
TLS_RSA_WITH_AES_256_CBC_SHA, Cipher Suite 70, 84
Transport Layer Security (TLS)
Übersicht 82

U

UPDATE, Zugriffsrecht 49
USAGE, Zugriffsrecht
Auslastungen 52
Details 52
USER_BASEDN 257
USER_OBJECTCLASS 257
USERID_ATTRIBUTE 257

V

VALIDATE-Ereignisse 376
Verschlüsselung
Daten 68
öffentlicher Schlüssel 84
Verschlüsselung mit öffentlichem Schlüssel 84
Vertrauensstellungen
Windows 387
Vista
Benutzerzugriffssteuerungsfunktion (User Access Control, UAC) 398

W

wechseln
Benutzer-IDs 154
Wechseln
Benutzer-IDs 147
Wiederholen
EXECUTE-Zeitmarke 128
Windows
Benutzerkonten
Zugriffstoken 388
erweiterte Sicherheit 394
lokales Systemkonto (LSA), Unterstützung 393
Szenarios
Clientauthentifizierung 385
Serverauthentifizierung 384

WITH ADMIN OPTION, Klausel
Rollenverwaltung delegieren 142
WLMADM (Workloadverwaltung), Berechtigung
Details 45
Übersicht 37
Write-Down
Details 197
Write-Up
Details 197

X

XQuery
dynamisch
EXECUTE, Zugriffsrecht 60
statisch
EXECUTE, Zugriffsrecht 60

Z

Zeilen
aktualisieren
LBAC-geschützte Daten 211
bei Verwendung von LBAC lesen 205
einfügen
LBAC-geschützte Daten 208
LBAC-Schutz entfernen 219
löschen
LBAC-geschützte Daten 216
mit LBAC schützen 204
Zertifizierungsstellen (CAs)
digitale Zertifikate 83
Zugriffsrechte
ALTER
Sequenzen 52
Tabellen 49
CONTROL 49
DELETE 49
durch Rolle des gesicherten Kontexts anfordern 153
Eigentumsrecht 23
entziehen
Rollen 141
Übersicht 57
erteilen
Rollen 143
EXECUTE
Routinen 52
GRANT, Anweisung 57
Hierarchie 23
INDEX 49
indirekt
Pakete mit Kurznamen 61
individuell 23
Indizes
Übersicht 52
Informationen zu erteilten Rechten
abrufen 222, 224
INSERT 49
Pakete
erstellen 51
implizit 23
planen 3
REFERENCES 49
Rollen 137
Schemata 47
SELECT 49
Zugriffsrechte *(Forts.)*
SETSESSIONUSER 47
Sichten 49
Systemkatalog
Berechtigungsinformationen 221
Zugriff beschränken 224
Tabellen 49
Tabellenbereiche 49
Übersicht 23
UPDATE 49
USAGE
Auslastungen 52
Sequenzen 52
Zugriffssteuerung
Authentifizierung 8
DBADM (Datenbankadministrator), Berechtigung 64
differenziert, Zeilen und Spalten
siehe RCAC 159
kennsatzbasierte Zugriffssteuerung (LBAC) 183
Sichten 62
spaltenspezifisch 183
Tabellen 62
zeilenspezifisch 183
Zugriffssteuerung für Zeilen und Spalten
siehe RCAC 159
Zugriffstoken
Windows 388

SC12-4693-00

Spine information:

IBM DB2 10.1 for Linux, UNIX and Windows

Datenbanksicherheit