



MobiLink™ Clientadministration

Version 16.0

Februar 2013

Version 16.0
Februar 2013

© 2013 SAP AG oder ein SAP-Konzernunternehmen. Alle Rechte vorbehalten.

Sie können diese Dokumentation (ganz oder teilweise) unter folgenden Bedingungen benutzen, reproduzieren und verteilen: 1) Sie müssen diese und alle anderen Urheberrechtsvermerke auf allen Kopien oder Auszügen der Dokumentation wiedergeben. 2) Sie dürfen die Dokumentation nicht verändern. 3) Sie dürfen nichts tun, aus dem abgeleitet werden könnte, dass Sie oder jemand anderer als SAP Verfasser oder Quelle der Dokumentation ist. Die hier enthaltenen Informationen können jederzeit ohne vorherigen Hinweis geändert werden.

Einige Softwareprodukte, die von der SAP AG oder einem ihrer Vertriebspartner vermarktet werden, enthalten Softwarekomponenten anderer Softwareanbieter. Die nationalen Produktspezifikationen können unterschiedlich sein.

Diese Dokumentationen werden von der SAP AG und ihren Tochtergesellschaften ("SAP Group") lediglich zu Informationszwecken bereitgestellt, ohne dass eine Gewährleistung oder eine Garantie irgendeiner Art gegeben wird. Die SAP Group übernimmt keine Verantwortung im Hinblick auf Fehler oder Auslassungen in den Dokumentationen. Die einzigen Garantien für Produkte und Dienstleistungen der SAP Group sind diejenigen, die in den mit den Produkten und Dienstleistungen eventuell gelieferten ausdrücklichen Garantieerklärungen enthalten sind. Keine der hier enthaltenen Informationen kann als Gewährung einer weitergehenden Garantie betrachtet werden.

SAP und weitere erwähnte SAP-Produkte und -Dienstleistungen sowie die entsprechenden Logos sind Marken oder eingetragene Marken der SAP AG in Deutschland und anderen Ländern. Weitere Hinweise finden Sie unter <http://www.sap.com/corporate-en/legal/copyright/index.epx#trademark>.

Inhalt

Über diese Dokumentation	v
Einführung in MobiLink-Clients	1
MobiLink-Clients	1
MobiLink-Benutzer	4
Dienstprogramme für MobiLink-Clients	19
Netzwerkprotokolloptionen des MobiLink-Clients	25
Schemaänderungen in entfernten MobiLink-Clients	62
SQL Anywhere-Clients für MobiLink	69
SQL Anywhere-Clients	69
MobiLink SQL Anywhere Client-Dienstprogramm (dbmlsync)	106
Erweiterte Optionen von MobiLink SQL Anywhere-Clients	143
MobiLink SQL-Anweisungen	181
MobiLink-Synchronisationsprofile	182
Ereignis-Hooks für SQL Anywhere-Clients	202
Dbmlsync C++-API-Referenz	267
Dbmlsync .NET-API-Referenz	291
DBTools-Schnittstelle für dbmlsync	317
Skriptgesteuerter Upload	323
Index	351

Über diese Dokumentation

In diesem Handbuch wird beschrieben, wie Sie MobiLink-Clients einrichten, konfigurieren und synchronisieren. MobiLink-Clients können SQL Anywhere- oder UltraLite-Datenbanken sein. Das Handbuch enthält auch Beschreibungen der Dbmlsync-API, mit der Sie die Synchronisation in Ihre C++- oder .NET-Clientanwendungen integrieren können.

Einführung in MobiLink-Clients

In diesem Abschnitt werden die Clients behandelt, die Sie bei der MobiLink-Synchronisation verwenden können. Er enthält außerdem Informationen, die für alle Typen von MobiLink-Clients gelten.

MobiLink-Clients

Der MobiLink-Server unterstützt derzeit die Verwendung von zwei Clients. Die unterstützten MobiLink-Clients sind folgende:

- SQL Anywhere
- UltraLite

Die folgenden Abschnitte enthalten Informationen über die unterstützten MobiLink-Clients sowie Informationen, die für alle MobiLink-Clients gelten.

SQL Anywhere-Clients

Um eine SQL Anywhere-Datenbank als MobiLink-Client zu verwenden, fügen Sie der Datenbank Synchronisationsobjekte hinzu. Die hinzuzufügenden Objekte sind Publikationen, MobiLink-Benutzer und Subskriptionen, die Benutzer mit Publikationen verbinden. Siehe:

- [„SQL Anywhere-Datenbanken als entfernte Datenbanken verwenden“ auf Seite 69](#)
- [„Publikationen“ auf Seite 74](#)
- [„MobiLink-Benutzer“ auf Seite 82](#)
- [„Erstellen von Synchronisationssubskriptionen“ auf Seite 86](#)

Die Synchronisation kann unter Verwendung der Dbmlsync-API, der SQL SYNCHRONIZE-Anweisung oder des Befehlszeilendienstprogramms dbmlsync initiiert werden. Die meisten Synchronisationen lesen Daten aus dem Transaktionslog der Datenbank, eine Transaktionslogdatei ist jedoch nicht für die skriptgesteuerte Upload-Synchronisation und die Synchronisation von reinen Download-Publikationen erforderlich.

Siehe [„Initiieren einer Synchronisation“ auf Seite 88](#).

Weitere Hinweise zu SQL Anywhere-Clients finden Sie unter [„SQL Anywhere-Clients“ auf Seite 69](#).

Hinweise zu den Befehlszeilenoptionen für dbmlsync finden Sie unter [„MobiLink SQL Anywhere Client-Dienstprogramm \(dbmlsync\)“ auf Seite 106](#).

Weitere Hinweise zur Anpassung der Synchronisation finden Sie unter [„Anpassen der dbmlsync-Synchronisation“ auf Seite 102](#).

UltraLite-Clients

UltraLite-Anwendungen werden automatisch mit MobiLink-Unterstützung versehen, sobald die Anwendung einen Aufruf für die entsprechende Synchronisationsfunktion enthält.

Die UltraLite-Anwendung und die UltraLite-Bibliotheken verarbeiten die Synchronisationsaktivitäten auf Anwendungsseite. Sie können Ihre UltraLite-Anwendung schreiben, ohne dabei auf die Synchronisation achten zu müssen. Die UltraLite-Laufzeitbibliothek protokolliert die Änderungen, die seit der vorhergehenden Synchronisation vorgenommen wurden.

Wenn Sie TCP/IP, HTTP, HTTPS oder Microsoft ActiveSync verwenden, wird die Synchronisation von Ihrer Anwendung durch einen einfachen Aufruf einer Synchronisationsfunktion initiiert.

Siehe auch

- „ActiveSync mit UltraLite unter Windows Mobile“ [[UltraLite - Datenbankverwaltung](#)]
- „UltraLite als MobiLink-Client“ [[UltraLite - Datenbankverwaltung](#)]
- [UltraLite - Datenbankverwaltung](#)

Netzwerkprotokolle für Clients

Der MobiLink-Server benutzt die Befehlszeilenoption `-x` zur Angabe des Netzwerkprotokolls bzw. -protokolle, über das bzw. die die Synchronisationsclients die Verbindung zum MobiLink-Server herstellen. Die Art des gewählten Netzwerkprotokolls muss mit dem vom Client benutzten Synchronisationsprotokoll übereinstimmen.

Die Syntax für Befehlszeilenoption `mlsrv16` lautet:

mlsrv16 -c "connection-string" -x protocol(options)

Im folgenden Beispiel wird das Protokoll TCP/IP ohne weitere Protokolloptionen gewählt.

```
mlsrv16 -c "DSN=SQL Anywhere 16 Demo" -x tcpip
```

Sie können Ihr Protokoll mithilfe von Optionen wie folgt konfigurieren:

(*keyword=value*;...)

Zum Beispiel:

```
mlsrv16 -c "DSN=SQL Anywhere 16 Demo" -x tcpip(  
    host=localhost;port=2439)
```

Siehe auch

Umfassende Hinweise zu MobiLink-Netzwerkprotokollen und Protokolloptionen finden Sie hier:

Thema	Siehe...
So stellen Sie Netzwerkoptionen für den MobiLink-Server ein	„mlsrv16-Option -x“ [MobiLink - Serveradministration]

Thema	Siehe...
Alle Netzwerkprotokolloptionen für MobiLink-Clientanwendungen	„Netzwerkprotokolloptionen des MobiLink-Clients“
So stellen Sie Optionen für SQL Anywhere-Clients ein	„Erweiterte Option CommunicationAddress (adr)“ „Erweiterte Option CommunicationType (ctp)“
So stellen Sie Optionen für UltraLite-Clients ein	„Synchronisationsparameter Stream Parameters“ [<i>UltraLite - Datenbankverwaltung</i>] „Synchronisationsparameter Stream Type“ [<i>UltraLite - Datenbankverwaltung</i>] „UltraLite-Synchronisationsdienstprogramm (ulsync)“ [<i>UltraLite - Datenbankverwaltung</i>]

Systemtabellen in MobiLink

MobiLink-Server-Systemtabellen

Wenn Sie eine Datenbank als konsolidierte Datenbank einrichten möchten, werden MobiLink-Systemtabellen erstellt, die vom MobiLink-Server benötigt werden.

Siehe „Systemtabellen des MobiLink-Servers“ [*MobiLink - Serveradministration*].

UltraLite-Systemtabellen

Das Schema einer UltraLite-Datenbank wird in einem systemeigenen Format gespeichert.

Weitere Hinweise zu den UltraLite-Systemtabellen finden Sie unter „UltraLite-Systemtabellen“ [*UltraLite - Datenbankverwaltung*].

SQL Anywhere-Systemtabellen

SQL Anywhere-Systemtabellen können nicht direkt aufgerufen werden, sondern über Systemansichten. Siehe „Systemansichten“ [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Von besonderem Interesse für MobiLink-Benutzer sind folgende SQL Anywhere-Systemansichten:

- „SYSSYNC-Systemansicht“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- „SYSPUBLICATION-Systemansicht“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- „SYSSUBSCRIPTION-Systemansicht“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- „SYSSYNCSCRIPT-Systemansicht“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- „SYSSYNCPROFILE-Systemansicht“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- „SYSARTICLE-Systemansicht“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- „SYSARTICLECOL-Systemansicht“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]

SQL Anywhere stellt ebenfalls konsolidierte Ansichten zur Verfügung, die Systemansichten abfragen, um dem Benutzer potentiell nützliche Informationen anzuzeigen. Siehe „[Konsolidierte Ansichten](#)“ [[SQL Anywhere Server - SQL-Referenzhandbuch](#)].

MobiLink-Benutzer

Ein **MobiLink-Benutzer**, auch **Synchronisationsbenutzer** genannt, ist der Name, mit dem Sie sich authentifizieren, wenn Sie sich mit dem MobiLink-Server verbinden.

Damit ein Benutzer Teil eines Synchronisationssystems ist, müssen folgende Voraussetzungen erfüllt sein:

- In der entfernten Datenbank muss ein MobiLink-Benutzername erstellt worden sein.
- Der MobiLink-Benutzername muss beim MobiLink-Server registriert sein.

MobiLink-Benutzernamen und Kennwörter unterscheiden sich von den Benutzernamen und Kennwörtern für Datenbanken. MobiLink-Benutzernamen dienen der Authentifizierung beim Verbinden der entfernten Datenbank mit dem MobiLink-Server.

Bei MobiLink-Benutzernamen wird immer zwischen Groß- und Kleinschreibung unterschieden, es sei denn, Sie verwenden benutzerdefinierte Authentifizierungsskripten. Dies bedeutet, dass der von der entfernten Datenbank bereitgestellte Benutzername auch hinsichtlich der Groß- und Kleinschreibung exakt mit dem in der konsolidierten Datenbank registrierten Benutzernamen übereinstimmen muss. Beachten Sie bei der Definition von MobiLink-Benutzernamen Folgendes:

- **In einem MobiLink-Synchronisationssystem können keine zwei Benutzernamen identisch sein, auch dann nicht, wenn sie sich nur hinsichtlich der Groß- und Kleinschreibung unterscheiden.** Beispiel: Sie können entweder den Benutzernamen **aA** oder den Benutzernamen **Aa**, nicht jedoch beide verwenden.
- **Bei allen Benutzernamen müssen Sie immer dieselbe Schreibung verwenden, sobald sie einmal in Gebrauch sind.** Beispiel: Wenn Sie einen Benutzer als **Aa** hinzufügen und eine Synchronisation damit durchführen, müssen Sie weiterhin mit **Aa** synchronisieren. Verwenden Sie dann **aA**, schlägt die Synchronisation fehl.

Bei der Verwendung von benutzerdefinierten Authentifizierungsskripten gibt das Skript die Verhaltensweise bezüglich Groß- und Kleinschreibung bei Benutzernamen vor.

Sie können die Benutzernamen auch einsetzen, um das Verhalten des MobiLink-Servers zu steuern. Dazu verwenden Sie den Parameter **username** in Synchronisationsskripten. Siehe „[Entfernte IDs und MobiLink-Benutzernamen in Skripten](#)“ auf Seite 11.

Der MobiLink-Benutzername wird in der Namensspalte der MobiLink-Systemtabelle **ml_user** in der konsolidierten Datenbank gespeichert.

Der MobiLink-Benutzername muss innerhalb des Synchronisationssystems nicht eindeutig sein. Wenn Sicherheit keine wichtige Rolle spielt, können Sie auch jeder entfernten Datenbank denselben MobiLink-Benutzernamen zuweisen.

UltraLite-Benutzerauthentifizierung

Die Benutzerauthentifizierungsschemata bei UltraLite und MobiLink sind getrennt. Sie können der Einfachheit halber die Werte der UltraLite-Benutzer-IDs als MobiLink-Benutzernamen verwenden. Dies funktioniert jedoch nur, wenn die UltraLite-Anwendung von nur einem Benutzer verwendet wird.

Siehe „UltraLite-Benutzer“ [[UltraLite - Datenbankverwaltung](#)].

Erstellen und Registrieren von MobiLink-Benutzern

Sie erstellen einen MobiLink-Benutzer in der entfernten Datenbank und registrieren ihn in der konsolidierten Datenbank.

Benutzer in der entfernten Datenbank erstellen

Sie können Benutzer wie folgt in die entfernte Datenbank einfügen:

- Für entfernte SQL Anywhere-Datenbanken verwenden Sie Sybase Central oder die Anweisung CREATE SYNCHRONIZATION USER.

Siehe „MobiLink-Benutzer“ auf Seite 82.

- Für entfernte UltraLite-Datenbanken verwenden Sie die Synchronisationsparameter für den Benutzernamen (z.B. Username) und für das Kennwort (Password).

Siehe „Synchronisationsparameter User Name“ [[UltraLite - Datenbankverwaltung](#)] und „Synchronisationsparameter Password“ [[UltraLite - Datenbankverwaltung](#)].

MobiLink-Benutzernamen zur konsolidierten Datenbank hinzufügen

Wenn Benutzernamen in der entfernten Datenbank erstellt wurden, können Sie sie mit einer der folgenden Methoden in der konsolidierten Datenbank registrieren:

- Verwenden Sie das mluser-Dienstprogramm.
Siehe „Dienstprogramm für die MobiLink-Benutzerauthentifizierung (mluser)“ [[MobiLink - Serveradministration](#)].
- Verwenden Sie Sybase Central.
- Implementieren Sie ein Skript für das authenticate_user- oder das authenticate_user_hashed-Ereignis. Wenn eines dieser Skripten aufgerufen wird, fügt der MobiLink-Server automatisch Benutzer hinzu, die erfolgreich authentifiziert werden.

Siehe „authenticate_user (Verbindungsereignis)“ [[MobiLink - Serveradministration](#)] oder „authenticate_user_hashed (Verbindungsereignis)“ [[MobiLink - Serveradministration](#)].

- Geben Sie die Befehlszeilenoption -zu+ mit mlsrv16 an. In diesem Fall werden alle bestehenden MobiLink-Benutzer, die der konsolidierten Datenbank noch nicht hinzugefügt wurden, bei der ersten Synchronisation hinzugefügt. Diese Option ist während der Entwicklung nützlich, aber für laufende Anwendungen nicht zu empfehlen.

Siehe „mlsrv16-Option -zu“ [[MobiLink - Serveradministration](#)].

Anfängliches MobiLink-Kennwort für einen Benutzer angeben (Sybase Central)

Das Kennwort für jeden Benutzer wird mit dem Benutzernamen in der Tabelle ml_user gespeichert. Sybase Central bietet eine praktische Möglichkeit, einzelne Benutzer und Kennwörter hinzuzufügen.

Voraussetzungen

Es gibt keine Voraussetzungen für diese Aufgabe.

Kontext und Bemerkungen

Wenn Sie einen Benutzer ohne Kennwort erstellen, authentifiziert MobiLink den Benutzer nicht und kein Kennwort ist erforderlich, um eine Verbindung herzustellen oder um zu synchronisieren.

Aufgabe

1. Doppelklicken Sie auf Ihr MobiLink-Projekt.
2. Doppelklicken Sie auf **Konsolidierte Datenbanken** und klicken Sie auf den Namen Ihrer konsolidierten Datenbank.
3. Klicken Sie auf **Benutzer**.
4. Klicken Sie auf **Datei**→**Neu**→**Benutzer**.
5. Befolgen Sie die Anweisungen des **Assistenten zum Erstellen von Benutzern**.

Ergebnisse

Der Benutzer wird mit dem angegebenen Kennwort erstellt.

Siehe auch

- „Dienstprogramm für die MobiLink-Benutzerauthentifizierung (mluser)“ [[MobiLink - Serveradministration](#)]

Anfängliche MobiLink-Kennwörter angeben (Dienstprogramm mluser)

Das Kennwort für jeden Benutzer wird mit dem Benutzernamen in der Tabelle ml_user gespeichert. Das Dienstprogramm mluser ist für das Hinzufügen von mehreren Benutzern und Kennwörtern im Batch geeignet.

Voraussetzungen

Es gibt keine Voraussetzungen für diese Aufgabe.

Kontext und Bemerkungen

Wenn Sie einen Benutzer ohne Kennwort erstellen, authentifiziert MobiLink den Benutzer nicht und kein Kennwort ist erforderlich, um eine Verbindung herzustellen oder um zu synchronisieren.

Aufgabe

1. Erstellen Sie eine Datei, die in jeder Zeile einen einzelnen Benutzernamen und ein Kennwort enthält, die durch eine Leerstelle getrennt sind.
2. Führen Sie an einer Eingabeaufforderung das Dienstprogramm `mluser` mit der Option `-f` aus, wenn Sie eine Datei mit Benutzernamen und Kennwörtern haben, oder mit den Optionen `-u` und `-p`, wenn Sie einen einzelnen Benutzernamen und ein Kennwort angeben.

Ergebnisse

Die angegebenen Benutzer und Kennwörter werden erstellt.

Beispiel

In der folgenden Befehlszeile wird mit der Option `-c` eine ODBC-Verbindung mit der konsolidierten Datenbank angegeben. Die Option `-f` gibt die Datei an, in der die Benutzernamen und Kennwörter gespeichert sind.

```
mluser -c "DSN=my_dsn" -f password-file
```

Siehe auch

- „Dienstprogramm für die MobiLink-Benutzerauthentifizierung (`mluser`)“ [[MobiLink - Serveradministration](#)]
- „Anfängliches MobiLink-Kennwort für einen Benutzer angeben (Sybase Central)“ auf Seite 6

Synchronisationen durch neue Benutzer

Normalerweise muss jeder MobiLink-Client einen gültigen MobiLink-Benutzernamen und ein Kennwort eingeben, wenn er eine Verbindung zu einem MobiLink-Server herstellen möchte.

Wenn Sie beim Start des MobiLink-Servers die Option `-zu+` verwenden, kann der Server Synchronisationsanforderungen von nicht registrierten Benutzern akzeptieren und durchführen. Wenn eine Anforderung von einem Benutzer eintrifft, der nicht in der `ml_user`-Tabelle registriert ist, wird die Anforderung ausgeführt und der Benutzer in die `ml_user`-Tabelle eingefügt.

Wenn Sie `-zu+` verwenden und ein MobiLink-Client mit einem Benutzernamen synchronisiert, der nicht in der aktuellen Tabelle `ml_user` enthalten ist, geht MobiLink standardmäßig wie folgt vor:

- **Neue Benutzer ohne Kennwort** Wenn der Benutzer kein Kennwort angegeben hat, wird der Benutzername in die Tabelle `ml_user` mit dem Kennwort `NULL` eingetragen. Dieser Benutzer kann ohne Kennwort synchronisieren.
- **Neue Benutzer mit Kennwort** Wenn der Benutzer einen Benutzernamen und ein Kennwort eingibt, werden beide in die Tabelle `ml_user` eingefügt und der Benutzername wird zu einem

bekannten Namen in Ihrem MobiLink-System. In Zukunft muss dieser Benutzer dasselbe Kennwort zum Synchronisieren eingeben.

- **Neue Benutzer mit einem neuen Kennwort** Ein neuer Benutzer kann Informationen in das Feld für das neue Kennwort oder in das Feld für das vorhandene Kennwort eingeben. In beiden Fällen haben die Einstellungen für das neue Kennwort Vorrang vor den alten Kennworteinstellungen und der neue Benutzer wird mit dem neuen Kennwort in das MobiLink-System aufgenommen. In Zukunft muss dieser Benutzer dasselbe Kennwort zum Synchronisieren eingeben.

Synchronisation durch unbekannte Benutzer verhindern

Standardmäßig erkennt der MobiLink-Server nur Benutzer, die in der ml_user-Tabelle registriert sind. Diese Standardeinstellung bietet zwei Vorteile. Erstens reduziert sie das Risiko eines nicht autorisierten Zugriffs auf den MobiLink-Server. Zweitens verhindert sie, dass autorisierte Benutzer versehentlich eine Verbindung mit einem falschen bzw. falsch geschriebenen Benutzernamen herstellen. Solche Zwischenfälle sollten vermieden werden, weil sie zu unerwartetem Verhalten seitens des MobiLink-Systems führen können.

Siehe auch

- „mlsrv16-Option -zu“ [\[MobiLink - Serveradministration\]](#)

Endbenutzerkennwörter

Jeder einzelne Endbenutzer muss jedes Mal, wenn er von einem MobiLink-Client aus synchronisiert, einen MobiLink-Benutzernamen und ein Kennwort eingeben, es sei denn, Sie deaktivieren die Benutzerauthentifizierung für Ihren MobiLink-Server.

Das Verfahren zur Angabe des Benutzernamens und des Kennworts ist für UltraLite-Clients und SQL Anywhere-Clients unterschiedlich.

- **UltraLite** Bei der Synchronisation muss der UltraLite-Client im Kennwortfeld der Synchronisationsstruktur ein gültiges Kennwort bereitstellen. Bei der integrierten MobiLink-Synchronisation ist ein Kennwort gültig, das mit dem Wert in der MobiLink-Systemtabelle ml_user übereinstimmt.

Ihre Anwendung sollte den Endbenutzer auffordern, vor der Synchronisation seinen MobiLink-Benutzernamen und sein Kennwort einzugeben.

- **SQL Anywhere** Benutzer können mithilfe der Option -mp ein gültiges Kennwort an der dbmlsync-Befehlszeile eingeben oder es mithilfe der erweiterten Option MobilinkPwd in der Datenbank mit der Synchronisationssubskription speichern. Andernfalls werden die Benutzer im dbmlsync-Verbindungsfenster aufgefordert, ein Kennwort anzugeben. Die zuletzt genannte Methode ist sicherer als die Angabe des Kennworts an der Befehlszeile, da Befehlszeilen für andere, auf demselben Computer laufende Prozesse einsehbar sind.

Wenn die Authentifizierung fehlschlägt, wird der Benutzer aufgefordert, den Benutzernamen und das Kennwort erneut einzugeben.

Siehe auch

- „UltraLite-Synchronisationsparameter“ [[UltraLite - Datenbankverwaltung](#)]
- „UltraLite-Netzwerkprotokolloptionen für dbmlsync“ [[UltraLite - Datenbankverwaltung](#)]
- „dbmlsync-Option -c“ auf Seite 116
- „dbmlsync-Option -mp“ auf Seite 125
- „Erweiterte Option MobiLinkPwd (mp)“ auf Seite 162

Kennwortänderungen

MobiLink bietet ein Verfahren, mit dem Endbenutzer ihr Kennwort ändern können. Die Schnittstelle ist für UltraLite- und SQL Anywhere-Clients unterschiedlich.

Das Verfahren zur Angabe des Benutzernamens und des Kennworts ist für UltraLite-Clients und SQL Anywhere-Clients unterschiedlich.

- **SQL Anywhere** Geben Sie in der dbmlsync-Befehlszeile ein gültiges vorhandenes Kennwort zusammen mit dem neuen Kennwort ein. Wenn Sie keine Befehlszeilenparameter verwenden, können Sie die Kennwörter im dbmlsync-Verbindungsfenster eingeben.

Siehe „dbmlsync-Option -mp“ auf Seite 125 und „dbmlsync-Option -mn“ auf Seite 124.

- **UltraLite** Bei der Synchronisation muss die Anwendung das vorhandene Kennwort im Kennwortfeld der Synchronisationsstruktur und das neue Kennwort im Feld new_password übergeben.

Siehe „Synchronisationsparameter Password“ [[UltraLite - Datenbankverwaltung](#)] und „Synchronisationsparameter New Password“ [[UltraLite - Datenbankverwaltung](#)].

Ein anfängliches Kennwort kann auf dem konsolidierten Datenbankserver oder beim ersten Synchronisationsversuch festgelegt werden. Siehe „Anfängliches MobiLink-Kennwort für einen Benutzer angeben (Sybase Central)“ auf Seite 6 und „Synchronisationen durch neue Benutzer“ auf Seite 7.

Wenn ein Kennwort zugeordnet wurde, können Sie es vom Client aus nicht auf den einen leeren Wert setzen.

Entfernte IDs

Die **entfernte ID** kann eine entfernte Datenbank in einem MobiLink-Synchronisationssystem eindeutig identifizieren.

Beim Erstellen einer SQL Anywhere- oder einer UltraLite-Datenbank wird die entfernte ID auf NULL gesetzt. Wenn die Datenbank mit MobiLink synchronisiert wird, sucht der MobiLink-Client nach einer entfernten ID NULL. Wenn MobiLink die entfernte ID NULL findet, weist sie eine GUID als entfernte ID zu. Sobald diese festgelegt ist, behält die Datenbank dieselbe entfernte ID bei, solange diese nicht manuell geändert wird (das manuelle Ändern einer entfernten ID wird nicht empfohlen).

Bei entfernten SQL Anywhere-Datenbanken verfolgt der MobiLink-Server den Synchronisationsfortschritt anhand der entfernten ID und der Subskription. Bei entfernten UltraLite-

Datenbanken verfolgt der MobiLink-Server den Synchronisationsfortschritt anhand der entfernten ID und der Publikation. Diese Information wird in der Systemtabelle `ml_subscription` gespeichert. Die entfernte ID wird auch im MobiLink-Serverlog für die einzelnen Synchronisationen aufgeführt.

Entfernte IDs müssen eindeutig sein.

Jede entfernte Datenbank muss durch ihre entfernte ID eindeutig identifiziert werden. Die integrierte entfernte GUID-ID erfüllt diese Anforderung. Für eine alternative, für Benutzer besser lesbare ID für entfernte Datenbanken in Skripten und Geschäftslogik sollten Sie die Verwendung des MobiLink-Benutzernamens und/oder eines eindeutigen Authentifizierungsparameters in Betracht ziehen. Wenn Sie Ihre eigenen entfernten IDs selbst zuordnen müssen, müssen Sie dabei sicherstellen, dass jeder entfernten Datenbank eine eindeutige entfernte ID zugeordnet ist.

Wenn dieselbe entfernte ID in zwei oder mehr gleichzeitigen Synchronisationen verwendet wird, kann sie möglicherweise zu Datenbeschädigungen und/oder Datenverlusten führen, je nach Beschaffenheit Ihrer Synchronisationsskripten und Geschäftslogik. Gleichzeitige Synchronisationen mit derselben entfernten ID können aus einem der folgenden Gründe stattfinden:

- Einer entfernten Datenbank wurde eine mehrfach vorhandene entfernte ID zugewiesen. In diesem Fall muss die mehrfach vorhandene entfernte ID auf eine eindeutige entfernte ID eingestellt werden.
- Ein Netzwerkfehler trennt die Verbindung des Clients, woraufhin dieser sich sofort erneut synchronisiert. Es ist möglich, dass die ursprüngliche und die neue Synchronisation gleichzeitig verarbeitet werden.

In beiden Fällen versucht der MobiLink-Server automatisch, eine Beschädigung der Daten oder Datenverluste zu verhindern. Um dies zu erreichen, bricht der MobiLink-Server in der Regel alle gleichzeitigen Synchronisationen bis auf eine mit einem Fehler ab.

Vorsicht

Seien Sie mit der Wiederverwendung entfernter IDs vorsichtig. Wenn Sie eine entfernte ID erneut verwenden, z.B. beim Ersetzen oder Wiederherstellen einer entfernten Datenbank, rufen Sie die gespeicherte Prozedur `ml_reset_sync_state` in der konsolidierten Datenbank für diese entfernte ID vor der ersten Synchronisation der neuen entfernten Datenbank auf. Weitere Informationen finden Sie unter [„ml_reset_sync_state-Systemprozedur“ \[MobiLink - Serveradministration\]](#).

Siehe auch

- „MobiLink-Benutzer“ auf Seite 4
- „Authentifizierungsparameter“ [\[MobiLink - Serveradministration\]](#)

Name der entfernten MobiLink-ID

Die entfernte ID wird als GUID eingestellt, aber sie können ihr einen beschreibenden Namen geben. Für SQL Anywhere- und UltraLite-Datenbanken wird die entfernte ID in der Datenbank als Eigenschaft mit dem Namen `ml_remote_id` gespeichert.

Weitere Hinweise zu SQL Anywhere-Clients finden Sie unter [„Einstellungen für die entfernte ID“ auf Seite 72](#) und [„ml_remote_id-Option“ \[SQL Anywhere Server - Datenbankadministration\]](#).

Weitere Hinweise zu UltraLite-Clients finden Sie unter „UltraLite ml_remote_id-Option“ [[UltraLite - Datenbankverwaltung](#)].

Wenn Sie die entfernte ID manuell festlegen und danach die entfernte Datenbank neu erstellen, müssen Sie entweder der neu erstellten entfernten Datenbank einen anderen Namen geben oder die Prozedur ml_reset_sync_state verwenden, um die Statusinformationen für die entfernte Datenbank in der konsolidierten Datenbank zurückzusetzen. Siehe „ml_reset_sync_state-Systemprozedur“ [[MobiLink - Serveradministration](#)].

Wenn Sie das Deployment einer Ausgangsdatenbank an mehreren Standorten durchführen, ist es am sichersten, beim Deployment Datenbanken zu verwenden, welche die entfernte ID NULL haben. Wenn Sie die Datenbanken synchronisiert haben, um sie vorab mit Daten zu füllen, können Sie die entfernte ID vor dem Deployment wieder auf NULL zurücksetzen. Auf diese Weise stellen Sie sicher, dass eine eindeutige entfernte ID verwendet wird, da bei der ersten Synchronisation der entfernten Datenbank eine eindeutige entfernte ID zugewiesen wird. Die entfernte ID kann auch beim Einrichten der entfernten Datenbank festgelegt werden, sie muss jedoch eindeutig sein.

Beispiel

Um Administrationsaufgaben bei der Definition eines MobiLink-Setups zu reduzieren, bei dem nur ein Benutzer pro entfernter Datenbank eingerichtet ist, können Sie dieselbe Nummer für alle drei MobiLink-Bezeichner für jede entfernte Datenbank verwenden. Beispiel: In einer entfernten SQL Anywhere-Datenbank können Sie sie wie folgt einstellen:

```
-- Set the MobiLink user name:
CREATE SYNCHRONIZATION USER "1" ... ;

-- Set the partition number for DEFAULT GLOBAL AUTOINCREMENT:
SET OPTION PUBLIC.GLOBAL_DATABASE_ID = '1';

-- Set the MobiLink remote ID:
SET OPTION PUBLIC.ml_remote_id = '1';
```

Entfernte IDs und MobiLink-Benutzernamen in Skripten

Der MobiLink-Benutzer identifiziert eine Person und wird für die Authentifizierung verwendet. Die entfernte MobiLink-Datenbank wird durch die entfernte ID eindeutig gekennzeichnet.

In vielen Synchronisationsskripten haben Sie die Möglichkeit, die entfernte Datenbank über die entfernte ID (mithilfe des benannten Parameters s.remote_id) oder über den MobiLink-Benutzernamen (mit s.username) zu identifizieren. Die Verwendung der entfernten ID bietet vor allem in UltraLite einige Vorteile.

Wenn Deployments eine Eins-zu-Eins-Relation zwischen einer entfernten Datenbank und einem MobiLink-Benutzer aufweisen, können Sie die entfernte ID ignorieren. In diesem Fall können MobiLink-Ereignisskripten den Parameter username referenzieren, welcher mit dem für die Authentifizierung verwendeten MobiLink-Benutzernamen übereinstimmt.

Wenn ein MobiLink-Benutzer Daten in unterschiedlichen Datenbanken synchronisieren möchte, aber jede entfernte Datenbank dieselben Daten enthält, können die Synchronisationsskripten den MobiLink-Benutzernamen referenzieren. Wenn der MobiLink-Benutzer jedoch unterschiedliche Datensätze in

unterschiedlichen Datenbanken synchronisieren möchte, sollten die Synchronisationsskripten die entfernte ID referenzieren.

In UltraLite-Datenbanken kann die gleiche Datenbank durch unterschiedliche Benutzer synchronisiert werden, sogar wenn der vorherige Upload-Status unbekannt ist, da der MobiLink-Server den Synchronisationsfortschritt anhand der entfernten ID verfolgt. In diesem Fall können Sie bei zeitstempelbasierten Downloads den MobiLink-Benutzernamen in Download-Skripten nicht mehr referenzieren, da für jeden der anderen Benutzer einige Zeilen fehlen könnten, die niemals abgerufen wurden. Um dies zu verhindern, müssen Sie eine Zuordnungstabelle in der konsolidierten Datenbank anlegen, die für jeden Benutzer, der dieselbe entfernte Datenbank verwendet, eine Zeile enthält. Um sicherzustellen, dass alle Daten für alle Benutzer heruntergeladen werden, können Sie die konsolidierte Tabelle und die Zuordnungstabelle, die auf der entfernten ID der aktuellen Synchronisation basiert, zusammenführen.

Sie können auch verschiedene Skriptversionen für verschiedene entfernte Datenbanken verwenden.

Siehe auch

- „Skriptversionen“ [[MobiLink - Serveradministration](#)]

Benutzerauthentifizierungsverfahren

Die Benutzerauthentifizierung ist ein Bestandteil eines Sicherheitssystems zum Schutz Ihrer Daten.

MobiLink stellt Ihnen verschiedene Benutzerauthentifizierungsverfahren zur Verfügung. Sie müssen nicht ein einziges Verfahren verwenden, das Ihre gesamte Installation schützt. Bei MobiLink können Sie innerhalb einer Installation für verschiedene Skriptversionen unterschiedliche Authentifizierungsverfahren flexibel einsetzen.

- **Keine MobiLink-Benutzerauthentifizierung** Wenn Sie für Ihre Daten keinen Kennwortschutz benötigen, können Sie sich dafür entscheiden, in Ihrer Installation keine Benutzerauthentifizierung zu verwenden. In diesem Fall muss der MobiLink-Benutzername dennoch in der ml_user-Tabelle enthalten sein, aber die hashed_password-Spalte ist NULL.
- **Integrierte MobiLink-Benutzerauthentifizierung** MobiLink benutzt die in der Systemtabelle ml_user gespeicherten Benutzernamen und Kennwörter für die Authentifizierung.

Das integrierte Verfahren ist in den folgenden Abschnitten beschrieben.

- **Angepasste Authentifizierung** Mit dem MobiLink-Skript authenticate_user können Sie das integrierte MobiLink-Benutzerauthentifizierungssystem durch ein eigenes ersetzen. Abhängig davon, welches Datenbankmanagementsystem Sie als konsolidierte Datenbank verwenden, können Sie beispielsweise anstelle der Benutzerauthentifizierung des MobiLink-Systems die Authentifizierung des jeweiligen Datenbanksystems verwenden.

Siehe „[Angepasste Benutzerauthentifizierung](#)“ auf Seite 15.

Hinweise zu weiteren sicherheitsbezogenen Funktionen von MobiLink und den dazugehörigen Produkten finden Sie an folgenden Stellen:

- „Verschlüsselung der MobiLink-Client/Server-Kommunikation“ [[SQL Anywhere Server - Datenbankadministration](#)]
- UltraLite-Clients: „Datenbanksicherheit“ [[UltraLite - Datenbankverwaltung](#)]
- SQL Anywhere-Clients: „Datensicherheit“ [[SQL Anywhere Server - Datenbankadministration](#)]

Architektur der Benutzerauthentifizierung

Das MobiLink-System zur Benutzerauthentifizierung verwendet Benutzernamen und Kennwörter. Sie können sich dafür entscheiden, dass der MobiLink-Server Benutzernamen und Kennwort mit einem integrierten Verfahren validiert, oder Ihr eigenes angepasstes Benutzerauthentifizierungsverfahren implementieren.

Im integrierten Authentifizierungssystem werden sowohl der Benutzername als auch das Kennwort in der MobiLink-Systemtabelle ml_user in der konsolidierten Datenbank gespeichert. Das Kennwort wird in verschlüsselter Form gespeichert, sodass nur der MobiLink-Server die Tabelle ml_user lesen und die ursprüngliche Form des Kennworts rekonstruieren kann. Benutzernamen und Kennwörter können zur konsolidierten Datenbank mit Sybase Central, dem Dienstprogramm mluser oder der Angabe von -zu+ beim Start des MobiLink-Servers hinzugefügt werden.

Siehe „Erstellen und Registrieren von MobiLink-Benutzern“ auf Seite 5.

Wenn ein MobiLink-Client eine Verbindung mit einem MobiLink-Server herstellt, gibt er die folgenden Werte an:

- **Benutzername** Der MobiLink-Benutzername. Obligatorisch. Für die Synchronisation muss der Benutzername in der Systemtabelle ml_user gespeichert sein. Alternativ können Sie den MobiLink-Server mit der Option -zu+ starten, sodass neue Benutzer zur Tabelle ml_user hinzugefügt werden.
- **password** Das MobiLink-Kennwort. Nur dann optional, wenn der Benutzer unbekannt ist oder wenn das entsprechende Kennwort in der MobiLink-Systemtabelle ml_user den Wert NULL hat.
- **Neues Kennwort** Ein neues MobiLink-Kennwort. Optional. MobiLink-Benutzer können ihr Kennwort ändern, indem sie diesen Wert festlegen.

Angepasste Authentifizierung

Sie haben auch die Möglichkeit, Ihr eigenes Benutzerauthentifizierungsverfahren einzusetzen.

Siehe „Angepasste Benutzerauthentifizierung“ auf Seite 15.

Authentifizierungsprozess

Im Folgenden finden Sie eine Beschreibung der Ereignisabfolge während der Authentifizierung.

1. Eine entfernte Anwendung startet eine Synchronisationsanforderung mit einer entfernten ID, einem MobiLink-Benutzernamen und wahlweise mit einem Kennwort und einem neuen Kennwort. Der

MobiLink-Server beginnt eine neue Transaktion und löst das Ereignis `begin_connection_autocommit` und `begin_connection` aus.

2. MobiLink stellt sicher, dass die entfernte ID derzeit nicht bereits eine Synchronisation gestartet hat und setzt `authentication_status` auf 4000.
3. Wenn Sie ein `authenticate_user`-Skript definiert haben, tritt Folgendes ein:
 - a. Wenn das `authenticate_user`-Skript in SQL geschrieben wurde, wird das Skript mit dem voreingestellten `authentication_status`-Parameter 4000, dem angegebenen MobiLink-Benutzernamen und optional mit dem Kennwort und dem neuen Kennwort aufgerufen.

Wenn das `authenticate_user`-Skript in Java oder .NET geschrieben wurde und eine SQL-Anweisung zurückgibt, wird diese SQL-Anweisung mit dem voreingestellten `authentication_status`-Parameter 4000, dem angegebenen MobiLink-Benutzernamen und optional mit dem Kennwort und dem neuen Kennwort aufgerufen.

- b. Falls das `authenticate_user`-Skript einen Ausnahmefehler auslöst oder bei der Ausführung des Skripts ein Fehler auftritt, stoppt der Synchronisationsprozess.

Das `authenticate_user`-Skript oder die zurückgegebene SQL-Anweisung müssen ein Aufruf einer gespeicherten Prozedur sein, die 2 bis 4 Argumente hat. Der voreingestellte Wert für `authentication_status` wird als erster Parameter übergeben und kann von der gespeicherten Prozedur aktualisiert werden. Der Rückgabewert des ersten Parameters ist der Wert von `authentication_status` aus dem Skript `authenticate_user`.

4. Wenn ein `authenticate_user_hashed`-Skript vorhanden ist, tritt Folgendes ein:
 - a. Falls ein Kennwort angegeben wurde, wird ein Hash-Wert dafür berechnet. Bei der Angabe eines neuen Kennworts, wird ein Hash-Wert dafür berechnet.
 - b. Das Skript `authenticate_user_hashed` wird mit dem aktuellen Wert von `authentication_status` (voreingestellter Wert von `authentication_status`, wenn das Skript `authenticate_user` nicht existiert, oder der Wert von `authentication_status`, den das Skript `authenticate_user` zurückgegeben hat) und den Hash-Kennwörtern aufgerufen. Das Verhalten entspricht Schritt 3. Der Rückgabewert des ersten Parameters wird als Wert von `authentication_status` des Skripts `authenticate_user_hashed` verwendet.
5. Der MobiLink-Server nimmt den größeren Wert des `auth_user`-Status aus den Skripten `authenticate_user` und `authenticate_user_hashed`, sofern vorhanden, oder den voreingestellten Status von Wert von `authentication_status`, wenn keines der Skripten existiert.
6. Der MobiLink-Server fragt den von Ihnen angegebenen MobiLink-Benutzernamen in der Tabelle `ml_user` ab.
 - a. Wenn eines der angepassten Skripten `authenticate_user` oder `authenticate_user_hashed` aufgerufen wurde, aber der von Ihnen angegebene MobiLink-Benutzername nicht in der Tabelle `ml_user` vorhanden und `authentication_status` gültig ist (1000 oder 2000), wird der MobiLink-Benutzername zur MobiLink-Systemtabelle `ml_user` hinzugefügt. Falls `authentication_status` nicht gültig ist, wird `ml_user` nicht aktualisiert und es kommt zu einem Fehler.

- b. Wenn die angepassten Skripten nicht aufgerufen wurden und der von Ihnen bereitgestellte MobiLink-Benutzername nicht in der Tabelle ml_user enthalten ist, wird der MobiLink-Benutzername zu ml_user hinzugefügt, sofern Sie den MobiLink-Server mit der Option -zu+ gestartet haben. Andernfalls tritt ein Fehler ein und authentication_status wird auf ungültig gesetzt.
 - c. Wenn die angepassten Skripten aufgerufen wurden und der von Ihnen bereitgestellte MobiLink-Benutzername in der Tabelle ml_user enthalten ist, geschieht nichts.
 - d. Wenn die angepassten Skripten **nicht** aufgerufen wurden und der von Ihnen angegebene MobiLink-Benutzername in der Tabelle ml_user enthalten ist, werden die Kennwörter mit dem Wert in der Tabelle ml_user verglichen. Wenn das Kennwort dem in der Tabelle ml_user für den MobiLink-Benutzer entspricht, wird authentication_status auf gültig gesetzt. Andernfalls wird authentication_status auf ungültig gesetzt.
- 7. Falls authentication_status gültig ist und weder das Skript authenticate_user noch authenticate_user_hashed aufgerufen wurde und Sie ein neues Kennwort in der Tabelle ml_user für diesen MobiLink-Benutzer bereitgestellt haben, wird das Kennwort auf das von Ihnen bereitgestellte Kennwort gesetzt.
 - 8. Wenn Sie ein authenticate_parameters-Skript definiert haben und authentication_status gültig ist (1000 oder 2000), tritt Folgendes ein:
 - a. Die Parameter werden an das Skript authentication_parameters übergeben.
 - b. Falls das Skript authenticate_parameters einen authentication_status-Wert zurückgibt, der größer als der aktuelle Wert von authentication_status ist, überschreibt der neue Wert von authentication_status den alten Wert.
 - 9. Falls authentication_status nicht zulässig ist, wird die Synchronisation abgebrochen.
 - 10. Falls Sie das modify_user-Skript definiert haben, wird es aufgerufen, um den von Ihnen angegebenen MobiLink-Benutzernamen durch einen neuen, von diesem Skript zurückgegebenen MobiLink-Benutzernamen zu ersetzen.
 - 11. Der MobiLink-Server führt die Transaktion immer nach der Benutzerauthentifizierung in MobiLink durch, unabhängig vom authentication_status. Wenn authentication_status einen gültigen Wert hat (1000 oder 2000), wird die Synchronisation fortgesetzt. Falls authentication_status nicht zulässig ist, wird die Synchronisation abgebrochen.

Angepasste Benutzerauthentifizierung

Sie können auch ein anderes als das in MobiLink integrierte Verfahren zur Benutzerauthentifizierung verwenden. Es gibt u.A. folgende Gründe, ein angepasstes Benutzerauthentifizierungsverfahren zu verwenden:

- Zur Integration mit vorhandenen Schemata zur Datenbank-Benutzerauthentifizierung oder externen Authentifizierungsverfahren

- Zur Bereitstellung von angepassten Funktionen, wie etwa einer Kennwort-Mindestlänge oder einem Kennwort-Ablaufdatum, die im integrierten MobiLink-Verfahren nicht enthalten sind

Es gibt drei Tools für die angepasste Authentifizierung:

- `mlsrv16 -zu+`-Option
- `authenticate_user`- oder `authenticate_user_hashed`-Skript
- `authenticate_parameters`-Skript

Mit der `mlsrv16 -zu+`-Option können Sie das automatische Hinzufügen von Benutzern steuern. Geben Sie z.B. `-zu+` an, wenn alle nicht erkannten MobiLink-Benutzernamen zur Tabelle `ml_user` hinzugefügt werden sollen, wenn sie erstmals synchronisieren. Die Option `-zu+` wird nur bei der integrierten MobiLink-Authentifizierung benötigt.

Die Skripten `authenticate_user`, `authenticate_user_hashed` und `authenticate_parameters` haben Vorrang vor dem Standardverfahren zur MobiLink-Benutzerauthentifizierung. Jeder Benutzer, dessen Authentifizierung erfolgreich ist, wird automatisch der `ml_user`-Tabelle hinzugefügt.

Sie können mit dem `authenticate_user`-Skript eine angepasste Authentifizierung von Benutzer-IDs und Kennwörtern erstellen. Wenn dieses Skript vorhanden ist, wird es anstelle des integrierten Kennwortvergleichs ausgeführt. Das Skript muss für den Erfolg bzw. den Fehlschlag der Authentifizierung jeweils einen Fehlercode zurückgeben.

Es gibt mehrere vordefinierte Skripten für das Ereignis `authenticate_user`, das mit MobiLink installiert wird. Sie erleichtern die Authentifizierung bei der Verwendung von LDAP-, POP3- und IMAP-Servern. Siehe „[Authentifizierung bei externen Servern](#)“ auf Seite 17.

Mit `authenticate_parameters` erstellen Sie eine angepasste Authentifizierung, die von anderen Werten als Benutzer-IDs und Kennwörtern abhängt.

Siehe auch

- „`mlsrv16`-Option `-zu`“ [[MobiLink - Serveradministration](#)]
- „`authenticate_user` (Verbindungsereignis)“ [[MobiLink - Serveradministration](#)]
- „`authenticate_user_hashed` (Verbindungsereignis)“ [[MobiLink - Serveradministration](#)]
- „`authenticate_parameters` (Verbindungsereignis)“ [[MobiLink - Serveradministration](#)]

Benutzerauthentifizierung mit Java und .NET

Die Benutzerauthentifizierung ist eine natürliche Anwendung der Java- und .NET-Synchronisationslogik, da Java- und .NET-Klassen die Möglichkeit bieten, auf andere Ressourcen mit Benutzernamen und Kennwörtern zuzugreifen, die im Umfeld Ihres Computersystems verwendet werden, wie etwa auf Anwendungsservern.

Ein einfaches Beispiel befindet sich im Verzeichnis `%SQLANYSAMPI6%\MobiLink\JavaAuthentication`. Der Beispielcode in `%SQLANYSAMPI6%\MobiLink\JavaAuthentication\CustEmpScripts.java` implementiert ein einfaches System zur Benutzerauthentifizierung. Bei der ersten Synchronisation wird in die Tabelle `login_added` ein MobiLink-Benutzer eingefügt. Bei nachfolgenden Synchronisationen wird

der Tabelle login_audit jeweils eine Zeile hinzugefügt. In diesem Beispiel wird vor dem Einfügen einer Benutzer-ID in die Tabelle login_added kein Test durchgeführt.

Ein .NET-Beispiel, das die Benutzerauthentifizierung beschreibt, finden Sie unter „[NET-Synchronisationsbeispiel](#)“ [[MobiLink - Serveradministration](#)].

Authentifizierung bei externen Servern

MobiLink verfügt über vordefinierte Java-Synchronisationsskripten, die Ihnen die Authentifizierung gegenüber externen Servern mithilfe des Ereignisses authenticate_user vereinfachen. Vordefinierte Skripten sind für folgende Authentifizierungsserver verfügbar:

- POP3- oder IMAP-Server mit der JavaMail 1.2 API-Schnittstelle
- LDAP-Server mit der JNDI-Schnittstelle (Java Naming and Directory Interface)

Wie Sie diese Skripten einsetzen, wird dadurch bestimmt, ob Ihre MobiLink-Benutzernamen direkt den Benutzer-IDs in Ihrem externen Authentifizierungssystem zugeordnet sind.

Hinweis

Sie können die Authentifizierung auch in externen Servern in Sybase Central über die Registerkarte "Authentifizierung" durchführen. Siehe „[MobiLink-Plug-In für Sybase Central](#)“ [[MobiLink - Erste Orientierung](#)].

Wenn Ihre MobiLink-Benutzernamen direkt Ihren Benutzer-IDs zugeordnet sind

Wenn der MobiLink-Benutzername einfach direkt einer gültigen Benutzer-ID in Ihrem Authentifizierungssystem zugeordnet ist, können die vordefinierten Skripten direkt als Reaktion auf das authenticate_user-Verbindungsereignis aufgerufen werden. Der Authentifizierungscode initialisiert sich selbst basierend auf den Eigenschaften, die in der ml_property-Tabelle gespeichert sind.

Fügen Sie das vordefinierte Java-Synchronisationsskript zur MobiLink-Systemtabelle ml_scripts hinzu (entweder mit einer gespeicherten Prozedur oder in Sybase Central)

- Bei der Verwendung der gespeicherten Prozedur ml_add_java_connection_script führen Sie folgenden Befehl aus:

```
call ml_add_java_connection_script(  
    'MyVersion',  
    'authenticate_user',  
    'ianywhere.ml.authentication.ServerType.authenticate' )
```

Dabei gilt: *MyVersion* ist der Name einer Skriptversion und *ServerType* ist **LDAP**, **POP3** oder **IMAP**.

- Bei der Verwendung des **Assistenten zum Hinzufügen von Verbindungsskripten** in Sybase Central wählen Sie den Skripttyp **authenticate_user** und geben Folgendes im Code-Editor ein:

```
ianywhere.ml.authentication.ServerType.authenticate
```

Dabei gilt: *ServerType* ist **LDAP**, **POP3** oder **IMAP**.

Siehe „[ml_add_java_connection_script-Systemprozedur](#)“ [[MobiLink - Serveradministration](#)].

Fügen Sie Eigenschaften für diesen Authentifizierungsserver hinzu Verwenden Sie für die Festlegung der verschiedenen Eigenschaften die gespeicherte Prozedur `ml_add_property`:

```
call ml_add_property(  
  'ScriptVersion',  
  'MyVersion',  
  'property_name',  
  'property_value' )
```

Dabei gilt: *MyVersion* ist der Name einer Skriptversion, *property_name* wird von Ihrem Authentifizierungsserver festgelegt und *property_value* ist ein Wert aus Ihrer Anwendung. Wiederholen Sie diesen Aufruf für jede festzulegende Eigenschaft.

Siehe „Eigenschaften für externe Authentifizierer“ auf Seite 19 und „`ml_add_property`-Systemprozedur“ [*MobiLink - Serveradministration*].

Wenn Ihre MobiLink-Benutzernamen nicht direkt Ihren Benutzer-IDs zugeordnet sind

Wenn Ihre MobiLink-Benutzernamen nicht den Benutzer-IDs entsprechen, muss der Code indirekt aufgerufen werden und Sie müssen die Benutzer-ID aus dem `ml_user`-Wert extrahieren bzw. eine Zuordnung vornehmen. Dazu schreiben Sie eine entsprechende Java-Klasse.

Siehe „Schreiben eines Synchronisationsskripts in Java“ [*MobiLink - Serveradministration*].

Nachfolgend finden Sie ein einfaches Beispiel hierzu. In diesem Beispiel wurde der Code in der `extractUserID`-Methode weggelassen, da er davon abhängt, wie die Zuordnung des `ml_user`-Werts zu einer `userid` erfolgt. Die Verarbeitung erfolgt komplett in der `authenticate`-Methode der `authentication`-Klasse.

```
package com.mycompany.mycode;  
  
import ianywhere.ml.authentication.*;  
import ianywhere.ml.script.*;  
  
public class MLEvents  
{  
    private DBConnectionContext _context;  
    private POP3 _pop3;  
  
    public MLEvents( DBConnectionContext context )  
    {  
        _context = context;  
        _pop3 = new POP3( context );  
    }  
  
    public void authenticateUser(  
        InOutInteger status,  
        String userID,  
        String password,  
        String newPassword )  
    {  
        String realUserID = extractUserID( userID );  
        _pop3.authenticate( status, realUserID, password, newPassword );  
    }  
  
    private String extractUserID( String userID )  
    {  
        // code here to map ml_user to a "real" POP3 user
```

```
}
}
```

In diesem Beispiel muss das POP3-Objekt mit dem DBConnectContext-Objekt initialisiert werden, damit es seine Initialisierungseigenschaften finden kann. Wenn Sie die Initialisierung nicht auf diese Weise vornehmen, müssen Sie die Eigenschaften im Code definieren. Beispiel:

```
POP3 pop3 = new POP3();
pop3.setServerName( "smtp.sybase.com" );
pop3.setServerPort( 25 );
```

Dies gilt für alle authentication-Klassen, obwohl die Eigenschaften nach Klasse unterschiedlich sind.

Eigenschaften für externe Authentifizierer

MobiLink stellt, vor allem für LDAP-Fälle, Standardwerte bereit. Die definierbaren Eigenschaften können unterschiedlich sein. Nachstehend finden Sie eine Liste der wichtigsten Eigenschaften.

POP3-Authenticator

mail.pop3.host	Der Hostname des Servers
mail.pop3.port	Die Portnummer (kann weggelassen werden, wenn Standardwert 110 benutzt wird)

IMAP-Authenticator

mail.imap.host	Der Hostname des Servers
mail.imap.port	Die Portnummer (kann weggelassen werden, wenn Standardwert 143 benutzt wird)

LDAP-Authenticator

java.naming.provider.url	URL des LDAP-Servers, z.B. ldap://ops-yourLocation/dn=sybase,dn=com
--------------------------	---

Weitere Hinweise finden Sie in der JNDI-Dokumentation.

Dienstprogramme für MobiLink-Clients

Die folgenden Dienstprogramme für MobiLink-Clients sind verfügbar

- „Dienstprogramm zur Installation des Microsoft ActiveSync-Providers (mlasinst)“
- „MobiLink-Dienstprogramm für die Dateiübertragung (mlfiletransfer)“

Siehe auch

- „UltraLite-Dienstprogramme“ [[UltraLite - Datenbankverwaltung](#)]
- „MobiLink-Dienstprogramme“ [[MobiLink - Serveradministration](#)]
- „Dienstprogramme für die Datenbankadministration“ [[SQL Anywhere Server - Datenbankadministration](#)]

Dienstprogramm zur Installation des Microsoft ActiveSync-Providers (mlasinst)

Installiert einen MobiLink-Provider für Microsoft ActiveSync (bekannt als Windows Mobile-Gerätecenter unter Windows Vista oder später) oder registriert und installiert UltraLite-Anwendungen auf Windows Mobile-Geräten.

Syntax

mlasinst [*options*] [[*src*] *dst name class* [*args*]]

Optionen	Beschreibung
-d	Deaktiviert die Anwendung zunächst.
-k path	Gibt den Standort des Desktop-Providers <i>mlasdesk.dll</i> an. Die Datei befindet sich im Verzeichnis <i>%SQLANY16%\bin32</i> . Endbenutzer (die normalerweise nicht über eine komplette SQL Anywhere-Installation verfügen) müssen -k angeben, wenn sie den Microsoft ActiveSync-Provider von MobiLink installieren.
-l filename	Gibt den Namen der Aktivitätslogdatei an.
-n	Registriert die Anwendung, ohne sie auf das Gerät zu kopieren. Diese Option registriert zusätzlich zur Installation des Microsoft ActiveSync-Provider von MobiLink eine Anwendung, ohne sie auf das Gerät zu kopieren. Dies ist das geeignete Verfahren, sofern die Anwendung mehr als eine Datei umfasst (z.B. wenn sie für die Verwendung der UltraLite-Laufzeitbibliothek-DLL kompiliert wurde und nicht für eine statische Bibliothek), oder wenn Sie eine alternative Methode zum Kopieren der Anwendung auf das Gerät verwenden.
-t n	Gibt an, wie lange, in Sekunden, der Desktop-Provider auf eine Antwort vom Client warten soll, bevor der Zeitablauf aktiviert wird. Der Standardwert beträgt 30 Sekunden.

Optionen	Beschreibung
-u	MobiLink-Provider für Microsoft ActiveSync deinstallieren. Diese Option trägt alle Anwendungen aus der Registrierung aus, die für den Einsatz mit dem Microsoft ActiveSync-Provider von MobiLink registriert wurden, und deinstalliert den Microsoft ActiveSync-Provider von MobiLink. Bei diesem Vorgang werden keine Dateien vom PC bzw. Gerät gelöscht. Wenn das Gerät nicht an den PC angeschlossen ist, wird ein Fehler gemeldet.
-v path	Gibt den Standort des Gerät-Providers <i>mlasdev.dll</i> an. Standardmäßig wird nach der Datei in einem plattformspezifischem Verzeichnis unter <i>%SQLANY16%\CE</i> gesucht. Endbenutzer (die normalerweise nicht über eine komplette SQL Anywhere-Installation verfügen) müssen -v angeben, wenn sie den Microsoft ActiveSync-Provider von MobiLink installieren.

Andere Parameter	Beschreibung
<i>src</i>	Gibt Namen und Suchpfad der Quelldatei zum Kopieren einer Anwendung auf das Gerät an. Geben Sie diesen Parameter nur an, wenn Sie eine Anwendung registrieren und auf das Gerät kopieren. Geben Sie den Parameter nicht an, wenn Sie die Option -n verwenden.
<i>dst</i>	Gibt Namen und Suchpfad der Zieldatei für eine Anwendung auf dem Gerät an.
<i>name</i>	Gibt den Namen an, mit dem sich Microsoft ActiveSync auf die Anwendung bezieht.
<i>class</i>	Gibt den registrierten Windows-Klassennamen der Anwendung an.
<i>args</i>	Gibt die Befehlszeilenargumente an, die an die Anwendung übergeben werden, wenn Microsoft ActiveSync die Anwendung startet.

Bemerkungen

Mit diesem Dienstprogramm wird ein MobiLink-Provider für Microsoft ActiveSync installiert. Der Provider enthält eine Komponente, die auf dem PC läuft (*mlasdesk.dll*), und eine Komponente, die auf dem Windows Mobile-Gerät bereitgestellt wird (*mlasdev.dll*). Das Dienstprogramm *mlasinst* erstellt einen Registrierungseintrag, der auf die aktuelle Speicherposition des Desktop-Providers zeigt, und kopiert den Geräte-Provider auf das Gerät.

Falls zusätzliche Argumente angegeben werden, kann das Dienstprogramm *mlasinst* auch benutzt werden, um UltraLite-Anwendungen auf einem Windows Mobile-Gerät zu registrieren und zu installieren. Sie haben außerdem die Möglichkeit, UltraLite-Anwendungen mit der Microsoft ActiveSync-Software zu installieren.

Abhängig von den jeweiligen Lizenzbedingungen können Sie diese Anwendung zusammen mit den Desktop- und den Gerätekomponenten an Endbenutzer weitergeben, damit diese Kopien Ihrer Anwendung für die Benutzung mit Microsoft ActiveSync anfertigen können.

Eine Verbindung mit einem entfernten Gerät ist erforderlich, um den Microsoft ActiveSync-Provider zu installieren.

Ausführliche Anweisungen zur Verwendung des Installationsprogramms für den Microsoft ActiveSync-Provider finden Sie unter

- SQL Anywhere: „[MobiLink-Provider für Microsoft ActiveSync installieren](#)“
- UltraLite: „[ActiveSync mit UltraLite unter Windows Mobile](#)“ [[UltraLite - Datenbankverwaltung](#)]

Beispiele

Mit dem folgenden Befehl wird der MobiLink-Provider für Microsoft ActiveSync mit den Standardargumenten installiert. Es wird keine Anwendung registriert. Es muss eine Verbindung vom PC zum Gerät bestehen, damit die Installation erfolgreich durchgeführt werden kann.

```
mlasinst
```

Mit dem folgenden Befehl wird der MobiLink-Provider für Microsoft ActiveSync deinstalliert. Es muss eine Verbindung vom PC zum Gerät bestehen, damit die Deinstallation erfolgreich durchgeführt werden kann.

```
mlasinst -u
```

Mit dem folgenden Befehl wird der MobiLink-Provider für Microsoft ActiveSync installiert, sofern dies noch nicht der Fall ist, und die Anwendung *myapp.exe* registriert. Außerdem kopiert er die Datei *c:\My Files\myapp.exe* auf dem Gerät in *\Programme\myapp.exe*. Die Argumente *-p* und *-x* sind Befehlszeilenoptionen für *myapp.exe*, wenn die Anwendung von Microsoft ActiveSync gestartet wird. Dieser Befehl muss in einer einzigen Zeile eingegeben werden.

```
mlasinst "C:\My Files\myapp.exe" "\Program Files\myapp.exe"  
"My Application" MYAPP -p -x
```

Siehe auch

- „[Synchronisation mit Microsoft ActiveSync](#)“ auf Seite 95
- „[UltraLite-Synchronisationsparameter](#)“ [[UltraLite - Datenbankverwaltung](#)]
- „[UltraLite-Netzwerkprotokolloptionen für dbmsync](#)“ [[UltraLite - Datenbankverwaltung](#)]

MobiLink-Dienstprogramm für die Dateiübertragung (mlfiletransfer)

Lädt eine Datei über MobiLink hoch oder herunter.

Syntax

```
mlfiletransfer [ options ] file
```

Option	Beschreibung
-ap <i>param1</i> , ...	MobiLink-Authentifizierungsparameter. Siehe „Authentifizierungsparameter“ [MobiLink - Serveradministration].
-g	Zeigt den Fortschritt der Übertragung an.
-i	Ignoriert partielle Übertragung von einem früheren Versuch.
-k	Entfernter Schlüssel zur Kennzeichnung von entfernten Objekten. Dies ist optional.
-lf <i>filename</i>	Der lokale Name der zu übertragenden Datei. Standardmäßig wird der Name, wie er vom Server erkannt wird (z.B. <i>file</i>) verwendet.
-lp <i>path</i>	Der lokale Pfad für die zu übertragende Datei. Standardmäßig ist der lokale Pfad das Stammverzeichnis unter Windows Mobile und auf anderen Plattformen das aktuelle Verzeichnis.
-p <i>password</i>	Das Kennwort für den MobiLink-Benutzernamen
-q	Dialogfreier Modus. Es werden keine Meldungen angezeigt.
-s	Upload einer Datei auf MobiLink. Die Standardeinstellung ist Download.
-u <i>username</i>	MobiLink-Benutzername. Diese Option ist erforderlich.
-v <i>version</i>	Die Skriptversion Diese Option ist erforderlich.
-x <i>protocol (options)</i>	<p>Das <i>protocol</i> kann vom Typ tcpip, tls, http oder https sein. Diese Option ist erforderlich.</p> <p>Die möglichen Protokolloptionen hängen vom Protokoll ab. Eine Liste der Optionen für jedes Protokoll finden Sie unter „Netzwerkprotokolloptionen des MobiLink-Clients“ auf Seite 25.</p>

Option	Beschreibung
<i>file</i>	<p>Die Datei, wie auf dem Server benannt, die übertragen werden soll. Beim Download sucht MobiLink die Datei im Unterverzeichnis <i>username</i> des Verzeichnisses, das mit der Option <code>-ftr</code> angegeben wurde. Kann sie dort nicht gefunden werden, wird im Verzeichnis, das mit der Option <code>-ftr</code> angegeben wurde, gesucht. Befindet sich die Datei an keinem der beiden Speicherorte, erzeugt MobiLink einen Fehler. Siehe „mlsrv16-Option -ftr“ [<i>MobiLink - Serveradministration</i>].</p> <p>Dateinamen dürfen weder Pfadangaben enthalten noch Auslassungszeichen (drei Punkte), Kommata, Schrägstriche (/) oder Backslashes (\).</p> <p>Beim Upload sucht MobiLink im mit der <code>mlsrv16-Option -ftru</code> angegebenen Verzeichnis nach den Dateien. Siehe „mlsrv16-Option -ftru“ [<i>MobiLink - Serveradministration</i>].</p>

Bemerkungen

Das Dienstprogramm ist hilfreich zum Herunterladen von Dateien, wenn Sie eine entfernte Datenbank zum ersten Mal erstellen, wenn Sie die Software auf dem entfernten Computer aktualisieren müssen und so weiter.

Um dieses Dienstprogramm für den Download von Dateien zu verwenden, müssen Sie den MobiLink-Server mit der Option `-ftr` starten. Die Option `-ftr` erzeugt ein Stammverzeichnis für die zu übertragende Datei und erstellt für jeden registrierten MobiLink-Benutzer ein Unterverzeichnis.

Um dieses Dienstprogramm für den Upload von Dateien zu verwenden, müssen Sie den MobiLink-Server mit der Option `-ftru` starten. Die `-ftru`-Option erstellt einen Speicherort für Dateien, für die ein Upload erfolgen soll.

Sie können `mlagent` als Alternative zu `mlfiletransfer` verwenden. Siehe „[mlagent-Befehl](#)“ [*MobiLink - Serveradministration*].

UltraLite-Benutzer können auch die Methoden `MLFileDownload` und `MLFileUpload` in der UltraLite-Laufzeitbibliothek verwenden. Siehe „[MobiLink-Dateiübertragungen](#)“ [*UltraLite - Datenbankverwaltung*].

Siehe auch

- „[mlsrv16-Option -ftr](#)“ [*MobiLink - Serveradministration*]
- „[authenticate_file_transfer](#) (Verbindungsereignis)“ [*MobiLink - Serveradministration*]

Beispiel

Mit dem folgenden Befehl wird der MobiLink-Server mit der Beispieldatenbank `CustDB` verbunden. Die `-ftr %SystemRoot%\system32`-Option weist den MobiLink-Server an, die Überwachung des Verzeichnisses `Windows\system32` für die angeforderten Dateien zu starten. In diesem Beispiel sucht der

MobiLink-Server zunächst im Verzeichnis `C:\Windows\System32\mobilink-username` nach der Datei. Wenn die Datei nicht existiert, wird im Verzeichnis `C:\Windows\system32` gesucht. Im Allgemeinen ist nicht erwünscht, dass der MobiLink-Server das Verzeichnis `C:\Windows\system32` nach Dateien durchsucht. In diesem Beispiel wird das Verzeichnis `Windows\system32` verwendet, sodass der MobiLink-Server das Editor-Dienstprogramm übertragen kann, das sich dort befindet.

```
mksrv16 -c "DSN=SQL Anywhere 16 CustDB" -zu+ -ftr %SystemRoot%\system32
```

Der folgende Befehl führt das Dienstprogramm `mlfiletransfer` aus. Er bewirkt, dass der MobiLink-Server `notepad.exe` in Ihr lokales Verzeichnis herunterlädt.

```
MLFileTransfer -u 1 -v "custdb 16.0" -x tcpip notepad.exe
```

Netzwerkprotokolloptionen des MobiLink-Clients

Dieser Abschnitt beschreibt die Netzwerkprotokolloptionen, die zur Verfügung stehen, wenn Sie eine Verbindung zwischen einem MobiLink-Client und einem MobiLink-Server herstellen. Mehrere MobiLink-Client-Dienstprogramme verwenden die MobiLink-Client-Netzwerkprotokolloptionen:

Verwenden von Client-Netzwerkprotokolloptionen mit...	Siehe...
dbmlsync	„Erweiterte Option CommunicationAddress (adr)“
UltraLite	„Synchronisationsparameter Stream Parameters“ [<i>UltraLite - Datenbankverwaltung</i>] oder Option -x in „UltraLite-Synchronisationsdienstprogramm (ulsync)“ [<i>UltraLite - Datenbankverwaltung</i>]
UltraLiteJ	<ul style="list-style-type: none"> „Netzwerkprotokolloptionen für UltraLiteJ-Synchronisationsdatenströme“ [<i>UltraLite® – Java-Programmierung</i>] StreamHTTTParms-Schnittstelle [UltraLiteJ] [<i>UltraLite® – Java-Programmierung</i>] StreamHTTSParms-Schnittstelle [UltraLiteJ] [<i>UltraLite® – Java-Programmierung</i>]
Relay Server	„Relay Server-Konfigurationsdatei“ [<i>Relay Server</i>]
MobiLink-Profiler	„MobiLink-Profiler starten (Administrationtools)“ [<i>MobiLink - Serveradministration</i>]
MobiLink-Dateiübertragung	„MobiLink-Dienstprogramm für die Dateiübertragung (mlfile-transfer)“
MobiLink Listener	-x in „MobiLink Listener-Dienstprogramm für Windows-Geräte (dblsn)“ [<i>MobiLink - Serverinitiierte Synchronisation</i>]

Die Art des gewählten Netzwerkprotokolls muss mit dem vom Client benutzten Synchronisationsprotokoll übereinstimmen. Hinweise zum Einstellen der Verbindungsoptionen für den MobiLink-Server finden Sie unter „[mlsrv16-Option -x](#)“ [*MobiLink - Serveradministration*].

Protokolloptionen

- **TCP/IP protocol options** Wenn Sie die Option **tcpip** angeben, können Sie optional folgende Protokolloptionen festlegen:

TCP/IP-Protokolloption	Weitere Hinweise finden Sie unter...
client_port = <i>nnnnn</i> [- <i>mmmmm</i>]	„client_port“
compression = <i>{ zlib none }</i>	„compression“
e2ee_public_key = <i>file</i>	„e2ee_public_key“
host = <i>hostname</i>	„host“
network_adapter_name = <i>name</i>	„network_adapter_name“
network_leave_open = <i>{ off on }</i>	„network_leave_open“
network_name = <i>name</i>	„network_name“
port = <i>portnumber</i>	„port“
timeout = <i>seconds</i>	„timeout“
zlib_download_window_size = <i>window-bits</i>	„zlib_download_window_size“
zlib_upload_window_size = <i>window-bits</i>	„zlib_upload_window_size“

- **TCP/IP-Protokoll mit Sicherheit** Wenn Sie die Option **tls** angeben (TCP/IP mit TLS-Sicherheit), können Sie optional folgende Protokolloptionen festlegen:

TLS-Protokolloption	Weitere Hinweise finden Sie unter...
certificate_company = <i>company_name</i>	„certificate_company“
certificate_name = <i>name</i>	„certificate_name“
certificate_unit = <i>company_unit</i>	„certificate_unit“
client_port = <i>nnnnn</i> [- <i>mmmmm</i>]	„client_port“
compression = <i>{ zlib none }</i>	„compression“
e2ee_public_key = <i>file</i>	„e2ee_public_key“

TLS-Protokolloption	Weitere Hinweise finden Sie unter...
fips = <i>{y n}</i>	„fips“
host = <i>hostname</i>	„host“
identity = <i>filename</i>	„identity“
identity_name = <i>name</i>	„identity_name“
identity_password = <i>password</i>	„identity_password“
network_adapter_name = <i>name</i>	„network_adapter_name“
network_leave_open = <i>{off on}</i>	„network_leave_open“
network_name = <i>name</i>	„network_name“
port = <i>portnumber</i>	„port“
timeout = <i>seconds</i>	„timeout“
trusted_certificates = <i>filename</i>	„trusted_certificates“
trusted_certificate_name = <i>name</i>	„trusted_certificate_name“ auf Seite 57
zlib_download_window_size = <i>window-bits</i>	„zlib_download_window_size“
zlib_upload_window_size = <i>window-bits</i>	„zlib_upload_window_size“

- **HTTP protocol** Wenn Sie die Option **http** angeben, können Sie optional folgende Protokolloptionen festlegen:

HTTP-Protokolloption	Weitere Hinweise finden Sie unter...
buffer_size = <i>number</i>	„buffer_size“
client_port = <i>nnnnn[-mmmmmm]</i>	„client_port“
compression = <i>{zlib none}</i>	„compression“
custom_header = <i>header</i>	„custom_header“
e2ee_public_key = <i>file</i>	„e2ee_public_key“
http_buffer_responses = <i>{on off}</i>	„http_buffer_responses“
http_password = <i>password</i>	„http_password“

HTTP-Protokoloption	Weitere Hinweise finden Sie unter...
http_proxy_password = <i>password</i>	„ http_proxy_password “
http_proxy_userid = <i>userid</i>	„ http_proxy_userid “
http_userid = <i>userid</i>	„ http_userid “
host = <i>hostname</i>	„ host “
network_adapter_name = <i>name</i>	„ network_adapter_name “
network_leave_open = <i>{ off on }</i>	„ network_leave_open “
network_name = <i>name</i>	„ network_name “
persistent = <i>{ off on }</i>	„ persistent “
port = <i>portnumber</i>	„ port “
proxy_host = <i>proxy-hostname-or-ip</i>	„ proxy_host “
proxy_port = <i>proxy-portnumber</i>	„ proxy_port “
set_cookie = <i>cookie-name=cookie-value</i>	„ set_cookie “
timeout = <i>seconds</i>	„ timeout “
url_suffix = <i>suffix</i>	„ url_suffix “
version = <i>HTTP-version-number</i>	„ version “
zlib_download_window_size = <i>window-bits</i>	„ zlib_download_window_size “
zlib_upload_window_size = <i>window-bits</i>	„ zlib_upload_window_size “

- **HTTPS protocol** Wenn Sie die Option **https** angeben (HTTP mit RSA-Verschlüsselung), können Sie optional folgende Protokoloptionen festlegen:

HTTPS-Protokoloption	Weitere Hinweise finden Sie unter...
buffer_size = <i>number</i>	„ buffer_size “
certificate_company = <i>company_name</i>	„ certificate_company “
certificate_name = <i>name</i>	„ certificate_name “
certificate_unit = <i>company_unit</i>	„ certificate_unit “

HTTPS-Protokolloption	Weitere Hinweise finden Sie unter...
client_port =nnnnn[-mmmmm]	„client_port“
compression = <i>{ zlib none }</i>	„compression“
custom_header = <i>header</i>	„custom_header“
e2ee_public_key = <i>file</i>	„e2ee_public_key“
fips = <i>{ y n }</i>	„fips“
host = <i>hostname</i>	„host“
http_buffer_responses = <i>{ off on }</i>	„http_buffer_responses“
http_password = <i>password</i>	„http_password“
http_proxy_password = <i>password</i>	„http_proxy_password“
http_proxy_userid = <i>userid</i>	„http_proxy_userid“
http_userid = <i>userid</i>	„http_userid“
identity = <i>filename</i>	„identity“
identity_name = <i>name</i>	„identity_name“
identity_password = <i>password</i>	„identity_password“
network_adapter_name = <i>name</i>	„network_adapter_name“
network_leave_open = <i>{ off on }</i>	„network_leave_open“
network_name = <i>name</i>	„network_name“
persistent = <i>{ off on }</i>	„persistent“
port = <i>portnumber</i>	„port“
proxy_host = <i>proxy-hostname-or-ip</i>	„proxy_host“
proxy_port = <i>proxy-portnumber</i>	„proxy_port“
set_cookie = <i>cookie-name=cookie-value</i>	„set_cookie“
timeout = <i>seconds</i>	„timeout“
trusted_certificates = <i>filename</i>	„trusted_certificates“

HTTPS-Protokolloption	Weitere Hinweise finden Sie unter...
trusted_certificate_name = <i>name</i>	„trusted_certificate_name“ auf Seite 57
url_suffix = <i>suffix</i>	„url_suffix“
version = <i>HTTP-version-number</i>	„version“
zlib_download_window_size = <i>window-size</i>	„zlib_download_window_size“
zlib_upload_window_size = <i>window-bits</i>	„zlib_upload_window_size“

Hinweis

Erforderliche getrennt lizenzierbare Komponenten.

FIPS-zertifizierte Verschlüsselung erfordert eine separate Lizenz. Alle Technologien für starke Verschlüsselungen unterliegen Exportbestimmungen.

Siehe „[Getrennt lizenzierbare Komponenten](#)“ [*SQL Anywhere 16 - Einführung*].

buffer_size

Legt die maximale Anzahl der Pufferbytes vor dem Schreiben in das Netzwerk fest. Für HTTP und HTTPS ist dies die maximale Größe der HTTP-Anforderung.

Syntax

buffer_size=*bytes*

Verfügbare Protokolle

- HTTP, HTTPS

Standardwert

- Mobile OS – 16K
- Desktop OS – 64K

Bemerkungen

Im Allgemeinen gilt für HTTP und HTTPS: Je größer die Puffergröße, desto geringer ist die Anzahl der HTTP-Anforderung/Antwort-Zyklen, dafür erhöht sich jedoch der Speicherbedarf.

Die Einheit ist Byte. Geben Sie K für Kilobyte, M für Megabyte und G für Gigabyte an.

Der Maximalwert ist "1G".

Diese Option steuert die Größe der Anforderungen vom Client und hat keine Auswirkungen auf die Größe der Antworten von MobiLink.

Hinweise zum Einstellen der Netzwerkprotokolloptionen mit dbmlsync finden Sie unter „[Erweiterte Option CommunicationAddress \(adr\)](#)“ auf Seite 149.

Hinweise zum Einstellen der Netzwerkprotokolloptionen mit UltraLite finden Sie unter „[UltraLite-Netzwerkprotokolloptionen für dbmlsync](#)“ [[UltraLite - Datenbankverwaltung](#)].

Beispiel

Im folgenden Beispiel wird die maximale Byte-Anzahl auf 32K festgelegt.

Auf einem SQL Anywhere-Client lautet die Implementierung folgendermaßen:

```
dbmlsync -e "adr=buffer_size=32K"
```

In einer UltraLite-Anwendung, die mit Embedded SQL oder C++ geschrieben wurde, lautet die Implementierung folgendermaßen:

```
synch_info.stream_parms = "buffer_size=32K";
```

certificate_company

Wenn angegeben, akzeptiert die Anwendung nur Serverzertifikate, wenn das Feld "Organisation" auf dem Zertifikat mit diesem Wert übereinstimmt.

Hinweis

Erforderliche getrennt lizenzierbare Komponenten.

FIPS-zertifizierte Verschlüsselung erfordert eine separate Lizenz. Alle Technologien für starke Verschlüsselungen unterliegen Exportbestimmungen.

Siehe „[Getrennt lizenzierbare Komponenten](#)“ [[SQL Anywhere 16 - Einführung](#)].

Syntax

certificate_company=*organization*

Verfügbare Protokolle

- TLS, HTTPS

Standardwert

Keine

Bemerkungen

MobiLink-Clients vertrauen allen Zertifikaten der entsprechenden Zertifizierungsstelle, daher akzeptieren sie möglicherweise auch Zertifikate, die diese Zertifizierungsstelle für andere Unternehmen ausgestellt hat. Unbeabsichtigt kann Ihr Client in diesem Fall den MobiLink-Server eines anderen Unternehmens für Ihren eigenen halten und vertrauliche Daten an diesen übermitteln. Diese Option legt als weitere

Überprüfungsebene fest, dass das Unternehmensfeld im Identitätsabschnitt des Zertifikats auch mit einem von Ihnen angegebenen Wert übereinstimmt.

Hinweise zum Einstellen der Netzwerkprotokolloptionen mit dbmlsync finden Sie unter „[Erweiterte Option CommunicationAddress \(adr\)](#)“ auf Seite 149.

Hinweise zum Einstellen der Netzwerkprotokolloptionen mit UltraLite finden Sie unter „[UltraLite-Netzwerkprotokolloptionen für dbmlsync](#)“ [[UltraLite - Datenbankverwaltung](#)].

Siehe auch

- „Verschlüsselung der MobiLink-Client/Server-Kommunikation“ [[SQL Anywhere Server - Datenbankadministration](#)]
- Zertifikatsfelder überprüfen [[SQL Anywhere Server - Datenbankadministration](#)]
- „mlsrv16-Option -x“ [[MobiLink - Serveradministration](#)]
- „trusted_certificates“ auf Seite 56
- „certificate_name“ auf Seite 33
- „certificate_unit“ auf Seite 34

Beispiel

Das folgende Beispiel legt die RSA-Verschlüsselung für ein HTTPS-Protokoll fest. Die Festlegung muss sowohl auf dem Server als auch auf dem Client durchgeführt werden. Jeder Befehl muss in einer einzigen Zeile eingegeben werden.

Die Serverimplementierung lautet:

```
mlsrv16
-c "DSN=SQL Anywhere 16 Demo;UID=DBA;PWD=sql"
-x https(
  identity=c:\%SQLANYSAMPl6%\Certificates\rsaserver.id;
  identity_password=test)
```

Auf einem SQL Anywhere-Client lautet die Implementierung folgendermaßen:

```
dbmlsync
-c "DSN=mydb;UID=DBA;PWD=sql"
-e "ctp=https;
  adr='trusted_certificates=c:\%SQLANYSAMPl6%\Certificates\rsaroot.crt;
  certificate_name=RSA Server;
  certificate_company=test;
  certificate_unit=test'"
```

Auf einem UltraLite-Client lautet die Implementierung folgendermaßen:

```
info.stream = "https";
info.stream_parms =
  "trusted_certificates=\%SQLANYSAMPl6%\Certificates\rsaroot.crt;"
  "certificate_name=RSA Server;"
  "certificate_company=test;"
  "certificate_unit=test";
```

certificate_name

Wenn angegeben, akzeptiert die Anwendung nur Serverzertifikate, wenn das Feld "Name" auf dem Zertifikat mit diesem Wert übereinstimmt.

Hinweis

Erforderliche getrennt lizenzierbare Komponenten.

FIPS-zertifizierte Verschlüsselung erfordert eine separate Lizenz. Alle Technologien für starke Verschlüsselungen unterliegen Exportbestimmungen.

Siehe „Getrennt lizenzierbare Komponenten“ [[SQL Anywhere 16 - Einführung](#)].

Syntax

certificate_name=*common-name*

Verfügbare Protokolle

- TLS, HTTPS

Standardwert

Keine

Bemerkungen

MobiLink-Clients vertrauen allen Zertifikaten der entsprechenden Zertifizierungsstelle, daher akzeptieren sie möglicherweise auch Zertifikate, die diese Zertifizierungsstelle für andere Unternehmen ausgestellt hat. Unbeabsichtigt kann Ihr Client in diesem Fall den MobiLink-Server eines anderen Unternehmens für Ihren eigenen halten und vertrauliche Daten an diesen übermitteln. Diese Option legt als weitere Überprüfungsebene fest, dass das Namensfeld ("Common Name") im Identitätsabschnitt des Zertifikats auch mit einem von Ihnen angegebenen Wert übereinstimmt.

Hinweise zum Einstellen der Netzwerkprotokolloptionen mit dbmlsync finden Sie unter „[Erweiterte Option CommunicationAddress \(adr\)](#)“ auf Seite 149.

Hinweise zum Einstellen der Netzwerkprotokolloptionen mit UltraLite finden Sie unter „[UltraLite-Netzwerkprotokolloptionen für dbmlsync](#)“ [[UltraLite - Datenbankverwaltung](#)].

Siehe auch

- „Verschlüsselung der MobiLink-Client/Server-Kommunikation“ [[SQL Anywhere Server - Datenbankadministration](#)]
- Zertifikatsfelder überprüfen [[SQL Anywhere Server - Datenbankadministration](#)]
- „mlsrv16-Option -x“ [[MobiLink - Serveradministration](#)]
- „trusted_certificates“ auf Seite 56
- „certificate_company“ auf Seite 31
- „certificate_unit“ auf Seite 34

Beispiel

Das folgende Beispiel legt die RSA-Verschlüsselung für ein HTTPS-Protokoll fest. Die Festlegung muss sowohl auf dem Server als auch auf dem Client durchgeführt werden. Jeder Befehl muss in einer einzigen Zeile eingegeben werden.

Die Serverimplementierung lautet:

```
mlsrv16
-c "DSN=SQL Anywhere 16 Demo;UID=DBA;PWD=sql"
-x https(
  identity=c:\%SQLANYAMP16%\Certificates\rsaserver.id;
  identity_password=test)
```

Auf einem SQL Anywhere-Client lautet die Implementierung folgendermaßen:

```
dbmlsync
-c "DSN=mydb;UID=DBA;PWD=sql"
-e "ctp=https;
  adr='trusted_certificates=c:\%SQLANYAMP16%\Certificates\rsaroot.crt;
  certificate_name=RSA Server"
  certificate_company=test;
  certificate_unit=test'"
```

Auf einem UltraLite-Client lautet die Implementierung folgendermaßen:

```
info.stream = "https";
info.stream_parms =
  "trusted_certificates=%SQLANYAMP16%\Certificates\rsaroot.crt;"
  "certificate_name=RSA Server;"
  "certificate_company=test;"
  "certificate_unit=test";
```

certificate_unit

Wenn angegeben, akzeptiert die Anwendung nur Serverzertifikate, wenn das Feld "Organisationseinheit" auf dem Zertifikat mit diesem Wert übereinstimmt.

Hinweis

Erforderliche getrennt lizenzierbare Komponenten.

FIPS-zertifizierte Verschlüsselung erfordert eine separate Lizenz. Alle Technologien für starke Verschlüsselungen unterliegen Exportbestimmungen.

Siehe „Getrennt lizenzierbare Komponenten“ [[SQL Anywhere 16 - Einführung](#)].

Syntax

certificate_unit=*organization-unit*

Verfügbare Protokolle

- TLS, HTTPS

Standardwert

Keine

Bemerkungen

MobiLink-Clients vertrauen allen Zertifikaten der entsprechenden Zertifizierungsstelle, daher akzeptieren sie möglicherweise auch Zertifikate, die diese Zertifizierungsstelle für andere Unternehmen ausgestellt hat. Unbeabsichtigt kann Ihr Client in diesem Fall den MobiLink-Server eines anderen Unternehmens für Ihren eigenen halten und vertrauliche Daten an diesen übermitteln. Diese Option legt als weitere Überprüfungsebene fest, dass das Feld der Organisationseinheit im Identitätsabschnitt des Zertifikats auch mit einem von Ihnen angegebenen Wert übereinstimmt.

Hinweise zum Einstellen der Netzwerkprotokolloptionen mit dbmlsync finden Sie unter „[Erweiterte Option CommunicationAddress \(adr\)](#)“ auf Seite 149.

Hinweise zum Einstellen der Netzwerkprotokolloptionen mit UltraLite finden Sie unter „[UltraLite-Netzwerkprotokolloptionen für dbmlsync](#)“ [[UltraLite - Datenbankverwaltung](#)].

Siehe auch

- „Verschlüsselung der MobiLink-Client/Server-Kommunikation“ [[SQL Anywhere Server - Datenbankadministration](#)]
- Zertifikatsfelder überprüfen [[SQL Anywhere Server - Datenbankadministration](#)]
- „mlsrv16-Option -x“ [[MobiLink - Serveradministration](#)]
- „trusted_certificates“ auf Seite 56
- „certificate_company“ auf Seite 31
- „certificate_name“ auf Seite 33

Beispiel

Das folgende Beispiel legt die RSA-Verschlüsselung für ein HTTPS-Protokoll fest. Die Festlegung muss sowohl auf dem Server als auch auf dem Client durchgeführt werden. Jeder Befehl muss in einer einzigen Zeile eingegeben werden.

Die Serverimplementierung lautet:

```
mlsrv16
-c "DSN=SQL Anywhere 16 Demo;UID=DBA;PWD=sql"
-x https(
  identity=c:\%SQLANYAMP16%\Certificates\rsaserver.id;
  identity_password=test)
```

Auf einem SQL Anywhere-Client lautet die Implementierung folgendermaßen:

```
dbmlsync
-c "DSN=mydb;UID=DBA;PWD=sql"
-e "ctp=https;
  adr='trusted_certificates=c:\%SQLANYAMP16%\Certificates\rsaroot.crt;
  certificate_name=RSA Server"
  certificate_company=test;
  certificate_unit=test'"
```

Auf einem UltraLite-Client lautet die Implementierung folgendermaßen:

```
info.stream = "https";
info.stream_parms =
```

```
"trusted_certificates=%SQLANYSAMPl6%\Certificates\rsaroot.crt;"
"certificate_name=RSA Server;"
"certificate_company=test;"
"certificate_unit=test";
```

client_port

Legt einen Bereich von Clientports für die Kommunikation fest.

Syntax

client_port=*nnnnn*[-*mmmmm*]

Verfügbare Protokolle

- TCPIP, TLS, HTTP, HTTPS

Standardwert

Keine

Bemerkungen

Geben Sie einen unteren und einen oberen Wert an, um einen Bereich für mögliche Portnummern zu erstellen. Um den Client auf eine bestimmte Portnummer festzulegen, geben Sie dieselbe Nummer für *nnnnn* und *mmmmm* an. Wenn Sie nur einen Wert definieren, ist das Ende des Bereichs um 100 größer als der Anfangswert, also insgesamt 101 Ports.

Diese Option kann für Clients innerhalb einer Firewall nützlich sein, die mit einem MobiLink-Server außerhalb der Firewall kommunizieren.

Hinweise zum Einstellen der Netzwerkprotokolloptionen mit dbmlsync finden Sie unter „[Erweiterte Option CommunicationAddress \(adr\)](#)“ auf Seite 149.

Hinweise zum Einstellen der Netzwerkprotokolloptionen mit UltraLite finden Sie unter „[UltraLite-Netzwerkprotokolloptionen für dbmlsync](#)“ [[UltraLite - Datenbankverwaltung](#)].

Beispiel

Im folgenden Beispiel wird der Bereich der zulässigen Clientports auf 10000 festgelegt.

Auf einem SQL Anywhere-Client lautet die Implementierung folgendermaßen:

```
dbmlsync -e "adr=client_port=10000-19999"
```

In einer UltraLite-Anwendung, die mit Embedded SQL oder C++ geschrieben wurde, lautet die Implementierung folgendermaßen:

```
synch_info.stream_parms = "client_port=10000-19999";
```

compression

Legt den Typ der Datenkomprimierung fest, der verwendet werden soll.

Syntax

compression= { **zlib** | **none** }

Verfügbare Protokolle

- TCPIP, TLS, HTTP, HTTPS

Standardwert

UltraLite verwendet standardmäßig keine Komprimierung.

Dbmlsync verwendet standardmäßig die zlib-Komprimierung.

Vorsicht

Wenn Sie in SQL Anywhere-Clients die Komprimierung deaktivieren, werden die Daten sichtbar. Verschlüsseln Sie daher den Datenstrom, wenn Sicherheit ein wichtiger Faktor ist.

Siehe „[Transportschichtssicherheit](#)“ [*SQL Anywhere Server - Datenbankadministration*].

Bemerkungen

Wenn Sie die zlib-Komprimierung verwenden, können Sie die Upload- und Download-Komprimierung mit den Optionen `zlib_download_window_size` und `zlib_upload_window_size` entsprechend konfigurieren. Mit diesen Optionen können Sie außerdem die Komprimierung für den Upload oder den Download ausschalten.

Wenn Sie Ihre Anwendung direkt mit der statischen UltraLite-Laufzeitbibliothek verknüpfen, rufen Sie `ULEnableZlibSyncCompression(sqlca)` auf, um die zlib-Funktionen direkt in Ihre Anwendung einzubinden. Andernfalls muss `mlczlib12.dll` bereitgestellt werden.

Hinweise zum Einstellen der Netzwerkprotokolloptionen mit dbmlsync finden Sie unter „[Erweiterte Option CommunicationAddress \(adr\)](#)“ auf Seite 149.

Hinweise zum Einstellen der Netzwerkprotokolloptionen mit UltraLite finden Sie unter „[UltraLite-Netzwerkprotokolloptionen für dbmlsync](#)“ [*UltraLite - Datenbankverwaltung*].

Siehe auch

- „[zlib_download_window_size](#)“ auf Seite 60
- „[zlib_upload_window_size](#)“ auf Seite 61

Beispiel

Die folgende Option stellt nur die Komprimierung für den Upload ein und legt die Größe des Upload-Fensters auf 9 fest:

```
"compression=zlib;zlib_download_window_size=0;zlib_upload_window_size=9"
```

custom_header

Legt einen benutzerdefinierten HTTP-Header fest

Syntax

custom_header=header

HTTP-Header haben die Form *header-name: header-value*.

Verfügbare Protokolle

- HTTP, HTTPS

Standardwert

Keine

Bemerkungen

Wenn Sie benutzerdefinierte HTTP-Header festlegen, bezieht der Client die Header in jede HTTP-Anforderung ein, die er versendet. Um mehrere benutzerdefinierte Header festzulegen, verwenden Sie **custom_header** mehrfach. Verwenden Sie dabei das Semikolon (;) als Trennzeichen. Beispiel:

custom_header=header1:value1; custom_header=header2:value2

Benutzerdefinierte Header sind nützlich, wenn Ihr Synchronisationsclient mit einem Tool anderer Hersteller interagiert, das benutzerdefinierte Header erfordert.

Hinweise zum Einstellen der Netzwerkprotokolloptionen mit dbmlsync finden Sie unter „[Erweiterte Option CommunicationAddress \(adr\)](#)“ auf Seite 149.

Hinweise zum Einstellen der Netzwerkprotokolloptionen mit UltraLite finden Sie unter „[UltraLite-Netzwerkprotokolloptionen für dbmlsync](#)“ [[UltraLite - Datenbankverwaltung](#)].

Beispiel

Manche HTTP-Proxyserver erfordern, dass alle Anforderungen spezielle Header enthalten. Das folgende Beispiel legt einen benutzerdefinierten HTTP-Header namens MyProxyHdr mit dem Wert ProxyUser in einer mit Embedded SQL- oder C++ programmierten UltraLite-Anwendung fest.

```
info.stream = "http";
info.stream_parms =
    "host=www.myhost.com;proxy_host=www.myproxy.com;
    custom_header=MyProxyHdr:ProxyUser";
```

e2ee_public_key

Gibt die Datei an, die den öffentlichen Schlüssel des Servers oder das X.509-Zertifikat für die Ende-zu-Ende-Verschlüsselung enthält.

Syntax

e2ee_public_key=file

Verfügbare Protokolle

TCPIP, TLS, HTTP, HTTPS

Standardwert

Keine

Bemerkungen

Die Datei kann entweder PEM- oder DER-kodiert sein.

Diese Option ist erforderlich, damit die Ende-zu-Ende-Verschlüsselung wirksam werden kann.

Die Ende-zu-Ende-Verschlüsselung kann auch mit der TLS/HTTPS-Protokolloption `fips` verwendet werden.

Hinweise zum Einstellen der Netzwerkprotokolloptionen mit `dbmlsync` finden Sie unter „[Erweiterte Option CommunicationAddress \(adr\)](#)“ auf Seite 149.

Hinweise zum Einstellen der Netzwerkprotokolloptionen mit `UltraLite` finden Sie unter „[UltraLite-Netzwerkprotokolloptionen für dbmlsync](#)“ [[UltraLite - Datenbankverwaltung](#)].

Siehe auch

- „`fips`“ auf Seite 39
- „`mlsrv16-Option -x`“ [[MobiLink - Serveradministration](#)]
- „Schlüsselpaargenerator-Dienstprogramm (`createkey`)“ [[SQL Anywhere Server - Datenbankadministration](#)]

Beispiel

Im folgenden Beispiel wird die Ende-zu-Ende-Verschlüsselung für einen `UltraLite`-Client dargestellt:

```
info.stream = "https";  
info.stream_parms =  
"trusted_certificates\=rsaroot.crt;e2ee_public_key=rsapublic.pem"
```

fips

Verwenden Sie FIPS-zertifizierte Verschlüsselungsimplementierungen für TLS-Verschlüsselung und Ende-zu-Ende-Verschlüsselung.

Hinweis

Erforderliche getrennt lizenzierte Komponenten.

FIPS-zertifizierte Verschlüsselung erfordert eine separate Lizenz. Alle Technologien für starke Verschlüsselungen unterliegen Exportbestimmungen.

Siehe „[Getrennt lizenzierte Komponenten](#)“ [[SQL Anywhere 16 - Einführung](#)].

Syntax

```
fips={ y | n }
```

Verfügbare Protokolle

HTTPS, TLS

Standardwert

Nein

Bemerkungen

Die fips-Option unterstützt nur die RSA-Verschlüsselung.

Clients, für die die fips-Option aktiviert ist, können Verbindungen mit Servern herstellen, für die die fips-Option deaktiviert ist, und umgekehrt.

Diese Option kann bei der Ende-zu-Ende-Verschlüsselung verwendet werden. Wenn fips auf **y** festgelegt ist, verwenden MobiLink-Clients FIPS 140-2-zertifizierte Implementierungen von RSA und AES. Siehe „[e2ee_public_key](#)“ auf Seite 38.

Hinweise zum Einstellen der Netzwerkprotokolloptionen mit dbmlsync finden Sie unter „[Erweiterte Option CommunicationAddress \(adr\)](#)“ auf Seite 149.

Hinweise zum Einstellen der Netzwerkprotokolloptionen mit UltraLite finden Sie unter „[UltraLite-Netzwerkprotokolloptionen für dbmlsync](#)“ [[UltraLite - Datenbankverwaltung](#)].

Beispiel

Im folgenden Beispiel wird die RSA-Verschlüsselung für ein TCP/IP-Protokoll eingerichtet. Die Festlegung muss sowohl auf dem Server als auch auf dem Client durchgeführt werden. Jeder Befehl muss in einer einzigen Zeile eingegeben werden.

Auf dem Server lautet die Implementierung folgendermaßen:

```
mlsrv16
-c "DSN=SQL Anywhere 16 Demo;UID=DBA;PWD=sql"
-x tls(
  fips=y;
  identity=c:\%SQLANYAMP16%\Certificates\rsaserver.id;
  identity_password=test)
```

Auf einem SQL Anywhere-Client lautet die Implementierung folgendermaßen:

```
dbmlsync -e
"CommunicationType=tls;
CommunicationAddress=
'fips=y;
trusted_certificates=\rsaroot.crt;
certificate_name=RSA Server'"
```

In einer UltraLite-Anwendung, die mit Embedded SQL, in C oder C++ geschrieben wurde, lautet die Implementierung folgendermaßen:

```
info.stream = "tls";
info.stream_parms =
"fips=y;
trusted_certificates=\rsaroot.crt;
certificate_name=RSA Server";
```

host

Gibt den Hostnamen oder die IP-Nummer des Computers an, auf dem der MobiLink-Server läuft, oder, falls Sie mit einem Webserver synchronisieren, den Computer, auf dem der Webserver läuft.

Syntax

host=*hostname-or-ip*

Verfügbare Protokolle

- TCPIP, TLS, HTTP, HTTPS

Standardwert

- Windows Mobile: Der Standardwert ist die IP-Adresse des PCs, mit dem Microsoft ActiveSync eine Partnerschaft eingegangen ist.
- Alle anderen Geräte: Standardwert ist **localhost**.

Bemerkungen

Verwenden Sie unter Windows Mobile localhost nicht, da sich dieser Wert auf das entfernte Gerät selbst bezieht. Mit dem Standardwert kann ein Windows Mobile-Gerät eine Verbindung zu einem MobiLink-Server auf dem PC einrichten, mit dem das Windows Mobile-Gerät eine Microsoft ActiveSync-Partnerschaft eingegangen ist.

Hinweise zum Einstellen der Netzwerkprotokolloptionen mit dbmlsync finden Sie unter „[Erweiterte Option CommunicationAddress \(adr\)](#)“ auf Seite 149.

Hinweise zum Einstellen der Netzwerkprotokolloptionen mit UltraLite finden Sie unter „[UltraLite-Netzwerkprotokolloptionen für dbmlsync](#)“ [[UltraLite - Datenbankverwaltung](#)].

Beispiel

Im folgenden Beispiel stellt der Client eine Verbindung mit einem Computer namens myhost an Port 1234 her.

Auf einem SQL Anywhere-Client lautet die Implementierung folgendermaßen:

```
dbmlsync -e "adr='host=myhost;port=1234' "
```

In einer UltraLite-Anwendung, die mit Embedded SQL oder C++ geschrieben wurde, lautet die Implementierung folgendermaßen:

```
synch_info.stream_parms = "host=myhost;port=1234";
```

http_buffer_responses

Ist diese Option aktiviert, sendet sie HTTP-Pakete vor der Verarbeitung per Datenstrom von MobiLink in einen Puffer, anstatt die empfangenen Byte sofort zu verarbeiten.

Syntax

http_buffer_responses={ **off** | **on** }

Verfügbare Protokolle

- HTTP, HTTPS

Standardwert

Off

Bemerkungen

Aufgrund des zusätzlich erforderlichen Speichers sollte diese Funktion nur verwendet werden, wenn Stabilitätsprobleme bei der HTTP-Synchronisation umgangen werden müssen. Insbesondere verwirft der Microsoft ActiveSync-Proxyserver für Windows Mobile-Geräte Daten, die nicht innerhalb von 15 Sekunden, nach dem der Server die Verbindung auf seiner Seite beendet hat, gelesen worden sind. Da MobiLink-Clients den Download beim Empfang direkt verarbeiten, besteht die Möglichkeit, dass sie ein HTTP-Paket innerhalb der zugeteilten 15 Sekunden nicht zu Ende lesen können. Dies führt dazu, dass die Synchronisation mit einem Datenstromfehler fehlschlägt, wenn die Synchronisation über eine nicht beständige HTTP-Verbindung erfolgt. Durch die Angabe **http_buffer_responses=on** liest der Client jedes HTTP-Paket in seiner Gesamtheit in einen Puffer ein, bevor er mit der Verarbeitung von Daten daraus beginnt. Auf diese Weise können Probleme durch Überschreitung des Timeouts von 15 Sekunden vermieden werden.

Hinweise zum Einstellen der Netzwerkprotokolloptionen mit dbmlsync finden Sie unter „[Erweiterte Option CommunicationAddress \(adr\)](#)“ auf Seite 149.

Hinweise zum Einstellen der Netzwerkprotokolloptionen mit UltraLite finden Sie unter „[UltraLite-Netzwerkprotokolloptionen für dbmlsync](#)“ [[UltraLite - Datenbankverwaltung](#)].

http_password

Authentifizierung gegenüber HTTP-Servern und Gateways anderer Hersteller, die RFC 2617 Basic- oder Digest-Authentifizierung verwenden

Syntax

http_password=*password*

Verfügbare Protokolle

- HTTP, HTTPS

Standardwert

Keine

Bemerkungen

Diese Funktion unterstützt Basic- und Digest-Authentifizierung, wie in RFC 2617 beschrieben.

Bei der Basic-Authentifizierung werden Kennwörter in HTTP-Headern in lesbarer Form gesendet. Sie können allerdings HTTPS zur Verschlüsselung der Header verwenden, um das Kennwort zu schützen. Bei der Digest-Authentifizierung werden Header nicht in lesbarer Form, sondern im Hash-Format gesendet.

Sie müssen `http_userid` mit dieser Option verwenden.

Hinweise zum Einstellen der Netzwerkprotokolloptionen mit `dbmlsync` finden Sie unter „[Erweiterte Option CommunicationAddress \(adr\)](#)“ auf Seite 149.

Hinweise zum Einstellen der Netzwerkprotokolloptionen mit UltraLite finden Sie unter „[UltraLite-Netzwerkprotokolloptionen für dbmlsync](#)“ [[UltraLite - Datenbankverwaltung](#)].

Siehe auch

- „`http_userid`“ auf Seite 45
- „`http_proxy_password`“ auf Seite 43
- „`http_proxy_userid`“ auf Seite 44

Beispiel

Das folgende Beispiel für eine in Embedded SQL- oder C++ programmierte UltraLite-Anwendung stellt einem Webserver eine Benutzer-ID und ein Kennwort für die Authentifizierung zur Verfügung.

```
synch_info.stream = "https";  
synch_info.stream_parms = "http_userid=user;http_password=pwd";
```

http_proxy_password

Authentifizierung gegenüber HTTP-Proxyservern anderer Hersteller, die RFC 2617 Basic- oder Digest-Authentifizierung verwenden

Syntax

`http_proxy_password=password`

Verfügbare Protokolle

- HTTP, HTTPS

Standardwert

Keine

Bemerkungen

Diese Funktion unterstützt Basic- und Digest-Authentifizierung, wie in RFC 2617 beschrieben.

Bei der Basic-Authentifizierung werden Kennwörter in HTTP-Headern in lesbarer Form gesendet. Sie können HTTPS verwenden, aber die erste Verbindungsaufnahme zum Proxyserver verwendet HTTP, also bleibt das Kennwort in lesbarer Form. Bei der Digest-Authentifizierung werden Header nicht in lesbarer Form, sondern im Hash-Format gesendet.

Sie müssen `http_proxy_userid` mit dieser Option verwenden.

Hinweise zum Einstellen der Netzwerkprotokolloptionen mit `dbmlsync` finden Sie unter „[Erweiterte Option CommunicationAddress \(adr\)](#)“ auf Seite 149.

Hinweise zum Einstellen der Netzwerkprotokolloptionen mit UltraLite finden Sie unter [„UltraLite-Netzwerkprotokolloptionen für dbmlsync“ \[UltraLite - Datenbankverwaltung\]](#).

Siehe auch

- [„http_password“](#) auf Seite 42
- [„http_userid“](#) auf Seite 45
- [„http_proxy_userid“](#) auf Seite 44

Beispiel

Das folgende Beispiel für eine in Embedded SQL oder C++ geschriebene UltraLite-Anwendung stellt einem Web-Proxyserver eine Benutzer-ID und ein Kennwort für die Authentifizierung zur Verfügung.

```
synch_info.stream = "https";  
synch_info.stream_parms = "http_proxy_userid=user;http_proxy_password=pwd";
```

http_proxy_userid

Authentifizierung gegenüber HTTP-Proxyservern anderer Hersteller, die RFC 2617 Basic- oder Digest-Authentifizierung verwenden

Syntax

`http_proxy_userid=userid`

Verfügbare Protokolle

- HTTP, HTTPS

Standardwert

Keine

Bemerkungen

Diese Funktion unterstützt Basic- und Digest-Authentifizierung, wie in RFC 2617 beschrieben.

Bei der Basic-Authentifizierung werden Kennwörter in HTTP-Headern in lesbarer Form einbezogen. Sie können HTTPS verwenden, aber die erste Verbindungsaufnahme zum Proxyserver verwendet HTTP, also bleibt das Kennwort in lesbarer Form. Bei der Digest-Authentifizierung werden Header nicht in lesbarer Form, sondern im Hash-Format gesendet.

Sie müssen `http_proxy_password` mit dieser Option verwenden.

Hinweise zum Einstellen der Netzwerkprotokolloptionen mit dbmlsync finden Sie unter [„Erweiterte Option CommunicationAddress \(adr\)“](#) auf Seite 149.

Hinweise zum Einstellen der Netzwerkprotokolloptionen mit UltraLite finden Sie unter [„UltraLite-Netzwerkprotokolloptionen für dbmlsync“ \[UltraLite - Datenbankverwaltung\]](#).

Siehe auch

- „http_password“ auf Seite 42
- „http_userid“ auf Seite 45
- „http_proxy_password“ auf Seite 43

Beispiel

Das folgende Beispiel für eine in Embedded SQL oder C++ geschriebene UltraLite-Anwendung stellt einem Web-Proxyserver eine Benutzer-ID und ein Kennwort für die Authentifizierung zur Verfügung.

```
synch_info.stream = "https";  
synch_info.stream_parms = "http_proxy_userid=user;http_proxy_password=pwd";
```

http_userid

Authentifizierung gegenüber HTTP-Servern und Gateways anderer Hersteller, die RFC 2617 Basic- oder Digest-Authentifizierung verwenden

Syntax

`http_userid=userid`

Verfügbare Protokolle

- HTTP, HTTPS

Standardwert

Keine

Bemerkungen

Diese Funktion unterstützt Basic- und Digest-Authentifizierung, wie in RFC 2617 beschrieben.

Bei der Basic-Authentifizierung werden Kennwörter in HTTP-Headern in lesbarer Form gesendet; Sie können allerdings HTTPS zur Verschlüsselung der Header verwenden, um das Kennwort zu schützen. Bei der Digest-Authentifizierung werden Header nicht in lesbarer Form, sondern im Hash-Format gesendet.

Sie müssen http_password mit dieser Option verwenden.

Hinweise zum Einstellen der Netzwerkprotokolloptionen mit dbmlsync finden Sie unter „[Erweiterte Option CommunicationAddress \(adr\)](#)“ auf Seite 149.

Hinweise zum Einstellen der Netzwerkprotokolloptionen mit UltraLite finden Sie unter „[UltraLite-Netzwerkprotokolloptionen für dbmlsync](#)“ [[UltraLite - Datenbankverwaltung](#)].

Siehe auch

- „http_password“ auf Seite 42
- „http_proxy_password“ auf Seite 43
- „http_proxy_userid“ auf Seite 44

Beispiel

Das folgende Beispiel für eine in Embedded SQL- oder C++ programmierte UltraLite-Anwendung stellt einem Webserver eine Benutzer-ID und ein Kennwort für die Authentifizierung zur Verfügung.

```
synch_info.stream = "https";  
synch_info.stream_parms = "http_userid=user;http_password=pwd";
```

identity

Verwenden Sie diese Option zum Aktivieren der Verwendung von clientseitigen Zertifikaten für die Authentifizierung von MobiLink-Clients für Server und Proxys anderer Hersteller.

Syntax

identity=*filename*

Verfügbare Protokolle

TLS, HTTPS

Standardwert

Keine

Bemerkungen

filename gibt die Datei mit der Identität des Clients an. Eine Identität besteht aus dem Clientzertifikat, dem entsprechenden privaten Schlüssel und (optional) den Zertifikaten der zwischengeschalteten Zertifizierungsstellen.

Wenn der private Schlüssel verschlüsselt ist, verwenden Sie die Option `identity_password` zur Festlegung eines Kennworts.

MobiLink-Clients können bei Verwendung von clientseitigen Zertifikaten nicht direkt bei MobiLink authentifiziert werden. Sie können nur für die Authentifizierung von Servern und Proxys anderer Hersteller verwendet werden, die so konfiguriert wurden, dass sie eine clientseitige Zertifikatsauthentifizierung akzeptieren, und sich zwischen dem Client und dem MobiLink-Server befinden.

Hinweise zum Einstellen der Netzwerkprotokolloptionen mit `dbmlsync` finden Sie unter „[Erweiterte Option CommunicationAddress \(adr\)](#)“ auf Seite 149.

Hinweise zum Einstellen der Netzwerkprotokolloptionen mit UltraLite finden Sie unter „[UltraLite-Netzwerkprotokolloptionen für dbmlsync](#)“ [[UltraLite - Datenbankverwaltung](#)].

Siehe auch

- „`identity_password`“ auf Seite 47

identity_name

Diese Funktion unterstützt eine Authentifizierung unter Verwendung von Client-Identitäten (ein Zertifikat samt einem privaten Schlüssel) von Common Access Cards (CACs). Diese Funktion wird nur für Windows Mobile unterstützt.

Dieser Parameter wird verwendet, um den allgemeinen Namen des öffentlichen Zertifikats anzugeben.

Hinweis

Erforderliche getrennt lizenzierbare Komponenten Diese Funktion ist Bestandteil der Zusatzkomponente CAC-Authentifizierung und erfordert eine separate Lizenz. Siehe „[Getrennt lizenzierbare Komponenten](#)“ [*SQL Anywhere 16 - Einführung*].

Syntax

`identity_name=name`

Verfügbare Protokolle

- TLS, HTTPS

Standardwert

Keine.

Bemerkungen

Dieser Parameter kann nur bei FIPS-zertifizierter RSA-Verschlüsselung verwendet werden.

Das öffentliche Zertifikat muss im Zertifikatspeicher des Geräts installiert sein.

Hinweis

Erforderliche getrennt lizenzierbare Komponenten.

FIPS-zertifizierte Verschlüsselung erfordert eine separate Lizenz. Alle Technologien für starke Verschlüsselungen unterliegen Exportbestimmungen.

Siehe „[Getrennt lizenzierbare Komponenten](#)“ [*SQL Anywhere 16 - Einführung*].

Hinweise zum Einstellen der Netzwerkprotokolloptionen mit dbmlsync finden Sie unter „[Erweiterte Option CommunicationAddress \(adr\)](#)“ auf Seite 149.

Hinweise zum Einstellen der Netzwerkprotokolloptionen mit UltraLite finden Sie unter „[UltraLite-Netzwerkprotokolloptionen für dbmlsync](#)“ [*UltraLite - Datenbankverwaltung*].

identity_password

Das für die Verschlüsselung des privaten Schlüssels verwendete Kennwort in der Identitätsdatei.

Syntax

identity_password=*password*

Verfügbare Protokolle

- TLS, HTTPS

Standardwert

Keine.

Bemerkungen

Diese Option ist nur erforderlich, wenn der private Schlüssel in der Identitätsdatei verschlüsselt ist. Siehe [„identity“ auf Seite 46](#).

Hinweise zum Einstellen der Netzwerkprotokolloptionen mit dbmlsync finden Sie unter [„Erweiterte Option CommunicationAddress \(adr\)“ auf Seite 149](#).

Hinweise zum Einstellen der Netzwerkprotokolloptionen mit UltraLite finden Sie unter [„UltraLite-Netzwerkprotokolloptionen für dbmlsync“ \[UltraLite - Datenbankverwaltung\]](#).

network_adapter_name

Mit dieser Option können MobiLink-Clients explizit den Namen des Netzwerkadapters für die Verwendung angeben, der bei der Herstellung der Verbindung mit MobiLink verwendet werden soll.

Syntax

network_adapter_name=*name*

Verfügbare Protokolle

- TCPIP, TLS, HTTP, HTTPS

Standardwert

Keine

Bemerkungen

Diese Option gilt nur für Windows Mobile und Windows-PC.

Hinweise zum Einstellen der Netzwerkprotokolloptionen mit dbmlsync finden Sie unter [„Erweiterte Option CommunicationAddress \(adr\)“ auf Seite 149](#).

Hinweise zum Einstellen der Netzwerkprotokolloptionen mit UltraLite finden Sie unter [„UltraLite-Netzwerkprotokolloptionen für dbmlsync“ \[UltraLite - Datenbankverwaltung\]](#).

network_leave_open

Wenn Sie `network_name` angeben, können Sie optional festlegen, dass die Netzwerkverbindung nach erfolgter Synchronisation bestehen bleiben soll.

Syntax

`network_leave_open={ off | on }`

Verfügbare Protokolle

- TCPIP, TLS, HTTP, HTTPS

Standardwert

Der Standardwert ist **off**.

Bemerkungen

Sie müssen `network_name` angeben, um diese Option verwenden zu können.

Wenn diese Option auf On eingestellt ist, bleibt die Netzwerkverbindung bestehen, nachdem die Synchronisation beendet ist.

Hinweise zum Einstellen der Netzwerkprotokolloptionen mit `dbmlsync` finden Sie unter „[Erweiterte Option CommunicationAddress \(adr\)](#)“ auf Seite 149.

Hinweise zum Einstellen der Netzwerkprotokolloptionen mit UltraLite finden Sie unter „[UltraLite-Netzwerkprotokolloptionen für dbmlsync](#)“ [[UltraLite - Datenbankverwaltung](#)].

Siehe auch

- „[network_name](#)“ auf Seite 49

Beispiel

Im folgenden Beispiel verwendet der Client den Netzwerknamen `MyNetwork` und legt fest, dass die Verbindung nach der Beendigung der Synchronisation geöffnet bleiben soll.

Auf einem SQL Anywhere-Client lautet die Implementierung folgendermaßen:

```
dbmlsync -e "adr='network_name=MyNetwork;network_leave_open=on' "
```

In einer UltraLite-Anwendung, die mit Embedded SQL oder C++ geschrieben wurde, lautet die Implementierung folgendermaßen:

```
synch_info.stream_parms = "network_name=MyNetwork;network_leave_open=on";
```

network_name

Legt den Netzwerknamen beim Starten fest, wenn die Verbindung zum Netzwerk fehlschlägt

Syntax

`network_name=name`

Verfügbare Protokolle

- TCPIP, TLS, HTTP, HTTPS

Standardwert

Keine

Bemerkungen

Geben Sie den Namen des Netzwerks an, um die MobiLink-Funktion der automatischen Einwahl verwenden zu können. Dies ermöglicht die Verbindung von einem Windows Mobile-Gerät oder Windows-PC aus, ohne dass Sie sich manuell einwählen müssen. Die Autoeinwahl ist ein weiterer Versuch, sich mit dem MobiLink-Server zu verbinden. Zunächst versucht der Client, sich ohne Einwahl zu verbinden. Schlägt dies fehl und wird ein Netzwerkname (`network_name`) eingerichtet, wird die Autoeinwahl aktiviert. Wenn die Abfolgeplanung verwendet wird, kann Ihre entfernte Datenbank unbeaufsichtigt synchronisiert werden. Wenn die Abfolgeplanungsfunktion nicht verwendet wird, können Sie mit dieser Option `dbmlsync` ausführen, ohne manuell eine Verbindung wählen zu müssen.

Unter Windows Mobile sollte *name* eines der Netzwerkprofile in der Dropdown-Liste in "Einstellungen→Verbindungen→Verbindungen" sein. Um die von Ihnen für das Internet oder das Arbeitsnetzwerk festgelegte Vorgabe zu verwenden, setzen Sie den Namen auf das Schlüsselwort **default_internet** oder **default_work**.

Bei Windows-Arbeitsplatzcomputern sollte *name* eines der Netzwerkprofile aus "Netzwerkverbindungen" sein.

Hinweise zum Einstellen der Netzwerkprotokolloptionen mit `dbmlsync` finden Sie unter „[Erweiterte Option CommunicationAddress \(adr\)](#)“ auf Seite 149.

Hinweise zum Einstellen der Netzwerkprotokolloptionen mit UltraLite finden Sie unter „[UltraLite-Netzwerkprotokolloptionen für dbmlsync](#)“ [[UltraLite - Datenbankverwaltung](#)].

Siehe auch

- „[Synchronisationszeitpläne](#)“ auf Seite 100
- „[network_leave_open](#)“ auf Seite 49

Beispiel

Im folgenden Beispiel verwendet der Client den Netzwerknamen `MyNetwork` und legt fest, dass die Verbindung nach der Beendigung der Synchronisation geöffnet bleiben soll.

Auf einem SQL Anywhere-Client lautet die Implementierung folgendermaßen:

```
dbmlsync -e "adr='network_name=MyNetwork;network_leave_open=on' "
```

In einer UltraLite-Anwendung, die mit Embedded SQL oder C++ geschrieben wurde, lautet die Implementierung folgendermaßen:

```
synch_info.stream_parms = "network_name=MyNetwork;network_leave_open=on";
```

persistent

Verwendet eine einzelne TCP/IP-Verbindung für alle HTTP-Anforderungen in einer Synchronisation

Syntax

persistent={ off | on }

Verfügbare Protokolle

- HTTP, HTTPS

Standardwert

On

Bemerkungen

Der Wert On bedeutet, dass der Client bei einer Synchronisation versuchen wird, für alle HTTP-Anfragen die gleiche TCP/IP-Verbindung zu verwenden.

Wenn ein zwischengeschalteter Server nicht-beständige Verbindungen anfordert oder der Client feststellt, dass ein zwischengeschalteter Server keine beständigen Verbindungen unterstützt, wird die Verbindung automatisch für den Rest der Synchronisationssitzung auf nicht-beständig heruntergestuft.

Hinweise zum Einstellen der Netzwerkprotokolloptionen mit dbmlsync finden Sie unter „[Erweiterte Option CommunicationAddress \(adr\)](#)“ auf Seite 149.

Hinweise zum Einstellen der Netzwerkprotokolloptionen mit UltraLite finden Sie unter „[UltraLite-Netzwerkprotokolloptionen für dbmlsync](#)“ [[UltraLite - Datenbankverwaltung](#)].

port

Legt die Socket-Portnummer des MobiLink-Servers fest

Syntax

port=*port-number*

Verfügbare Protokolle

- TCPIP, TLS, HTTP, HTTPS

Standardwert

Bei TCP/IP ist der Standardwert **2439**. Dies ist die registrierte IANA-Portnummer für den MobiLink-Server.

Bei HTTP ist der Standardwert **80**.

Bei HTTPS ist der Standardwert **443**.

Bemerkungen

Die Portnummer muss eine Dezimalzahl sein, die mit dem Port übereinstimmt, den der MobiLink-Server überwachen soll.

Wenn Sie über einen Webserver synchronisieren, geben Sie den Webserverport an, der HTTP- oder HTTPS-Anforderungen annimmt.

Hinweise zum Einstellen der Netzwerkprotokolloptionen mit dbmlsync finden Sie unter „[Erweiterte Option CommunicationAddress \(adr\)](#)“ auf Seite 149.

Hinweise zum Einstellen der Netzwerkprotokolloptionen mit UltraLite finden Sie unter „[UltraLite-Netzwerkprotokolloptionen für dbmlsync](#)“ [[UltraLite - Datenbankverwaltung](#)].

Beispiel

Im folgenden Beispiel stellt der Client eine Verbindung mit einem Computer namens myhost an Port 1234 her.

Auf einem SQL Anywhere-Client lautet die Implementierung folgendermaßen:

```
dbmlsync -e "adr='host=myhost;port=1234' "
```

In einer UltraLite-Anwendung, die mit Embedded SQL oder C++ geschrieben wurde, lautet die Implementierung folgendermaßen:

```
synch_info.stream_parms = "host=myhost;port=1234";
```

proxy_host

Legt den Hostnamen oder die IP-Adresse des Proxyservers fest

Syntax

proxy_host=*proxy-hostname-or-ip*

Verfügbare Protokolle

- HTTP, HTTPS

Standardwert

Keine

Bemerkungen

Nur anwendbar, wenn Sie einen HTTP-Proxy verwenden

Hinweise zum Einstellen der Netzwerkprotokolloptionen mit dbmlsync finden Sie unter „[Erweiterte Option CommunicationAddress \(adr\)](#)“ auf Seite 149.

Hinweise zum Einstellen der Netzwerkprotokolloptionen mit UltraLite finden Sie unter „[UltraLite-Netzwerkprotokolloptionen für dbmlsync](#)“ [[UltraLite - Datenbankverwaltung](#)].

Beispiel

Im folgenden Beispiel stellt der Client eine Verbindung zu einem Proxy-Server her, der auf einem Computer namens myproxyhost an Port 1234 ausgeführt wird.

Auf einem SQL Anywhere-Client lautet die Implementierung folgendermaßen:

```
dbmlsync -e "adr='proxy_host=myproxyhost;proxy_port=1234' "
```

In einer UltraLite-Anwendung, die mit Embedded SQL oder C++ geschrieben wurde, lautet die Implementierung folgendermaßen:

```
synch_info.stream_parms = "proxy_host=myproxyhost;proxy_port=1234";
```

proxy_port

Legt die Portnummer des Proxyservers fest

Syntax

proxy_port=*proxy-port-number*

Verfügbare Protokolle

- HTTP, HTTPS

Standardwert

Keine

Bemerkungen

Nur anwendbar, wenn Sie einen HTTP-Proxy verwenden

Hinweise zum Einstellen der Netzwerkprotokolloptionen mit dbmlsync finden Sie unter „[Erweiterte Option CommunicationAddress \(adr\)](#)“ auf Seite 149.

Hinweise zum Einstellen der Netzwerkprotokolloptionen mit UltraLite finden Sie unter „[UltraLite-Netzwerkprotokolloptionen für dbmlsync](#)“ [[UltraLite - Datenbankverwaltung](#)].

Beispiel

Im folgenden Beispiel stellt der Client eine Verbindung zu einem Proxy-Server her, der auf einem Computer namens myproxyhost an Port 1234 ausgeführt wird.

Auf einem SQL Anywhere-Client lautet die Implementierung folgendermaßen:

```
dbmlsync -e "adr='proxy_host=myproxyhost;proxy_port=1234' "
```

In einer UltraLite-Anwendung, die mit Embedded SQL oder C++ geschrieben wurde, lautet die Implementierung folgendermaßen:

```
synch_info.stream_parms = "proxy_host=myproxyhost;proxy_port=1234";
```

set_cookie

Legt benutzerdefinierte HTTP-Cookies in HTTP-Anforderungen fest, die während der Synchronisation verwendet werden

Syntax

set_cookie=*cookie-name=cookie-value*,...

Verfügbare Protokolle

- HTTP, HTTPS

Standardwert

Keine

Bemerkungen

Benutzerdefinierte HTTP-Cookies sind nützlich, wenn Ihr Synchronisationsclient mit Tools anderer Hersteller, z.B. mit einem Authentifizierungstool, die Cookies zur Identifizierung von Sitzungen verwenden, interagiert. So könnte in Ihrem System ein Benutzeragent eine Verbindung mit einem Webserver, Proxyserver oder Gateway aufnehmen und sich authentifizieren. Bei Erfolg erhält der Agent ein oder mehrere Cookies vom Server. Der Agent startet dann die Synchronisation und übergibt seine Sitzungs-Cookies durch die `set_cookie`-Option.

Wenn Sie mehrere Paare mit Namen und Werten haben, trennen Sie diese durch Kommas.

Hinweise zum Einstellen der Netzwerkprotokolloptionen mit `dbmlsync` finden Sie unter „[Erweiterte Option CommunicationAddress \(adr\)](#)“ auf Seite 149.

Hinweise zum Einstellen der Netzwerkprotokolloptionen mit `UltraLite` finden Sie unter „[UltraLite-Netzwerkprotokolloptionen für dbmlsync](#)“ [[UltraLite - Datenbankverwaltung](#)].

Beispiel

Das folgende Beispiel setzt zwei benutzerdefinierte HTTP-Cookies in eine in Embedded SQL- oder C++ programmierte `UltraLite`-Anwendung.

```
info.stream = "http";
info.stream_parms =
    "host=www.myhost.com;
    set_cookie=MySessionID=12345, enabled=yes;"
```

timeout

Legt die Zeitspanne in Sekunden fest, die der Client auf erfolgreiche Netzwerkprozesse wartet, bevor er abbricht

Syntax

timeout=*seconds*

Verfügbare Protokolle

- TCPIP, TLS, HTTP, HTTPS

Standardwert

240 Sekunden

Bemerkungen

Wenn ein Verbindungs-, Lese- oder Schreibversuch innerhalb der festgelegten Zeit fehlschlägt, bricht der Client die Synchronisation ab.

Während der gesamten Synchronisation sendet der Client Aktualisierungen zur Verfügbarkeit innerhalb des festgelegten Intervalls, um den MobiLink-Server darüber zu informieren, dass er noch aktiv ist. MobiLink sendet seinerseits Verfügbarkeitsaktualisierungen zurück, um den Client darüber zu informieren, dass MobiLink noch aktiv ist. Um zu verhindern, dass langsame Netzwerke den Timeout über die festgelegte Zeit hinaus verzögern, sendet der MobiLink-Clients in einem Intervall des halben Timeoutwerts Keepalive-Bytes an den MobiLink-Server. Wenn dieser Wert beispielsweise auf 240 Sekunden festgelegt ist, werden alle 120 Sekunden Keepalive-Nachrichten gesendet.

Setzen Sie den Timeoutwert nicht zu niedrig. Die Verfügbarkeitsüberprüfung erhöht die Netzwerkbelastung, weil der MobiLink-Server und der Client innerhalb jedes Timeouts miteinander kommunizieren müssen, um sicherzustellen, dass die Verbindung noch besteht. Bei hoher Netzwerk- oder Serverauslastung und kurzem Timeout wird eine aktive Verbindung möglicherweise abgebrochen, da der MobiLink-Server und dbmlsync die Verfügbarkeit der Verbindung nicht bestätigen können. Der Verfügbarkeits-Timeout sollte in der Regel nicht unter 30 Sekunden liegen.

Der maximale Timeout beträgt 10 Minuten. Sie können einen größeren Wert als 600 Sekunden angeben, doch er wird als 600 Sekunden interpretiert.

Der Wert 0 bedeutet, dass der Timeout 10 Minuten beträgt.

Hinweise zum Einstellen der Netzwerkprotokolloptionen mit dbmlsync finden Sie unter „[Erweiterte Option CommunicationAddress \(adr\)](#)“ auf Seite 149.

Hinweise zum Einstellen der Netzwerkprotokolloptionen mit UltraLite finden Sie unter „[UltraLite-Netzwerkprotokolloptionen für dbmlsync](#)“ [[UltraLite - Datenbankverwaltung](#)].

Beispiel

Das folgende Beispiel setzt den Timeout auf 300 Sekunden.

Auf einem SQL Anywhere-Client lautet die Implementierung folgendermaßen:

```
dbmlsync -e "adr=timeout=300"
```

In einer UltraLite-Anwendung, die mit Embedded SQL oder C++ geschrieben wurde, lautet die Implementierung folgendermaßen:

```
synch_info.stream_parms = "timeout=300";
```

trusted_certificates

Gibt eine Datei an, die eine Liste von vertrauenswürdigen Stammzertifikaten für die sichere Synchronisation enthält.

Hinweis

Erforderliche getrennt lizenzierbare Komponenten.

FIPS-zertifizierte Verschlüsselung erfordert eine separate Lizenz. Alle Technologien für starke Verschlüsselungen unterliegen Exportbestimmungen.

Siehe „[Getrennt lizenzierbare Komponenten](#)“ [*SQL Anywhere 16 - Einführung*].

Syntax

trusted_certificates=*filename*

Verfügbare Protokolle

- TLS, HTTPS

Standardwert

Keine

Bemerkungen

Wenn über einen TLS-Synchronisationsdatenstrom synchronisiert wird, sendet der MobiLink-Server sein Zertifikat an den Client und das Zertifikat der Einheit, die es signiert hat, und so weiter bis zu einem selbstsignierten Stamm.

Der Client überprüft, ob die Kette gültig und das Stammzertifikat in der Kette vertrauenswürdig ist. Mit dieser Funktion können Sie angeben, für welches Stammzertifikat dies zutrifft.

Zertifikate werden gemäß den folgenden Prioritätsregeln verwendet:

- Wenn Zertifikate für UltraLite-Clients durch ulinit oder uload in der Datenbank festgelegt wurden, werden diese Zertifikate verwendet.
- Wenn der trusted_certificates-Parameter angegeben ist, werden die Zertifikate aus der angegebenen Datei verwendet. Sie ersetzen vertrauenswürdige Zertifikate, die mit ulinit oder uload in der Datenbank gespeichert wurden.
- Wenn Zertifikate weder durch den trusted_certificates-Parameter noch durch ulinit oder uload angegeben wurden und Sie Windows, Windows Mobile oder Android verwenden, werden Zertifikate aus dem Speicher für vertrauenswürdigen Zertifikate des Betriebssystems gelesen. Der Zertifikatsspeicher wird von Webbrowsern bei der Verbindung mit sicheren Webservern über HTTPS verwendet.

Sie können nicht sowohl die trusted_certificates-Option als auch die trusted_certificate_name-Option in derselben Gruppe von Datenstromoptionen verwenden.

Hinweise zum Einstellen der Netzwerkprotokolloptionen mit dbmlsync finden Sie unter „Erweiterte Option CommunicationAddress (adr)“ auf Seite 149.

Hinweise zum Einstellen der Netzwerkprotokolloptionen mit UltraLite finden Sie unter „UltraLite-Netzwerkprotokolloptionen für dbmlsync“ [*UltraLite - Datenbankverwaltung*].

Siehe auch

- „trusted_certificate_name“ auf Seite 57
- „UltraLite-Dateipfadformate in Verbindungsparametern“ [*UltraLite - Datenbankverwaltung*]
- „Verschlüsselung der MobiLink-Client/Server-Kommunikation“ [*SQL Anywhere Server - Datenbankadministration*]
- „mlsrv16-Option -x“ [*MobiLink - Serveradministration*]
- „certificate_company“ auf Seite 31
- „certificate_name“ auf Seite 33
- „certificate_unit“ auf Seite 34

Beispiel

Das folgende Beispiel legt die RSA-Verschlüsselung für ein HTTPS-Protokoll fest. Die Festlegung muss sowohl auf dem Server als auch auf dem Client durchgeführt werden. Jeder Befehl muss in einer einzigen Zeile eingegeben werden.

Die Serverimplementierung lautet:

```
mlsrv16
-c "DSN=SQL Anywhere 16 Demo;UID=DBA;PWD=sql"
-x https(
  identity=c:\%SQLANYAMP16%\Certificates\rsaserver.id;
  identity_password=test)
```

Auf einem SQL Anywhere-Client lautet die Implementierung folgendermaßen:

```
dbmlsync
-c "DSN=mydb;UID=DBA;PWD=sql"
-e "ctp=https;
  adr='trusted_certificates=c:\%SQLANYAMP16%\Certificates\rsaroot.crt;
  certificate_name=RSA Server"
  certificate_company=test;
  certificate_unit=test'"
```

Auf einem UltraLite-Client lautet die Implementierung folgendermaßen:

```
info.stream = "https";
info.stream_parms =
  "trusted_certificates=%SQLANYAMP16%\Certificates\rsaroot.crt;"
  "certificate_name=RSA Server;"
  "certificate_company=test;"
  "certificate_unit=test";
```

trusted_certificate_name

Geben Sie einen eindeutigen Namen für ein Zertifikat ein, das in der entfernten Datenbank gespeichert ist. Diese Option ist nur für SQL Anywhere-Clients verfügbar.

Hinweis

Erforderliche getrennt lizenzierbare Komponenten.

FIPS-zertifizierte Verschlüsselung erfordert eine separate Lizenz. Alle Technologien für starke Verschlüsselungen unterliegen Exportbestimmungen.

Siehe „[Getrennt lizenzierbare Komponenten](#)“ [*SQL Anywhere 16 - Einführung*].

Syntax

`trusted_certificate_name=name`

Verfügbare Protokolle

- TLS, HTTPS

Standardwert

Keine

Bemerkungen

Wenn diese Option angegeben ist, ruft dbmlsync das angegebene Zertifikat aus der Datenbank ab und behandelt es für den Zweck der Validierung der Identität des Servers als vertrauenswürdigen Zertifikat.

Sie können nicht sowohl die `trusted_certificate_name`-Option als auch die `trusted_certificates`-Option in derselben Gruppe von Datenstromoptionen verwenden.

Hinweise zum Einstellen der Netzwerkprotokolloptionen mit dbmlsync finden Sie unter „[Erweiterte Option CommunicationAddress \(adr\)](#)“ auf Seite 149.

Hinweise zum Einstellen der Netzwerkprotokolloptionen mit UltraLite finden Sie unter „[UltraLite-Netzwerkprotokolloptionen für dbmlsync](#)“ [*UltraLite - Datenbankverwaltung*].

Siehe auch

- „`trusted_certificates`“ auf Seite 56
- „[UltraLite-Dateipfadformate in Verbindungsparametern](#)“ [*UltraLite - Datenbankverwaltung*]
- „[Verschlüsselung der MobiLink-Client/Server-Kommunikation](#)“ [*SQL Anywhere Server - Datenbankadministration*]
- „`mlsrv16-Option -x`“ [*MobiLink - Serveradministration*]
- „`certificate_company`“ auf Seite 31
- „`certificate_name`“ auf Seite 33
- „`certificate_unit`“ auf Seite 34

Beispiel

Im folgenden Beispiel wird angenommen, dass `root.crt` ein Zertifikat enthält, das Sie als vertrauenswürdigen Stammzertifikat verwenden möchten. Verwenden Sie die folgende Syntax, um das Zertifikat in die Datenbank einzufügen:

```
CREATE CERTIFICATE myroot FROM FILE 'root.crt'
```

Verwenden Sie anschließend die folgenden dbmlsync-Optionen, um auf das Zertifikat zuzugreifen.

```
ALTER SYNCHRONIZATION SUBSCRIPTION mysub  
TYPE tls  
ADDRESS 'trusted_certificate_name=myroot;...'
```

url_suffix

Gibt das Suffix an, das dem URL in der ersten Zeile aller HTTP-Anforderungen, die während der Synchronisation gesendet werden, hinzugefügt werden soll

Syntax

`url_suffix=suffix`

Die Syntax von *suffix* hängt davon ab, ob Sie IIS oder den Apache Relay Server verwenden:

Redirector	Syntax von <i>suffix</i>
IIS	Der Standardpfad für Windows ist <code>/rs/client/rs_client.dll</code> .
Apache	Der Standardpfad für Linux ist <code>/cli/iarelayserver</code> .

Verfügbare Protokolle

- HTTP, HTTPS

Standardwert

Der Standardwert ist `/`.

Bemerkungen

Wenn Sie über einen Proxy- oder Webserver synchronisieren, kann das `url_suffix` benötigt werden, um den MobiLink-Server zu suchen.

Hinweise zum Einstellen dieser Option bei Verwendung des Relay Servers finden Sie unter „[Relay Server-Konfigurationsdatei](#)“ [*Relay Server*].

Hinweise zum Einstellen der Netzwerkprotokolloptionen mit dbmlsync finden Sie unter „[Erweiterte Option CommunicationAddress \(adr\)](#)“ auf Seite 149.

Hinweise zum Einstellen der Netzwerkprotokolloptionen mit UltraLite finden Sie unter „[UltraLite-Netzwerkprotokolloptionen für dbmlsync](#)“ [*UltraLite - Datenbankverwaltung*].

Siehe auch

- „[Relay Server-Konfigurationsdatei](#)“ [*Relay Server*]

version

Gibt die HTTP-Version an, die bei der Synchronisation verwendet werden soll

Syntax

version=*HTTP-version-number*

Verfügbare Protokolle

- HTTP, HTTPS

Standardwert

Der Standardwert ist **1.1**.

Bemerkungen

Diese Option ist nützlich, wenn Ihre HTTP-Infrastruktur eine bestimmte HTTP-Version erfordert. Die Werte können **1.0** oder **1.1** lauten .

Hinweise zum Einstellen der Netzwerkprotokolloptionen mit dbmlsync finden Sie unter „[Erweiterte Option CommunicationAddress \(adr\)](#)“ auf Seite 149.

Hinweise zum Einstellen der Netzwerkprotokolloptionen mit UltraLite finden Sie unter „[UltraLite-Netzwerkprotokolloptionen für dbmlsync](#)“ [[UltraLite - Datenbankverwaltung](#)].

Beispiel

Im folgenden Beispiel wird die HTTP-Version auf 1.0 festgelegt.

Auf einem SQL Anywhere-Client lautet die Implementierung folgendermaßen:

```
dbmlsync -e "adr=version=1.0"
```

In einer UltraLite-Anwendung, die mit Embedded SQL oder C++ geschrieben wurde, lautet die Implementierung folgendermaßen:

```
synch_info.stream_parms = "version=1.0";
```

zlib_download_window_size

Wenn Sie die Komprimierungsoption auf 'zlib' setzen, können Sie mit dieser Option die Größe des Komprimierungsfensters für den Download festlegen.

Syntax

zlib_download_window_size=*window-bits*

Verfügbare Protokolle

- TCPIP, TLS, HTTP, HTTPS

Standardwert

Mobile OS – 12

Desktop OS – 15

Bemerkungen

Um die Komprimierung für Downloads zu deaktivieren, setzen Sie *window-bits* auf "0". Ansonsten kann der Wert für die Fenstergröße zwischen 9 und einschließlich 15 liegen. Im Allgemeinen können mit einem größeren Fenster bessere Komprimierungsraten erreicht werden. Dies erfordert aber auch mehr Speicherkapazität.

window-bits ist der Logarithmus zur Basis 2 der Fenstergröße (Größe des Verlaufspuffers). Mit den folgenden Formeln können Sie festlegen, wieviel Speicher auf dem Client für jedes *window-bits* verwendet wird:

upload (compress): $\text{memory} = 2^{(\text{window-bits} + 3)}$

download (decompress): $\text{memory} = 2^{(\text{window-bits})}$

Wenn Sie Ihre Anwendung direkt mit der statischen UltraLite-Laufzeitbibliothek verknüpfen, rufen Sie `ULEnableZlibSyncCompression(sqlca)` auf, um die zlib-Funktionen direkt in Ihre Anwendung einzubinden. Andernfalls muss *mlczlib12.dll* bereitgestellt werden.

Hinweise zum Einstellen der Netzwerkprotokolloptionen mit dbmlsync finden Sie unter „[Erweiterte Option CommunicationAddress \(adr\)](#)“ auf Seite 149.

Hinweise zum Einstellen der Netzwerkprotokolloptionen mit UltraLite finden Sie unter „[UltraLite-Netzwerkprotokolloptionen für dbmlsync](#)“ [[UltraLite - Datenbankverwaltung](#)].

Siehe auch

- „compression“ auf Seite 36
- „zlib_upload_window_size“ auf Seite 61

Beispiel

Die folgende Option steuert die Komprimierung für den Upload:

```
"compression=zlib;zlib_download_window_size=0"
```

zlib_upload_window_size

Wenn Sie die Komprimierungsoption auf "zlib" setzen, können Sie mit dieser Option die Größe des Komprimierungsfensters für den Upload festlegen.

Syntax

zlib_upload_window_size=*window-bits*

Verfügbare Protokolle

- TCPIP, TLS, HTTP, HTTPS

Standardwert

12 auf mobilen Betriebssystemen, 15 auf Desktop-Betriebssystemen

Bemerkungen

Um die Komprimierung für Uploads auszuschalten, setzen Sie die Fenstergröße auf "0". Ansonsten kann der Wert für die Fenstergröße zwischen 9 und einschließlich 15 liegen. Im Allgemeinen können mit einem größeren Fenster bessere Komprimierungsraten erreicht werden. Dies erfordert aber auch mehr Speicherkapazität.

window-bits ist der Logarithmus zur Basis 2 der Fenstergröße (Größe des Verlaufspuffers). Mit den folgenden Formeln können Sie festlegen, wieviel Speicher auf dem Client für jedes *window-bits* verwendet wird:

upload (compress): $\text{memory} = 2^{(\text{window-size} + 3)}$

download (decompress): $\text{memory} = 2^{(\text{window-size})}$

Wenn Sie Ihre Anwendung direkt mit der statischen UltraLite-Laufzeitbibliothek verknüpfen, rufen Sie `ULEnableZlibSyncCompression(sqlca)` auf, um die zlib-Funktionen direkt in Ihre Anwendung einzubinden. Andernfalls muss *mlczlib12.dll* bereitgestellt werden.

Hinweise zum Einstellen der Netzwerkprotokolloptionen mit dbmlsync finden Sie unter „Erweiterte Option CommunicationAddress (adr)“ auf Seite 149.

Hinweise zum Einstellen der Netzwerkprotokolloptionen mit UltraLite finden Sie unter „UltraLite-Netzwerkprotokolloptionen für dbmlsync“ [[UltraLite - Datenbankverwaltung](#)].

Siehe auch

- „compression“ auf Seite 36
- „zlib_download_window_size“ auf Seite 60

Beispiel

Die folgende Option steuert die Komprimierung für den Download:

```
"compression=zlib;zlib_upload_window_size=0"
```

Schemaänderungen in entfernten MobiLink-Clients

Wenn Ihre Anforderungen sich ändern, können bereitgestellte entfernte Datenbanken Schemaänderungen erforderlich machen. Die häufigsten Schemaänderungen sind das Einfügen neuer Spalten in eine vorhandene Tabelle oder das Hinzufügen einer neuen Tabelle zur Datenbank.

In früheren Versionen erforderten Schemaänderungen, die sich auf die Synchronisation auswirken, eine erfolgreiche Synchronisation unmittelbar vor der Schemaänderung. Dies ist nun nicht mehr erforderlich. Dafür müssen Sie die neue SQL-Syntax zum Speichern der Skriptversion in der Synchronisationssubskription statt der erweiterten Option ScriptVersion verwenden.

Die SQL-Syntax zur Unterstützung dieser Funktion hat folgenden Aufbau:

- **CREATE SYNCHRONIZATION SUBSCRIPTION-Anweisung** Verwenden Sie die SCRIPT VERSION-Klausel zum Festlegen der Skriptversion, die bei der Synchronisation verwendet werden soll.

- **ALTER SYNCHRONIZATION SUBSCRIPTION-Anweisung** Verwenden Sie die SET SCRIPT VERSION-Klausel zum Festlegen der Skriptversion, die bei der Synchronisation verwendet werden soll.

Siehe auch

- „CREATE SYNCHRONIZATION SUBSCRIPTION-Anweisung [MobiLink]“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- „ALTER SYNCHRONIZATION SUBSCRIPTION-Anweisung [MobiLink]“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]

Skriptversionen und Subskriptionen

Ab Version 12.0.0 ist die Durchführung von Schemaänderungen an entfernten Datenbanken bedeutend einfacher. Um diese Funktionalität nutzen zu können, müssen Sie die Verwendung der erweiterten dbmlsync-Option ScriptVersion beenden. Stattdessen sollten Sie Ihre Skriptversion direkt der Synchronisationssubskription zuordnen und dafür die Klauseln verwenden, die den Anweisungen CREATE SYNCHRONIZATION SUBSCRIPTION und ALTER SYNCHRONIZATION SUBSCRIPTION hinzugefügt wurden.

Wenn Sie die neue Syntax verwenden, wird jede Datenbanktransaktion mit der Skriptversion hochgeladen, die der Subskription zum Zeitpunkt der Transaktion zugeordnet wurde. Dadurch kann eine Schemaänderung, die eine Skriptversionsänderung erfordert, ohne Synchronisation durchgeführt werden.

Bei Verwendung der zuvor verwendeten erweiterten Option ScriptVersion wird die Skriptversion der Transaktion während der Synchronisation zugeordnet. Aus diesem Grund müssen Sie vor jeder Schemaänderung eine Synchronisation durchführen.

Bei einigen bestehenden Synchronisationssystemen besteht aus anderen Gründen als Schemaänderungen weiterhin eine Abhängigkeit von der Änderung der von einer Subskription verwendeten Skriptversion zwischen den Synchronisationen. Möglicherweise ist es nicht möglich, diese Systeme so zu aktualisieren, dass sie die neue Funktionalität nutzen können.

Es ist weiterhin immer empfehlenswert, bei der Erstellung einer Synchronisationssubskription die SCRIPT VERSION-Klausel anzugeben. Vorhandene Subskriptionen können anhand der Anweisungen im Beispiel aktualisiert werden.

Beispiel: Skriptversionen Subskriptionen zuordnen

Wenn eine Subskription mit dem Namen **my_sub** vorhanden ist, die Sie mit der erweiterten dbmlsync-Option ScriptVersion synchronisieren, finden Sie nachstehend die Schritte, mit denen Sie Ihre Skriptversion direkt Ihrer Subskription zuordnen können.

1. Ermitteln Sie die Skriptversion, die derzeit zum Synchronisieren von **my_sub** verwendet wird. Die einfachste Möglichkeit hierzu ist folgende:
 - a. Fügen Sie die Option -v+ in Ihrer vorhandenen dbmlsync-Befehlszeile hinzu und führen Sie eine Synchronisation durch.

- b. Suchen Sie in Ihrer dbmlsync-Ausgabedatei nach einer Zeile, die die verwendete Skriptversion ausweist. Suchen Sie nach einem Ausdruck, der folgendem Beispiel ähnelt:

```
Script version: my_script_ver_1
```

2. Ordnen Sie die aktuelle Skriptversion der Subskription zu:

```
ALTER SYNCHRONIZATION SUBSCRIPTION <sub_name>  
SET SCRIPT VERSION = <ver>
```

wobei <sub_name> der Name der Subskription ist, in diesem Fall **my_sub**, und <ver> die aktuelle Skriptversion, die in Schritt 1 ermittelt wurde.

Alle Transaktionen, die nach diesem Zeitpunkt stattfinden, werden der Skriptversion zugeordnet.

3. Führen Sie ein letztes Mal eine Synchronisation mit den alten Optionen durch. Damit stellen Sie sicher, dass alle Transaktionen, die vor Abschluss von Schritt 2 stattgefunden haben, mit der korrekten Skriptversion hochgeladen werden.
4. Entfernen Sie die erweiterte Option ScriptVersion überall, wo sie für diese Subskription angegeben ist. Die erweiterte Option kann in der dbmlsync-Befehlszeile, in einem Synchronisationsprofil oder in Verbindung mit einer Subskription, Publikation oder einem MobiLink-Benutzer in der entfernten Datenbank angegeben sein.

Bei Datenbanken, die mehr als eine Subskription enthalten, wiederholen Sie die Prozedur für jede Subskription.

Tabellen zu bereitgestellten entfernten SQL Anywhere-Datenbanken hinzufügen

Sie können entfernten SQL Anywhere-Datenbanken nach ihrem Deployment Tabellen hinzufügen.

Voraussetzungen

Sie müssen Eigentümer der Publikation sein oder eine der folgenden Berechtigungen haben:

- ALTER-Privileg für die Publikation
- SYS_REPLICATION_ADMIN_ROLE-Systemrolle

Kontext und Bemerkungen

Hinweis

Wenn Sie sich sicher sein können, dass keine anderen Verbindungen zu der entfernten Datenbank existieren, können Sie die Anweisung ALTER PUBLICATION manuell benutzen, um neue oder geänderte Tabellen Ihren Publikationen hinzuzufügen. Sonst müssen Sie den Hook `sp_hook_dbmlsync_schema_upgrade` verwenden, um ein Upgrade Ihres Schemas vorzunehmen.

Aufgabe

1. Fügen Sie die zugeordneten Tabellenskripten in der konsolidierten Datenbank hinzu.

Für die entfernte Datenbank ohne die neue Tabelle und die entfernte Datenbank mit der neuen Tabelle kann dieselbe Skriptversion verwendet werden. Wenn jedoch das Vorhandensein der neuen Tabelle eine Änderung der Prozedur bewirkt, wie vorhandene Tabellen synchronisiert werden, müssen Sie eine neue Skriptversion erstellen und für alle Tabellen, die mit der neuen Skriptversion synchronisiert werden, neue Skripten anlegen.

2. Führen Sie eine normale Synchronisation aus. Vergewissern Sie sich, dass die Synchronisation erfolgreich verlaufen ist, bevor Sie fortsetzen.
3. Fügen Sie die Tabelle mit der Anweisung ALTER PUBLICATION hinzu. Beispiel:

```
ALTER PUBLICATION your_pub  
ADD TABLE table_name;
```

Sie können diese Anweisung in einem sp_hook_dbmsync_schema_upgrade-Hook verwenden.

4. Führen Sie die Synchronisation durch.

Ergebnisse

Die Tabellen werden zur entfernten Datenbank hinzugefügt.

Nächste Schritte

Verwenden Sie die neue Skriptversion, falls erforderlich.

Siehe auch

- „sp_hook_dbmsync_schema_upgrade“ auf Seite 249
- „Skriptversionen“ [*MobiLink - Serveradministration*]
- „ALTER PUBLICATION-Anweisung [MobiLink] [SQL Remote]“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- „ALTER TABLE-Anweisung“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]

Publizierte Tabellen in bereitgestellten entfernten SQL Anywhere-Datenbanken ändern

Sie können eine Tabellendefinition in einer bereitgestellten entfernten SQL Anywhere-Datenbank ändern.

Voraussetzungen

Sie müssen Eigentümer der Publikation sein oder eine der folgenden Berechtigungen haben:

- ALTER-Privileg für die Publikation
- SYS_REPLICATION_ADMIN_ROLE-Systemrolle

Kontext und Bemerkungen

Wenn ein MobiLink-Client mit einem neuen Schema synchronisiert, erwartet er Skripten wie etwa `upload_update` oder `download_cursor`, die Parameter für alle Spalten in der entfernten Tabelle enthalten. Ältere entfernte Datenbanken erwarten Skripten, die nur die ursprünglichen Spalten enthalten.

Aufgabe

1. Erstellen Sie in der konsolidierten Datenbank eine neue Skriptversion.
2. Erstellen Sie für Ihre neue Skriptversion Skripten für alle Tabellen in den Publikationen, die die zu ändernde Tabelle enthalten und mit der alten Skriptversion synchronisiert werden.
3. Führen Sie eine normale Synchronisation der entfernten Datenbank mit der alten Skriptversion aus. Vergewissern Sie sich, dass die Synchronisation erfolgreich verlaufen ist, bevor Sie fortsetzen.
4. Verwenden Sie die Anweisung `ALTER PUBLICATION` in der entfernten Datenbank, um die Tabelle temporär aus der Publikation zu löschen. Beispiel:

```
ALTER PUBLICATION your_pub  
DROP TABLE table_name;
```

Sie können diese Anweisung in einem `sp_hook_dbmlsync_schema_upgrade`-Hook verwenden.

5. Verwenden Sie die Anweisung `ALTER TABLE` in der entfernten Datenbank, um die Tabelle zu ändern.
6. Verwenden Sie die Anweisung `ALTER PUBLICATION` in der entfernten Datenbank, um die Tabelle wieder in die Publikation aufzunehmen.

Sie können diese Anweisung in einem `sp_hook_dbmlsync_schema_upgrade`-Hook verwenden.

7. Synchronisieren Sie mit der neuen Skriptversion.

Ergebnisse

Die publizierte Tabelle wird geändert.

Siehe auch

- „`ALTER PUBLICATION`-Anweisung [MobiLink] [SQL Remote]“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- „`ALTER TABLE`-Anweisung“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- „`sp_hook_dbmlsync_schema_upgrade`“ auf Seite 249

Schema-Upgrades für entfernte UltraLite-Datenbanken

Sie können das Schema einer entfernten UltraLite-Datenbank ändern, indem Sie Ihre bestehende Anwendung DDL ausführen lassen.

- Wenn Sie das Deployment einer neuen Anwendung mit einer neuen Datenbank durchführen, müssen Sie die UltraLite-Datenbank erneut mit Daten füllen, indem Sie mit dem MobiLink-Server synchronisieren.
- Wenn Sie das Deployment einer neuen Anwendung mit DDL für die Aktualisierung der Datenbank durchführen, bleiben Ihre Daten erhalten.
- Wenn die bestehende Anwendung DDL-Anweisungen auf generischem Weg erhält, wird die DDL auf Ihre Datenbank angewendet und die Daten bleiben erhalten.

Es ist gewöhnlich nicht zu empfehlen, dass alle Benutzer gleichzeitig ein Upgrade auf die neue Version der Anwendung vornehmen. Sie müssen daher in der Lage sein, beide Versionen gleichzeitig in der Produktionsumgebung laufen zu lassen und mit einer einzigen konsolidierten Datenbank zu synchronisieren. Sie können zwei oder mehr Versionen der Synchronisationsskripten erstellen, die in der konsolidierten Datenbank gespeichert sind und die Aktionen des MobiLink-Servers steuern. Jede Version der Anwendung kann dann die entsprechenden Synchronisationsskripten auswählen, indem sie zu Beginn der Synchronisation den korrekten Versionsnamen angibt.

Weitere Hinweise zu UltraLite DDL finden Sie unter „[UltraLite-SQL-Anweisungen](#)“ [[UltraLite - Datenbankverwaltung](#)].

Siehe auch

- „[Deployment von UltraLite-Datenbankschema-Upgrades](#)“ [[UltraLite - Datenbankverwaltung](#)]

SQL Anywhere-Clients für MobiLink

In diesem Abschnitt wird beschrieben, wie Sie SQL Anywhere-Clients für die MobiLink-Synchronisation einrichten und ausführen.

SQL Anywhere-Clients

In den folgenden Abschnitten wird die Verwendung von SQL Anywhere-Clients in Verbindung mit MobiLink erläutert.

SQL Anywhere-Datenbanken als entfernte Datenbanken verwenden

Jede SQL Anywhere-Datenbank kann in einem MobiLink-System als entfernte Datenbank verwendet werden. Sie müssen eine Publikation, einen MobiLink-Benutzer und eine Synchronisationssubskription erstellen sowie den Benutzer bei der konsolidierten Datenbank registrieren.

Voraussetzungen

Es gibt keine Voraussetzungen für das Ausführen dieser Aufgabe.

Kontext und Bemerkungen

Wenn Sie den **Assistenten zum Erstellen eines Synchronisationsmodells** verwenden, um die MobiLink-Clientanwendung zu erstellen, werden diese Objekte erstellt, wenn Sie das Deployment für das Modell ausführen.

Hinweis

Eine Datenbank ohne Transaktionslog kann nur als entfernte Datenbank für skriptgesteuerte Uploads und für reine Downloadpublikationen verwendet werden.

Aufgabe

1. Starten Sie mit einer vorhandenen SQL Anywhere-Datenbank oder erstellen Sie eine neue Datenbank und fügen Sie Ihre Tabellen hinzu.
2. Erstellen Sie eine oder mehrere Publikationen in der entfernten Datenbank.

Siehe [„Publikationen“ auf Seite 74](#).

3. Erstellen Sie in der entfernten Datenbank MobiLink-Benutzer

Siehe [„MobiLink-Benutzer“ auf Seite 82](#).

4. Registrieren Sie die Benutzer in der konsolidierten Datenbank.

Siehe [MobiLink-Benutzernamen zur konsolidierten Datenbank hinzufügen auf Seite 5](#).

5. Erstellen von Synchronisationsskripten in der entfernten Datenbank.

Siehe [„Erstellen von Synchronisationssubskriptionen“ auf Seite 86](#).

Ergebnisse

Die SQL Anywhere-Datenbank wird als entfernte Datenbank eingerichtet.

Siehe auch

- „Skriptgesteuerter Upload“ auf Seite 323
- „Reine Download-Publikationen“ auf Seite 78

Entfernte MobiLink-Datenbanken durch Anpassen eines Prototyps bereitstellen

Für das Deployment entfernter SQL Anywhere-Datenbanken legen Sie die Datenbanken an und fügen die betreffenden Publikationen hinzu. Hierzu können Sie einen Prototypen einer entfernten Datenbank anpassen.

Voraussetzungen

Es gibt keine Voraussetzungen für das Ausführen dieser Aufgabe.

Kontext und Bemerkungen

Wenn Sie das Deployment einer Ausgangsdatenbank an mehreren Standorten durchführen, ist es am sichersten, beim Deployment Datenbanken zu verwenden, welche die entfernte ID NULL haben. Wenn Sie die Datenbanken synchronisiert haben, um sie vorab mit Daten zu füllen, können Sie die entfernte ID vor dem Deployment wieder auf NULL zurücksetzen. Auf diese Weise stellen Sie sicher, dass eine eindeutige entfernte ID verwendet wird, da bei der ersten Synchronisation der entfernten Datenbank eine eindeutige entfernte ID zugewiesen wird. Die entfernte ID kann auch beim Einrichten der entfernten Datenbank festgelegt werden, sie muss jedoch eindeutig sein.

Wenn Sie den **Assistenten zum Erstellen eines Synchronisationsmodells** verwenden, um Ihre MobiLink-Clientanwendung zu erstellen, können Sie das Deployment der Datenbank mithilfe eines Assistenten ausführen.

Aufgabe

1. Erstellen Sie eine entfernte Prototypdatenbank.

Die Prototypdatenbank muss alle erforderlichen Tabellen und Publikationen, jedoch nicht die Daten der einzelnen Datenbanken enthalten. Diese Daten umfassen normalerweise Folgendes:

- Der MobiLink-Benutzername
- Synchronisationssubskriptionen.

- Die Option `global_database_id`, die den Ausgangspunkt für globale Autoinkrement-Schlüsselwerte liefert
2. Führen Sie für jede entfernte Datenbank die folgenden Vorgänge aus:
 - a. Erstellen Sie ein Verzeichnis für die entfernte Datenbank.
 - b. Kopieren Sie die entfernte Prototypdatenbank in das Verzeichnis.
 - c. Falls sich das Transaktionslog in demselben Verzeichnis befindet wie die entfernte Datenbank, braucht der Dateiname des Transaktionslogs nicht geändert zu werden.
 - d. Führen Sie ein SQL-Skript aus, das die individuellen Daten in die Datenbank einfügt.
Bei dem SQL-Skript kann es sich um ein Skript mit Parametern handeln.

Ergebnisse

Das Deployment der entfernten Datenbanken wird durchgeführt.

Beispiel

Das folgende SQL-Skript stammt aus dem Beispiel Contact. Es befindet sich in `%SQLANYWAMP16%\MobiLink\Contact\customize.sql`.

```
PARAMETERS ml_userid, db_id;
go
SET OPTION PUBLIC.global_database_id = {db_id}
go
CREATE SYNCHRONIZATION USER {ml_userid}
      TYPE 'TCPIP'
      ADDRESS 'host=localhost;port=2439'
go
CREATE SYNCHRONIZATION SUBSCRIPTION TO "DBA"."Product"
      FOR {ml_userid}
go
CREATE SYNCHRONIZATION SUBSCRIPTION TO "DBA"."Contact"
      FOR {ml_userid}
go
commit work
go
```

Mit dem folgenden Befehl wird das Skript für eine entfernte Datenbank mit der Datenquelle `dsn_remote_1` ausgeführt:

```
dbisql -c "DSN=dsn_remote_1" read customize.sql [SSinger] [2]
```

Siehe auch

- „Einstellungen für die entfernte ID“ auf Seite 72
- „Deployment des Synchronisationsmodells“ [*MobiLink - Erste Orientierung*]
- „Bereitstellung von SQL Anywhere MobiLink-Clients“ [*MobiLink - Serveradministration*]
- Erste Synchronisation erfolgreich auf Seite 73
- „PARAMETERS-Anweisung [Interactive SQL]“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- „SQL-Skriptdateien“ [*SQL Anywhere Server - SQL-Benutzerhandbuch*]

Einstellungen für die entfernte ID

Die entfernte ID identifiziert eine entfernte Datenbank in einem MobiLink-Synchronisationssystem eindeutig. Beim Erstellen einer SQL Anywhere-Datenbank wird die entfernte ID auf NULL gesetzt. Wenn die Datenbank mit MobiLink synchronisiert wird, sucht MobiLink nach der entfernten ID NULL. Falls diese gefunden wird, weist MobiLink eine GUID als entfernte ID zu. Einmal gesetzt, behält die Datenbank die entfernte ID bei, solange diese nicht manuell geändert wird.

Wenn Sie entfernte IDs beispielsweise in MobiLink-Ereignisskripten referenzieren, können Sie der entfernten ID einen beschreibenden Namen zuweisen. Hierfür stellen Sie die Datenbankoption `ml_remote_id` für die entfernte Datenbank ein. Bei der Option `ml_remote_id` handelt es sich um eine benutzerdefinierte Option, die in der Systemtabelle `SYSOPTION` gespeichert wird. Sie können sie mithilfe der Anweisung `SET OPTION` oder des SQL Anywhere 16-Plug-Ins in Sybase Central ändern.

Die entfernte ID muss innerhalb des Synchronisationssystems eindeutig sein.

Wenn Sie die entfernte ID manuell festlegen und danach die entfernte Datenbank neu erstellen, müssen Sie entweder der neu erstellten entfernten Datenbank einen anderen Namen geben oder die Prozedur `ml_reset_sync_state` verwenden, um die Statusinformationen für die entfernte Datenbank in der konsolidierten Datenbank zurückzusetzen.

Vorsicht

In den meisten Fällen müssen Sie weder die entfernte ID festlegen noch deren Wert kennen. Falls Sie die entfernte ID jedoch ändern müssen, sollten Sie dies vorzugsweise vor der ersten Synchronisation tun. Falls Sie die entfernte ID später ändern, müssen Sie direkt vor dieser Änderung eine vollständige, erfolgreiche Synchronisation durchgeführt haben. Andernfalls könnten Sie Daten verlieren und die Datenbank könnte inkonsistent werden.

Siehe auch

- „`ml_remote_id`-Option“ [[SQL Anywhere Server - Datenbankadministration](#)]
- „`ml_reset_sync_state`-Systemprozedur“ [[MobiLink - Serveradministration](#)]
- „`SET OPTION`-Anweisung“ [[SQL Anywhere Server - SQL-Referenzhandbuch](#)]
- „Datenbankoptionen mit der `SET OPTION`-Anweisung setzen“ [[SQL Anywhere Server - Datenbankadministration](#)]
- „`SYSOPTION`-Systemansicht“ [[SQL Anywhere Server - SQL-Referenzhandbuch](#)]
- „Entfernte IDs“ auf Seite 9

Beispiel

Die folgende SQL-Anweisung setzt die entfernte ID auf den Wert 'HR001':

```
SET OPTION PUBLIC.ml_remote_id = 'HR001'
```

Upgrades von entfernten Datenbanken

Wenn Sie eine neue entfernte SQL Anywhere-Datenbank über eine ältere Version installieren, sind die Angaben über den Synchronisationsverlauf in der konsolidierten Datenbank falsch. Sie können diesen Fehler beheben, indem Sie mit der gespeicherten Prozedur `ml_reset_sync_state` die Statusinformationen für die entfernte Datenbank in der konsolidierten Datenbank zurücksetzen.

Siehe auch

- „ml_reset_sync_state-Systemprozedur“ [*MobiLink - Serveradministration*]
- „Upgrades von SQL Anywhere MobiLink-Clients“ [*SQL Anywhere 16 - Änderungen und Upgrades*]

Offsets für den Verarbeitungsfortschritt

Der Offset für den Verarbeitungsfortschritt ist ein Ganzzahlwert, der den Zeitpunkt angibt, bis zu dem das Upload und die Bestätigung aller Vorgänge für die Subskription erfolgt sind. Das Dienstprogramm dbmlsync verwendet den Offset, um zu entscheiden, welche Daten hochgeladen werden sollen. In der entfernten Datenbank wird der Offset in der progress-Spalte der Systemtabelle SYS.ISYSSYNC gespeichert. In der konsolidierten Datenbank wird der Offset in der progress-Spalte der Tabelle ml_subscription gespeichert.

Für jede entfernte Datenbank merken sich die entfernte und die konsolidierte Datenbank einen Offset für jede Subskription. Wenn ein MobiLink-Benutzer synchronisiert, werden die Offsets bei allen Subskriptionen bestätigt, die dem MobiLink-Benutzer zugeordnet sind, auch wenn sie zu diesem Zeitpunkt nicht synchronisiert werden. Dies ist erforderlich, da mehrere Publikationen dieselben Daten enthalten können. Die einzige Ausnahme ist, dass dbmlsync den Offset für den Verarbeitungsfortschritt erst überprüft, wenn ein Upload versucht wurde.

Wenn eine Unvereinbarkeit zwischen den Offsets der entfernten und der konsolidierten Datenbank besteht, gilt als Standardverhalten, dass die Offsets in der entfernten Datenbank durch die Werte der konsolidierten Datenbank ersetzt werden und ab diesen Offsets ein neuer Upload erfolgt. In der Regel ist diese Standardvorgabe ausreichend. So ist zum Beispiel dieser Vorgang in der Regel geeignet, wenn die konsolidierte Datenbank aus einer Sicherung wiederhergestellt wird und das Transaktionslog der entfernten Datenbank intakt ist oder wenn ein Upload erfolgreich verläuft, durch einen Kommunikationsfehler die Uploadbestätigung aber nicht abgesendet werden kann.

Die meisten Konflikte bei Offsets für den Verarbeitungsfortschritt werden automatisch gelöst, indem die Verarbeitungsfortschrittswerte der konsolidierten Datenbank verwendet werden. In den seltenen Fällen, in denen Sie eingreifen müssen, um ein Problem mit Offsets für den Verarbeitungsfortschritt zu beheben, können Sie die Option -r von dbmlsync verwenden.

Erste Synchronisation erfolgreich

Wenn Sie eine neu erstellte Subskription zum ersten Mal synchronisieren wollen, werden die Offsets für den Verarbeitungsfortschritt nicht mit denen der konsolidierten Datenbank abgeglichen. Dieses Merkmal ermöglicht die Neuerstellung und Synchronisation einer entfernten Datenbank, ohne dass ihre Statusinformationen, die in der konsolidierten Datenbank gespeichert sind, gelöscht werden müssen.

Das Dienstprogramm dbmlsync erkennt eine erste Synchronisation daran, dass die Spalten in der Systemtabelle SYS.ISYSSYNC der entfernten Datenbank folgendermaßen aussehen: Der Wert für die **progress**-Spalte ist derselbe wie für die **created**-Spalte, und der Wert für die **log_sent**-Spalte ist NULL.

Wenn Sie allerdings in demselben Upload zwei oder mehr Subskriptionen synchronisieren und eine der Subskriptionen nicht zum ersten Mal synchronisiert wird, werden Offsets für den Verarbeitungsfortschritt bei allen synchronisierten Subskriptionen überprüft, auch bei denen, die zum ersten Mal synchronisiert werden. Beispiel: Wenn Sie die Option dbmlsync -s mit zwei Subskriptionen (-s pub1,pub2) verwenden

und sub1 vorher synchronisiert wurde, aber sub2 nicht, werden die Offsets für den Verarbeitungsfortschritt beider Subskriptionen mit den Werten der konsolidierten Datenbank abgeglichen.

Siehe auch

- „dbmlsync-Option -r“ auf Seite 132
- „ISYSSYNC-Systemtabelle“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- „Transaktionslogdateien“ auf Seite 93

Publikationen

Eine Publikation ist ein Datenbankobjekt, das die zu synchronisierenden Daten identifiziert. Sie definiert die zu heraufzuladenden Daten und begrenzt die Tabellen, in die Änderungen heruntergeladen werden können. (Der Download wird durch das download_cursor-Skript definiert.)

Eine Publikation besteht aus einem oder mehreren Artikeln. Jeder Artikel definiert eine Teilmenge einer Tabelle, die auf diese Weise synchronisiert wird. Die Teilmenge kann die ganze Tabelle oder einen Teil ihrer Zeilen bzw. Spalten umfassen. Jeder Artikel in einer Publikation muss eine andere Tabelle referenzieren.

Sie erstellen eine Subskription, um eine Publikation mit einem Benutzer zu verknüpfen.

Sie können Publikationen mithilfe von Sybase Central oder mit der Anweisung CREATE PUBLICATION erstellen.

In Sybase Central erscheinen alle Publikationen und Artikel im Ordner **Publikationen**.

Standardmäßig werden Trigger-Aktionen in einem SQL Anywhere-Client nicht mit dem MobiLink-Server synchronisiert, unter der Annahme, dass identische Trigger in der Backend-Datenbank vorhanden sind, mit der die Daten synchronisiert werden. Weitere Hinweise zu diesem Verhalten finden Sie unter „Erweiterte Option SendTriggers (st)“ auf Seite 169.

Hinweise zu Publikationen

- Die SYS_REPLICATION_ADMIN_ROLE-Systemrolle ist zum Erstellen und Löschen von Publikationen erforderlich.
- Sie können nicht zwei Publikationen mit unterschiedlichen Spalten-Teilmengen für dieselbe Tabelle erstellen.
- Die Publikation legt fest, welche Spalten ausgewählt werden, sie legt aber nicht die Reihenfolge ihres Versendens fest. Spalten werden immer in der Reihenfolge gesendet, in der sie in der CREATE TABLE-Anweisung definiert wurden.
- Jeder Artikel muss alle Spalten im Primärschlüssel der Tabelle enthalten, die er referenziert.
- Ein Artikel kann die synchronisierbaren Spalten einer Tabelle begrenzen. Mit einer WHERE-Klausel können auch die Zeilen begrenzt werden.
- Ansichten und gespeicherte Prozeduren können nicht in Publikationen einbezogen werden.

- Publikationen und Subskriptionen werden auch von der nachrichtenbasierten Replikationstechnologie SQL Remote benutzt. SQL Remote erfordert Publikationen und Subskriptionen sowohl in der konsolidierten als auch in der entfernten Datenbank. Im Gegensatz dazu sind MobiLink-Publikationen nur in entfernten SQL Anywhere-Datenbanken erforderlich. Konsolidierte MobiLink-Datenbanken werden mithilfe von Synchronisationsskripten konfiguriert.

Siehe auch

- „CREATE PUBLICATION-Anweisung [MobiLink] [SQL Remote]“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- „Erweiterte Option SendTriggers (st)“ auf Seite 169

Ganze Tabellen publizieren

Die einfachste Publikation, die Sie erstellen können, besteht aus einer Gruppe von Artikeln, von denen jede alle Zeilen und Spalten einer Tabelle enthält.

Voraussetzungen

Sie müssen die SYS_REPLICATION_ADMIN_ROLE-Systemrolle haben.

Die zu publizierenden Tabellen müssen bereits vorhanden sein.

Aufgabe

1. Stellen Sie eine Verbindung zur entfernten Datenbank mithilfe des SQL Anywhere 16-Plug-Ins her.
2. Doppelklicken Sie auf **Publikationen**.
3. Klicken Sie auf **Datei**→**Neu**→**Publikation**.
4. Geben Sie in das Feld **Wie lautet der Name der neuen Publikation?** einen Namen für die neue Publikation ein. Klicken Sie auf **Weiter**.
5. Klicken Sie auf **Weiter**.
6. In der Liste **Verfügbare Tabellen** wählen Sie eine Tabelle aus. Klicken Sie auf **Hinzufügen**.
7. Klicken Sie auf **Fertig stellen**.

Ergebnisse

Eine Publikation wird erstellt.

Siehe auch

- „CREATE PUBLICATION-Anweisung [MobiLink] [SQL Remote]“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]

Nur einige Spalten einer Tabelle publizieren

Sie können eine Publikation erstellen, die alle Zeilen, aber nur einige der Spalten einer Tabelle enthält.

Voraussetzungen

Eine entfernte Datenbank ist vorhanden und Sie haben die SYS_REPLICATION_ADMIN_ROLE-Systemrolle.

Kontext und Bemerkungen

Hinweis

- Wenn Sie zwei Publikationen mit derselben Tabelle, jedoch mit unterschiedlichen Spalten publizieren, können Sie nur für eine davon eine Synchronisationssubskription erstellen.
- Ein Artikel muss alle Primärschlüsselspalten in der Tabelle enthalten.

Aufgabe

1. Stellen Sie eine Verbindung zur entfernten Datenbank mithilfe des SQL Anywhere 16-Plug-Ins her.
2. Doppelklicken Sie auf **Publikationen**.
3. Klicken Sie auf **Datei→Neu→Publikation**.
4. Geben Sie in das Feld **Wie lautet der Name der neuen Publikation?** einen Namen für die neue Publikation ein. Klicken Sie auf **Weiter**.
5. Klicken Sie auf **Weiter**.
6. In der Liste **Verfügbare Tabellen** wählen Sie eine Tabelle aus. Klicken Sie auf **Hinzufügen**.
7. Klicken Sie auf **Weiter**.
8. In der Liste **Verfügbare Spalten** erweitern Sie die Liste der verfügbaren Spalten. Wählen Sie eine Spalte aus und klicken Sie auf **Hinzufügen**.
9. Klicken Sie auf **Fertig stellen**.

Ergebnisse

Die ausgewählten Tabellenspalten werden publiziert.

Siehe auch

- „CREATE PUBLICATION-Anweisung [MobiLink] [SQL Remote]“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]

Publikationen mit WHERE-Klauseln erstellen

Wenn in einer Artikeldefinition keine WHERE-Klausel festgelegt wurde, erfasst der Upload alle geänderten Zeilen in der Tabelle. Fügen Sie WHERE-Klauseln zu Artikeln in der Publikation hinzu, um die Zeilen für den Upload auf diejenigen zu beschränken, die geändert wurden und den Suchbedingungen der WHERE-Klausel entsprechen.

Voraussetzungen

Eine entfernte Datenbank ist vorhanden und Sie haben die SYS_REPLICATION_ADMIN_ROLE-Systemrolle.

Kontext und Bemerkungen

Die Suchbedingung in der WHERE-Klausel kann nur Spalten referenzieren, die im Artikel enthalten sind. Außerdem ist es nicht möglich, folgende Elemente in der WHERE-Klausel zu verwenden:

- Unterabfragen
- Variablen
- Nicht-deterministische Funktionen

Diese Bedingungen werden nicht erzwungen, doch wenn gegen sie verstoßen wird, kann es zu unerwarteten Ergebnissen kommen. Fehler im Zusammenhang mit der WHERE-Klausel werden generiert, wenn die DML mit einer von der WHERE-Klausel referenzierten Tabelle ausgeführt wird, und nicht, wenn die Publikation definiert wird.

Aufgabe

1. Stellen Sie eine Verbindung zur entfernten Datenbank mithilfe des SQL Anywhere 16-Plug-Ins her.
2. Doppelklicken Sie auf **Publikationen**.
3. Klicken Sie auf **Datei**→**Neu**→**Publikation**.
4. Geben Sie in das Feld **Wie lautet der Name der neuen Publikation?** einen Namen für die neue Publikation ein. Klicken Sie auf **Weiter**.
5. Klicken Sie auf **Weiter**.
6. In der Liste **Verfügbare Tabellen** wählen Sie eine Tabelle aus. Klicken Sie auf **Hinzufügen**.
7. Klicken Sie auf **Weiter**.
8. Klicken Sie auf **Weiter**.
9. In der **Artikelliste** wählen Sie eine Tabelle aus und geben eine Suchbedingung in den Fensterausschnitt **Der markierte Artikel hat folgende WHERE-Klausel** ein.
10. Klicken Sie auf **Fertig stellen**.

Ergebnisse

Die neue Publikation wird erstellt.

Siehe auch

- [„CREATE PUBLICATION-Anweisung \[MobiLink\] \[SQL Remote\]“ \[SQL Anywhere Server - SQL-Referenzhandbuch\]](#)

Reine Download-Publikationen

Sie können eine Publikation für den reinen Download in entfernte Datenbanken ohne Upload von Daten aus diesen Datenbanken erstellen. Reine Download-Publikationen verwenden kein Transaktionslog beim Client.

Unterschiedliche Methoden für reinen Download

Es gibt zwei Möglichkeiten, um den Download ohne Upload zu erzwingen:

- **Reine Download-Synchronisationen** Verwenden Sie die dbmlsync-Optionen -e DownloadOnly oder -ds.
- **Reine Download-Publikationen** Erstellen Sie die Publikation mit dem Schlüsselwort FOR DOWNLOAD ONLY.

Die zwei Methoden unterscheiden sich deutlich:

Reine Download-Synchronisationen	Reine Download-Publikationen
Wenn beim Download versucht wird, Zeilen zu ändern, die in der entfernten Datenbank geändert wurden, ohne dass bisher ein Upload erfolgt ist, wird der Download abgebrochen.	Beim Download können Zeilen überschrieben werden, die in der entfernten Datenbank geändert, aber noch nicht heraufgeladen wurden.
Benutzt eine normale Publikation für den Upload oder Download. Eine reine Download-Synchronisation wird mit den dbmlsync-Befehlszeilenoptionen oder erweiterten Optionen definiert.	Benutzt eine Publikation nur für Download. Alle Synchronisationen für diese Publikationen werden nur im Download durchgeführt. Sie können eine normale Publikation nicht in eine reine Download-Publikation umwandeln.
Erfordert eine Logdatei	Erfordert keine Logdatei

Reine Download-Synchronisationen	Reine Download-Publikationen
Die Logdatei wird nicht gekürzt, wenn für diese Subskriptionen über einen längeren Zeitraum kein Upload durchgeführt wird, und kann daher viel Speicherplatz belegen.	Wenn eine Logdatei vorhanden ist, betrifft die Synchronisation nicht die Trennstelle der Synchronisation. Deshalb kann die Logdatei immer noch gekürzt werden, auch wenn die Publikation über einen längeren Zeitraum hinweg nicht synchronisiert wird. Reine Download-Publikationen beeinflussen die Logdateikürzung nicht.
Sie müssen gelegentlich einen Upload durchführen, um das Volumen des von der reinen Download-Synchronisation durchsuchten Logs zu verringern. Sonst würde die reine Download-Synchronisation zunehmend mehr Zeit brauchen.	Ein Upload ist nie erforderlich.

Siehe auch

- „CREATE PUBLICATION-Anweisung [MobiLink] [SQL Remote]“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- „Reine Upload- und reine Download-Synchronisation“ [*MobiLink - Serveradministration*]

Eigenschaften von vorhandenen Publikationen oder Artikeln ändern

Nachdem Sie eine Publikation erstellt haben, können Sie sie ändern, indem Sie Artikel hinzufügen, ändern bzw. löschen oder indem Sie die Publikation umbenennen. Wenn ein Artikel geändert wird, muss die gesamte Spezifikation des geänderten Artikels eingegeben werden.

Voraussetzungen

Eine entfernte Datenbank ist vorhanden und Sie haben die SYS_REPLICATION_ADMIN_ROLE-Systemrolle.

Publikationen können nur vom DBA oder vom Publikationseigentümer geändert werden.

Kontext und Bemerkungen

Gehen Sie hierbei mit großer Sorgfalt vor. In einem laufenden MobiLink-Setup kann das Ändern von Publikationen zu Fehlern und Datenverlust führen. Wenn die von Ihnen geänderte Publikation Subskriptionen enthält, müssen Sie diese Änderung als Schema-Upgrade behandeln. Siehe „Schemaänderungen in entfernten MobiLink-Clients“ auf Seite 62.

Aufgabe

1. Stellen Sie eine Verbindung zur entfernten Datenbank her.

2. Im linken Fensterausschnitt klicken Sie auf die Publikation oder den Artikel. Die Eigenschaften erscheinen im rechten Fensterausschnitt.
3. Konfigurieren Sie die Eigenschaften.

Ergebnisse

Die Publikation oder der Artikel wird geändert.

Artikel hinzufügen

Artikel können zu einer Publikation hinzugefügt werden, nachdem sie erstellt wurde.

Voraussetzungen

Eine entfernte Datenbank ist vorhanden und Sie haben die SYS_REPLICATION_ADMIN_ROLE-Systemrolle.

Publikationen können nur vom DBA oder vom Publikationseigentümer geändert werden.

Aufgabe

1. Stellen Sie eine Verbindung zur entfernten Datenbank mithilfe des SQL Anywhere 16-Plug-Ins her.
2. Doppelklicken Sie auf **Publikationen**.
3. Wählen Sie eine Publikation aus.
4. Klicken Sie auf **Datei→Neu→Artikel**.
5. Gehen Sie im **Assistenten zum Erstellen von Artikeln** folgendermaßen vor:
 - In der Liste **Welche Tabelle wollen Sie für diesen Artikel verwenden?** wählen Sie eine Tabelle aus. Klicken Sie auf **Weiter**.
 - Klicken Sie auf **Ausgewählte Spalten** und wählen Sie die Spalten aus. Klicken Sie auf **Weiter**.
 - Im Fensterausschnitt **Sie können eine WHERE-Klausel für diesen Artikel festlegen** geben Sie eine optionale WHERE-Klausel ein. Klicken Sie auf **Fertig stellen**.

Ergebnisse

Der angegebene Artikel wird zur Publikation hinzugefügt.

Artikel entfernen

Ein Artikel kann gelöscht werden, wenn er nicht mehr benötigt wird.

Voraussetzungen

Eine entfernte Datenbank ist vorhanden und Sie haben die SYS_REPLICATION_ADMIN_ROLE-Systemrolle.

Publikationen können nur vom DBA oder vom Publikationseigentümer geändert werden.

Aufgabe

1. Verbinden Sie sich über das SQL Anywhere 16-Plug-In mit der Datenbank.
2. Doppelklicken Sie auf **Publikationen**.
3. Rechtsklicken Sie auf die Publikation und wählen Sie **Löschen**.
4. Klicken Sie auf **Ja**.

Ergebnisse

Der ausgewählte Artikel wird gelöscht.

Vorhandene Publikationen mit SQL ändern

Artikel und Publikationen können geändert werden, nachdem sie erstellt wurden.

Voraussetzungen

Eine entfernte Datenbank ist vorhanden und Sie haben die SYS_REPLICATION_ADMIN_ROLE-Systemrolle.

Publikationen können nur vom DBA oder vom Publikationseigentümer geändert werden.

Aufgabe

1. Stellen Sie eine Verbindung zur entfernten Datenbank her.
2. Führen Sie eine ALTER PUBLICATION-Anweisung aus.

Ergebnisse

Die Publikation oder der Artikel wird geändert.

Beispiel

Die folgende Anweisung fügt die Tabelle 'Customers' zur pub_contact-Publikation hinzu.

```
ALTER PUBLICATION pub_contact  
ADD TABLE Customers
```

Siehe auch

- „ALTER PUBLICATION-Anweisung [MobiLink] [SQL Remote]“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]

Publikation löschen

Sie können eine Publikation löschen, wenn sie nicht mehr benötigt wird.

Voraussetzungen

Sie müssen entweder eine Benutzer-ID mit der SYS_REPLICATION_ADMIN_ROLE-Systemrolle haben oder Eigentümer der betreffenden Publikation sein, um eine Publikation löschen zu können.

Aufgabe

1. Stellen Sie eine Verbindung zur entfernten Datenbank mithilfe des SQL Anywhere 16-Plug-Ins her.
2. Doppelklicken Sie auf **Publikationen**.
3. Rechtsklicken Sie auf eine Publikation und wählen Sie **Löschen**.

Ergebnisse

Die ausgewählte Publikation wird gelöscht.

Siehe auch

- „DROP PUBLICATION-Anweisung [MobiLink] [SQL Remote]“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]

MobiLink-Benutzer

Ein MobiLink-Benutzername dient der Authentifizierung beim Anmelden am MobiLink-Server. Sie müssen MobiLink-Benutzer in der entfernten Datenbank erstellen und sie dann in der konsolidierten Datenbank registrieren.

MobiLink-Benutzer dürfen nicht mit Datenbankbenutzern verwechselt werden. Sie können einen MobiLink-Benutzernamen einrichten, der mit dem Namen eines Datenbankbenutzers übereinstimmt, aber weder MobiLink noch SQL Anywhere werden davon berührt.

Siehe auch

- „CREATE SYNCHRONIZATION USER-Anweisung [MobiLink]“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- „Erweiterte Optionen für MobiLink-Benutzer speichern“ auf Seite 84
- MobiLink-Benutzernamen zur konsolidierten Datenbank hinzufügen auf Seite 5

MobiLink-Benutzer mit Sybase Central zu einer entfernten Datenbank hinzufügen

Erstellen Sie einen MobiLink-Benutzer in der entfernten Datenbank, der zum Authentifizieren verwendet werden soll, wenn Sie sich in der konsolidierten Datenbank mit dem MobiLink-Server verbinden.

Voraussetzungen

Sie müssen eine vorhandene Datenbank haben und eine Benutzer-ID mit der SYS_REPLICATION_ADMIN_ROLE-Systemrolle.

Aufgabe

1. Verbinden Sie sich über das SQL Anywhere 16-Plug-In mit der Datenbank.
2. Doppelklicken Sie auf **MobiLink-Benutzer**.
3. Klicken Sie auf **Datei→Neu→MobiLink-Benutzer**.
4. Geben Sie in das Feld **Wie lautet der Name des neuen MobiLink-Benutzers?** einen Namen für den MobiLink-Benutzer ein.
5. Klicken Sie auf **Fertig stellen**.

Ergebnisse

Der MobiLink-Benutzer wird erstellt.

Nächste Schritte

Registrieren Sie den MobiLink-Benutzer in der konsolidierten Datenbank. Siehe [MobiLink-Benutzernamen zur konsolidierten Datenbank hinzufügen auf Seite 5](#).

MobiLink-Benutzer mithilfe von SQL zu einer entfernten Datenbank hinzufügen

Erstellen Sie einen MobiLink-Benutzer in der entfernten Datenbank, der zur Authentifizierung verwendet werden soll, wenn Sie sich in der konsolidierten Datenbank mit dem MobiLink-Server verbinden.

Voraussetzungen

Sie müssen eine vorhandene Datenbank haben und eine Benutzer-ID mit der SYS_REPLICATION_ADMIN_ROLE-Systemrolle.

Aufgabe

1. Stellen Sie eine Verbindung mit der Datenbank her.

2. Führen Sie die Anweisung `CREATE SYNCHRONIZATION USER` aus. Der MobiLink-Benutzername kennzeichnet eindeutig eine entfernte Datenbank und muss daher auch innerhalb Ihres Synchronisationssystems eindeutig sein.

In der Anweisung `CREATE SYNCHRONIZATION USER` können Sie Eigenschaften für den MobiLink-Benutzer angeben oder mit der Anweisung `ALTER SYNCHRONIZATION USER` separat festlegen.

Ergebnisse

Der MobiLink-Benutzer wird erstellt.

Nächste Schritte

Registrieren Sie den MobiLink-Benutzer in der konsolidierten Datenbank. Siehe [MobiLink-Benutzernamen zur konsolidierten Datenbank hinzufügen auf Seite 5](#).

Beispiel

Mit dem folgenden Beispiel wird ein MobiLink-Benutzer mit dem Namen "SSinger" hinzugefügt:

```
CREATE SYNCHRONIZATION USER SSinger
```

Siehe auch

- „`CREATE SYNCHRONIZATION USER`-Anweisung [MobiLink]“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- „`ALTER SYNCHRONIZATION USER`-Anweisung [MobiLink]“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]

Erweiterte Optionen für MobiLink-Benutzer speichern

Sie können mithilfe erweiterter Optionen die Optionen für jeden MobiLink-Benutzer in der entfernten Datenbank definieren. Erweiterte Optionen können in der Befehlszeile eingegeben, in der Datenbank gespeichert oder mit der Hook-Prozedur `sp_hook_dbmlsync_set_extended_options` angegeben werden.

Voraussetzungen

Sie müssen eine vorhandene Datenbank haben und eine Benutzer-ID mit der `SYS_REPLICATION_ADMIN_ROLE`-Systemrolle.

Kontext und Bemerkungen

Sie können das Verhalten einer bevorstehenden Synchronisation programmatisch anpassen.

Weitere Hinweise finden Sie unter „`sp_hook_dbmlsync_set_extended_options`“ auf Seite 252.

Aufgabe

1. Doppelklicken Sie auf **Benutzer**.

2. Doppelklicken Sie auf den MobiLink-Benutzernamen und wählen Sie **Eigenschaften**.
3. Ändern Sie die Eigenschaften je nach Bedarf.

Ergebnisse

Die angegebenen Optionen für den MobiLink-Benutzer werden gespeichert.

Siehe auch

- „Erweiterte Optionen von MobiLink SQL Anywhere-Clients“ auf Seite 143
- „Erweiterte Optionen für dbmlsync“ auf Seite 92
- „ALTER SYNCHRONIZATION USER-Anweisung [MobiLink]“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- „CREATE SYNCHRONIZATION USER-Anweisung [MobiLink]“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]

Erweiterte Optionen für MobiLink-Benutzer mit SQL speichern

Sie können mithilfe erweiterter Optionen die Optionen für jeden MobiLink-Benutzer in der entfernten Datenbank definieren. Erweiterte Optionen können in der Befehlszeile eingegeben, in der Datenbank gespeichert oder mit der Hook-Prozedur `sp_hook_dbmlsync_set_extended_options` angegeben werden.

Voraussetzungen

Sie müssen eine vorhandene Datenbank haben und eine Benutzer-ID mit der `SYS_REPLICATION_ADMIN_ROLE`-Systemrolle.

Kontext und Bemerkungen

Sie können das Verhalten einer bevorstehenden Synchronisation programmatisch anpassen.

Weitere Hinweise finden Sie unter „`sp_hook_dbmlsync_set_extended_options`“ auf Seite 252.

Aufgabe

1. Stellen Sie eine Verbindung mit der Datenbank her.
2. Führen Sie die Anweisung `ALTER SYNCHRONIZATION USER` aus.

Sie können auch Eigenschaften angeben, wenn Sie den MobiLink-Benutzernamen erstellen.

Ergebnisse

Ergebnis

Nächste Schritte

Die angegebenen Optionen für den MobiLink-Benutzer werden gespeichert.

Beispiel

Im folgenden Beispiel werden die erweiterten Optionen für den MobiLink-Benutzer mit dem Namen "SSinger" auf die Standardwerte gesetzt:

```
ALTER SYNCHRONIZATION USER SSinger  
DELETE ALL OPTION
```

Siehe auch

- „Erweiterte Optionen von MobiLink SQL Anywhere-Clients“ auf Seite 143
- „Erweiterte Optionen für dbmlsync“ auf Seite 92
- „ALTER SYNCHRONIZATION USER-Anweisung [MobiLink]“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- „CREATE SYNCHRONIZATION USER-Anweisung [MobiLink]“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]

Erstellen von Synchronisationssubskriptionen

Nachdem Sie MobiLink-Benutzer und -Publikationen erstellt haben, müssen Sie mindestens einen MobiLink-Benutzer für eine oder mehrere vorhandene Publikationen subscribieren. Dies erfolgt, indem Sie Synchronisationssubskriptionen erstellen.

Hinweis

Sie müssen darauf achten, dass alle Subskriptionen für einen MobiLink-Benutzer nur mit einer konsolidierten Datenbank synchronisiert werden. Sonst könnte es zu Datenverlust und nicht vorhersagbarem Verhalten kommen.

Eine Synchronisationssubskription verknüpft einen bestimmten MobiLink-Benutzer mit einer Publikation. Sie kann auch andere Informationen enthalten, die für die Synchronisation benötigt werden. Zum Beispiel können Sie die Adresse des MobiLink-Servers und Optionen für eine Synchronisationssubskription angeben. Werte für eine bestimmte Synchronisationssubskription heben die für MobiLink-Benutzer festgelegten Werte auf.

Synchronisationssubskriptionen sind nur in entfernten MobiLink SQL Anywhere-Datenbanken erforderlich. Serverlogik wird über Synchronisationsskripte implementiert, die in den MobiLink-Systemtabellen in der konsolidierten Datenbank gespeichert sind.

Eine einzelne SQL Anywhere-Datenbank kann mit mehreren MobiLink-Servern synchronisiert werden. Für die Synchronisation mit mehreren Servern erstellen Sie für jeden einzelnen Server einen eigenen MobiLink-Benutzer.

Siehe „CREATE SYNCHRONIZATION SUBSCRIPTION-Anweisung [MobiLink]“ [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Beispiel

Wenn Sie die Tabellen Customers und SalesOrders in der SQL Anywhere-Beispieldatenbank synchronisieren möchten, können Sie die folgenden Anweisungen verwenden.

1. Erstellen Sie zunächst eine Publikation, die die Tabellen Customers und SalesOrders enthält. Geben Sie der Publikation den Namen testpub.

```
CREATE PUBLICATION testpub
(TABLE Customers, TABLE SalesOrders)
```

2. Als Nächstes erstellen Sie einen MobiLink-Benutzer. In diesem Fall ist der MobiLink-Benutzer demo_ml_user.

```
CREATE SYNCHRONIZATION USER demo_ml_user
```

3. Um diesen Vorgang abzuschließen, erstellen Sie eine Synchronisationssubskription mit dem Namen my_sub, die den Benutzer und die Publikation miteinander verknüpft.

```
CREATE SYNCHRONIZATION SUBSCRIPTION my_sub TO testpub
FOR demo_ml_user
TYPE tcpip
ADDRESS 'host=localhost;port=2439;'
SCRIPT VERSION 'version1'
```

Siehe auch

- „Publikationen“ auf Seite 74
- „MobiLink-Benutzer“ auf Seite 82

MobiLink-Synchronisationssubskriptionen ändern

Sie können eine bestehende Synchronisationssubskription ändern.

Voraussetzungen

Sie müssen eine vorhandene Datenbank und die SYS_REPLICATION_ADMIN_ROLE-Systemrolle haben.

Aufgabe

1. Stellen Sie eine Verbindung mit der Datenbank her.
2. Doppelklicken Sie auf **Benutzer**.
3. Doppelklicken Sie auf einen Benutzer.
4. Rechtsklicken Sie auf die Subskription, die Sie ändern möchten, und wählen Sie **Eigenschaften** aus.
5. Ändern Sie die Eigenschaften je nach Bedarf.

Ergebnisse

Die Synchronisationssubskription wird geändert.

Siehe auch

- „ALTER SYNCHRONIZATION SUBSCRIPTION-Anweisung [MobiLink]“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- „ALTER SYNCHRONIZATION USER-Anweisung [MobiLink]“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- „CREATE SYNCHRONIZATION SUBSCRIPTION-Anweisung [MobiLink]“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]

MobiLink-Subskriptionen löschen

Sie können eine Synchronisationssubskription für einen MobiLink-Benutzer löschen, wenn sie nicht mehr benötigt wird.

Voraussetzungen

Sie müssen die SYS_REPLICATION_ADMIN_ROLE-Systemrolle haben, um eine Synchronisationssubskription löschen zu können.

Aufgabe

1. Stellen Sie eine Verbindung mit der Datenbank her.
2. Doppelklicken Sie auf **Benutzer**.
3. Doppelklicken Sie auf einen MobiLink-Benutzer.
4. Rechtsklicken Sie auf eine Subskription und wählen Sie **Löschen**.

Ergebnisse

Die ausgewählte Synchronisationssubskription wird gelöscht.

Siehe auch

- „DROP SYNCHRONIZATION SUBSCRIPTION-Anweisung [MobiLink]“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]

Initiieren einer Synchronisation

Die MobiLink-Synchronisation wird immer vom Client initiiert. Bei SQL Anywhere-Clients kann die Synchronisation über das Dienstprogramm dbmlsync, die dbmlsync-API oder die SQL SYNCHRONIZE-Anweisung initiiert werden. Die eingesetzte Semantik ist bei allen Möglichkeiten ähnlich, sie bieten jedoch unterschiedliche Schnittstellen für die Synchronisation und verschiedene Fähigkeiten zur Integration der Synchronisation in Ihren eigenen Anwendungen.

Es gibt viele Optionen für das Anpassen des Synchronisationsverhaltens. Einige davon sind jedoch besonders wichtig, da sie für so gut wie alle Synchronisationen erforderlich sind. Diese werden nachfolgend beschrieben.

Mit der Option -c können Sie Verbindungsparameter festlegen, die steuern, wie dbmlsync eine Verbindung mit der entfernten Datenbank herstellt. Diese Informationen sind bei Verwendung der SQL-Synchronisationsanweisung nicht erforderlich, da die Verbindungsinformationen der Datenbankverbindung, die die Anweisung ausführt, entnommen werden.

Mit der Subskriptionsoption (Option -s) können Sie festlegen, welche in der entfernten Datenbank definierte Subskription synchronisiert wird.

Mit den erweiterten Optionen CommunicationAddress und CommunicationType können Sie Netzwerkprotokolloptionen angeben, die bestimmen, wie dbmlsync während der Synchronisation die Verbindung mit dem MobiLink-Server herstellt.

Mit der Skriptversionsklausel in der CREATE SYNCHRONIZATION SUBSCRIPTION SQL-Anweisung können Sie festlegen, welche Skriptversion bei der Synchronisation einer Subskription verwendet werden soll. Die Skriptversion legt fest, welche Skripten vom MobiLink-Server für die Steuerung und Verarbeitung der Synchronisation verwendet werden.

Privilegien für dbmlsync

Für die Synchronisation muss dbmlsync eine Verbindung mit der entfernten Datenbank mithilfe einer Benutzer-ID herstellen, die die SYS_RUN_REPLICATION_ROLE-Systemrolle hat. Siehe [„Sicherheitshinweise zu rollenbasierter Zugriffssteuerung und Synchronisation“ auf Seite 89](#).

Synchronisation anpassen

Siehe [„Anpassen der dbmlsync-Synchronisation“ auf Seite 102](#).

Siehe auch

- „MobiLink SQL Anywhere Client-Dienstprogramm (dbmlsync)“ auf Seite 106
- „Dbmlsync-API“ auf Seite 103
- „SYNCHRONIZE-Anweisung [MobiLink]“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- „dbmlsync-Option -c“ auf Seite 116
- „dbmlsync-Option -s“ auf Seite 133
- „Erweiterte Option CommunicationAddress (adr)“ auf Seite 149
- „Erweiterte Option CommunicationType (ctp)“ auf Seite 150
- „CREATE SYNCHRONIZATION SUBSCRIPTION-Anweisung [MobiLink]“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- „GRANT ROLE SYS_REPLICATION_ADMIN_ROLE-Anweisung [MobiLink] [SQL Remote]“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- „GRANT ROLE SYS_RUN_REPLICATION_ROLE-Anweisung [MobiLink] [SQL Remote]“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]

Sicherheitshinweise zu rollenbasierter Zugriffssteuerung und Synchronisation

Ein Benutzer muss SYS_RUN_REPLICATION_ROLE haben, um die Synchronisation ausführen zu können. Mit SYS_RUN_REPLICATION_ROLE werden einem Benutzer Systemprivilegien erteilt. Diese Privilegien können jedoch nur genutzt werden, wenn sich der Benutzer über ein authentifiziertes Tool wie

dbmlsync oder SQL Remote angemeldet hat. Dies funktioniert ähnlich wie in früheren Versionen bei REMOTE DBA.

Die SYS_RUN_REPLICATION_ROLE-Systemrolle umfasst standardmäßig die SYS_AUTH_DBA_ROLE-Systemrolle. Die SYS_AUTH_DBA_ROLE-Systemrolle kann der SYS_RUN_REPLICATION_ROLE-Systemrolle entzogen werden. Die SYS_AUTH_DBA_ROLE-Systemrolle ist die einzige Berechtigung, die aus der SYS_RUN_REPLICATION_ROLE-Systemrolle entfernt werden kann.

Die SYS_AUTH_DBA_ROLE-Systemrolle hat normalerweise umfangreichere Berechtigungen als für eine Synchronisation benötigt werden. Um eine sicherere Synchronisationsumgebung einzurichten, verwenden Sie eine der folgenden Methoden:

- Entziehen Sie der SYS_RUN_REPLICATION_ROLE-Systemrolle SYS_AUTH_DBA_ROLE und erteilen Sie die folgenden Systemprivilegien:
 - INSERT ANY TABLE
 - UPDATE ANY TABLE
 - UPDATE ANY TABLE
 - DELETE ANY TABLE
 - EXECUTE ANY PROCEDURE
 - Alle Systemprivilegien, die für Anweisungen in Hooks erforderlich sind.
 - Alle Systemprivilegien, die für Anweisungen in gespeicherten Prozeduren erforderlich sind, mit denen skriptgesteuerte Uploads definiert werden.

Die Vorteile dieser Methode sind ihre Einfachheit und die Tatsache, dass diese Privilegien auf Systemebene nur genutzt werden können, wenn der Benutzer über ein authentifiziertes Tool wie dbmlsync oder SQL REMOTE verbunden ist. Der Nachteil dieser Vorgehensweise besteht darin, dass die SYS_RUN_REPLICATION_ROLE-Systemrolle mehr Privilegien erhält als für die Synchronisation erforderlich. Sie erhält die Privilegien INSERT, UPDATE, DELETE und ALTER für alle Tabellen und EXECUTE für alle Prozeduren, obwohl sie diese Privilegien nur für wenige Tabellen und Prozeduren benötigt.

- Entziehen Sie der SYS_RUN_REPLICATION_ROLE-Systemrolle SYS_AUTH_DBA_ROLE, erstellen Sie eine benutzererweiterte Rolle, die die SYS_RUN_REPLICATION_ROLE-Systemrolle hat, und weisen Sie diese benutzererweiterte Rolle allen Benutzern zu, denen die Synchronisation der Datenbank erlaubt werden soll. Erteilen Sie der benutzererweiterten Rolle die folgenden Privilegien auf Objektebene:
 - INSERT, UPDATE, DELETE und ALTER für alle Tabellen, die synchronisiert werden sollen.
 - EXECUTE für alle Hook-Prozeduren und gespeicherten Prozeduren, mit denen skriptgesteuerte Uploads definiert werden.
 - SELECT, INSERT, UPDATE und DELETE für die Tabellen dbo.synchronize_results und dbo.synchronize_parameters. Dies ist nur erforderlich, wenn die SQL-Anweisung SYNCHRONIZE verwendet wird.
 - Alle Privilegien, die für Anweisungen in Hooks erforderlich sind.
 - Alle Privilegien, die für Anweisungen in gespeicherten Prozeduren erforderlich sind, mit denen skriptgesteuerte Uploads definiert werden.

Der Vorteil dieser Vorgehensweise besteht darin, dass Sie eine sehr genaue Kontrolle über die Privilegien haben, die dem Benutzer erteilt werden. Die erteilten Privilegien stehen den Benutzern jedoch unabhängig davon zur Verfügung, wie sie angemeldet sind. Die Benutzer sind nicht auf Verbindungen beschränkt, die durch dbmlsync und SQL Remote hergestellt wurden.

Siehe auch

- „GRANT ROLE SYS_RUN_REPLICATION_ROLE-Anweisung [MobiLink] [SQL Remote]“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- „REVOKE ROLE SYS_RUN_REPLICATION_ROLE-Anweisung [MobiLink] [SQL Remote]“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- „Systemprivilegien“ [*SQL Anywhere Server - Datenbankadministration*]
- „Benutzererweiterte Rollen“ [*SQL Anywhere Server - Datenbankadministration*]
- „Benutzersicherheit (Rollen und Privilegien)“ [*SQL Anywhere Server - Datenbankadministration*]

Beispiele

Das folgende Beispiel zeigt, wie Sie der SYS_RUN_REPLICATION_ROLE-Systemrolle die SYS_AUTH_DBA_ROLE-Systemrolle entziehen und die notwendigen Privilegien für die Synchronisation erteilen.

```
REVOKE ROLE SYS_AUTH_DBA_ROLE FROM SYS_RUN_REPLICATION_ROLE;
GRANT INSERT ANY TABLE TO SYS_RUN_REPLICATION_ROLE;
GRANT UPDATE ANY TABLE TO SYS_RUN_REPLICATION_ROLE;
GRANT DELETE ANY TABLE TO SYS_RUN_REPLICATION_ROLE;
GRANT ALTER ANY TABLE TO SYS_RUN_REPLICATION_ROLE;
GRANT EXECUTE ANY PROCEDURE TO SYS_RUN_REPLICATION_ROLE;
```

Das folgende Beispiel zeigt, wie Sie der SYS_RUN_REPLICATION_ROLE-Systemrolle die SYS_AUTH_DBA_ROLE-Systemrolle entziehen und eine benutzererweiterte Rolle namens **SYNC_USER** erstellen, deren Privilegien ausreichen, um die Tabellen **T1** und **T2** zu synchronisieren und den sp_hook_dbmlsync_begin-Hook auszuführen. Danach wird die SYNC_USER-Rolle dem Benutzer **user1** erteilt.

```
REVOKE ROLE SYS_AUTH_DBA_ROLE FROM SYS_RUN_REPLICATION_ROLE;

// Create user-extended role SYNC_USER
CREATE USER SYNC_USER IDENTIFIED BY 'sql';
GRANT ROLE SYS_RUN_REPLICATION_ROLE TO SYNC_USER;
CREATE ROLE FOR USER SYNC_USER;

// Grant privileges on table T1 to SYNC_USER
GRANT INSERT ON T1 TO SYNC_USER;
GRANT UPDATE ON T1 TO SYNC_USER;
GRANT DELETE ON T1 TO SYNC_USER;
GRANT ALTER ON T1 TO SYNC_USER;

// Grant privileges on table T2 to SYNC_USER
GRANT INSERT ON T2 TO SYNC_USER;
GRANT UPDATE ON T2 TO SYNC_USER;
GRANT DELETE ON T2 TO SYNC_USER;
GRANT ALTER ON T2 TO SYNC_USER;

// Grant privileges on any hooks to SYNC_USER
GRANT EXECUTE ON dba.sp_hook_dbmlsync_begin TO SYNC_USER;

// Grant privileges on the synchronize_results and synchronize_parameters
tables
```

```
// to SYNC_USER so that it can use the SYNCHRONIZE statement
GRANT SELECT ON dbo.synchronize_results TO SYNC_USER;
GRANT INSERT ON dbo.synchronize_results TO SYNC_USER;
GRANT UPDATE ON dbo.synchronize_results TO SYNC_USER;
GRANT DELETE ON dbo.synchronize_results TO SYNC_USER;

GRANT SELECT ON dbo.synchronize_parameters TO SYNC_USER;
GRANT INSERT ON dbo.synchronize_parameters TO SYNC_USER;
GRANT UPDATE ON dbo.synchronize_parameters TO SYNC_USER;
GRANT DELETE ON dbo.synchronize_parameters TO SYNC_USER;

// Grant SYNC_USER to user1 so that user1 can synchronize the database

GRANT ROLE SYNC_USER to user1;
```

Erweiterte Optionen für dbmlsync

MobiLink bietet eine Reihe von erweiterten Optionen, mit denen Sie den Synchronisationsprozess anpassen können. Die erweiterten Optionen können in den Publikationen, den Benutzerdaten und in den Subskriptionen festgelegt werden. Darüber hinaus können Werte von erweiterten Optionen mithilfe von Optionen an der dbmlsync-Befehlszeile oder der Einstiegsprozedur `sp_hook_dbmlsync_set_extended_options` überschrieben werden.

- **Aufheben einer erweiterten Option in der dbmlsync-Befehlszeile** Geben Sie die Werte der erweiterten Optionen mit den dbmlsync-Optionen `-e` bzw. `-eu` in der Form *option-name=value* ein. Beispiel:

```
dbmlsync -e "v=on;sc=low"
```

- **Festlegen einer erweiterten Option für eine Subskription, eine Publikation oder einen Benutzer** Fügen Sie die Option der Anweisung `CREATE SYNCHRONIZATION SUBSCRIPTION` oder `CREATE SYNCHRONIZATION USER` in der entfernten SQL Anywhere-Datenbank hinzu.

Beim Hinzufügen einer erweiterten Option für eine Publikation ist die Vorgehensweise anders. Sie fügen eine erweiterte Option für eine Publikation mit der Anweisung `ALTER/CREATE SYNCHRONIZATION SUBSCRIPTION` ohne `FOR`-Klausel hinzu.

Siehe auch

- „Erweiterte Optionen von MobiLink SQL Anywhere-Clients“ auf Seite 143
- „`sp_hook_dbmlsync_set_extended_options`“ auf Seite 252

Beispiel

Mit der folgenden Anweisung wird eine Synchronisationssubskription erstellt, die mithilfe der erweiterten Optionen die Cachegröße zum Vorbereiten des Uploads auf 3 MB und die Upload-Inkrementgröße auf 3 kB setzt.

```
CREATE SYNCHRONIZATION SUBSCRIPTION TO my_pub
FOR ml_user
ADDRESS 'host=test.internal;port=2439;'
OPTION memory='3m',increment='3k'
```

Die Optionswerte müssen in Apostrophen stehen, aber die Optionsnamen dürfen nicht in Apostrophen oder Anführungszeichen stehen.

Netzwerkprotokolloptionen für dbmlsync

dbmlsync-Verbindungsinformationen umfassen das für die Kommunikation mit dem Server zu verwendende Protokoll, die Adresse des MobiLink-Servers sowie andere Verbindungsparameter.

Siehe auch

- „Erweiterte Option `CommunicationType (ctp)`“ auf Seite 150
- „Erweiterte Option `CommunicationAddress (adr)`“ auf Seite 149

Transaktionslogdateien

Normalerweise ermittelt dbmlsync den Inhalt des Uploads, indem das SQL Anywhere-Transaktionslog herangezogen wird.

Standardmäßig benutzen SQL Anywhere-Datenbanken ein Transaktionslog. Beim Erstellen der Datenbank oder, zu einem späteren Zeitpunkt, mit dem dblog-Dienstprogramm können Sie bestimmen, ob ein Transaktionslog erstellt und wo dieses gespeichert werden soll.

Das Transaktionslog ist nicht erforderlich, wenn Sie Publikationen mit skriptgesteuertem Upload synchronisieren oder ausschließlich reine Download-Publikationen verwenden.

Zur Vorbereitung des Uploads erfordert das dbmlsync-Dienstprogramm Zugriff auf alle Transaktionslogs, die seit der letzten erfolgreichen Synchronisation aller Subskriptionen für den synchronisierenden MobiLink-Benutzer geschrieben wurden. SQL Anywhere-Logdateien werden jedoch normalerweise im Rahmen der Datenbankpflege verkürzt und umbenannt. In solch einem Fall müssen alte Logdateien umbenannt und in einem separaten Verzeichnis gespeichert werden, bis alle beschriebenen Änderungen erfolgreich synchronisiert wurden.

Sie können in der dbmlsync-Befehlszeile das Verzeichnis angeben, in dem sich die umbenannten Logdateien befinden. Sie können diesen Parameter auslassen, wenn die Arbeitslogdatei seit der letzten Synchronisation nicht gekürzt und umbenannt wurde oder wenn dbmlsync aus dem Verzeichnis ausgeführt wird, das die umbenannten Logdateien enthält.

Siehe auch

- „Sicherung und Datenwiederherstellung“ [*SQL Anywhere Server - Datenbankadministration*]
- „Offsets für den Verarbeitungsfortschritt“ auf Seite 73
- „Das Transaktionslog“ [*SQL Anywhere Server - Datenbankadministration*]
- „Dienstprogramm Initialisierung (dbinit)“ [*SQL Anywhere Server - Datenbankadministration*]
- „Skriptgesteuerter Upload“ auf Seite 323

Beispiel

Angenommen, die alten Logdateien werden im Verzeichnis `c:\oldlogs` gespeichert. Mit dem folgenden Befehl können Sie die entfernte Datenbank synchronisieren.

```
dbmlsync -c "dbn=remote;uid=syncuser" c:\oldlogs
```

Der Pfad zum alten Logverzeichnis muss das letzte Argument in der Befehlszeile sein.

Parallelität während der Synchronisation

Um die Integrität von Synchronisationen sicherzustellen, muss dbmlsync dafür sorgen, dass keine vom Server heruntergeladenen Änderungen Zeilen in der entfernten Datenbank ändern, die seit dem Senden des letzten Uploads ihrerseits geändert wurden. Standardmäßig erfolgt dies ohne Sperren von Tabellen, sodass die Auswirkungen auf andere gleichzeitige Benutzer der Datenbank minimal sind. Die Tabellen werden bei der Synchronisation einer Publikation mit skriptgesteuertem Upload oder bei definiertem `sp_hook_dbmlsync_schema_upgrade`-Hook gesperrt (IN SHARE MODE).

Sind die Tabellen nicht gesperrt, protokolliert dbmlsync nach Erstellung des Upload alle geänderten Zeilen. Wenn der Download eine Änderung für eine dieser Zeilen enthält, wird dies als ein Konflikt gewertet.

Bei der Erkennung eines Konflikts wird der Download abgebrochen und die Downloadvorgänge werden zurückgesetzt, um ein Überschreiben der neuen Änderung zu verhindern. Das Dienstprogramm dbmlsync versucht dann erneut, die Synchronisation einschließlich des Upload-Schritts durchzuführen. Da die Zeile nun zu Beginn des Synchronisationsvorgangs vorhanden ist, wird sie in den Upload einbezogen.

Standardmäßig versucht dbmlsync, die Synchronisation auszuführen, bis sie erfolgreich ist. Sie können die Anzahl der Neuversuche mit der erweiterten Option `ConflictRetries` begrenzen. Wenn Sie `ConflictRetries` auf -1 setzen, unternimmt dbmlsync so lange Synchronisationsversuche, bis die Synchronisation erfolgreich ist. Wenn Sie die Option auf eine nicht negative Ganzzahl setzen, führt dbmlsync eine entsprechende Anzahl an Neuversuchen aus.

Option -d

Wenn bei der Verwendung des Sperrmechanismus weitere Verbindungen zur Datenbank vorhanden sind und diese Verbindungen Sperren für die Synchronisationstabellen besitzen, schlägt die Synchronisation fehl. Damit die Synchronisation auch dann sofort ausgeführt wird, wenn andere Sperren vorhanden sind, verwenden Sie die dbmlsync-Option -d. Wenn diese Option angegeben ist, wird jede Verbindung mit Sperren, die die Synchronisation behindern könnten, von der Datenbank getrennt, sodass die Synchronisation vorgenommen werden kann. Nicht festgeschriebene Änderungen der getrennten Verbindungen werden zurückgesetzt.

Option LockTables

Sie können mit der erweiterten Option `LockTables` erzwingen, dass dbmlsync Tabellen während der Synchronisation sperrt. Das Sperren von Tabellen während der Synchronisation kann wünschenswert sein, um die in Hook-Prozeduren eingesetzte Logik zu vereinfachen.

Siehe auch

- „Erweiterte Option `ConflictRetries` (cr)“ auf Seite 150
- „dbmlsync-Option -d“ auf Seite 118
- „Erweiterte Option `LockTables` (lt)“ auf Seite 160

Initiieren einer Synchronisation von einer Anwendung aus

Es kann sinnvoll sein, die Funktionen von dbmlsync in eine Anwendung zu integrieren, anstatt dem Kunden ein separates Programm zu liefern.

Sie haben hierzu drei Möglichkeiten:

- Dbmlsync-API

Weitere Hinweise finden Sie unter [„Dbmlsync-API“ auf Seite 103](#).

- SQL-Anweisung SYNCHRONIZE

Weitere Hinweise finden Sie unter [„SYNCHRONIZE-Anweisung \[MobiLink\]“ \[SQL Anywhere Server - SQL-Referenzhandbuch\]](#).

- Wenn Sie eine Entwicklung in einer Sprache vornehmen, die eine DLL aufrufen kann, können Sie dbmlsync über die DBTools-Schnittstelle aufrufen. Wenn Sie in C oder C++ programmieren, können Sie die *dbtools.h*-Headerdatei im Unterverzeichnis *SDK\Include* Ihres SQL Anywhere 16-Verzeichnisses einbeziehen. Die Datei enthält eine Beschreibung der Struktur *a_sync_db* und der Funktion *DBSynchronizeLog*, die Sie verwenden müssen, um diese Funktion Ihrer Anwendung hinzuzufügen. Diese Lösung ist auf allen unterstützten Plattformen verfügbar, einschließlich Windows und UNIX.

Die Dbmlsync-API und die SQL-Anweisung SYNCHRONIZE sind beide einfacher zu verwenden als die DBTools-Schnittstelle. Ihre Verwendung wird deshalb stark empfohlen.

Weitere Hinweise finden Sie unter:

- [„DBTools-Schnittstelle für dbmlsync“ auf Seite 317](#)
- [DBSynchronizeLog-Methode \[Datenbanktools\] \[SQL Anywhere Server - Programmierung\]](#)
- [a_sync_db-Struktur \[Datenbanktools\] \[SQL Anywhere Server - Programmierung\]](#)

Synchronisation mit Microsoft ActiveSync

Microsoft ActiveSync ist eine Synchronisationssoftware für Microsoft Windows Mobile-Handhelds. Microsoft ActiveSync steuert die Synchronisation zwischen einem Windows Mobile-Gerät und einem PC. Ein MobiLink-Provider für Microsoft ActiveSync koordiniert die Synchronisation mit dem MobiLink-Server.

Eine Microsoft ActiveSync-Synchronisation für SQL Anywhere-Clients wird mit folgenden Schritten eingerichtet:

- Entfernte SQL Anywhere-Datenbank für die Microsoft ActiveSync-Synchronisation konfigurieren
Siehe [„Entfernte SQL Anywhere-Datenbanken für Microsoft ActiveSync konfigurieren“ auf Seite 96](#).
- MobiLink-Provider für Microsoft ActiveSync installieren
Siehe [„MobiLink-Provider für Microsoft ActiveSync installieren“ auf Seite 97](#).
- SQL Anywhere-Client für den Einsatz mit Microsoft ActiveSync registrieren

Siehe „[SQL Anywhere-Clients für Microsoft ActiveSync registrieren](#)“ auf Seite 98.

Wenn Sie Microsoft ActiveSync-Synchronisation einsetzen, muss die Synchronisation von der Microsoft ActiveSync-Software initiiert werden. Der MobiLink-Provider für Microsoft ActiveSync kann dbmlsync starten oder das Programm nach Zeitplan mit einer Terminzeichenfolge aus dem Ruhezustand wecken.

Sie können dbmlsync auch mithilfe eines Verzögerungs-Hooks in der entfernten Datenbank in den Ruhezustand versetzen, dabei kann aber der MobiLink-Provider für Microsoft ActiveSync aus diesem Zustand die Synchronisation nicht aktivieren.

Weitere Hinweise zur Abfolgeplanung für die Synchronisation finden Sie unter „[Synchronisationszeitpläne](#)“ auf Seite 100.

Entfernte SQL Anywhere-Datenbanken für Microsoft ActiveSync konfigurieren

MobiLink-Benutzer, Publikationen und Subskriptionen können für die Verwendung von Microsoft ActiveSync konfiguriert werden.

Voraussetzungen

Es gibt keine Voraussetzungen für diese Aufgabe.

Kontext und Bemerkungen

Diese Informationen sind nur relevant, wenn Sie Microsoft ActiveSync verwenden.

Aufgabe

1. Wählen Sie einen Synchronisationstyp (TCP/IP, TLS, HTTP oder HTTPS).

Der Synchronisationstyp kann für eine Synchronisationspublikation, für einen Synchronisationsbenutzer oder für eine Synchronisationssubskription festgelegt werden. In allen Fällen geschieht die Einrichtung auf ähnliche Weise. Es folgt ein Teil einer typischen CREATE SYNCHRONIZATION USER-Anweisung:

```
CREATE SYNCHRONIZATION USER SSinger  
TYPE tcpip  
...
```

2. Geben Sie eine ADDRESS-Klausel an, wenn Sie die Kommunikation zwischen dem MobiLink-Provider für Microsoft ActiveSync und dem MobiLink-Server definieren wollen.

Bei HTTP- oder TCP/IP-Synchronisation wird mit der ADDRESS-Klausel der Anweisungen CREATE SYNCHRONIZATION USER oder CREATE SYNCHRONIZATION SUBSCRIPTION die Kommunikation zwischen MobiLink-Client und -Server festgelegt. Bei Microsoft ActiveSync erfolgt die Kommunikation in zwei Phasen: Vom Dienstprogramm dbmlsync auf dem Gerät bis zum MobiLink-Provider für Microsoft ActiveSync auf dem PC und dann vom PC bis zum MobiLink-Synchronisationsserver. Mit der ADDRESS-Klausel wird die Kommunikation zwischen dem MobiLink-Provider für Microsoft ActiveSync und dem MobiLink-Server angegeben.

Ergebnisse

Der angegebene Benutzer, die angegebene Publikation oder die angegebene Subskription wird für die Verwendung von Microsoft ActiveSync konfiguriert.

Beispiel

Mit der folgenden Anweisung wird auf einem Computer mit dem Namen "kangaroo" die TCP/IP-Kommunikation mit einem MobiLink-Server festgelegt:

```
CREATE SYNCHRONIZATION USER SSinger  
TYPE tcpip  
ADDRESS 'host=kangaroo;port=2439';
```

Siehe auch

- „CREATE SYNCHRONIZATION USER-Anweisung [MobiLink]“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]

MobiLink-Provider für Microsoft ActiveSync installieren

Bevor Sie Ihren SQL Anywhere-MobiLink-Client für die Verwendung mit Microsoft ActiveSync registrieren, müssen Sie den MobiLink-Provider für Microsoft ActiveSync mit dem Installationsprogramm (*mlasinst.exe*) installieren.

Voraussetzungen

Es gibt keine Voraussetzungen für diese Aufgabe.

Kontext und Bemerkungen

Das Installationsprogramm von SQL Anywhere für Windows Mobile installiert den MobiLink-Provider für Microsoft ActiveSync. Wenn Sie SQL Anywhere für Windows Mobile installieren, brauchen Sie die Schritte in diesem Abschnitt nicht auszuführen.

Aufgabe

1. Vergewissern Sie sich, dass Microsoft ActiveSync auf Ihrem Computer installiert ist und eine Verbindung zum Windows Mobile-Gerät besteht.
2. Führen Sie den folgenden Befehl aus, um den MobiLink-Provider zu installieren:

```
mlasinst -k desk-path -v dev-path
```

Dabei gilt: *desk-path* ist der Speicherort der Desktop-Komponente des Providers (*mlasdesk.dll*) und *dev-path* der Speicherort der Gerätekomponente (*mlasdev.dll*).

Wenn Sie SQL Anywhere auf Ihrem Computer installiert haben, befindet sich *mlasdesk.dll* im Verzeichnis *%SQLANY16%\bin32*. *mlasdev.dll* im Verzeichnis *%SQLANY16%\CE*. Wenn Sie *-v* oder *-k* nicht angeben, werden diese Verzeichnisse standardmäßig durchsucht.

Wenn Sie eine Meldung erhalten, dass der entfernte Provider nicht geöffnet werden konnte, setzen Sie das Gerät warm zurück und wiederholen den Befehl.

3. Starten Sie den Computer neu.

Microsoft ActiveSync erkennt neue Provider erst, nachdem der Computer neu gestartet wurde.

4. Aktivieren Sie den MobiLink-Provider.

Weitere Hinweise zu Windows-Versionen vor Vista:

- Klicken Sie im Microsoft ActiveSync-Fenster auf **Optionen**.
- Prüfen Sie den MobiLink-Eintrag in der Liste und klicken Sie auf **OK**, damit der Provider aktiviert wird.
- Sie können eine Liste der registrierten Anwendungen anzeigen, wenn Sie nochmals auf **Optionen** klicken, den MobiLink-Provider wählen und auf **Einstellungen** klicken.

Für Windows Vista oder höher:

- Klicken Sie im Fenster des Windows Mobile-Gerätecenters auf **Einstellungen Mobilgerät** und dann auf **Inhaltseinstellungen ändern**.
- Wählen Sie **MobiLink-Clients** und **Speichern**, um den Provider zu aktivieren.
- Um eine Liste der registrierten Anwendungen anzuzeigen, klicken Sie nacheinander auf **Inhaltseinstellungen ändern**, **MobiLink-Clients** und **Synchronisationseinstellungen**.

Ergebnisse

Der MobiLink-Provider für Microsoft ActiveSync wird installiert.

Nächste Schritte

Wenn Sie den MobiLink-Provider für Microsoft ActiveSync installiert haben, müssen Sie jede Anwendung einzeln registrieren.

Siehe auch

- „SQL Anywhere-Clients für Microsoft ActiveSync registrieren“ auf Seite 98
- „Dienstprogramm zur Installation des Microsoft ActiveSync-Providers (mlasinst)“ auf Seite 20

SQL Anywhere-Clients für Microsoft ActiveSync registrieren

In diesem Abschnitt wird beschrieben, wie die Microsoft ActiveSync-Software verwendet wird, um einen SQL Anywhere-Client für die Verwendung mit Microsoft ActiveSync zu registrieren.

Voraussetzungen

Sie haben den MobiLink-Provider für Microsoft ActiveSync installiert.

Kontext und Bemerkungen

Sie können Ihre Anwendung für die Verbindung mit Microsoft ActiveSync mithilfe des Dienstprogramms zur Installation des Microsoft ActiveSync-Providers oder mit der Microsoft ActiveSync-Software selbst registrieren.

Aufgabe

1. Vergewissern Sie sich, dass der MobiLink-Provider für Microsoft ActiveSync installiert ist.
2. Starten Sie die Microsoft ActiveSync-Software auf Ihrem PC.
3. Weitere Hinweise zu Windows-Versionen vor Vista:
 - Klicken Sie im **Microsoft ActiveSync**-Fenster auf **Optionen**.
 - Klicken Sie in der Liste der Informationstypen auf **MobiLink** und anschließend auf **Einstellungen**.
 - Klicken Sie im Dialogfeld **MobiLink-Synchronisation** auf **Neu**.

Für Windows Vista oder höher:

- Klicken Sie im **Windows Mobile-Gerätecenter**-Fenster auf **Einstellungen des Mobilgeräts** und dann auf **Inhaltseinstellungen ändern**.
 - Klicken Sie auf **Inhaltseinstellungen ändern**.
 - Klicken Sie auf **MobiLink-Clients**.
 - Klicken Sie auf **Synchronisationseinstellungen**.
4. Geben Sie die folgenden Daten für Ihre Anwendung ein:
 - **Anwendungsname** Ein Name, der die in der Benutzerschnittstelle von Microsoft ActiveSync anzuzeigende Anwendung kennzeichnet
 - **Klassenname** Der Klassenname für den dbmlsync-Client, wie er mit der Befehlszeilenoption -wc festgelegt wurde
 - **Pfad** Der Speicherort der Anwendung dbmlsync auf dem Gerät
 - **Argumente** Alle Befehlszeilenargumente, die verwendet werden sollen, wenn Microsoft ActiveSync das Dienstprogramm dbmlsync startet

Sie starten dbmlsync auf eine der beiden folgenden Weisen:

- Wenn Sie Planungsoptionen angeben, wechselt dbmlsync in den Wartemodus (Hover). In diesem Fall verwenden Sie die dbmlsync-Option -wc mit einem passenden Wert in der Einstellung für den Klassennamen.
 - Andernfalls wechselt dbmlsync nicht in den Hovering-Modus. In diesem Fall verwenden Sie die Option -k, um dbmlsync zu beenden.
5. Klicken Sie auf **OK**, damit die Anwendung registriert wird.

Ergebnisse

Der SQL Anywhere-Client wird für die Verwendung mit Microsoft ActiveSync registriert.

Siehe auch

- „Dienstprogramm zur Installation des Microsoft ActiveSync-Providers (mlasinst)“ auf Seite 20
- „MobiLink-Provider für Microsoft ActiveSync installieren“ auf Seite 97
- „dbmlsync-Option -wc“ auf Seite 142
- „dbmlsync-Option -k (nicht mehr empfohlen)“ auf Seite 124
- „dbmlsync-Option -wc“ auf Seite 142
- „Synchronisationszeitpläne“ auf Seite 100

Synchronisationszeitpläne

Sie können dbmlsync so konfigurieren, dass eine Synchronisation in festgelegten Abständen gemäß definierten Regeln durchgeführt wird. Dazu stehen Ihnen zwei Möglichkeiten zur Verfügung:

- Benutzen Sie die erweiterte dbmlsync-Option SCHEDULE zum Start der Synchronisation zu bestimmten Tageszeiten, Wochenzeiten oder in regelmäßigen Intervallen. In diesem Fall läuft dbmlsync weiter, bis es vom Benutzer gestoppt wird.

Siehe „[Abfolgeplanung mit dbmlsync-Optionen](#)“ auf Seite 101.

- Benutzen Sie Ereignis-Hooks zum Start von Synchronisationen aufgrund einer von Ihnen definierten Programmlogik. Dies ist die beste Methode zur Implementierung von Synchronisationen in unregelmäßigen Intervallen oder als Antwort auf ein Ereignis. In diesem Fall können Sie dbmlsync programmgesteuert aus dem Hook-Programmcode stoppen.

Siehe „[Initiieren einer Synchronisation mit Ereignis-Hooks](#)“ auf Seite 101.

Diese Methode steht nicht zur Verfügung, wenn die Dbmlsync-API oder die SQL-Anweisung SYNCHRONIZE verwendet wird.

Hovering

Beim Hovering scannt dbmlsync das Transaktionslog der Datenbank und erstellt während der Zeit zwischen zwei Synchronisationen ein Upload. Dies ermöglicht eine schnellere Synchronisation, nachdem sie ausgelöst wurde, da ein Teil der Arbeit bereits getan ist.

Beim Hovering scannt dbmlsync bis zum Ende des Transaktionslogs und durchsucht das Log dann regelmäßig nach neuen Transaktionen. Sie können das Intervall zwischen Abfragen mit der erweiterten Option PollingPeriod der Option -pp steuern. Siehe „[Erweiterte Option PollingPeriod \(pp\)](#)“ auf Seite 165.

Beim Hovering in Verbindung mit zwei oder mehr Subskriptionen zur gleichen Zeit können Sie mit der erweiterten Option HoverRescanThreshold oder dem Ereignis-Hook sp_hook_dbmlsync_log_rescan die Speichernutzung einschränken, indem andernfalls verlorener Speicher wiederhergestellt wird.

Hovering kann mit der erweiterten Option DisablePolling oder der Option -p deaktiviert werden.

Siehe auch

- „Erweiterte Option HoverRescanThreshold (hrt)“ auf Seite 157
- „dbmlsync-Option -p“ auf Seite 127
- „Erweiterte Option DisablePolling (p)“ auf Seite 152
- „sp_hook_dbmlsync_log_rescan“ auf Seite 235

Abfolgeplanung mit dbmlsync-Optionen

Anstatt dbmlsync im Batchmodus auszuführen, wobei das Dienstprogramm synchronisiert und dann herunterfährt, können Sie einen SQL Anywhere-Client so einrichten, dass dbmlsync ständig läuft und zu festgelegten Zeiten synchronisiert.

Der Synchronisationsplan wird als erweiterte Option festgelegt. Er kann in der dbmlsync-Befehlszeile angegeben oder in der Datenbank für den Synchronisationsbenutzer, die Subskription oder die Publikation gespeichert werden.

Diese Methode steht nicht zur Verfügung, wenn die Dbmlsync-API oder die SQL-Anweisung SYNCHRONIZE verwendet wird.

- **Hinzufügen einer Abfolgeplanung zur Synchronisationsdefinition** Legen Sie die erweiterte Option "Schedule" in der Synchronisationssubskription fest. Beispiel:

```
CREATE SYNCHRONIZATION SUBSCRIPTION TO mypub
FOR mluser
ADDRESS 'host=localhost'
OPTION schedule='weekday@11:30am-12:30pm'
```

Sie können die Abfolgeplanung und Synchronisation mit der dbmlsync-Option -is sofort aufheben. Die Option -is gibt dbmlsync vor, den Zeitplan zu ignorieren, der durch die erweiterte Option zur Abfolgeplanung festgelegt wurde.

- **Hinzufügen einer Abfolgeplanung über die dbmlsync-Befehlszeile** Geben Sie die erweiterte Option für die Abfolge an. Erweiterte Optionen werden mit -e oder -eu gesetzt. Beispiel:

```
dbmlsync -e "sch=weekday@11:30am-12:30pm" ...
```

Wenn die geplante Synchronisation an einer dieser beiden Positionen festgelegt wird, fährt dbmlsync nicht herunter, wenn die Synchronisation abgeschlossen ist, sondern läuft kontinuierlich weiter.

Siehe auch

- „Erweiterte Option Schedule (sch)“ auf Seite 166
- „Erweiterte Optionen von MobiLink SQL Anywhere-Clients“ auf Seite 143
- „dbmlsync-Option -eu“ auf Seite 123
- „dbmlsync-Option -is“ auf Seite 123

Initiieren einer Synchronisation mit Ereignis-Hooks

Es gibt dbmlsync-Ereignis-Hooks, die Sie implementieren können, um zu steuern, wann eine Synchronisation erfolgen soll.

Mit dem `sp_hook_dbmsync_end`-Hook können Sie die Restart-Zeile in der Tabelle `#hook_dict` verwenden, um am Ende einer Synchronisation zu entscheiden, ob `dbmsync` die Synchronisation wiederholen soll.

Mit dem `sp_hook_dbmsync_delay`-Hook können Sie eine Verzögerung am Beginn jeder Synchronisation festlegen, die es Ihnen ermöglicht, den Zeitpunkt zu wählen, zu dem die Synchronisation beginnen soll. Mit diesem Hook können Sie eine Verzögerung über ein bestimmtes Intervall einrichten oder periodisch den Eintritt einer Bedingung prüfen.

Diese Methode steht nicht zur Verfügung, wenn die `Dbmsync-API` oder die `SQL-Anweisung SYNCHRONIZE` verwendet wird.

Siehe auch

- [„sp_hook_dbmsync_end“ auf Seite 232](#)
- [„sp_hook_dbmsync_delay“ auf Seite 218](#)

Anpassen der dbmsync-Synchronisation

dbmsync-Client Ereignis-Hooks

Mit Ereignis-Hooks können Sie mit `SQL` gespeicherte Prozeduren verwenden, um den clientseitigen Synchronisationsprozess für `dbmsync` zu verwalten. Sie können Client-Ereignis-Hooks über das Befehlszeilendienstprogramm oder die `dbmsync`-Programmierschnittstelle einsetzen.

Mit Ereignis-Hooks können Sie Synchronisationsereignisse protokollieren und verarbeiten. Sie können beispielsweise Synchronisationen basierend auf logischen Ereignissen planen, fehlgeschlagene Verbindungen erneut aufbauen oder bestimmte Fehler und Verletzungen der referenziellen Integrität verarbeiten.

dbmsync-Programmierschnittstellen

Mit der folgenden Programmierschnittstelle können Sie `MobiLink`-Clients in Ihre Anwendungen integrieren und Synchronisationen starten. Diese Schnittstellen sind eine Alternative zum `dbmsync`-Befehlszeilendienstprogramm.

- **dbmsync-API** Die `Dbmsync-API` stellt eine Programmierschnittstelle bereit, die es `MobiLink`-Clients, die in `C++` oder `.Net` geschrieben sind, gestattet, Synchronisationen zu starten und Rückmeldungen über den Fortschritt der angeforderten Synchronisationen zu empfangen. Diese neue Programmierschnittstelle bietet Zugriff auf eine Vielzahl weiterer Informationen über die Synchronisationsergebnisse. Außerdem können Sie Synchronisationen in eine Warteschlange stellen und sie so einfacher verwalten.
- **DBTools-Schnittstelle für dbmsync** Mit der `DBTools`-Schnittstelle für `dbmsync` können Sie Synchronisationsfunktionen in Ihre Clientanwendungen für die `SQL Anywhere`-Synchronisation integrieren. Alle `SQL Anywhere`-Datenbankverwaltungsdienstprogramme sind auf `DBTools` aufgebaut.

Skriptgesteuerter Upload

Sie können das Client-Transaktionslog außer Kraft setzen und einen eigenen Upload-Datenstrom definieren.

Siehe auch

- „Ereignis-Hooks für SQL Anywhere-Clients“ auf Seite 202
- „Dbmsync-API“ auf Seite 103
- „DBTools-Schnittstelle für dbmsync“ auf Seite 317
- „Skriptgesteuerter Upload“ auf Seite 323

Dbmsync-API

Die Dbmsync-API stellt eine Programmierschnittstelle bereit, die es in C++ oder .NET geschriebenen MobiLink-Clientanwendungen gestattet, Synchronisationen zu starten und Rückmeldungen über den Fortschritt der angeforderten Synchronisationen zu empfangen. Die API dient dazu, die Synchronisation nahtlos in Ihre Anwendungen zu integrieren.

Diese Programmierschnittstelle bietet Zugriff auf eine Vielzahl weiterer Informationen über die Synchronisationsergebnisse. Außerdem können Sie Synchronisationen in eine Warteschlange stellen und sie so einfacher verwalten.

Vorsicht

Die Dbmsync-API ist nicht threadsicher. Alle Aufrufe einer einzigen Instanz der DbmsyncClient-Klasse müssen auf dem gleichen Thread erfolgen. Wenn Funktionen für eine einzelne DbmsyncClient-Instanz über verschiedene Threads aufgerufen werden, kann dies zu unerwarteten Fehlern und unzuverlässigen Ergebnissen führen.

Siehe auch

- „Dbmsync .NET-API-Referenz“ auf Seite 291
- „Dbmsync C++-API-Referenz“ auf Seite 267

SQL Anywhere-Clientlogs

Wenn Sie MobiLink-Anwendungen mit entfernten SQL Anywhere-Datenbanken erstellen, gibt es zwei wichtige Client-Logdateitypen:

- dbmsync-Meldungslog
- SQL Anywhere-Transaktionslog

dbmsync-Meldungslog

Standardmäßig werden dbmsync-Meldungen an das dbmsync-Meldungsfenster gesendet. Außerdem können Sie die Ausgabe mithilfe der Befehlszeilenoption -o oder -ot an eine Meldungslogdatei senden. Der folgende Ausschnitt aus einer Befehlszeile sendet die Ausgabe an eine Logdatei mit dem Namen *dbmsync.dbs*.

```
dbmlsync -o dbmlsync.dbs ...
```

Die dbmlsync-Aktivität ist vor allem bei der Entwicklung und der Fehlerbehandlung hilfreich.

Sie können die Größe der Logdateien steuern und festlegen, was geschehen soll, wenn eine Datei die Maximalgröße erreicht:

- Mit der Option -o können Sie eine Logdatei festlegen und die Ausgabe daran anhängen.
- Mit der Option -ot können Sie eine Logdatei festlegen, aber Sie müssen den Inhalt der Datei löschen, bevor Sie die Ausgabe daran anhängen.
- Zusätzlich zur Option -o oder -ot können Sie mit der Option -os die Größe angeben, bei der die Logdatei umbenannt und eine neue Datei mit dem ursprünglichen Namen gestartet wird.

Wenn keine Meldungslogdatei angegeben wird, werden alle Ausgaben im dbmlsync-Meldungsfenster angezeigt. Wenn eine Meldungslogdatei angegeben wird, werden weniger Ausgabeinformationen im dbmlsync-Meldungsfenster angezeigt.

Mit der Option -v können Sie steuern, welche Informationen in der Meldungslogdatei protokolliert und im dbmlsync-Meldungsfenster angezeigt werden. Für die normale Verwendung in einer Produktionsumgebung wird die ausführliche Ausgabe nicht empfohlen, da sie die Performance beeinträchtigen kann.

SQL Anywhere-Transaktionslog

Siehe „[Transaktionslogdateien](#)“ auf Seite 93.

Siehe auch

- „[dbmlsync-Option -o](#)“ auf Seite 126
- „[dbmlsync-Option -ot](#)“ auf Seite 127
- „[dbmlsync-Option -os](#)“ auf Seite 127
- „[dbmlsync-Option -v](#)“ auf Seite 141

MobiLink auf Mac OS X ausführen

Sie können den MobiLink-Server und den SQL Anywhere MobiLink-Client unter Mac OS X ausführen. Es ist nicht möglich, UltraLite unter Mac OS X auszuführen.

Voraussetzungen

Es gibt keine Voraussetzungen für diese Aufgabe.

Kontext und Bemerkungen

Um eine konsolidierte MobiLink-Datenbank auf Mac OS X zu synchronisieren, können Sie den ODBC-Treiber von SQL Anywhere als Treibermanager verwenden. Siehe „[Erstellen einer ODBC-Datenquelle \(Mac OS X\)](#)“ [[SQL Anywhere Server - Datenbankadministration](#)].

Aufgabe

1. Starten des MobiLink-Servers unter Mac OS X.

a. Starten Sie SyncConsole.

Im Finder doppelklicken Sie auf "SyncConsole". Die SyncConsole-Anwendung befindet sich unter */Anwendungen/SQLAnywhere16*.

b. Klicken Sie auf **Datei**→**Neu**→**MobiLink-Server**.

c. Konfigurieren Sie den MobiLink-Server:

i. Geben Sie im Feld **Verbindungsparameter** folgende Zeichenfolge ein:

`DSN=dsn-name`

Der *dsn-name* ist ein ODBC-Datenquellenname in SQL Anywhere. Weitere Hinweise zum Erstellen von ODBC-Datenquellen finden Sie unter „[ODBC-Datenquellen](#)“ [[SQL Anywhere Server - Datenbankadministration](#)].

Wenn der *dsn-name* Leerzeichen enthält, setzen Sie die Zeichenfolge in Anführungszeichen. Beispiel:

`DSN="SQL Anywhere 16 Demo"`

ii. Legen Sie im Feld **Optionen** die gewünschten Optionen fest.

Mit dem Feld **Optionen** können Sie verschiedene Aspekte des MobiLink-Serververhaltens steuern. Eine vollständige Liste der Optionen finden Sie unter „[mlsrv16-Syntax](#)“ [[MobiLink - Serveradministration](#)].

d. Klicken Sie auf **Start**, um den MobiLink-Server zu starten.

Das Servermeldungsfenster zeigt an, dass der Server zum Annehmen von Synchronisationsanforderungen bereit ist.

2. Starten von dbmlsync unter Mac OS X.

a. Starten Sie SyncConsole.

Doppelklicken Sie im **Finder** auf **SyncConsole**. Die SyncConsole-Anwendung befindet sich unter */Anwendungen/SQLAnywhere16*.

b. Klicken Sie auf **Datei**→**Neu**→**MobiLink-Client**.

Das Fenster mit Clientoptionen wird angezeigt. Es enthält zahlreiche Konfigurationsoptionen, die den dbmlsync-Befehlszeilenoptionen entsprechen. Eine vollständige Auflistung finden Sie unter „[dbmlsync-Syntax](#)“ auf [Seite 106](#).

Mit den Optionen auf den Registerkarten **Login**, **Datenbank**, **Netzwerk** und **Erweitert** wird die Verbindung vom MobiLink-Client zur entfernten SQL Anywhere-Datenbank definiert. Häufig müssen Sie nur eine ODBC-Datenquelle auf der Registerkarte **Login** festlegen, um eine Verbindung herzustellen.

Mit den Optionen auf der Registerkarte **DBMLSync** werden Einzelheiten zur Verbindung mit dem MobiLink-Server festgelegt. Wenn diese Funktionen in der Publikation und Subskription einer entfernten Datenbank definiert sind, können Sie die Optionen auf dieser Registerkarte leer lassen.

3. Ausführen der Beispieldatenbank unter Mac OS X.

- a. Rufen Sie das *sa_config*-Konfigurationsskript auf.

Weitere Hinweise finden Sie unter „Unix- und Mac OS X-Umgebungsvariablen“ [[SQL Anywhere Server - Datenbankadministration](#)].

- b. Richten Sie eine ODBC-Datenquelle ein. Beispiel:

```
dbdsn -w "SQL Anywhere 16 Demo"  
-c "UID=DBA;PWD=sql;DBF=/Applications/SQLAnywhere12/System/demo.db"
```

- c. Führen Sie den MobiLink-Server aus. Beispiel:

```
mlsrv16 -c "DSN=SQL Anywhere 16 Demo"
```

Ergebnisse

Der MobiLink-Server und der SQL Anywhere MobiLink-Client werden unter Mac OS X ausgeführt.

Versionshinweise

Damit dbmlsync einwandfrei ausgeführt werden kann, müssen die Haupt- und Unterversion von dbmlsync.exe mit jenen des Datenbankservers übereinstimmen. Außerdem muss die Hauptversion der Datenbankdatei mit der Version von dbmlsync.exe übereinstimmen, und die Unterversion der Datenbankdatei darf nicht höher als die Unterversion von *dbmlsync.exe* sein. Die Version der Datenbankdatei ist die höchste Version, auf die sie aktualisiert wurde.

Beispiel: Die Version 9.0.2 von dbmlsync darf nur zusammen mit der Version 9.0.2 des Datenbankservers (*dbeng9.exe*) verwendet werden. Sie kann mit Datenbankdateien der Versionen 9.0.0, 9.0.1 und 9.0.2 ausgeführt werden.

MobiLink SQL Anywhere Client-Dienstprogramm (dbmlsync)

dbmlsync-Syntax

Verwenden Sie das Dienstprogramm dbmlsync, um entfernte SQL Anywhere-Datenbanken mit einer konsolidierten Datenbank zu synchronisieren. Um dbmlsync ausführen zu können, müssen Sie das SYS_RUN_REPLICATION_ROLE-Systemprivileg haben.

Syntax

dbmlsync [*options*] [*transaction-logs-directory*]

Option	Beschreibung
@data	Liest Optionen aus der angegebenen Umgebungsvariablen oder Konfigurationsdatei ein. Siehe „@data dbmlsync-Option“ auf Seite 111.

Option	Beschreibung
-a	Fordert bei Fehler nicht erneut zur Eingabe auf. Siehe „ dbmlsync-Option -a “ auf Seite 112.
-ap <i>parameter,...</i>	Gibt Authentifizierungsparameter an. Siehe „ dbmlsync-Option -ap “ auf Seite 112.
-ba <i>filename</i>	Übernimmt eine Download-Datei. Siehe „ dbmlsync-Option -ba “ auf Seite 112.
-bc <i>filename</i>	Eine Download-Datei erstellen Siehe „ dbmlsync-Option -bc “ auf Seite 113.
-be <i>string</i>	Fügt eine Zeichenfolge hinzu, wenn eine Download-Datei erstellt wird. Siehe „ dbmlsync-Option -be “ auf Seite 113.
-bg	Passt eine Download-Datei für entfernte Datenbanken an, wenn eine Download-Datei erstellt wird. Siehe „ dbmlsync-Option -bg “ auf Seite 114.
-bk	Aktiviert die Hintergrundsynchroisation. Siehe „ dbmlsync-Option -bk “ auf Seite 115.
-bkr <i>number</i>	Legt fest, wie oft dbmlsync erneut versucht, die Synchronisation auszuführen, nachdem eine Hintergrundsynchroisation unterbrochen wurde. Siehe „ dbmlsync-Option -bkr “ auf Seite 115.
-c <i>connection-string</i>	Geben Sie Parameter für die Verbindung zur entfernten Datenbank an und zwar in der Form <i>parm1=value1; parm2=value2 ...</i> Andernfalls erscheint ein Fenster und Sie müssen die Verbindungsinformationen eingeben. Siehe „ dbmlsync-Option -c “ auf Seite 116.
-ci <i>size</i>	Legt die anfängliche Größe des dbmlsync-Cache fest. Siehe „ dbmlsync-Option -ci “ auf Seite 116.
-cl <i>size</i>	Legen Sie die Mindestgröße der dbmlsync-Cachedatei fest. Siehe „ dbmlsync-Option -cl “ auf Seite 117.
-cm <i>size</i>	Legen Sie die maximale Größe für die dbmlsync-Cachedatei fest. Siehe „ dbmlsync-Option -cm “ auf Seite 117.
-d	Löscht alle anderen Verbindungen zur Datenbank, deren Sperren mit den zu synchronisierenden Artikeln kollidieren. Siehe „ dbmlsync-Option -d “ auf Seite 118.
-dc	Setzt einen fehlgeschlagenen Download fort. Siehe „ dbmlsync-Option -dc “ auf Seite 118.

Option	Beschreibung
-dl	Zeigt Logmeldungen im dbmlsync-Meldungsfenster an. Siehe „dbmlsync-Option -dl“ auf Seite 119 .
-do	Deaktiviert das Durchsuchen von Offline-Transaktionslogs. Diese Dienstprogrammoption kann nicht mit der Dienstprogrammoption -x verwendet werden. Siehe „dbmlsync-Option -do“ auf Seite 119 .
-drs bytes	Legt bei neu startbaren Downloads den maximalen Umfang der Daten fest, die nach einem Kommunikationsfehler erneut gesendet werden müssen. Siehe „dbmlsync-Option -drs“ auf Seite 120 .
-ds	Führt eine reine Download-Synchronisation durch. Siehe „dbmlsync-Option -ds“ auf Seite 120 .
-e "keyword=value"...	Gibt erweiterte Optionen an. Siehe „Erweiterte Optionen von MobiLink SQL Anywhere-Clients“ auf Seite 143 .
-eh	Ignoriert alle Fehler, die in Hook-Funktionen auftreten. Siehe „dbmlsync-Option -eh“ auf Seite 122 .
-ek Schlüssel	Geben Sie den Chiffrierschlüssel der entfernten Datenbank an. Siehe „dbmlsync-Option -ek“ auf Seite 122 .
-ep "keyword=value"	Fordert zur Eingabe des Chiffrierschlüssels auf. Siehe „dbmlsync-Option -ep“ auf Seite 123 .
-eu	Gibt erweiterte Optionen für den letzten von der Option -n definierten Upload an. Siehe „dbmlsync-Option -eu“ auf Seite 123 .
-is	Ignoriert die Abfolgeplanung. Siehe „dbmlsync-Option -is“ auf Seite 123 .
-k	Schließt das Fenster nach Ausführung. Siehe „dbmlsync-Option -k (nicht mehr empfohlen)“ auf Seite 124 .
-l	Listet verfügbare erweiterte Optionen auf. Siehe „dbmlsync-Option -l“ auf Seite 124 .
-mn password	Gibt ein neues MobiLink-Kennwort an. Siehe „dbmlsync-Option -mn“ auf Seite 124 .
-mp password	Gibt ein MobiLink-Kennwort an. Siehe „dbmlsync-Option -mp“ auf Seite 125 .
-n "name,..."	Gibt Synchronisationspublikationsnamen an. Siehe „dbmlsync-Option -n (nicht mehr empfohlen)“ auf Seite 125 .

Option	Beschreibung
-o <i>filename</i>	Protokolliert Ausgabemeldungen in dieser Datei. Siehe „ dbmlsync-Option -o “ auf Seite 126.
-os <i>size</i>	Gibt eine Maximalgröße für die Meldungslogdatei an, bei der das Log umbenannt wird. Siehe „ dbmlsync-Option -os “ auf Seite 127.
-ot <i>Logdatei</i>	Löscht den Inhalt der Meldungslogdatei und fügt anschließend die ausgegebenen Meldungen an. Siehe „ dbmlsync-Option -ot “ auf Seite 127.
-p	Deaktiviert den Logscan-Abruf. Siehe „ dbmlsync-Option -p “ auf Seite 127.
-pc [+ -]	Erhält eine offene Verbindung mit dem MobiLink-Server zwischen den Synchronisationen aufrecht. Siehe „ dbmlsync-Option -pc “ auf Seite 128.
-pd <i>dllname</i> ;...	Lädt angegebene DLLs für Windows Mobile im Voraus. Siehe „ dbmlsync-Option -pd “ auf Seite 129.
-pi	Testet, ob Sie sich mit MobiLink verbinden können. Siehe „ dbmlsync-Option -pi “ auf Seite 129.
-po <i>port</i>	Gibt den Port an, auf dem dbmlsync auf Verbindungsanforderungen wartet. Siehe „ dbmlsync-Option -po “ auf Seite 130.
-pp <i>number</i>	Legt die Periode für den Logscan-Abruf fest. Siehe „ dbmlsync-Option -pp “ auf Seite 130.
-q	Führt mit minimiertem Fenster aus. Siehe „ dbmlsync-Option -q “ auf Seite 131.
-qc	Fährt dbmlsync herunter, wenn die Synchronisation beendet ist. Siehe „ dbmlsync-Option -qc “ auf Seite 131.
-qi	Startet dbmlsync im dialogfreien Modus, wobei das Fenster vollständig ausgeblendet wird. Siehe „ dbmlsync-Option -qi “ auf Seite 131.
-r [a b]	Verwendet Client-Fortschrittswerte für einen Upload-Neustart. Siehe „ dbmlsync-Option -r “ auf Seite 132.
-s <i>name</i>	Gibt Synchronisationssubskriptionsnamen an. Siehe „ dbmlsync-Option -s “ auf Seite 133.
-sc	Lädt Schemainformationen vor jeder Synchronisation neu. Siehe „ dbmlsync-Option -sc “ auf Seite 134.

Option	Beschreibung
-sm	Mit dieser Option wird dbmlsync im Servermodus gestartet. Siehe „dbmlsync-Option -sm“ auf Seite 134.
-sp <i>sync-profile</i>	Fügt den in der Befehlszeile angegebenen Synchronisationsoptionen weitere Optionen aus dem Synchronisationsprofil hinzu. Siehe „dbmlsync-Option -sp“ auf Seite 135.
-ts <i>session-name(session-option=[option-value;...])</i>	Richtet eine Protokollierungssitzung für den MobiLink-Client ein. Siehe „dbmlsync-Option -ts“ auf Seite 135.
-tu	Führt einen transaktionalen Upload durch. Siehe „dbmlsync-Option -tu“ auf Seite 136.
-u <i>ml_username</i>	Legt den zu synchronisierenden MobiLink-Benutzer fest. Siehe „dbmlsync-Option -u (nicht mehr empfohlen)“ auf Seite 138.
-ud	Nur für Unix. Führt dbmlsync als Daemon aus. Siehe „dbmlsync-Option -ud“ auf Seite 139.
-ui	Startet dbmlsync unter Linux mit X Window im Shell-Modus, wenn keine nutzbare Anzeige verfügbar ist. Siehe „dbmlsync-Option -ui“ auf Seite 139.
-uo	Führt eine reine Upload-Synchronisation durch. Siehe „dbmlsync-Option -uo“ auf Seite 139.
-urc <i>row-estimate</i>	Legt eine geschätzte Zeilenanzahl für den Upload fest. Siehe „dbmlsync-Option -urc“ auf Seite 140.
-ux	Öffnet unter Linux und Solaris das dbmlsync-Meldungsfenster. Siehe „dbmlsync-Option -ux“ auf Seite 140.
-v [<i>Stufen</i>]	Vorgang ausführlich darstellen. Siehe „dbmlsync-Option -v“ auf Seite 141.
-wc <i>classname</i>	Legt einen Fensterklassennamen fest. Siehe „dbmlsync-Option -wc“ auf Seite 142.
-x [<i>size</i>]	Benennt das Transaktionslog um und startet es neu. Verwenden Sie die optionalen size-Parameter mit der Befehlszeilenoption -x, um die Größe des Transaktionslogs zu steuern. Siehe „dbmlsync-Option -x“ auf Seite 143.
<i>transaction-logs-directory</i>	Legt den Standort des Transaktionslogs fest. Siehe unter "Transaktionslogsdatei" weiter unten.

Bemerkungen

Führen Sie dbmlsync aus, um eine entfernte SQL Anywhere-Datenbank mit einer konsolidierten Datenbank zu synchronisieren.

Um den MobiLink-Server zu finden und sich mit ihm zu verbinden, nutzt dbmlsync die Informationen aus Publikation, Synchronisationsbenutzer und Synchronisationssubskription oder aus der dbmlsync-Befehlszeile.

Transaktionslogdatei Das *transaction-logs-directory* ist das Verzeichnis, das das Transaktionslog für die entfernte SQL Anywhere-Datenbank enthält. Es gibt ein aktives Transaktionslog und null oder mehr Transaktionslog-Archivdateien, die dbmlsync gegebenenfalls benötigt, um festzulegen, welche Daten für den Upload ausgewählt werden sollen. Sie müssen diesen Parameter angeben, falls alle nachstehenden Elemente zutreffen:

- Der Inhalt der aktiven Logdatei wurde gekürzt und die Datei wurde seit der letzten Synchronisation umbenannt.
- Sie führen das Dienstprogramm dbmlsync nicht in dem Verzeichnis aus, in dem die umbenannten Logdateien gespeichert sind.

dbmlsync-Ereignis-Hooks Sie können einige gespeicherte dbmlsync-Clientprozeduren verwenden, um den Synchronisationsprozess anzupassen.

dbmlsync verwenden Weitere Hinweise über dbmlsync finden Sie unter „[Initiieren einer Synchronisation](#)“ auf Seite 88.

Siehe auch

- „GRANT ROLE SYS_RUN_REPLICATION_ROLE-Anweisung [MobiLink] [SQL Remote]“ [[SQL Anywhere Server - SQL-Referenzhandbuch](#)]
- „Transaktionslogdateien“ auf Seite 93
- „Ereignis-Hooks für SQL Anywhere-Clients“ auf Seite 202
- „Initiieren einer Synchronisation“ auf Seite 88
- „Ereignis-Hooks für SQL Anywhere-Clients“ auf Seite 202
- „Dbmlsync-API“ auf Seite 103
- „DBTools-Schnittstelle für dbmlsync“ auf Seite 317

@data dbmlsync-Option

Liest Optionen aus der angegebenen Umgebungsvariablen oder Konfigurationsdatei ein

Syntax

dbmlsync @data ...

Bemerkungen

Mit dieser Option können Sie Befehlszeilenoptionen in einer Umgebungsvariablen oder Konfigurationsdatei verfügbar machen. Wenn beide mit dem von Ihnen angegebenen Namen vorhanden sind, wird die Umgebungsvariable verwendet.

Weitere Hinweise über Konfigurationsdateien finden Sie unter „[Konfigurationsdateien](#)“ [*SQL Anywhere Server - Datenbankadministration*].

Wenn Sie Kennwörter oder andere Informationen in einer Konfigurationsdatei schützen möchten, können Sie das Dienstprogramm zum Ausblenden von Dateien verwenden, um den Inhalt von Konfigurationsdateien zu verschleiern.

Siehe „[Dienstprogramm zum Verschleiern von Dateien \(dbfhide\)](#)“ [*SQL Anywhere Server - Datenbankadministration*].

dbmlsync-Option -a

Bestimmte Arten von Fehlern (wie etwa ein falsches MobiLink-Kennwort) führen in der Regel dazu, dass dbmlsync den Benutzer in einem Fenster zur Eingabe der richtigen Werte auffordert. Die Option -a verhindert solche Aufforderungen.

Syntax

dbmlsync -a ...

dbmlsync-Option -ap

Übergibt Parameter an das Skript `authenticate_parameters` und an Authentifizierungsparameter auf dem MobiLink-Server.

Syntax

dbmlsync -ap "parameters,..." ...

Bemerkungen

Benutzen Sie diese Option mit dem Verbindungsskript `authenticate_parameters` oder Authentifizierungsparametern auf dem Server. Beispiel:

```
dbmlsync -ap "parm1,parm2,parm3"
```

Die Parameter werden an den MobiLink-Server geschickt und an das `authenticate_parameters`-Skript oder andere Ereignisse in der konsolidierten Datenbank übergeben.

Siehe auch

- „[Authentifizierungsparameter](#)“ [*MobiLink - Serveradministration*]
- „[authenticate_parameters](#) (Verbindungsereignis)“ [*MobiLink - Serveradministration*]
- „[AuthParms-Synchronisationsprofiloption](#)“ auf Seite 186

dbmlsync-Option -ba

Übernimmt eine Download-Datei.

Syntax

dbmlsync -ba "filename" ...

Bemerkungen

Geben Sie den Namen einer bestehenden Download-Datei an, die in die entfernte Datenbank übernommen werden soll. Optional können Sie auch einen Pfad eingeben. Wenn Sie keinen Pfad eingeben, ist der Standardspeicherort das Verzeichnis, von dem aus dbmlsync gestartet wurde.

Siehe auch

- „MobiLink - dateibasierte Downloads“ [[MobiLink - Serveradministration](#)]
- „dbmlsync-Option -bc“ auf Seite 113
- „dbmlsync-Option -be“ auf Seite 113
- „dbmlsync-Option -bg“ auf Seite 114
- „ApplyDnldFile-Synchronisationsprofiloption“ auf Seite 186

dbmlsync-Option -bc

Erstellt eine Download-Datei.

Syntax

dbmlsync -bc "filename" ...

Bemerkungen

Erstellen Sie eine Download-Datei mit dem angegebenen Namen. Für Download-Dateien müssen Sie die Erweiterung *.df* verwenden.

Optional können Sie auch einen Pfad eingeben. Wenn Sie keinen Pfad eingeben, ist der Standardort das aktuelle Arbeitsverzeichnis von dbmlsync (das Verzeichnis, in dem dbmlsync gestartet wurde).

Sie können auch die Option -be verwenden, um eine Zeichenfolge anzugeben, die in der entfernten Datenbank validiert wird, sowie die Option -bg, um eine Download-Datei für eine neue entfernte Datenbank anzulegen.

Siehe auch

- „MobiLink - dateibasierte Downloads“ [[MobiLink - Serveradministration](#)]
- „dbmlsync-Option -ba“ auf Seite 112
- „dbmlsync-Option -be“ auf Seite 113
- „dbmlsync-Option -bg“ auf Seite 114
- „CreateDnldFile-Synchronisationsprofiloption“ auf Seite 188

dbmlsync-Option -be

Beim Erstellen einer Download-Datei gibt diese Option eine zusätzliche Zeichenfolge an, die in die Datei aufgenommen werden soll.

Syntax

dbmlsync -bc "filename" -be "string" ...

Bemerkungen

Die Zeichenfolge kann für die Authentifizierung oder andere Zwecke verwendet werden. Sie wird an die gespeicherte Prozedur `sp_hook_dbmlsync_validate_download_file` in der entfernten Datenbank übergeben, wenn die Download-Datei übernommen wird.

Siehe auch

- „`sp_hook_dbmlsync_validate_download_file`“ auf Seite 264
- „MobiLink - dateibasierte Downloads“ [*MobiLink - Serveradministration*]
- „dbmlsync-Option -bc“ auf Seite 113
- „dbmlsync-Option -ba“ auf Seite 112
- „DnldFileExtra-Synchronisationsprofiloption“ auf Seite 189

dbmlsync-Option -bg

Beim Erstellen einer Download-Datei erstellt diese Option eine Datei, die von noch nicht synchronisierten entfernten Datenbanken verwendet werden kann.

Syntax

dbmlsync -bc "filename" -bg ...

Bemerkungen

Die Option -bg sorgt dafür, dass die Download-Datei die Generationsnummern in der entfernten Datenbank aktualisiert.

Mit dieser Option können Sie eine Download-Datei einrichten, die in bisher noch nie synchronisierten entfernten Datenbanken übernommen werden kann. Sonst müssen Sie eine Synchronisation vornehmen, bevor Sie eine Download-Datei übernehmen.

Download-Dateien, die mit der Option -bg erstellt wurden, sollten Snapshot-Downloads sein. Zeitstempelbasierte Downloads funktionieren nicht mit entfernten Datenbanken, die nicht synchronisiert wurden. Dies liegt daran, dass der Zeitstempel des letzten Downloads bei einer neuen entfernten Datenbank standardmäßig auf den 1. Januar 1900 gesetzt wird und damit vor dem Zeitstempel des letzten Downloads in der Download-Datei liegt. Damit zeitstempelbasierte dateibasierte Downloads funktionieren, darf der Zeitstempel des letzten Downloads in der Download-Datei höchstens gleich demjenigen in der entfernten Datenbank sein.

Übernehmen Sie keine mit der Option -bg erstellten Download-Dateien in entfernte Datenbanken, die bereits synchronisiert wurden, wenn Ihr System von Funktionalitäten abhängt, die durch Generierungsnummern gesteuert werden, da diese Option diese Funktionalität umgeht.

Siehe auch

- „MobiLink - dateibasierte Downloads“ [[MobiLink - Serveradministration](#)]
- „dbmlsync-Option -ba“ auf Seite 112
- „dbmlsync-Option -bc“ auf Seite 113
- „MobiLink-Generierungsnummern“ [[MobiLink - Serveradministration](#)]
- „Synchronisation neuer entfernter Datenbanken“ [[MobiLink - Serveradministration](#)]
- „UpdateGenNum-Synchronisationsprofiloption“ auf Seite 199

dbmlsync-Option -bk

Aktiviert die Hintergrundsynchro­nisation.

Syntax

dbmlsync -bk "*connection-string*" ...

Bemerkungen

Während einer Hintergrundsynchro­nisation trennt die Datenbank-Engine die dbmlsync-Verbindung mit der entfernten Datenbank und setzt alle noch nicht festgeschriebenen dbmlsync-Vorgänge zurück, wenn eine andere Verbindung darauf wartet, auf eine von dbmlsync gesperrte Datenbankressource zugreifen zu können. Dadurch können andere Verbindungen genutzt werden, ohne auf den Abschluss der Synchronisation zu warten. Abhängig von den Vorgängen, die für dbmlsync ausstanden, als die Verbindung getrennt wurde, kann es trotzdem zu einer längeren Verzögerung für die wartende Verbindung kommen, da die Datenbank die nicht festgeschriebenen dbmlsync-Änderungen zurücksetzt.

Wenn die dbmlsync-Verbindung getrennt wird, schlägt die gerade ausgeführte Synchronisation fehl und gibt Fehler aus.

Siehe auch

- „dbmlsync-Option -bkr“ auf Seite 115
- „MobiLink-Synchronisationsprofile“ auf Seite 182
- „Background-Synchronisationsprofiloption“ auf Seite 186

dbmlsync-Option -bkr

Steuert das Verhalten von dbmlsync nach der Unterbrechung einer Hintergrundsynchro­nisation.

Syntax

dbmlsync -bkr *num*...

Bemerkungen

num ist eine Ganzzahl größer oder gleich -1.

Wenn *num* -1 ist, dann wiederholt dbmlsync eine unterbrochene Synchronisation, bis sie ohne Unterbrechung mit oder ohne Erfolg abgeschlossen werden kann. Wenn *num* 0 ist, dann wiederholt dbmlsync die unterbrochene Synchronisation nicht. Wenn *num* größer als 0 ist, wiederholt dbmlsync die

Synchronisation so oft, wie durch *num* angegeben ist. Wenn die Synchronisation nach der angegebenen *num* von Versuchen noch nicht abgeschlossen wurde, wird sie als Vordergrundssynchronisation ausgeführt, damit sie ohne Unterbrechung abgeschlossen werden kann.

Standardmäßig ist BackgroundRetry auf 0 gesetzt. Es wird ein Fehler erzeugt, wenn BackgroundRetry auf einen anderen Wert als null gesetzt wird, ohne die Background-Option auf TRUE zu setzen. Siehe „dbmlsync-Option -bk“ auf Seite 115.

BackgroundRetry wird ignoriert, wenn die dbmlsync-API oder die SQL SYNCHRONIZE-Anweisung verwendet wird.

Siehe auch

- „dbmlsync-Option -bk“ auf Seite 115
- „BackgroundRetry-Synchronisationsprofiloption“ auf Seite 187

dbmlsync-Option -c

Legt die Parameter für die entfernte Datenbank fest

Syntax

dbmlsync -c "*connection-string*" ...

Bemerkungen

Die Verbindungszeichenfolge muss dbmlsync die Berechtigung erteilen, eine Verbindung mit der entfernten SQL Anywhere-Datenbank als Benutzer mit der SYS_REPLICATION_ADMIN_ROLE-Systemrolle oder gleichwertigen Privilegien herzustellen.

Geben Sie die Verbindungszeichenfolge in der Form *keyword=value* ein, wobei mehrere Parameter durch Semikola getrennt werden. Wenn einer der Parameternamen Leerzeichen enthält, müssen Sie die Verbindungszeichenfolge zwischen Anführungszeichen setzen.

Wenn Sie -c nicht angeben, erscheint ein dbmlsync-Setupfenster. Sie können die verbleibenden Befehlszeilenoptionen in den Feldern des Verbindungsfensters angeben.

Eine komplette Liste der Verbindungsparameter für die Verbindung mit SQL Anywhere-Datenbanken finden Sie unter „Verbindungsparameter“ [[SQL Anywhere Server - Datenbankadministration](#)].

dbmlsync-Option -ci

Legt die anfängliche Größe des dbmlsync-Cache fest.

Syntax

dbmlsync -ci *size* [**K** | **M** | **P**]...

Bemerkungen

Die *size* ist die anfängliche Cachegröße in Byte, die von dbmlsync zum Speichern von Synchronisationsdaten benutzt wird. Optional können Sie das Suffix "K" oder "M" verwenden, um kB bzw. MB als Einheit anzugeben.

Um die Größe als einen Prozentsatz des gesamten physischen Speichers im System anzugeben, geben Sie eine Zahl zwischen 0 und 100 ein, gefolgt von dem Buchstaben p. Mit `-ci 30p` wird die Cachegröße beispielsweise auf 30% des physischen Speichers gesetzt.

Siehe auch

- „dbmlsync-Option -cm“ auf Seite 117
- „dbmlsync-Option -cl“ auf Seite 117

dbmlsync-Option -cl

Legen Sie die Mindestgröße fest, auf die die dbmlsync-Cachedatei reduziert werden kann.

Syntax

dbmlsync -cl *size* [**K** | **M** | **P**]...

Bemerkungen

Die *size* ist die kleinste Größe (in Byte), auf die der dbmlsync-Cache reduziert werden kann. Optional können Sie das Suffix "K" oder "M" verwenden, um kB bzw. MB als Einheit anzugeben.

Um die Größe als einen Prozentsatz des gesamten physischen Speichers im System anzugeben, geben Sie eine Zahl zwischen 0 und 100 ein, gefolgt von dem Buchstaben p. Mit `-cl 5p` wird beispielsweise sichergestellt, dass die Cachegröße nicht weniger als 5% des physischen Speichers beträgt.

Siehe auch

- „dbmlsync-Option -cm“ auf Seite 117
- „dbmlsync-Option -ci“ auf Seite 116

dbmlsync-Option -cm

Legen Sie die maximale Größe für den dbmlsync-Cache fest.

Syntax

dbmlsync -cm *size* [**K** | **M** | **P**]...

Bemerkungen

Die *size* ist die höchste Größe (in Byte), auf die der dbmlsync-Cache anwachsen kann. Optional können Sie das Suffix "K" oder "M" verwenden, um kB bzw. MB als Einheit anzugeben.

Um die Größe als einen Prozentsatz des gesamten physischen Speichers im System anzugeben, geben Sie eine Zahl zwischen 0 und 100 ein, gefolgt von dem Buchstaben p. Mit `-cm 60p` wird die maximale Cachegröße beispielsweise auf 60% des physischen Speichers begrenzt.

Siehe auch

- „dbmlsync-Option -ci“ auf Seite 116
- „dbmlsync-Option -cl“ auf Seite 117

dbmlsync-Option -d

Löscht kollidierende Sperren in der entfernten Datenbank.

Syntax

dbmlsync -d ...

Bemerkungen

In Fällen, in denen dbmlsync zu synchronisierende Tabellen sperren muss, die bereits von einer anderen Verbindung gesperrt werden, kann die Synchronisation fehlschlagen oder verzögert werden. Wenn Sie diese Option angeben, zwingen Sie SQL Anywhere, alle anderen Verbindungen zur entfernten Datenbank zu löschen, die Konfliktsperren beinhalten, damit die Synchronisation sofort fortgesetzt werden kann.

Siehe auch

- „Parallelität während der Synchronisation“ auf Seite 94
- „KillConnections-Synchronisationsprofiloption“ auf Seite 192

dbmlsync-Option -dc

Startet einen fehlgeschlagenen Download erneut

Syntax

dbmlsync -dc ...

Bemerkungen

Falls dbmlsync während eines Downloads einen Fehler feststellt, wird standardmäßig kein Teil des Downloads in die entfernte Datenbank übernommen. Der empfangene Teil des Downloads wird jedoch in einer temporären Datei auf dem entfernten Gerät gespeichert, sodass Sie den Download bei der nächsten Synchronisation mit `-dc` schneller abschließen können. Bei der Verwendung von `-dc` startet dbmlsync den Download neu und versucht, den Teil des Downloads einzulesen, der beim letzten Downloadversuch nicht empfangen wurde. Wenn die verbliebenden Daten eingelesen werden können, wird der gesamte Download in Ihre entfernte Datenbank übernommen. Andernfalls schlägt die Synchronisation fehl.

Falls Sie `-dc` verwenden und neue Daten für die Übertragung anstehen, schlägt der neu startbare Download fehl.

Sie können auch einen fehlgeschlagenen Netzwerk-Download mit der erweiterten Option ContinueDownload oder dem Hook sp_hook_dbmlsync_end neu starten.

Siehe auch

- „Wiederaufnahme fehlgeschlagener Downloads“ [*MobiLink - Serveradministration*]
- „Erweiterte Option ContinueDownload (cd)“ auf Seite 151
- „sp_hook_dbmlsync_end“ auf Seite 232
- „Erweiterte Option DownloadReadSize (drs)“ auf Seite 154
- „ContinueDownload-Synchronisationsprofiloption“ auf Seite 188

dbmlsync-Option -dl

Zeigt die Meldungen im dbmlsync-Meldungsfenster oder an der Eingabeaufforderung und in der Meldungslogdatei an.

Syntax

dbmlsync -dl ...

Bemerkungen

Wenn die Ausgabe in einer Datei protokolliert wird, schreibt das System normalerweise mehr Meldungen in die Logdatei als in das dbmlsync-Fenster. Diese Option zwingt dbmlsync, Informationen, die normalerweise nur in die Datei geschrieben werden, auch am Bildschirm auszugeben. Wenn Sie diese Option verwenden, verringert sich möglicherweise die Geschwindigkeit der Synchronisation.

dbmlsync-Option -do

Deaktiviert das Durchsuchen von Offline-Transaktionslogs.

Syntax

dbmlsync -do ...

Bemerkungen

Wenn Transaktionslogdateien für mehrere Datenbanken in einem einzelnen Verzeichnis gespeichert werden, kann mit dbmlsync möglicherweise von keiner dieser Datenbanken aus synchronisiert werden, selbst wenn für keine dieser Datenbanken eine Offline-Transaktionslogdatei vorhanden ist. Wenn dbmlsync mit der Option -d verwendet wird, versucht dbmlsync nicht, Offline-Transaktionslogs zu durchsuchen und kann mit einer Datenbank synchronisiert werden, die mit allen anderen Datenbanken in einem einzigen Verzeichnis gespeichert ist.

Wenn diese Option verwendet wird und Offline-Transaktionslogs erforderlich sind, kann dbmlsync keine Synchronisation ausführen.

Diese Option kann nicht mit der Option -x verwendet werden.

dbmlsync-Option -drs

Bei neu startbaren Downloads legt diese Option die maximale Anzahl von Byte fest, die nach einem Kommunikationsfehler erneut gesendet werden müssen.

Syntax

dbmlsync -drs *bytes* ...

Bemerkungen

Die Option -drs legt die Download-Lesegröße fest und ist nur bei der Verwendung von neu startbaren Downloads sinnvoll.

Dbmlsync liest den Download in Abschnitten. DownloadReadSize (Download-Lesegröße) legt die Größe dieser Abschnitte fest. Wenn ein Kommunikationsfehler auftritt, verliert dbmlsync den gesamten bearbeiteten Abschnitt. Abhängig davon, wann der Fehler auftritt, liegt die Anzahl der verlorenen Byte zwischen 0 und der Größe des Downloadlesevorgangs (DownloadReadSize) -1. Beispiel: Wenn DownloadReadSize 100 Byte beträgt und nach dem Lesen von 497 Byte ein Fehler auftritt, gehen die letzten 97 gelesenen Byte verloren. Auf diese Weise verlorene Byte werden erneut gesendet, wenn der Download erneut gestartet wird.

Im Allgemeinen führt eine größere Download-Lesegröße zu einer besseren Performance bei erfolgreichen Synchronisationen, aber zu einer Übertragung von mehr Daten, wenn ein Fehler auftritt.

Diese Option wird typischerweise verwendet, wenn die Standardgröße bei einer unzuverlässigen Verbindung verkleinert werden soll.

Der Standardwert ist **32767**. Wenn der Wert dieser Option über 32767 liegt, wird der Wert 32767 verwendet.

Sie können auch die Download-Lesegröße mit der erweiterten Option DownloadReadSize festlegen.

Siehe auch

- „Erweiterte Option DownloadReadSize (drs)“ auf Seite 154
- „Wiederaufnahme fehlgeschlagener Downloads“ [*MobiLink - Serveradministration*]
- „Erweiterte Option ContinueDownload (cd)“ auf Seite 151
- „sp_hook_dbmlsync_end“ auf Seite 232
- „dbmlsync-Option -dc“ auf Seite 118

Beispiel

Die folgende dbmlsync-Befehlszeile zeigt, wie Sie diese Option beim Start von dbmlsync verwenden können.

```
dbmlsync -drs 100
```

dbmlsync-Option -ds

Führt eine reine Download-Synchronisation durch.

Syntax

dbmlsync -ds ...

Bemerkungen

Wenn eine reine Download-Synchronisation vorgenommen wird, führt dbmlsync keinen Upload von Datenbankänderungen durch. Allerdings wird ein Upload von Informationen über das Schema und das Offset für den Verarbeitungsfortschritt vorgenommen.

Außerdem sorgt dbmlsync dafür, dass Änderungen in der entfernten Datenbank während einer reinen Download-Synchronisation nicht überschrieben werden. Dazu wird das Log gescannt, um Zeilen zu ermitteln, für die Vorgänge auf einen Upload warten. Wenn eine dieser Zeilen durch den Download geändert würde, wird der Download zurückgesetzt und die Synchronisation schlägt fehl. Wenn die Synchronisation aus diesem Grund fehlschlägt, müssen Sie eine vollständige Synchronisation vornehmen, um das Problem zu beheben.

Wenn Sie mit entfernten Datenbanken arbeiten, die durch reine Download-Synchronisation synchronisiert werden, sollten Sie regelmäßig eine vollständige bidirektionale Synchronisation durchführen, um das Volumen des Logs zu verkleinern, das von der reinen Download-Synchronisation gescannt werden muss. Sonst würden die reinen Download-Synchronisationen zunehmend mehr Zeit brauchen.

Bei aktivierter Option -ds wird die erweiterte Option ConflictRetries ignoriert. Dbmlsync versucht nie, eine reine Download-Synchronisation nochmals durchzuführen. Wenn eine reine Download-Synchronisation fehlschlägt, wird sich dieser Fehlschlag wiederholen, bis eine vollständige Synchronisation durchgeführt wird.

Eine Liste aller Skripten, die für reine Download-Synchronisationen definiert werden müssen, finden Sie unter „Erforderliche Skripten“ [[MobiLink - Serveradministration](#)].

Siehe auch

- „Reine Upload- und reine Download-Synchronisation“ [[MobiLink - Serveradministration](#)]
- „Erweiterte Option DownloadOnly (ds)“ auf Seite 153
- „Reine Download-Publikationen“ auf Seite 78
- „DownloadOnly-Synchronisationsprofiloption“ auf Seite 190

dbmlsync-Option -e

Legt erweiterte Optionen fest.

Syntax

dbmlsync -e *extended-option=value*; ...

Bemerkungen

Erweiterte Optionen können in ihrer Langform oder ihrer Kurzform angegeben werden.

Mit der dbmlsync-Option -l erhalten Sie eine Liste aller erweiterten Optionen.

Optionen, die mit der Option -e in der Befehlszeile angegeben wurden, betreffen alle in der Befehlszeile angeforderten Synchronisationen. Im folgenden Beispiel für eine Befehlszeile bezieht sich die erweiterte Option drs=512 auf die Synchronisation von "sub1" und "sub2".

```
dbmlsync -e "drs=512" -s sub1 -s sub2
```

Sie können erweiterte Optionen im dbmlsync-Meldungslog und in der Systemansicht SYSSYNC prüfen.

Wenn Sie erweiterte Optionen für einen einzelnen Upload eingeben wollen, benutzen Sie die Option -eu.

Siehe auch

- „Erweiterte Optionen von MobiLink SQL Anywhere-Clients“ auf Seite 143
- „dbmlsync-Option -l“ auf Seite 124
- „dbmlsync-Option -eu“ auf Seite 123
- „SYSSYNC-Systemansicht“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- „sp_hook_dbmlsync_set_extended_options“ auf Seite 252
- „ExtOpt-Synchronisationsprofiloption“ auf Seite 191

Beispiel

Die folgende dbmlsync-Befehlszeile zeigt, wie Sie beim Start von dbmlsync erweiterte Optionen verwenden können.

```
dbmlsync -e "adr=host=localhost;dir=c:\db\logs"...
```

dbmlsync-Option -eh

Ignoriert alle Fehler, die in Hook-Funktionen auftreten.

Syntax

```
dbmlsync -eh ...
```

Siehe auch

- „IgnoreHookErrors-Synchronisationsprofiloption“ auf Seite 192

dbmlsync-Option -ek

Ermöglicht die Angabe des Chiffrierschlüssels für stark verschlüsselte Datenbanken direkt in der Befehlszeile.

Syntax

```
dbmlsync -ek key ...
```

Bemerkungen

Wenn Sie mit einer stark verschlüsselten entfernten Datenbank arbeiten, müssen Sie einen Chiffrierschlüssel für die Datenbank bzw. das Transaktionslog angeben, auch für Offline-Transaktionen. Bei stark verschlüsselten Datenbanken müssen Sie entweder -ek oder -ep, dürfen aber nicht beide

angeben. Wenn Sie bei einer stark verschlüsselten Datenbank keinen Schlüssel angeben, schlägt der Befehl fehl.

dbmlsync-Option -ep

Fordert zur Eingabe des Chiffrierschlüssels für die entfernte Datenbank auf.

Syntax

dbmlsync -ep ...

Bemerkungen

Diese Option öffnet ein Fenster, in das Sie den Chiffrierschlüssel eingeben. Diese zusätzliche Sicherheitsmaßnahme verhindert, dass der Chiffrierschlüssel in lesbarer Form angezeigt wird. Bei stark verschlüsselten entfernten Datenbanken müssen Sie entweder -ek oder -ep angeben, dürfen aber nicht beides verwenden. Wenn Sie bei einer stark verschlüsselten Datenbank keinen Schlüssel angeben, schlägt der Befehl fehl.

dbmlsync-Option -eu

Gibt erweiterte Upload-Optionen an

Syntax

dbmlsync -s *subscription-name* -eu keyword=value;...

dbmlsync -n *publication-name* -eu keyword=value;...

Bemerkungen

Erweiterte Optionen, die in der Befehlszeile mit der Option -eu angegeben wurden, betreffen nur die mit der Option -n oder -s angegebene Synchronisation, der sie folgen. In der folgenden Zeile bezieht sich die erweiterte Option eh=on nur auf die Synchronisation von Subskription "sub2".

```
dbmlsync -s sub1 -s sub2 -eu eh=on
```

Ausführliche Erläuterungen dazu, wie erweiterte Optionen verarbeitet werden, wenn sie an mehr als einer Stelle gesetzt wurden, sowie eine vollständige Liste der erweiterten Optionen finden Sie unter „[Erweiterte Optionen von MobiLink SQL Anywhere-Clients](#)“ auf Seite 143.

dbmlsync-Option -is

Ignoriert die erweiterte Option für die Abfolgeplanung.

Syntax

dbmlsync -is ...

Bemerkungen

Ignorieren Sie erweiterte Optionen, die die Synchronisation zeitlich einplanen.

Weitere Hinweise zur Zeitplanung finden Sie unter „[Synchronisationszeitpläne](#)“ auf Seite 100.

Siehe auch

- „[IgnoreScheduling-Synchronisationsprofiloption](#)“ auf Seite 192

dbmlsync-Option -k (nicht mehr empfohlen)

Führt dbmlsync herunter, wenn die Synchronisation beendet ist. Dbmlsync wird bei einem Fehler während der Synchronisation nicht heruntergefahren, es sei denn eine der Optionen -c oder -ot ist ebenfalls angegeben.

Diese Option wird nicht mehr empfohlen. Benutzen Sie anstelle dessen -qc.

Syntax

dbmlsync -k ...

Siehe auch

- „[dbmlsync-Option -qc](#)“ auf Seite 131

dbmlsync-Option -l

Listet verfügbare erweiterte Optionen auf

Syntax

dbmlsync -l ...

Siehe auch

- „[Erweiterte Optionen von MobiLink SQL Anywhere-Clients](#)“ auf Seite 143

dbmlsync-Option -mn

Gibt ein neues Kennwort für den synchronisierten MobiLink-Benutzer an.

Syntax

dbmlsync -mn *password* ...

Bemerkungen

Ändert das Kennwort eines MobiLink-Benutzers

Siehe auch

- „MobiLink-Benutzer“ auf Seite 4
- „Erweiterte Option MobiLinkPwd (mp)“ auf Seite 162
- „Erweiterte Option NewMobiLinkPwd (mn)“ auf Seite 163
- „dbmlsync-Option -mp“ auf Seite 125
- „NewMobiLinkPwd-Synchronisationsprofiloption“ auf Seite 195

dbmlsync-Option -mp

Gibt das Kennwort des synchronisierten MobiLink-Benutzers an

Syntax

dbmlsync -mp *password* ...

Bemerkungen

Gibt das Kennwort für die Authentifizierung des MobiLink-Benutzers an

Siehe auch

- „MobiLink-Benutzer“ auf Seite 4
- „Erweiterte Option MobiLinkPwd (mp)“ auf Seite 162
- „Erweiterte Option NewMobiLinkPwd (mn)“ auf Seite 163
- „dbmlsync-Option -mn“ auf Seite 124

dbmlsync-Option -n (nicht mehr empfohlen)

Hinweis

Diese Option wird nicht mehr empfohlen. Verwenden Sie stattdessen die dbmlsync-Option -s. Siehe „dbmlsync-Option -s“ auf Seite 133.

Um die dbmlsync-Option -s zu verwenden, sollten Sie den Namen der Subskription, die Sie synchronisieren wollen, überprüfen. Sie können den Subskriptionsnamen mit der folgenden Abfrage ermitteln:

```
SELECT subscription_name
FROM syssync JOIN sys.syspublication
WHERE site_name = <ml_user> AND publication_name = <pub_name>;
```

Ersetzen Sie <ml_user> durch den MobiLink-Benutzer, den Sie synchronisieren. Hierbei handelt es sich um den Wert, der durch die Option -u der dbmlsync-Befehlszeile festgelegt wird. Siehe „dbmlsync-Option -u (nicht mehr empfohlen)“ auf Seite 138.

Ersetzen Sie <pub_name> durch den Namen der zu synchronisierenden Publikation. Hierbei handelt es sich um den Wert, der mit der Option -n in der dbmlsync-Befehlszeile angegeben wird. Siehe „dbmlsync-Option -n (nicht mehr empfohlen)“ auf Seite 125.

Gibt die zu synchronisierenden Publikationen an

Syntax

dbmlsync -n *pubname* ...

Bemerkungen

Sie können mehrere Optionen -n angeben, wenn mehr als eine Synchronisationspublikation synchronisiert werden soll.

Es gibt zwei Möglichkeiten, um mehrere Publikationen mithilfe der Option -n zu synchronisieren:

- Angabe von -n *pub1* , *pub2* , *pub3* zur Übertragung von *pub1*, *pub2* und *pub3* in einem Upload, gefolgt von einem Download.

In diesem Fall werden nur die Optionen verwendet, die für die erste Publikation in der Liste und deren Subskriptionen angegeben wurden, sofern Sie erweiterte Optionen für die Publikationen oder Subskriptionen festgelegt haben. Erweiterte Optionen für nachfolgende Publikationen und Subskriptionen werden ignoriert.

- Angabe von -n *pub1* -n *pub2* -n *pub3* zur Synchronisation von *pub1*, *pub2* und *pub3* in drei getrennten, aufeinander folgenden Synchronisationen.

Wenn aufeinander folgende Synchronisationen schnell hintereinander erfolgen, etwa wenn Sie -n *pub1* -n *pub2* angeben, kann es vorkommen, dass dbmlsync mit der Verarbeitung einer Synchronisation beginnt, während der Server noch mit der Verarbeitung der vorherigen Synchronisation beschäftigt ist. In diesem Fall schlägt die zweite Synchronisation mit einer Fehlermeldung fehl, die besagt, dass gleichzeitige Synchronisationen nicht zulässig sind. Wenn eine solche Situation eintritt, können Sie definieren, dass eine gespeicherte Prozedur `sp_hook_dbmlsync_delay` eine Verzögerung vor jeder Synchronisation durchführt. In der Regel reichen ein paar Sekunden bis zu einer Minute als Verzögerung aus.

Siehe auch

- „[sp_hook_dbmlsync_delay](#)“ auf Seite 218
- „[Publication-Synchronisationsprofiloption \(nicht mehr empfohlen\)](#)“ auf Seite 196

dbmlsync-Option -o

Gibt den Namen der dbmlsync-Meldungslogdatei an.

Syntax

dbmlsync -o *filename* ...

Bemerkungen

Ausgabe an eine Logdatei anfügen. Der Standardwert sendet die Ausgabe zum Bildschirm.

Siehe auch

- „[dbmlsync-Option -os](#)“ auf Seite 127
- „[dbmlsync-Option -ot](#)“ auf Seite 127

dbmlsync-Option -os

Gibt eine Maximalgröße für die dbmlsync-Meldungslogdatei an, bei der das Log umbenannt wird.

Syntax

dbmlsync -os *size* [**K** | **M** | **G**]...

Bemerkungen

Die *size* ist die maximale Dateigröße für dbmlsync-Meldungslogs, angegeben in Byte-Einheiten. Verwenden Sie das Suffix "k", "m" oder "g", um die Einheit in kB, MB oder GB anzugeben. Standardmäßig gibt es keine Größenbegrenzung. Die minimale Größe beträgt 10 kB.

Bevor das Dienstprogramm dbmlsync Ausgabemeldungen in einer Datei protokolliert, prüft es die aktuelle Dateigröße. Wenn aufgrund der Logmeldung die Dateigröße den festgelegten Wert überschreitet, benennt das Dienstprogramm dbmlsync die Ausgabedatei in *yymmddxx.dbs*, wobei *yymmdd* Jahr, Monat und Tag darstellt und *xx* eine Zahl ist, die bei 00 beginnt und fortlaufend erhöht wird.

Mit dieser Option können Sie alte Logdateien manuell löschen und Plattenspeicher frei geben.

Siehe auch

- „dbmlsync-Option -o“ auf Seite 126
- „dbmlsync-Option -ot“ auf Seite 127

dbmlsync-Option -ot

Löscht den Inhalt der angegebenen Datei und fügt anschließend die ausgegebenen Meldungen ein.

Syntax

dbmlsync -ot *logfile* ...

Bemerkungen

Die Funktion ist die gleiche wie bei der Option -o, außer dass der Inhalt der Meldungslogdatei beim Start von dbmlsync gelöscht wird, bevor Meldungen darin eingetragen werden.

Siehe auch

- „dbmlsync-Option -o“ auf Seite 126
- „dbmlsync-Option -os“ auf Seite 127

dbmlsync-Option -p

Deaktiviert den Logscan-Abruf

Syntax

dbmlsync -p ...

Bemerkungen

Um einen Upload einzurichten, muss dbmlsync das Transaktionslog scannen. In der Regel erfolgt dies zu Beginn der Synchronisation. Wenn Synchronisationen in einem Zeitplan ablaufen oder `sp_hook_dbmlsync_delay` benutzt wird, scannt dbmlsync das Log standardmäßig in der Pause, die vor der Synchronisation eintritt. Dieses Verhalten ist effizienter, da das Log zumindest bereits teilweise gescannt ist, wenn die Synchronisation beginnt. Dieses Standardverhalten wird als Logscan-Abruf (logscan polling) bezeichnet.

Logscan-Abruf ist standardmäßig aktiviert, hat aber nur dann Wirkung, wenn Synchronisationen mit der erweiterten Option für die Abfolgeplanung geplant sind oder der Hook `sp_hook_dbmlsync_delay` benutzt wird. Wenn der Abruf aktiviert ist, wird er in bestimmten Intervallen durchgeführt. Das Standardintervall ist 1 Minute, kann aber mit der dbmlsync-Option `-pp` geändert werden.

Diese Option ist identisch mit der erweiterten Option `DisablePolling=on`.

Siehe auch

- „Erweiterte Option `DisablePolling (p)`“ auf Seite 152
- „Erweiterte Option `PollingPeriod (pp)`“ auf Seite 165
- „dbmlsync-Option `-pp`“ auf Seite 130

dbmlsync-Option -pc

Erhält eine dauerhafte Verbindung mit dem MobiLink-Server zwischen den Synchronisationen aufrecht

Syntax

dbmlsync -pc [+ | -] ...

Bemerkungen

Wenn `-pc+` verwendet wird, verbindet sich dbmlsync normal mit dem MobiLink-Server. Die Verbindung bleibt anschließend jedoch für weitere Synchronisationen geöffnet. Eine dauerhafte Verbindung wird in folgenden Fällen abgebrochen:

- Eine Synchronisation schlägt aufgrund eines Fehlers fehl
- Zeitüberschreitung bei der Verfügbarkeitsüberprüfung
Siehe „[timeout](#)“ auf Seite 54.
- Eine Synchronisation wird eingeleitet, in der der Verbindungstyp oder die Adresse unterschiedlich sind. Dies kann sich auf unterschiedliche Einstellungen (z.B. ein anderer Host wurde festgelegt) oder auf unterschiedliche Festlegungen (z.B. gleicher Host und Port in unterschiedlicher Reihenfolge) beziehen.

Wenn eine dauerhafte Verbindung abgebrochen wird, wird eine neue, ebenfalls dauerhafte Verbindung hergestellt.

Die Option `-pc+` ist besonders hilfreich, wenn der Client häufig synchronisiert und die Kosten für das Herstellen einer Verbindung mit dem Server hoch sind.

Wenn -pc- verwendet wird, wird die Verbindung am Ende jeder Synchronisation geschlossen und am Anfang der nächsten Synchronisation erneut geöffnet. Standardmäßig werden dauerhafte Verbindungen nicht gepflegt.

dbmlsync-Option -pd

Lädt angegebene DLLs für Windows Mobile im Voraus.

Syntax

dbmlsync -pd *dllname*;

Bemerkungen

Wenn Sie dbmlsync unter Windows Mobile ausführen und verschlüsselte Kommunikationsdatenströme verwenden, müssen Sie die Option -pd benutzen, um sicherzustellen, dass die richtigen DLLs beim Start geladen werden. Sonst lädt dbmlsync die DLLs erst dann, wenn sie benötigt werden. Wenn diese DLLs zu spät geladen werden, kann es aufgrund der Ressourcenknappheit unter Windows Mobile zu einem Fehlschlag kommen.

Nachstehend finden Sie die DLLs, die für jedes Kommunikationsprotokoll geladen werden müssen:

Protokoll	DLL
RSA	mlcrsa16.dll
FIPS	mlcrsafips16.dll

Mehrere DLLs müssen in einer durch Semikola getrennten Liste angegeben werden. Beispiel:

```
-pd mlcrsafips16.dll;mlcrsa16.dll
```

dbmlsync-Option -pi

Pingt einen MobiLink-Server

Syntax

dbmlsync -pi -c *connection_string* ...

Bemerkungen

Wenn Sie die Option -pi verwenden, verbindet sich dbmlsync mit der entfernten Datenbank, ruft die erforderlichen Informationen ab, um sich mit dem MobiLink-Server zu verbinden, verbindet sich mit dem Server und authentifiziert den angegebenen MobiLink-Benutzer. Wenn der MobiLink-Server eine ping-Anfrage erhält, verbindet er sich mit der konsolidierten Datenbank, authentifiziert den Benutzer und sendet dann den Benutzerauthentifizierungs-Status und -Wert an den Client zurück. Wenn der MobiLink-Benutzername nicht in der Systemtabelle ml_user gefunden werden kann und der MobiLink-Server mit

der Befehlszeilenoption `-zu+` ausgeführt wird, fügt er den Benutzer der MobiLink-Systemtabelle `ml_user` hinzu.

Um die Verbindung adäquat zu testen, sollten Sie die Option `-pi` mit allen Synchronisationsoptionen nutzen, die Sie für die Synchronisation mit `dbmlsync` verwenden möchten. Wenn Sie `-pi` verwenden, führt `dbmlsync` keine Synchronisation durch.

Wird der `ping`-Parameter erfolgreich ausgeführt, gibt der MobiLink-Server eine Informationsmeldung aus. Wird der `ping`-Parameter erfolglos ausgeführt, gibt der Server eine Fehlermeldung aus.

Wenn Sie `dbmlsync` mit `-pi` starten, kann der MobiLink-Server folgende Skripten ausführen, sofern sie vorhanden sind:

- `begin_connection`
- `authenticate_user`
- `authenticate_user_hashed`
- `authenticate_parameters`
- `end_connection`

Siehe auch

- [„Ping-Synchronisationsprofiloption“ auf Seite 195](#)

dbmlsync-Option -po

Wenn `dbmlsync` im Servermodus ausgeführt wird, gibt diese Option den Port an, auf dem `dbmlsync` auf Verbindungen von Clients wartet.

Syntax

dbmlsync -po *port number* ...

Bemerkungen

Diese Option kann nur mit der Option `-sm` verwendet werden.

Siehe auch

- [„dbmlsync-Option -sm“ auf Seite 134](#)

dbmlsync-Option -pp

Gibt die Häufigkeit der Logscans an

Syntax

dbmlsync -pp *number* [**h** | **m** | **s**]...

Bemerkungen

Um einen Upload einzurichten, muss dbmlsync das Transaktionslog scannen. In der Regel erfolgt dies zu Beginn der Synchronisation. Wenn Synchronisationen in einem Zeitplan ablaufen oder `sp_hook_dbmlsync_delay` benutzt wird, scannt dbmlsync das Log standardmäßig in der Pause, die vor der Synchronisation eintritt. Dieses Verhalten ist effizienter, da das Log zumindest bereits teilweise gescannt ist, wenn die Synchronisation beginnt. Dieses Standardverhalten wird als Logscan-Abruf (logscan polling) bezeichnet.

Diese Option legt das Intervall zwischen Logscans fest. Benutzen Sie die Suffixe "s", "m", "h" oder "d", um Sekunden, Minuten, Stunden oder Tage anzugeben. Standardwert ist **1** Minute. Wenn Sie kein Suffix eingeben, wird als Standardzeiteinheit die Minute verwendet.

Siehe auch

- „Erweiterte Option `PollingPeriod (pp)`“ auf Seite 165
- „Erweiterte Option `DisablePolling (p)`“ auf Seite 152
- „dbmlsync-Option `-p`“ auf Seite 127

dbmlsync-Option -q

Startet den MobiLink-Synchronisationsclient in einem minimierten Fenster

Syntax

`dbmlsync -q ...`

dbmlsync-Option -qc

Führt dbmlsync herunter, wenn die Synchronisation beendet ist

Syntax

`dbmlsync -qc ...`

Bemerkungen

Bei dieser Option wird dbmlsync beendet, nachdem die Synchronisation abgeschlossen wurde, sofern sie erfolgreich war oder wenn eine Meldungslogdatei mit der Option `-o` oder `-ot` angegeben wurde.

Siehe auch

- „dbmlsync-Option `-o`“ auf Seite 126
- „dbmlsync-Option `-ot`“ auf Seite 127

dbmlsync-Option -qi

Steuert, ob das dbmlsync-Symbol auf der Taskleiste und das Meldungsfenster angezeigt werden.

Syntax

dbmlsync -qi ...

Bemerkungen

Bei dieser Option gibt es nach dem Start keinen Hinweis darauf, dass dbmlsync läuft, abgesehen von eventuellen Fehlermeldungen beim Start. Sie können die Logdateien für -o verwenden, um Fehler zu diagnostizieren.

Wenn die Option -qi angegeben ist, wird dbmlsync nach Abschluss der Synchronisation beendet.

Siehe auch

- „dbmlsync-Option -o“ auf Seite 126
- „dbmlsync-Option -ot“ auf Seite 127

dbmlsync-Option -r

Legt fest, dass der Offset der entfernten Datenbank verwendet werden soll, wenn die Offsets in der entfernten und der konsolidierten Datenbank nicht übereinstimmen.

Die Option -rb gibt an, dass der Offset der entfernten Datenbank benutzt werden soll, wenn er niedriger ist als der der konsolidierten Datenbank (z.B. wenn die entfernte Datenbank aus einer Sicherung wiederhergestellt wurde). Die Option -r wird für die Abwärtskompatibilität bereitgestellt und ist mit -rb identisch. Die Option -ra zeigt an, dass der Offset der entfernten Datenbank verwendet werden soll, wenn er höher ist als der der konsolidierten Datenbank. Diese Option ist nur in sehr seltenen Fällen anwendbar und kann zu Datenverlust führen.

Syntax

dbmlsync { -r | -ra | -rb } ...

Bemerkungen

Hinweise zu Offsets für den Verarbeitungsfortschritt finden Sie unter „[Offsets für den Verarbeitungsfortschritt](#)“ auf Seite 73.

-rb Wenn die entfernte Datenbank aus einer Sicherung wiederhergestellt wird, kann das Standardverhalten zu Datenverlust führen. In diesem Fall müssen Sie beim ersten Mal, wenn Sie dbmlsync nach der Wiederherstellung der entfernten Datenbank ausführen, die Option -rb verwenden. Wenn Sie -rb verwenden, wird der Upload ab dem Offset fortgesetzt, der in der entfernten Datenbank erfasst wurde, sofern dieser Offset kleiner ist als derjenige, der aus der konsolidierten Datenbank abgerufen wurde. Wenn Sie -rb verwenden und der Offset in der entfernten Datenbank nicht kleiner ist als der Offset aus der konsolidierten Datenbank, wird ein Fehler gemeldet und die Synchronisation wird abgebrochen.

Die Option -rb kann dazu führen, dass ein Upload von Daten erfolgt, die bereits versendet wurden. Daraus können sich Konflikte in der konsolidierten Datenbank ergeben, die mit entsprechenden Skripten zur Konfliktlösung verarbeitet werden sollten.

-ra Die Option -ra darf nur in seltenen Fällen verwendet werden. Wenn Sie -ra verwenden, wird der Upload ab dem Offset nochmals versucht, der aus der entfernten Datenbank bezogen wurde, wenn der Offset der entfernten Datenbank größer ist als der aus der konsolidierten Datenbank. Wenn Sie -ra verwenden und der Offset in der entfernten Datenbank nicht größer ist als der Offset aus der konsolidierten Datenbank, wird ein Fehler gemeldet und die Synchronisation wird abgebrochen.

Die Option -ra muss umsichtig eingesetzt werden. Wenn die Offsets aufgrund einer Wiederherstellung der konsolidierten Datenbank nicht übereinstimmen, gehen die Änderungen verloren, die in der entfernten Datenbank in der Lücke zwischen den beiden erfassten Offsets vorgenommen wurden. Die Option -ra kann sinnvoll sein, wenn die konsolidierte Datenbank aus einer Sicherung wiederhergestellt wurde und das Transaktionslog der entfernten Datenbank an derselben Stelle gekürzt wurde wie der Offset der entfernten Datenbank. In diesem Fall gehen alle Daten, für die ein Upload aus der entfernten Datenbank bereits erfolgt ist, ab dem Punkt in der konsolidierten Datenbank bis zu dem Punkt des Offsets der entfernten Datenbank verloren.

Siehe auch

- „RemoteProgressGreater-Synchronisationsprofiloption“ auf Seite 197
- „RemoteProgressLess-Synchronisationsprofiloption“ auf Seite 197

dbmlsync-Option -s

Legt die zu synchronisierende(n) Subskription(en) fest.

Syntax

dbmlsync -s *subname* ...

Bemerkungen

Diese Option ersetzt die dbmlsync-Option -n.

Es gibt zwei Möglichkeiten, um mehrere Subskriptionen mithilfe der Option -s zu synchronisieren:

- Angabe von **-s sub1, sub2, sub3** zur Synchronisation von sub1, sub2 und sub3 in einem Upload, gefolgt von einem Download.

In diesem Fall werden nur die Optionen verwendet, die für die erste Subskription in der Liste angegeben wurden, sofern Sie erweiterte Optionen für die Subskriptionen festgelegt haben. Erweiterte Optionen für nachfolgende Subskriptionen werden ignoriert.

- Geben Sie **-s sub1 -s sub2 -s sub3** zur Synchronisation von sub1, sub2 und sub3 in drei getrennten, aufeinander folgenden Synchronisationen an, von denen jede über einen eigenen Upload und Download verfügt.

Wenn aufeinander folgende Synchronisationen schnell hintereinander erfolgen, etwa wenn Sie **-s sub1 -s sub2** angeben, kann es vorkommen, dass dbmlsync mit der Verarbeitung einer Synchronisation beginnt, während der Server noch mit der Verarbeitung der vorherigen Synchronisation beschäftigt ist. In diesem Fall schlägt die zweite Synchronisation mit einer Fehlermeldung fehl, die besagt, dass gleichzeitige Synchronisationen nicht zulässig sind. Wenn eine solche Situation eintritt, können Sie definieren, dass eine gespeicherte Prozedur

sp_hook_dbmlsync_delay eine Verzögerung vor jeder Synchronisation durchführt. In der Regel reichen ein paar Sekunden bis zu einer Minute als Verzögerung aus.

Siehe auch

- „sp_hook_dbmlsync_delay“ auf Seite 218
- „Subscription-Synchronisationsprofiloption“ auf Seite 198

dbmlsync-Option -sc

Legt fest, dass dbmlsync Schemainformationen vor jeder Synchronisation neu laden muss

Syntax

dbmlsync -sc ...

Bemerkungen

Vor Version 9.0 hat dbmlsync die Schemainformationen vor jeder Synchronisation neu aus der Datenbank geladen. Die neu geladenen Informationen waren Fremdschlüsselbeziehungen, Publikationsdefinitionen, in der Datenbank gespeicherte erweiterte Optionen und Informationen über die Datenbankeinstellungen. Das Laden dieser Informationen ist zeitaufwändig und oft ändert sich der Informationsgehalt zwischen Synchronisationen nicht.

Ab Version 9.0 lädt dbmlsync standardmäßig Schemainformationen nur beim Start. Geben Sie -sc an, wenn Sie wollen, dass die Informationen vor jeder Synchronisation geladen werden sollen.

dbmlsync-Option -sm

Mit dieser Option wird dbmlsync im Servermodus gestartet.

Syntax

dbmlsync -sm ...

Bemerkungen

Im Servermodus wird dbmlsync gestartet und wartet auf Verbindungen von Anwendungen, die die Dbmlsync-API oder die SQL SYNCHRONIZE-Anweisung verwenden.

Diese Option sollte nur verwendet werden, wenn ein dbmlsync-Server von der Befehlszeile aus gestartet wird.

Normalerweise wird dbmlsync direkt über die Dbmlsync-API oder die SQL SYNCHRONIZE-Anweisung gestartet. Diese Option **sollte nicht** verwendet werden, wenn eine dieser Methoden eingesetzt wird.

Siehe auch

- „dbmlsync-Option -po“ auf Seite 130

dbmlsync-Option -sp

Wenn -sp verwendet wird, werden die im angegebenen Synchronisationsprofil enthaltenen Optionen zu den in der Befehlszeile für die Synchronisation angegebenen Optionen hinzugefügt.

Syntax

dbmlsync -sp *sync profile ...*

Bemerkungen

Wenn in der Befehlszeile und im Synchronisationsprofil gleichbedeutende Optionen angegeben werden, haben die Optionen in der Befehlszeile Vorrang vor denen im Profil.

Siehe auch

- „CREATE SYNCHRONIZATION PROFILE-Anweisung [MobiLink]“ [[SQL Anywhere Server - SQL-Referenzhandbuch](#)]
- „ALTER SYNCHRONIZATION PROFILE-Anweisung [MobiLink]“ [[SQL Anywhere Server - SQL-Referenzhandbuch](#)]
- „DROP SYNCHRONIZATION PROFILE-Anweisung [MobiLink]“ [[SQL Anywhere Server - SQL-Referenzhandbuch](#)]
- „MobiLink-Synchronisationsprofile“ auf Seite 182

dbmlsync-Option -ts

Richtet eine Protokollierungssitzung für den MobiLink-Client ein

Syntax

dbmlsync -ts *session-name*(*session-option*=*[option-value;...]*)

Der session-name muss **logging** lauten.

session-option	Wert der Option
events	Kommagetrennte Liste der System-Trace-Ereignisse. Die unterstützten Ereignisse sind "Info", "Warning" und "Error".
targets	<i>target-type</i> (<i>target-option</i> = <i>value[;...]</i>), wobei <i>target-type</i> nur file sein kann.

Die Zielloptionen werden als Namen-Wert-Paare angegeben. Die Zieldatei kann die folgenden Optionen aufweisen:

Name für Zielloption	Erwarteter Wert	Beschreibung
filename_prefix	Zeichenfolge	Ein ETD-Dateinamenpräfix mit oder ohne Pfad. Alle ETD-Dateien haben die Erweiterung <i>.etd</i> . Dieser Parameter ist erforderlich.

Name für Zieloption	Erwarteter Wert	Beschreibung
max_size	Ganzzahl	Die maximale Größe der Datei in Byte. Der Standardwert ist 0, was bedeutet, dass es keine Grenze für die Dateigröße gibt und die Datei größer wird, solange Speicherplatz verfügbar ist. Sobald die angegebene Größe erreicht ist, wird eine neue Datei gestartet.
num_files	Ganzzahl	Die Anzahl der Dateien, in die Informationen zur Ereignisprotokollierung geschrieben werden. Diese Option wird nur verwendet, wenn max_size eingestellt ist. Wenn alle Dateien die maximale angegebene Größe erreichen, beginnt der MobiLink-Client, die älteste Datei zu überschreiben.
flush_on_write	YES, TRUE, NO, FALSE	Ein Wert, der steuert, ob Festplattenpuffer für jedes protokollierte Ereignis geleert werden. Die Werte YES, TRUE, NO und FALSE werden akzeptiert. Der Standardwert ist FALSE. Bei aktiviertem Parameter kann die Performance des MobiLink-Clients vermindert sein, wenn viele Trace-Ereignisse protokolliert werden.
compressed	YES, TRUE, NO, FALSE	Ein Wert, der die Komprimierung der ETD-Datei zum Einsparen von Speicherplatz steuert. Der Standardwert ist FALSE.

Bemerkungen

Alle Daten, die nach dem Teil `-ts logging` der Option angegeben werden, müssen ohne Leerzeichen angegeben werden.

Siehe auch

- „Dienstprogramm für die Verwaltung von ETD-Dateien (dbmanageetd)“ [[SQL Anywhere Server - Datenbankadministration](#)]

Beispiel

Im Folgenden finden Sie ein Beispiel der Option `-ts`:

```
-ts
logging{events=Info,warning,Error;targets=file{filename_prefix=mls_etd;max_size=10000000;num_files=10;flush_on_write=true}}
```

dbmlsync-Option -tu

Legt fest, dass jede Transaktion in der entfernten Datenbank als eigene Transaktion in einer eigenen Synchronisation ausgelesen werden soll.

Syntax

dbmlsync -tu ...

Bemerkungen

Wenn Sie `-tu` verwenden, erstellen Sie einen **transaktionalen Upload**: dbmsync liest jede Transaktion in die entfernte Datenbank als einzelne Transaktion aus. Im MobiLink-Server werden alle Transaktionen beim Empfang einzeln angewendet und festgeschrieben.

Bei der Verwendung von `-tu` wird die Reihenfolge der Transaktionen in der entfernten Datenbank in der konsolidierten Datenbank immer beibehalten. Die Reihenfolge der Vorgänge in einer Transaktion wird jedoch aus folgenden zwei Gründen möglicherweise nicht bewahrt:

- MobiLink übernimmt immer Aktualisierungen basierend auf den Fremdschlüsselbeziehungen. Wenn beispielsweise Daten in Fremdschlüssel- und Primärschlüsseltabelle geändert werden, fügt MobiLink Daten zuerst in die Primärschlüsseltabelle und danach in die Fremdschlüsseltabelle ein, löscht aber Daten zuerst aus der Fremdschlüsseltabelle und danach aus der Primärschlüsseltabelle. Falls Ihre entfernten Vorgänge diese Reihenfolge nicht befolgen, ist die Reihenfolge der Vorgänge in der konsolidierten Datenbank unterschiedlich.
- Vorgänge innerhalb einer Transaktion werden zusammengefügt. Wenn Sie daher eine Zeile in einer Transaktion dreimal ändern, wird nur die endgültige Form der Zeile hochgeladen.

Wenn ein transaktionaler Upload unterbrochen wird, werden die Daten, die nicht gesendet wurden, mit der nächsten Synchronisation gesendet. In der Regel werden zu diesem Zeitpunkt nur die Transaktionen gesendet, die nicht erfolgreich abgeschlossen wurden. In einigen Fällen, beispielsweise wenn der Upload während der ersten Synchronisation einer Subskription fehlschlägt, sendet dbmsync alle Transaktionen erneut.

Wenn Sie `-tu` nicht verwenden, fügt MobiLink alle Änderungen in der entfernten Datenbank im Upload zu einer Transaktion zusammen. Wenn Sie also eine Zeile zwischen Synchronisationen drei Mal ändern, wird unabhängig von der Anzahl von entfernten Transaktionen nur die endgültige Form der Zeile hochgeladen. Dieses Standardverhalten ist effizient und in vielen Fällen optimal.

In bestimmten Situationen möchten Sie aber vielleicht entfernte Transaktionen in der konsolidierten Datenbank beibehalten. Sie möchten z.B. Trigger für die konsolidierte Datenbank definieren, die Transaktionen bearbeiten, wenn sie in der entfernten Datenbank ausgeführt werden.

Außerdem bietet die Aufteilung des Uploads in kleinere Transaktionen gewisse Vorteile. Viele konsolidierte Datenbanken werden für kleine Transaktionen optimiert, sodass das Senden großer Transaktionen nicht effizient ist oder zu zahlreichen Konflikten führen kann. Wenn Sie `-tu` verwenden, können Sie nicht den gesamten Upload verlieren, wenn während des Uploads Kommunikationsfehler auftreten. Wenn Sie `-tu` verwenden und ein Upload-Fehler auftritt, werden alle erfolgreich übertragenen Transaktionen übernommen.

Die Option `-tu` bewirkt, dass MobiLink sich ähnlich wie SQL Remote verhält. Im Unterschied zu MobiLink repliziert jedoch SQL Remote alle Änderungen auf die entfernte Datenbank in der Reihenfolge ihres Auftretens ohne Zusammenfügung. Um dieses Verhalten zu imitieren, müssen Sie nach jedem Datenbankvorgang in der entfernten Datenbank eine Festschreibeweisung ausführen.

Es ist nicht möglich, `-tu` zusammen mit der erweiterten Option `Increment` oder mit skriptgesteuerten Uploads zu verwenden.

Siehe auch

- „mlsrv16-Option -tx“ [*MobiLink - Serveradministration*]
- „Selbstreferenzierende Tabellen“ [*MobiLink - Serveradministration*]
- „TransactionalUpload-Synchronisationsprofiloption“ auf Seite 198

dbmlsync-Option -u (nicht mehr empfohlen)

Hinweis

Diese Option wird nicht mehr empfohlen. Verwenden Sie stattdessen die dbmlsync-Option -s. Siehe „dbmlsync-Option -s“ auf Seite 133.

Um die dbmlsync-Option -s zu verwenden, sollten Sie den Namen der Subskription, die Sie synchronisieren wollen, überprüfen. Sie können den Subskriptionsnamen mit der folgenden Abfrage ermitteln:

```
SELECT subscription_name
FROM syssync JOIN sys.syspublication
WHERE site_name = <ml_user> AND publication_name = <pub_name>;
```

Ersetzen Sie <ml_user> durch den MobiLink-Benutzer, den Sie synchronisieren. Hierbei handelt es sich um den Wert, der durch die Option -u der dbmlsync-Befehlszeile festgelegt wird. Siehe „dbmlsync-Option -u (nicht mehr empfohlen)“ auf Seite 138.

Ersetzen Sie <pub_name> durch den Namen der zu synchronisierenden Publikation. Hierbei handelt es sich um den Wert, der mit der Option -n in der dbmlsync-Befehlszeile angegeben wird. Siehe „dbmlsync-Option -n (nicht mehr empfohlen)“ auf Seite 125.

Gibt den MobiLink-Benutzernamen an.

Syntax

dbmlsync -u *ml_username* ...

Bemerkungen

In der dbmlsync-Befehlszeile können Sie nur einen Benutzer angeben, wobei *ml_username* der Name ist, der in der FOR-Klausel der Anweisung CREATE SYNCHRONIZATION SUBSCRIPTION verwendet wird, die der zu verarbeitenden Subskription entspricht.

Diese Option sollte mit der Option -n *publication* verwendet werden, um die Subskription anzugeben, mit der dbmlsync arbeiten soll. Jede Subskription ist eindeutig identifiziert mit einem Paar aus *ml_username* und *publication*.

Sie können nur einen Benutzernamen in der Befehlszeile angeben. Alle in einem Durchgang zu synchronisierenden Subskriptionen müssen denselben Benutzer betreffen. Die Option -u kann ausgelassen werden, wenn jede einzelne in der Befehlszeile mit -n angegebene Publikation nur über eine Subskription verfügt.

Siehe auch

- „MLUser-Synchronisationsprofiloption (nicht mehr empfohlen)“ auf Seite 194

dbmlsync-Option -ud

Nur für Unix-Plattformen, legt fest, dass dbmlsync als Daemon ausgeführt wird.

Syntax

dbmlsync -ud ...

Bemerkungen

Nur auf UNIX-Plattformen.

Wenn Sie dbmlsync als Daemon ausführen, müssen Sie auch die Option -o oder -ot benutzen, um die Ausgabeinformationen zu protokollieren.

Starten Sie dbmlsync als Daemon, werden seine Berechtigungen durch die umask-Einstellung des aktuellen Benutzers gesteuert. Es wird empfohlen, den umask-Wert vor dem Start von dbmlsync festzulegen, um sicherzustellen, dass dbmlsync die entsprechenden Berechtigungen hat.

Siehe auch

- „dbmlsync-Option -o“ auf Seite 126
- „dbmlsync-Option -ot“ auf Seite 127

dbmlsync-Option -ui

Startet dbmlsync unter Linux mit X Window-Serverunterstützung im Shell-Modus, wenn keine nutzbare Anzeige verfügbar ist.

Syntax

dbmlsync -ui ...

Bemerkungen

Wenn diese Option verwendet wird, versucht dbmlsync, mit X Window zu starten. Falls dieser Versuch fehlschlägt, startet das Programm im Shell-Modus.

Wenn -ui angegeben wird, versucht dbmlsync, eine verwendbare Anzeige zu finden. Wenn keine gefunden wird, z.B. weil der X Window-Server nicht läuft, startet dbmlsync im Shell-Modus.

dbmlsync-Option -uo

Legt fest, dass die Synchronisation nur einen Uploadvorgang enthält.

Syntax

dbmlsync -uo...

Bemerkungen

Während einer reinen Upload-Synchronisation bereitet dbmlsync einen Upload für den MobiLink-Synchronisationsserver vor und versendet ihn genau so wie bei einer vollständigen Synchronisation. Anstatt aber einen Download zurückzusenden, sendet MobiLink nur eine Bestätigung, in der festgestellt wird, ob der Upload erfolgreich festgeschrieben wurde.

Eine Liste aller Skripten, die für reine Upload-Synchronisationen definiert werden müssen, finden Sie unter „Erforderliche Skripten“ [[MobiLink - Serveradministration](#)].

Siehe auch

- „Reine Upload- und reine Download-Synchronisation“ [[MobiLink - Serveradministration](#)]
- „Erweiterte Option DownloadOnly (ds)“ auf Seite 153
- „Erweiterte Option UploadOnly (uo)“ auf Seite 173
- „UploadOnly-Synchronisationsprofiloption“ auf Seite 200

dbmlsync-Option -urc

Gibt eine Schätzung der Anzahl der Zeilen an, für die bei einer Synchronisation ein Upload durchgeführt werden soll.

Syntax

dbmlsync -urc *row-estimate* ...

Bemerkungen

Um die Performance zu steigern, können Sie eine Schätzung der Zeilenzahl angeben, die bei einer Synchronisation hochgeladen werden sollen. Diese Einstellung ist vor allem sinnvoll, wenn Sie viele Zeilen aktualisieren. Eine höhere Schätzung führt zu schnelleren Uploads, aber auch zu mehr Speicherbedarf.

Die Synchronisation wird einwandfrei ausgeführt, unabhängig von der Schätzung.

Siehe auch

- [Anzahl der Zeilen bei großen Uploads schätzen](#) [[MobiLink - Serveradministration](#)]
- „UploadRowCnt-Synchronisationsprofiloption“ auf Seite 200

dbmlsync-Option -ux

Öffnet unter Linux ein dbmlsync-Meldungsfenster, in dem Meldungen angezeigt werden.

Syntax

dbmlsync -ux...

Bemerkungen

Wenn -ux festgelegt wurde, muss dbmlsync in der Lage sein, eine verwendbare Anzeige zu finden. Wenn keine gefunden wird, weil z.B. die DISPLAY-Umgebungsvariable nicht eingestellt ist oder der X Window-Server nicht läuft, schlägt dbmlsync fehl.

Um das dbmlsync-Meldungsfenster im dialogfreien Modus auszuführen, verwenden Sie -q.

Unter Windows erscheint das dbmlsync-Meldungsfenster automatisch.

Siehe auch

- „dbmlsync-Option -q“ auf Seite 131

dbmlsync-Option -v

Ermöglicht die Angabe, welche Informationen in der Nachrichten-Logdatei protokolliert und im Synchronisationsfenster angezeigt werden. Eine hohe Stufe der Ausführlichkeit kann die Performance beeinflussen und sollte normalerweise nur in der Entwicklungsphase verwendet werden.

Syntax

dbmlsync -v [*levels*] ...

Bemerkungen

Die Optionen -v haben Einfluss auf die Meldungslogdatei und das Synchronisationsfenster. Sie haben nur dann ein Meldungslog, wenn Sie in der dbmlsync-Befehlszeile -o oder -ot angeben.

Wenn Sie nur -v angeben, wird eine geringe Menge von Informationen protokolliert, aber immerhin mehr Informationen als ohne -v.

Folgende Werte für *levels* sind zulässig: Sie können eine oder mehrere dieser Optionen gleichzeitig verwenden, z.B. -vnrsu oder -v+cp.

- **+** Alle Protokolloptionen ausgenommen c und p aktivieren
- **c** Die Verbindungszeichenfolge im Log aufzeichnen
- **p** Das MobiLink-Kennwort im Log aufzeichnen
- **n** Die Anzahl der Zeilen aus dem Upload bzw. Download protokollieren
- **o** Informationen über die von Ihnen angegebenen Befehlszeilenooptionen und erweiterte Optionen protokollieren
- **r** Die Werte der Zeilen aus dem Upload bzw. Download protokollieren
- **s** Hook-Skript-bezogene Meldungen protokollieren
- **u** Informationen über den Upload protokollieren

Es gibt erweiterte Optionen, die eine ähnliche Funktionalität haben wie die Optionen -v. Wenn Sie sowohl -v als auch die erweiterten Optionen angeben und Konflikte vorliegen, hebt die Option -v die erweiterte Option auf. Wenn kein Konflikt vorliegt, gelten die Ausführlichkeitsoptionen zusätzlich zu allen von Ihnen verwendeten Optionen. Wenn Sie die Befehlszeilenoption -v verwenden, werden die Ausführlichkeitsoptionen sofort wirksam. Wenn Sie die erweiterte Option verwenden, treten die Ausführlichkeitsoptionen erst dann in Kraft, wenn die erste Synchronisation beginnt. Nach der ersten Synchronisation sollte das Verhalten dasselbe sein, unabhängig davon, wie die Option angegeben wurde.

Siehe auch

- „Erweiterte Option Verbose (v)“ auf Seite 174
- „Erweiterte Option VerboseHooks (vs)“ auf Seite 175
- „Erweiterte Option VerboseMin (vm)“ auf Seite 176
- „Erweiterte Option VerboseOptions (vo)“ auf Seite 176
- „Erweiterte Option VerboseRowCounts (vn)“ auf Seite 177
- „Erweiterte Option VerboseRowValues (vr)“ auf Seite 178
- „dbmlsync-Option -o“ auf Seite 126
- „dbmlsync-Option -ot“ auf Seite 127
- „Verbosity-Synchronisationsprofiloption“ auf Seite 201

dbmlsync-Option -wc

Legt einen Fensterklassennamen fest

Syntax

dbmlsync -wc *class-name* ...

Bemerkungen

Diese Option legt einen Fensterklassennamen fest, der verwendet werden kann, um dbmlsync aus dem Wartemodus zu aktivieren, wie etwa, wenn die Abfolgeplanung aktiviert wird oder wenn Sie die serverinitiierte Synchronisation benutzen.

Außerdem kennzeichnet der Fensterklassenname die Anwendung für die Microsoft ActiveSync-Synchronisation. Der Klassenname muss angegeben werden, wenn die Anwendung zur Verwendung mit der Microsoft ActiveSync-Synchronisation registriert wird.

Diese Option steht nur für Windows zur Verfügung.

Siehe auch

- „SQL Anywhere-Clients für Microsoft ActiveSync registrieren“ auf Seite 98
- „Synchronisation mit Microsoft ActiveSync“ auf Seite 95
- INFINITE-Schlüsselwort in „Erweiterte Option Schedule (sch)“ auf Seite 166
- „Synchronisationszeitpläne“ auf Seite 100

Beispiel

```
dbmlsync -wc dbmlsync_$message_end...
```

dbmlsync-Option -x

Damit wird das Transaktionslog umbenannt und neu gestartet.

Syntax

dbmlsync -x [size [**K** | **M** | **G**]] ...

Bemerkungen

Es wird dringend empfohlen, dass Sie mit der Option -x immer einen Größenwert angeben. Insbesondere wird empfohlen, dass Sie **-x 0** angeben. Wenn Sie den Größe angeben, vermeiden Sie damit mögliche Zweideutigkeit und unbeabsichtigtes Verhalten. Wenn die Option -x unmittelbar vor dem Offline-Log-Verzeichnis angegeben wird, muss die Größe angegeben werden, um Fehler oder unbeabsichtigtes Verhalten zu vermeiden.

Wenn die optionale Größe angegeben ist, wird das Log bei Überschreitung dieser Größe (in Byte) umbenannt. Verwenden Sie das Suffix "k", "m" oder "g", um die Einheit in kB, MB oder GB anzugeben. Die Standardgröße ist 0.

Wenn von der entfernten Datenbank nicht regelmäßig Sicherungskopien angelegt werden, wächst das Transaktionslog stetig an. Zur Steuerung der Größe des Transaktionslogs kann alternativ zur Option -x auch eine Ereignisverarbeitungsroutine von SQL Anywhere verwendet werden.

Siehe auch

- „Aufgabenautomatisierung mit Abfolgeplanung und Ereignissen“ [*SQL Anywhere Server - Datenbankadministration*]
- „delete_old_logs-Option [SQL Remote]“ [*SQL Anywhere Server - Datenbankadministration*]
- „CREATE EVENT-Anweisung“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]

Erweiterte Optionen von MobiLink SQL Anywhere-Clients

Erweiterte Optionen können in der dbmlsync-Befehlszeile mit den Optionen -e oder -eu angegeben oder in der Datenbank gespeichert werden. Sie speichern erweiterte Optionen in der Datenbank mithilfe von Sybase Central mit dem Ereignis-Hook "sp_hook_dbmlsync_set_extended_options" oder mit der Klausel OPTION in einer der folgenden Anweisungen:

- CREATE SYNCHRONIZATION SUBSCRIPTION
- ALTER SYNCHRONIZATION SUBSCRIPTION
- CREATE SYNCHRONIZATION USER
- ALTER SYNCHRONIZATION USER
- CREATE SYNCHRONIZATION SUBSCRIPTION ohne Angabe eines Synchronisationsbenutzers (die die Werte mit einer Publikation verbindet)

Prioritätenfolge

Dbmlsync kombiniert Optionen, die in der Datenbank gespeichert sind, mit den in der Befehlszeile angegebenen Optionen. Wenn miteinander in Konflikt stehende Optionen angegeben werden, löst dbmlsync sie wie nachstehend beschrieben auf. In der folgenden Liste haben Optionen, die durch Methoden definiert werden, die in der Liste weiter oben stehen, Vorrang vor den Methoden weiter unten in der Liste.

1. Optionen, die im Ereignis-Hook `sp_hook_dbmlsync_set_extended_options` festgelegt sind.
2. Optionen in der Befehlszeile, bei denen es sich nicht um erweiterte Optionen handelt. (Beispiel: `-ds` setzt `-e "ds=off"` außer Kraft.)
3. In der Befehlszeile mit der Option `-eu` angegebene Optionen.
4. In der Befehlszeile mit der Option `-e` angegebene Optionen.
5. Optionen, die für die Subskription angegeben wurden (in SQL-Anweisungen oder mit Sybase Central). Wenn Sie den **Assistenten zum Bereitstellen eines Synchronisationsmodells** verwenden, um das Deployment eines MobiLink-Modells durchzuführen, werden erweiterte Optionen für Sie festgelegt und in der Subskription angegeben.
6. Optionen, die für den MobiLink-Benutzer angegeben wurden (in SQL-Anweisungen oder mit Sybase Central)
7. Optionen, die für die Publikation angegeben wurden (in SQL-Anweisungen oder mit Sybase Central)

Hinweis

Diese Prioritätenfolge wirkt sich auch auf Verbindungsparameter aus, beispielsweise auf diejenigen, die in den oben erwähnten SQL-Anweisungen mit den Optionen `TYPE` und `ADDRESS` angegeben wurden.

Sie können erweiterte Optionen im Log und der Systemansicht `SYSSYNC` prüfen.

Weitere Hinweise zur Verwendung der erweiterten Optionen zum Optimieren der Synchronisation finden Sie unter [„Erweiterte Optionen für dbmlsync“](#) auf Seite 92.

Übersicht über erweiterte dbmlsync-Optionen

Im Folgenden finden Sie eine Liste der erweiterten Optionen von dbmlsync.

Option	Beschreibung
BufferDownload ={ ON OFF }; ...	Diese Option legt fest, ob der gesamte Download vom MobiLink-Server in den Cache eingelesen werden soll, bevor er auf die entfernte Datenbank angewendet wird. Siehe „Erweiterte Option BufferDownload (bd)“ auf Seite 148.

Option	Beschreibung
CommunicationAddress = <i>protocol-option</i> ; ...	Gibt Netzwerkprotokolloptionen für das Verbinden mit dem MobiLink-Server an. Siehe „ Erweiterte Option CommunicationAddress (adr) “ auf Seite 149.
CommunicationType = <i>network-protocol</i> ; ...	Gibt den Typ des Netzwerkprotokolls an, das für das Verbinden mit dem MobiLink-Server verwendet werden soll. Siehe „ Erweiterte Option CommunicationType (ctp) “ auf Seite 150.
ConflictRetries = <i>number</i> ; ...	Legt die Anzahl der Neuversuche fest, falls der Download aufgrund von Konflikten fehlschlägt. Siehe „ Erweiterte Option ConflictRetries (cr) “ auf Seite 150.
ContinueDownload = { ON OFF } ; ...	Startet einen fehlgeschlagenen Download erneut. Siehe „ Erweiterte Option ContinueDownload (cd) “ auf Seite 151.
DisablePolling = { ON OFF } ; ...	Deaktiviert den automatischen Log-Scan-Abruf. Siehe „ Erweiterte Option DisablePolling (p) “ auf Seite 152.
DownloadOnly = { ON OFF } ; ...	Legt fest, dass die Synchronisation ein reiner Download sein soll. Siehe „ Erweiterte Option DownloadOnly (ds) “ auf Seite 153.
DownloadReadSize = <i>number</i> [K] ; ...	Bei neu startbaren Downloads legt diese Option den maximalen Umfang der Daten fest, die nach einem Kommunikationsfehler erneut gesendet werden müssen. Siehe „ Erweiterte Option DownloadReadSize (drs) “ auf Seite 154.
ErrorLogSendLimit = <i>number</i> [K M] ; ...	Legt fest, welcher Anteil der Meldungslogdatei der entfernten Datenbank von dbmlsync an den Server geschickt werden soll, wenn ein Synchronisationsfehler auftritt. Siehe „ Erweiterte Option ErrorLogSendLimit (el) “ auf Seite 155.
FireTriggers = { ON OFF } ; ...	Legt fest, dass Trigger in der entfernten Datenbank ausgelöst werden sollen, wenn der Download übernommen wird. Siehe „ Erweiterte Option FireTriggers (ft) “ auf Seite 156.
HoverRescanThreshold = <i>number</i> [K M] ; ...	Wenn Sie eine geplante Synchronisation verwenden, wird hiermit die Menge des verworfenen Speichers limitiert, die sich ansammeln darf, bevor ein Re-Scan durchgeführt wird. Siehe „ Erweiterte Option HoverRescanThreshold (hrt) “ auf Seite 157.
IgnoreHookErrors = { ON OFF } ; ...	Legt fest, dass Fehler in Hook-Funktionen ignoriert werden sollen. Siehe „ Erweiterte Option IgnoreHookErrors (eh) “ auf Seite 158.
IgnoreScheduling = { ON OFF } ; ...	Gibt an, dass die erweiterte Option für die Abfolgeplanung ignoriert werden soll. Siehe „ Erweiterte Option IgnoreScheduling (isc) “ auf Seite 158.

Option	Beschreibung
Increment = <i>number</i> [K M]; ...	Erlaubt inkrementelle Uploads und steuert die Größe der Upload-Inkremente. Siehe „ Erweiterte Option Increment (inc) “ auf Seite 159.
LockTables = { ON OFF SHARE EXCLUSIVE } ; ...	Legt fest, dass Tabellen in der synchronisierten Publikation gesperrt werden müssen, bevor die Synchronisation erfolgt. Siehe „ Erweiterte Option LockTables (lt) “ auf Seite 160.
MirrorLogDirectory = <i>dir</i> ; ...	Gibt den Speicherort alter Transaktionslogspiegeldateien an, damit sie gelöscht werden können. Siehe „ Erweiterte Option MirrorLogDirectory (mld) “ auf Seite 161.
MobiLinkPwd = <i>password</i> ; ...	Legt das MobiLink-Kennwort fest. Siehe „ Erweiterte Option MobiLinkPwd (mp) “ auf Seite 162.
NewMobiLinkPwd = <i>new-password</i> ; ...	Legt ein neues MobiLink-Kennwort fest. Siehe „ Erweiterte Option NewMobiLinkPwd (mn) “ auf Seite 163.
NoSyncOnStartup = { on off } ; ...	Verhindert eine Synchronisation durch dbmlsync beim Start, wenn dies normalerweise durch eine Abfolgeplanungsoption ausgelöst würde. Siehe „ Erweiterte Option NoSyncOnStartup (nss) “ auf Seite 163.
OfflineDirectory = <i>path</i> ; ...	Legt den Pfad fest, der Offline-Transaktionslogs enthält. Siehe „ Erweiterte Option OfflineDirectory (dir) “ auf Seite 164.
PollingPeriod = <i>number</i> [S M H D]; ...	Legt die Log-Scan-Abrufsperiode fest. Siehe „ Erweiterte Option PollingPeriod (pp) “ auf Seite 165.
Schedule = <i>schedule</i> ; ...	Legt eine Zeitplanung für die Synchronisation fest. Siehe „ Erweiterte Option Schedule (sch) “ auf Seite 166.
ScriptVersion = <i>version-name</i> ; ...	Legt eine Skriptversion fest. Siehe „ Erweiterte Option ScriptVersion (sv) “ auf Seite 168.
SendDownloadAck = { ON OFF } ; ...	Legt fest, dass eine Downloadbestätigung vom Client an den Server gesendet werden soll. Siehe „ Erweiterte Option SendDownloadAck (sa) “ auf Seite 169.
SendTriggers = { ON OFF } ; ...	Legt fest, dass Triggeraktionen beim Upload gesendet werden sollen. Siehe „ Erweiterte Option SendTriggers (st) “ auf Seite 169.
TableOrder = <i>tables</i> ; ...	Legt die Reihenfolge der Tabellen im Upload fest. Siehe „ Erweiterte Option TableOrder (tor) “ auf Seite 171.

Option	Beschreibung
TableOrderChecking = { OFF ON }; ...	Hiermit können Sie die Prüfung der referenziellen Integrität für die Tabellenfolge deaktivieren, die durch die erweiterte TableOrder-Option festgelegt wird. Siehe „ Erweiterte Option TableOrderChecking (toc) “ auf Seite 172.
UploadOnly = { ON OFF }; ...	Legt fest, dass die Synchronisation nur einen Uploadvorgang enthalten sollte. Siehe „ Erweiterte Option UploadOnly (uo) “ auf Seite 173.
Verbose = { ON OFF }; ...	Legt fest, dass in aller Ausführlichkeit protokolliert wird. Siehe „ Erweiterte Option Verbose (v) “ auf Seite 174.
VerboseHooks = { ON OFF }; ...	Legt fest, dass Meldungen im Zusammenhang mit Hook-Prozedurskripten protokolliert werden sollen. Siehe „ Erweiterte Option VerboseHooks (vs) “ auf Seite 175.
VerboseMin = { ON OFF }; ...	Legt fest, dass eine geringe Menge von Informationen protokolliert werden soll. Siehe „ Erweiterte Option VerboseMin (vm) “ auf Seite 176.
VerboseOptions = { ON OFF }; ...	Legt fest, dass Informationen über die von Ihnen festgelegten Befehlszeilenoptionen (einschließlich erweiterten Optionen) protokolliert werden sollen. Siehe „ Erweiterte Option VerboseOptions (vo) “ auf Seite 176.
VerboseRowCounts = { ON OFF }; ...	Legt fest, dass die Anzahl von Zeilen für den Upload und den Download protokolliert werden sollen. Siehe „ Erweiterte Option VerboseRowCounts (vn) “ auf Seite 177.
VerboseRowValues = { ON OFF }; ...	Legt fest, dass die Werte von Zeilen für den Upload und den Download protokolliert werden sollen. Siehe „ Erweiterte Option VerboseRowValues (vr) “ auf Seite 178.
VerboseUpload = { ON OFF }; ...	Legt fest, dass Informationen über den Upload-Datenstrom protokolliert werden sollen. Siehe „ Erweiterte Option VerboseUpload (vu) “ auf Seite 179.

Siehe auch

- „dbmlsync-Option -e“ auf Seite 121
- „dbmlsync-Option -eu“ auf Seite 123
- „CREATE SYNCHRONIZATION SUBSCRIPTION-Anweisung [MobiLink]“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- „ALTER SYNCHRONIZATION SUBSCRIPTION-Anweisung [MobiLink]“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- „CREATE SYNCHRONIZATION USER-Anweisung [MobiLink]“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- „ALTER SYNCHRONIZATION USER-Anweisung [MobiLink]“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- „CREATE PUBLICATION-Anweisung [MobiLink] [SQL Remote]“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- „ALTER PUBLICATION-Anweisung [MobiLink] [SQL Remote]“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- „SYSSYNC-Systemansicht“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- „sp_hook_dbmlsync_set_extended_options“ auf Seite 252

Beispiel

Die folgende dbmlsync-Befehlszeile zeigt, wie Sie beim Start von dbmlsync erweiterte Optionen verwenden können.

```
dbmlsync -e "adr=host=localhost;dir=c:\db\logs"...
```

Die folgende SQL-Anweisung zeigt, wie Sie erweiterte Optionen in der Datenbank speichern können.

```
CREATE SYNCHRONIZATION SUBSCRIPTION TO mypub
FOR mluser
ADDRESS 'host=localhost'
OPTION schedule='weekday@11:30am-12:30pm', dir='c:\db\logs'
```

Die folgende dbmlsync-Befehlszeile öffnet die Syntaxanzeige, in der Optionen und ihre Syntax angezeigt werden:

```
dbmlsync -l
```

Erweiterte Option BufferDownload (bd)

Diese Option legt fest, ob der gesamte Download vom MobiLink-Server in den Cache eingelesen werden soll, bevor er auf die entfernte Datenbank angewendet wird.

Syntax

bd={ON | OFF}; ...

BufferDownload={ON | OFF}; ...

Bemerkungen

Der Standardwert ist ON. Der Standardwert führt zu einer geringeren Belastung des MobiLink-Servers und sollte den serverseitigen Durchsatz verbessern.

Wenn BufferDownload auf Off gesetzt ist, übernimmt dbmlsync den Download so, wie er gelesen wird.

Siehe auch

- [Übersicht über erweiterte dbmlsync-Optionen auf Seite 144](#)

Erweiterte Option CommunicationAddress (adr)

Gibt Netzwerkprotokolloptionen für das Verbinden mit dem MobiLink-Server an

Syntax

adr=*protocol-option*; ...

CommunicationAddress=*protocol-option*; ...

Bemerkungen

Hinweise zu den Parametern finden Sie unter „Netzwerkprotokolloptionen des MobiLink-Clients“ auf Seite 25.

Sie müssen darauf achten, dass alle Subskriptionen für einen MobiLink-Benutzer nur mit einer konsolidierten Datenbank synchronisiert werden. Sonst könnte es zu Datenverlust und nicht vorhersagbarem Verhalten kommen.

Verwenden Sie die erweiterte Option "CommunicationType", um den Typ des Netzwerkprotokolls anzugeben.

Siehe auch

- „Netzwerkprotokolloptionen des MobiLink-Clients“ auf Seite 25
- [Übersicht über erweiterte dbmlsync-Optionen auf Seite 144](#)
- „Relay Server-Konfigurationsdatei“ [*Relay Server*]
- „Erweiterte Option CommunicationType (ctp)“ auf Seite 150

Beispiel

Die folgende dbmlsync-Befehlszeile zeigt, wie Sie diese Option beim Start von dbmlsync verwenden können.

```
dbmlsync -e "adr=host=localhost"
```

Um mehrere Netzwerkprotokolloptionen in der Befehlszeile anzugeben, setzen Sie sie in Apostrophe. Beispiel:

```
dbmlsync -e "adr='host=somehost;port=5001'"
```

Um die Adresse oder den Verbindungstyp in der Datenbank zu speichern, können Sie eine erweiterte Option oder die ADDRESS-Klausel verwenden. Beispiel:

```
CREATE SYNCHRONIZATION SUBSCRIPTION  
  TO sales_publication  
  FOR ml_user1  
  ADDRESS 'host=localhost;port=2439'
```

Erweiterte Option CommunicationType (ctp)

Gibt den Typ des Netzwerkprotokolls an, das für das Verbinden mit dem MobiLink-Server verwendet werden soll.

Syntax

ctp=*network-protocol*; ...

CommunicationType=*network-protocol*; ...

Bemerkungen

network-protocol kann vom Typ **tcpip**, **tls**, **http** oder **https** sein. Der Standardwert ist **tcpip**.

Sie müssen darauf achten, dass alle Subskriptionen für einen MobiLink-Benutzer nur mit einer konsolidierten Datenbank synchronisiert werden. Sonst könnte es zu Datenverlust und nicht vorhersagbarem Verhalten kommen.

Siehe auch

- [Übersicht über erweiterte dbmlsync-Optionen auf Seite 144](#)
- [„Verschlüsselung der MobiLink-Client/Server-Kommunikation“ \[SQL Anywhere Server - Datenbankadministration\]](#)
- [„Erweiterte Option CommunicationAddress \(adr\)“ auf Seite 149](#)

Beispiel

Die folgende dbmlsync-Befehlszeile zeigt, wie Sie diese Option beim Start von dbmlsync verwenden können.

```
dbmlsync -e "ctp=https"
```

Die folgende SQL-Anweisung zeigt, wie Sie diese Option in der Datenbank speichern können.

```
CREATE SYNCHRONIZATION SUBSCRIPTION  
TO sales_publication  
FOR ml_user1  
OPTION ctp='tcpip'
```

Erweiterte Option ConflictRetries (cr)

Legt die Anzahl der Neuversuche fest, falls der Download aufgrund von Konflikten fehlschlägt.

Syntax

cr=*number*; ...

ConflictRetries=*number*; ...

Bemerkungen

Wenn Tabellen nicht während der Synchronisation gesperrt werden, können Vorgänge auf die Datenbank zwischen der Erstellung des Uploads und der Übernahme des Downloads angewendet werden. Wenn diese Änderungen Zeilen betreffen, die auch durch den Download geändert werden, ermittelt dbmlsync einen

Konflikt und übernimmt den Download-Datenstrom nicht. In diesem Fall führt dbmlsync die gesamte Synchronisation erneut durch. Normalerweise ist die Synchronisation beim nächsten Versuch erfolgreich, ein erneuter Konflikt kann jedoch zu einem wiederholten Versuch führen. Diese Option legt fest, wie viele Neuversuche maximal durchgeführt werden sollen.

Diese Option ist nur sinnvoll, wenn die Option LockTables auf OFF gesetzt ist, was standardmäßig der Fall ist.

Der Standardwert lautet **-1** (Neuversuche werden unendlich fortgesetzt).

Siehe auch

- [Übersicht über erweiterte dbmlsync-Optionen auf Seite 144](#)
- „Konfliktverarbeitung“ [*MobiLink - Serveradministration*]

Beispiel

Die folgende dbmlsync-Befehlszeile zeigt, wie Sie diese Option beim Start von dbmlsync verwenden können.

```
dbmlsync -e "cr=5"
```

Die folgende SQL-Anweisung zeigt, wie Sie diese Option in der Datenbank speichern können.

```
CREATE SYNCHRONIZATION SUBSCRIPTION  
  TO sales_publication  
  FOR ml_user1  
  OPTION cr='5';
```

Erweiterte Option ContinueDownload (cd)

Startet einen fehlgeschlagenen Download erneut.

Syntax

cd={ ON | OFF }; ...

ContinueDownload={ ON | OFF }; ...

Bemerkungen

Wenn dbmlsync nicht den gesamten Download des Servers empfängt, wird kein Teil des Downloads in die entfernte Datenbank übernommen. Der Teil des Downloads, der empfangen wurde, wird jedoch auf dem entfernten Gerät in einer temporären Datei gespeichert, sodass er später neu gestartet werden kann. Wenn Sie die erweiterte Option cd=on setzen, startet dbmlsync den Download neu und versucht, den Teil des Downloads einzulesen, der beim letzten Downloadversuch nicht empfangen wurde. Wenn die verbliebenden Daten eingelesen werden können, wird der gesamte Download in Ihre entfernte Datenbank übernommen.

Falls Sie cd=on verwenden und neue Daten für die Übertragung anstehen, schlägt die Synchronisation ohne Neustart des Downloads fehl. Wenn der Download nicht neu gestartet werden kann, schlägt die Synchronisation fehl.

Sie können auch einen Download mit der erweiterten Option `-dc` oder dem Hook `sp_hook_dbmlsync_end` neu starten.

Siehe auch

- [Übersicht über erweiterte dbmlsync-Optionen auf Seite 144](#)
- [„Wiederaufnahme fehlgeschlagener Downloads“ \[MobiLink - Serveradministration\]](#)
- [„sp_hook_dbmlsync_set_extended_options“ auf Seite 252](#)
- [„dbmlsync-Option -dc“ auf Seite 118](#)
- [„sp_hook_dbmlsync_end“ auf Seite 232](#)

Beispiel

Die folgende dbmlsync-Befehlszeile zeigt, wie Sie diese Option beim Start von dbmlsync verwenden können.

```
dbmlsync -e "cd=on"
```

Die folgende SQL-Anweisung zeigt, wie Sie diese Option in der Datenbank speichern können.

```
CREATE SYNCHRONIZATION SUBSCRIPTION  
  TO sales_publication  
  FOR ml_user1  
  OPTION cd='on';
```

Erweiterte Option DisablePolling (p)

Deaktiviert den automatischen Log-Scan-Abruf.

Syntax

p={ON | OFF}; ...

DisablePolling={ON | OFF}; ...

Bemerkungen

Um einen Upload einzurichten, muss dbmlsync das Transaktionslog scannen. Normalerweise führt es dies kurz vor der Synchronisation durch. Wenn Synchronisationen in einem Zeitplan ablaufen oder der Hook `sp_hook_dbmlsync_delay` benutzt wird, scannt dbmlsync das Log standardmäßig zwischen den Synchronisationen. Dieses Verhalten ist effizienter, da das Log dann zumindest bereits teilweise gescannt ist, wenn die Synchronisation beginnt. Dieses Standardverhalten wird als Logscan-Abruf (logscan polling) bezeichnet.

Logscan-Abruf ist standardmäßig aktiviert, hat aber nur dann Wirkung, wenn Synchronisationen geplant sind oder `sp_hook_dbmlsync_delay` benutzt wird. Wenn der Abruf aktiviert ist, wird er in bestimmten Intervallen durchgeführt: Das Programm dbmlsync scannt bis zum Ende des Logs, wartet die Logscan-Abrufperiode ab und scannt dann eventuelle neue Transaktionen im Log. Standardmäßig ist die Abrufperiode auf 1 Minute gesetzt, kann aber mit der dbmlsync Option `-pp` oder der erweiterten Option `PollingPeriod` geändert werden.

Standard ist, Logscan-Abruf nicht zu deaktivieren (**OFF**).

Diese Option ist identisch mit **dbmlsync -p**.

Siehe auch

- [Übersicht über erweiterte dbmlsync-Optionen auf Seite 144](#)
- [„Erweiterte Option PollingPeriod \(pp\)“ auf Seite 165](#)
- [„dbmlsync-Option -p“ auf Seite 127](#)
- [„dbmlsync-Option -pp“ auf Seite 130](#)

Beispiel

Die folgende dbmlsync-Befehlszeile zeigt, wie Sie diese Option beim Start von dbmlsync verwenden können.

```
dbmlsync -e "p=on"
```

Die folgende SQL-Anweisung zeigt, wie Sie diese Option in der Datenbank speichern können.

```
CREATE SYNCHRONIZATION SUBSCRIPTION  
  TO sales_publication  
  FOR ml_user1  
  OPTION p='on' ;
```

Erweiterte Option DownloadOnly (ds)

Legt fest, dass die Synchronisation ein reiner Download sein soll.

Syntax

ds={ ON | OFF }; ...

DownloadOnly={ ON | OFF }; ...

Bemerkungen

Wenn eine reine Download-Synchronisation vorgenommen wird, führt dbmlsync keinen Upload von Zeilenvorgängen oder Daten durch. Allerdings wird ein Upload von Informationen über das Schema und das Offset für den Verarbeitungsfortschritt vorgenommen.

Außerdem sorgt dbmlsync dafür, dass Änderungen in der entfernten Datenbank, die noch nicht hochgeladen wurden, während einer reinen Download-Synchronisation nicht überschrieben werden. Dazu wird das Log gescannt, um Zeilen zu ermitteln, für die Vorgänge auf einen Upload warten. Wenn eine dieser Zeilen durch den Download geändert würde, wird der Download zurückgesetzt und die Synchronisation schlägt fehl. Wenn die Synchronisation aus diesem Grund fehlschlägt, müssen Sie eine vollständige Synchronisation vornehmen, um das Problem zu beheben.

Wenn Sie mit entfernten Datenbanken arbeiten, die durch reine Download-Synchronisation synchronisiert werden, sollten Sie regelmäßig eine vollständige Synchronisation durchführen, um das Volumen des Logs zu verkleinern, das von der reinen Download-Synchronisation gescannt werden muss. Sonst würden die reinen Download-Synchronisationen zunehmend mehr Zeit brauchen. Wenn dies ein Problem darstellt, können Sie alternativ eine Publikation mit reinem Download verwenden, um Logprobleme bei der Synchronisation zu vermeiden.

Eine Liste aller Skripten, die für reine Download-Synchronisationen definiert werden müssen, finden Sie unter „Erforderliche Skripten“ [[MobiLink - Serveradministration](#)].

Standardwert ist **OFF** (Upload und Download durchführen).

Siehe auch

- [Übersicht über erweiterte dbmlsync-Optionen auf Seite 144](#)
- [„dbmlsync-Option -ds“ auf Seite 120](#)
- [„Reine Download-Publikationen“ auf Seite 78](#)
- [„Reine Upload- und reine Download-Synchronisation“ \[\[MobiLink - Serveradministration\]\(#\)\]](#)

Beispiel

Die folgende dbmlsync-Befehlszeile zeigt, wie Sie diese Option beim Start von dbmlsync verwenden können.

```
dbmlsync -e "ds=on"
```

Die folgende SQL-Anweisung zeigt, wie Sie diese Option in der Datenbank speichern können.

```
CREATE SYNCHRONIZATION SUBSCRIPTION  
  TO sales_publication  
  FOR ml_user1  
  OPTION ds='ON';
```

Erweiterte Option DownloadReadSize (drs)

Bei neu startbaren Downloads legt diese Option den maximalen Umfang der Daten fest, die nach einem Kommunikationsfehler erneut gesendet werden müssen.

Syntax

drs=*number*[**K**]; ...

DownloadReadSize=*number*[**K**]; ...

Bemerkungen

Die Option DownloadReadSize ist nur bei der Verwendung von neu startbaren Downloads sinnvoll.

Die Download-Lesegröße wird in Einheiten von Byte festgelegt. Verwenden Sie das Suffix "k", um die Einheit in kB anzugeben.

Dbmlsync liest den Download in Abschnitten. DownloadReadSize legt die Größe dieser Abschnitte fest. Wenn ein Kommunikationsfehler auftritt, verliert dbmlsync den gesamten bearbeiteten Abschnitt. Abhängig davon, wann der Fehler auftritt, liegt die Anzahl der verlorenen Byte zwischen 0 und der Größe des Downloadlesevorgangs (DownloadReadSize) -1. Beispiel: Wenn DownloadReadSize 100 Byte beträgt und nach dem Lesen von 497 Byte ein Fehler auftritt, gehen die letzten 97 gelesenen Byte verloren. Auf diese Weise verlorene Byte werden erneut gesendet, wenn der Download erneut gestartet wird.

Im Allgemeinen führt ein größerer Wert von DownloadReadSize zu einer besseren Performance bei erfolgreichen Synchronisationen, aber zu einer Übertragung von mehr Daten, wenn ein Fehler auftritt.

Diese Option wird typischerweise verwendet, wenn die Standardgröße bei einer unzuverlässigen Verbindung verkleinert werden soll.

Der Standardwert ist **32767**. Wenn der Wert dieser Option über 32767 liegt, wird der Wert 32767 verwendet.

Siehe auch

- [Übersicht über erweiterte dbmlsync-Optionen auf Seite 144](#)
- [„dbmlsync-Option -drs“ auf Seite 120](#)
- [„Wiederaufnahme fehlgeschlagener Downloads“ \[*MobiLink - Serveradministration*\]](#)
- [„Erweiterte Option ContinueDownload \(cd\)“ auf Seite 151](#)
- [„sp_hook_dbmlsync_end“ auf Seite 232](#)
- [„dbmlsync-Option -dc“ auf Seite 118](#)

Beispiel

Die folgende dbmlsync-Befehlszeile zeigt, wie Sie diese Option beim Start von dbmlsync verwenden können.

```
dbmlsync -e "drs=100"
```

Die folgende SQL-Anweisung zeigt, wie Sie diese Option in der Datenbank speichern können.

```
CREATE SYNCHRONIZATION SUBSCRIPTION  
  TO sales_publication  
  FOR ml_user1  
  OPTION drs='100';
```

Erweiterte Option ErrorLogSendLimit (el)

Legt fest, welcher Anteil der Meldungslogdatei der entfernten Datenbank von dbmlsync an den Server geschickt werden soll, wenn ein Synchronisationsfehler auftritt.

Syntax

el=*number*[**K** | **M**]; ...

ErrorLogSendLimit=*number*[**K** | **M**]; ...

Bemerkungen

Diese Option wird in Byte angegeben. Stellen Sie ein "k" oder "m" dahinter, um die Einheit in kB oder MB anzugeben.

Diese Option legt die Anzahl der Byte des Meldungslogs fest, die dbmlsync an den MobiLink-Server sendet, wenn während der Synchronisation Fehler auftreten. Setzen Sie diese Option auf **0**, wenn das dbmlsync-Meldungslog gar nicht versendet werden soll.

Der Standardwert ist **32K**.

Ist diese Option nicht Null, wird ein Fehlerlog heraufgeladen, wenn auf der Clientseite ein Fehler auftritt. Nicht alle Fehler auf der Clientseite lösen ein Fehlerlog aus: Bei Verbindungsfehlern oder wenn dbmlsync

keine Verbindung mit dem MobiLink-Server aufbauen kann, wird kein Log gesendet. Tritt der Fehler auf, nachdem der Upload versendet wurde, wird das Fehlerlog nur hochgeladen, wenn die erweiterte Option `SendDownloadAck` aktiviert ist.

Wenn `ErrorLogSendLimit` groß genug eingestellt wird, sendet `dbmlsync` das gesamte Meldungslog der aktuellen Sitzung an den MobiLink-Server. Wenn z.B. die Meldungen des Meldungslogs an die alte Meldungslogdatei angefügt wurden, sendet `dbmlsync` nur die in der aktuellen Sitzung neu erzeugten Meldungen. Wenn die Gesamtlänge der neuen Meldungen größer ist als `ErrorLogSendLimit`, lädt `dbmlsync` das Meldungslog nur bis zur angegebenen Größe hoch.

Die Größe des Nachrichtenlogs wird durch Ihre Einstellungen für die Ausführlichkeitsstufe beeinflusst. Sie können diese Einstellung mit der Option `-v` von `dbmlsync` oder erweiterte `dbmlsync`-Optionen verändern, die mit "verbose" beginnen.

Siehe auch

- [Übersicht über erweiterte dbmlsync-Optionen auf Seite 144](#)
- [„dbmlsync-Option -v“ auf Seite 141](#)
- [„dbmlsync-Option -e“ auf Seite 121](#)
- [„dbmlsync-Option -eu“ auf Seite 123](#)
- [„Erweiterte Option Verbose \(v\)“ auf Seite 174](#)
- [„Erweiterte Option VerboseHooks \(vs\)“ auf Seite 175](#)
- [„Erweiterte Option VerboseMin \(vm\)“ auf Seite 176](#)
- [„Erweiterte Option VerboseOptions \(vo\)“ auf Seite 176](#)
- [„Erweiterte Option VerboseRowCounts \(vn\)“ auf Seite 177](#)
- [„Erweiterte Option VerboseRowValues \(vr\)“ auf Seite 178](#)
- [„Erweiterte Option VerboseUpload \(vu\)“ auf Seite 179](#)

Beispiel

Die folgende `dbmlsync`-Befehlszeile zeigt, wie Sie diese Option beim Start von `dbmlsync` verwenden können.

```
dbmlsync -e "el=32k"
```

Die folgende SQL-Anweisung zeigt, wie Sie diese Option in der Datenbank speichern können.

```
CREATE SYNCHRONIZATION SUBSCRIPTION  
  TO sales_publication  
  FOR ml_user1  
  OPTION el='32k';
```

Erweiterte Option FireTriggers (ft)

Legt fest, dass Trigger in der entfernten Datenbank ausgelöst werden sollen, wenn der Download übernommen wird.

Syntax

ft={ **ON** | **OFF** }; ...

FireTriggers={ **ON** | **OFF** }; ...

Bemerkungen

Der Standardwert ist **ON**.

Siehe auch

- [Übersicht über erweiterte dbmlsync-Optionen auf Seite 144](#)

Beispiel

Die folgende dbmlsync-Befehlszeile zeigt, wie Sie diese Option beim Start von dbmlsync verwenden können.

```
dbmlsync -e "ft=off"
```

Die folgende SQL-Anweisung zeigt, wie Sie diese Option in der Datenbank speichern können.

```
CREATE SYNCHRONIZATION SUBSCRIPTION  
  TO sales_publication  
  FOR ml_user1  
  OPTION ft='off';
```

Erweiterte Option HoverRescanThreshold (hrt)

Wenn Sie eine geplante Synchronisation verwenden, wird hiermit die Menge des verworfenen Speichers limitiert, die sich ansammeln darf, bevor ein Re-Scan durchgeführt wird.

Syntax

hrt=*number*[**K** | **M**]; ...

HoverRescanThreshold=*number*[**K** | **M**]; ...

Bemerkungen

Gibt den Speicher in Byteeinheiten ein. Stellen Sie ein "k" oder "m" dahinter, um die Einheit in kB oder MB anzugeben. Der Standardwert ist **1m**.

Wenn mehr als eine Option -n oder -s in der Befehlszeile angegeben wird, können in dbmlsync Fragmentierungen auftreten, sodass Speicherinhalt gelöscht wird. Der gelöschte Speicherinhalt kann nur durch erneutes Scannen des Datenbank-Transaktionslogs wiederhergestellt werden. Mit dieser Option können Sie die maximale Größe festlegen, die der gelöschte Speicher annehmen darf, bevor das Log erneut gescannt und der Speicher wiederhergestellt wird. Die Wiederherstellung des gelöschten Speichers kann auch durch Anwendung der gespeicherten Prozedur sp_hook_dbmlsync_log_rescan geregelt werden.

Siehe auch

- [Übersicht über erweiterte dbmlsync-Optionen auf Seite 144](#)
- [„sp_hook_dbmlsync_log_rescan“ auf Seite 235](#)

Beispiel

Die folgende dbmlsync-Befehlszeile zeigt, wie Sie diese Option beim Start von dbmlsync verwenden können.

```
dbmlsync -e "hrt=2m"
```

Die folgende SQL-Anweisung zeigt, wie Sie diese Option in der Datenbank speichern können.

```
CREATE SYNCHRONIZATION SUBSCRIPTION  
  TO sales_publication  
  FOR ml_user1  
  OPTION hrt='2m';
```

Erweiterte Option IgnoreHookErrors (eh)

Legt fest, dass Fehler in Hook-Funktionen ignoriert werden sollen.

Syntax

```
eh={ ON | OFF }; ...
```

```
IgnoreHookErrors={ ON | OFF }; ...
```

Bemerkungen

Der Standardwert ist **OFF**.

Diese Option ist gleichwertig mit der Option -eh von dbmlsync.

Siehe auch

- [Übersicht über erweiterte dbmlsync-Optionen auf Seite 144](#)

Beispiel

Die folgende dbmlsync-Befehlszeile zeigt, wie Sie diese Option beim Start von dbmlsync verwenden können.

```
dbmlsync -e "eh=off"
```

Die folgende SQL-Anweisung zeigt, wie Sie diese Option in der Datenbank speichern können.

```
CREATE SYNCHRONIZATION SUBSCRIPTION  
  TO sales_publication  
  FOR ml_user1  
  OPTION eh='off';
```

Erweiterte Option IgnoreScheduling (isc)

Gibt an, dass die erweiterte Option für die Abfolgeplanung ignoriert werden soll.

Syntax

```
isc={ ON | OFF }; ...
```

```
IgnoreScheduling={ ON | OFF }; ...
```

Bemerkungen

Wenn die Option auf ON gesetzt ist, ignoriert dbmlsync die erweiterte Option für die Abfolgeplanung und synchronisiert sofort. Der Standardwert ist **OFF**.

Diese Option ist gleichwertig mit der Option -is von dbmlsync.

Siehe auch

- [Übersicht über erweiterte dbmlsync-Optionen auf Seite 144](#)
- [„Synchronisationszeitpläne“ auf Seite 100](#)
- [„Erweiterte Option Schedule \(sch\)“ auf Seite 166](#)

Beispiel

Die folgende dbmlsync-Befehlszeile zeigt, wie Sie diese Option beim Start von dbmlsync verwenden können.

```
dbmlsync -e "isc=off"
```

Die folgende SQL-Anweisung zeigt, wie Sie diese Option in der Datenbank speichern können.

```
CREATE SYNCHRONIZATION SUBSCRIPTION  
  TO sales_publication  
  FOR ml_user1  
  OPTION isc='off';
```

Erweiterte Option Increment (inc)

Erlaubt inkrementelle Uploads und steuert die Größe der Upload-Inkmente.

Syntax

inc=*number*[**K** | **M**]; ...

Increment=*number*[**K** | **M**]; ...

Bemerkungen

Der Wert dieser Option gibt annähernd die Größe der einzelnen Upload-Teile in Byte an. Stellen Sie ein "k" oder "m" dahinter, um die Einheit in kB oder MB anzugeben.

Wenn diese Option auf einen anderen Wert als null gesetzt ist, werden Uploads in einem oder mehreren Teilen an MobiLink gesendet. Dies kann sinnvoll sein, wenn dbmlsync Probleme hat, eine Verbindung zum MobiLink-Server während der kompletten Uploadzeit aufrechtzuerhalten. Der Standardwert ist 0.

Der Wert der Option steuert die Größe der einzelnen Upload-Teile wie folgt: Dbmlsync erzeugt den Upload, indem das Datenbank-Transaktionslog gescannt wird. Wenn diese Option gesetzt ist, scannt dbmlsync die Anzahl von Byte, die in der Option festgelegt ist, und setzt dann den Scan bis zum ersten Punkt fort, an dem keine weiteren ausstehenden Teiltransaktionen mehr vorhanden sind, also zum nächsten Punkt, an dem alle Transaktionen festgeschrieben oder zurückgesetzt wurden. Danach sendet das Programm die gescannten Daten als Upload-Teil und setzt den Scan des Logs an der Stelle fort, an der er vorher beendet wurde.

Es ist nicht möglich, die erweiterte Option Increment zusammen mit skriptgesteuerten oder transaktionalen Uploads zu verwenden.

Siehe auch

- [Übersicht über erweiterte dbmlsync-Optionen auf Seite 144](#)

Beispiel

Die folgende dbmlsync-Befehlszeile zeigt, wie Sie diese Option beim Start von dbmlsync verwenden können.

```
dbmlsync -e "inc=32000"
```

Die folgende SQL-Anweisung zeigt, wie Sie diese Option in der Datenbank speichern können.

```
CREATE SYNCHRONIZATION SUBSCRIPTION  
  TO sales_publication  
  FOR ml_user1  
  OPTION inc='32k';
```

Erweiterte Option LockTables (lt)

Legt fest, dass Tabellen in der synchronisierten Publikation gesperrt werden müssen, bevor die Synchronisation erfolgt.

Syntax

lt={ ON | OFF | SHARE | EXCLUSIVE }; ...

LockTables={ ON | OFF | SHARE | EXCLUSIVE }; ...

Bemerkungen

SHARE bedeutet, dass dbmlsync alle Synchronisationstabellen im gemeinsam genutzten Modus sperrt. EXCLUSIVE bedeutet, dass dbmlsync alle Synchronisationstabellen im Exklusivmodus sperrt. Bei allen Plattformen außer Windows Mobile ist ON dasselbe wie SHARE. Für Windows Mobile-Geräte ist ON dasselbe wie EXCLUSIVE.

Die Option ist standardmäßig deaktiviert (OFF). Standardmäßig sperrt dbmlsync keine Synchronisationstabellen, außer in folgenden Situationen:

- Wenn eine Publikation existiert, die in der aktuellen Synchronisation skriptbasierte Uploads verwendet, oder wenn in der entfernten Datenbank ein sp_hook_dbmlsync_schema_upgrade-Hook definiert ist, sperrt dbmlsync die Synchronisationstabellen mit SHARE.

Wenn die Einstellung ON ist, sind während der Synchronisation keine Änderungen erlaubt.

Weitere Hinweise zu gemeinsamen und Exklusivsperrern finden Sie unter „[Funktionsweise von Sperren](#)“ [*SQL Anywhere Server - SQL-Benutzerhandbuch*] und „[LOCK TABLE-Anweisung](#)“ [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Weitere Hinweise zu Tabellensperren in MobiLink-Anwendungen finden Sie unter „[Parallelität während der Synchronisation](#)“ auf Seite 94.

Wenn Synchronisationstabellen im Exklusivmodus gesperrt sind (Standard für Windows Mobile), können keine anderen Verbindungen auf Tabellen zugreifen, und gespeicherte Prozeduren von dbmlsync, die eine eigene Verbindung benötigen, werden daher nicht ausgeführt, sofern Sie den Zugriff auf eine der synchronisierten Tabellen benötigen.

Weitere Informationen über Hooks mit eigenen Verbindungen finden Sie unter „[Ereignis-Hooks für SQL Anywhere-Clients](#)“ auf Seite 202.

Siehe auch

- [Übersicht über erweiterte dbmlsync-Optionen auf Seite 144](#)

Beispiel

Die folgende dbmlsync-Befehlszeile zeigt, wie Sie diese Option beim Start von dbmlsync verwenden können.

```
dbmlsync -e "lt=on"
```

Die folgende SQL-Anweisung zeigt, wie Sie diese Option in der Datenbank speichern können.

```
CREATE SYNCHRONIZATION SUBSCRIPTION  
  TO sales_publication  
  FOR ml_user1  
  OPTION lt='on';
```

Erweiterte Option MirrorLogDirectory (mld)

Gibt den Speicherort alter Transaktionslog-Spiegeldateien an, damit sie gelöscht werden können.

Syntax

mld=*dir*, ...

MirrorLogDirectory=*dir*, ...

Bemerkungen

Mit dieser Option kann dbmlsync alte Transaktionslog-Spiegeldateien löschen, wenn einer der beiden folgenden Umstände eintritt:

- Der Offline-Transaktionslogspiegel befindet sich in einem anderen Verzeichnis als der Transaktionslogspiegel
oder
- dbmlsync läuft nicht auf demselben Computer wie der entfernte Datenbankserver

In einer normalen Installation befinden sich der aktive Transaktionslogspiegel und die unbenannten Transaktionslog-Spiegeldateien in demselben Verzeichnis und dbmlsync läuft auf demselben Computer wie die entfernte Datenbank, sodass diese Option nicht erforderlich ist und alte Transaktionslog-Spiegeldateien automatisch gelöscht werden.

Transaktionslogs in diesem Verzeichnis sind nur betroffen, wenn die Datenbankoption `delete_old_logs` auf ON, DELAY oder *n* Tage gesetzt ist.

Siehe auch

- [Übersicht über erweiterte dbmlsync-Optionen auf Seite 144](#)
- [„delete_old_logs-Option \[SQL Remote\]“ \[SQL Anywhere Server - Datenbankadministration\]](#)

Beispiel

Die folgende dbmlsync-Befehlszeile zeigt, wie Sie diese Option beim Start von dbmlsync verwenden können.

```
dbmlsync -e "mld=c:\tmp\file"
```

Die folgende SQL-Anweisung zeigt, wie Sie diese Option in der Datenbank speichern können.

```
CREATE SYNCHRONIZATION SUBSCRIPTION  
  TO sales_publication  
  FOR ml_user1  
  OPTION mld='c:\tmp\file';
```

Erweiterte Option MobiLinkPwd (mp)

Legt das MobiLink-Kennwort fest.

Syntax

mp=password; ...

MobiLinkPwd=password; ...

Bemerkungen

Legt das Kennwort fest, das zum Verbinden mit dem MobiLink-Server verwendet wird. Dieses Kennwort muss das richtige Kennwort für den MobiLink-Benutzer sein, dessen Subskriptionen synchronisiert werden. Der Standardwert ist NULL.

Wenn der MobiLink-Benutzer bereits ein Kennwort hat, benutzen Sie die erweiterte Option **-e mn**, um es zu ändern.

Siehe auch

- [Übersicht über erweiterte dbmlsync-Optionen auf Seite 144](#)
- [„Erweiterte Option NewMobiLinkPwd \(mn\)“ auf Seite 163](#)
- [„dbmlsync-Option -mn“ auf Seite 124](#)
- [„dbmlsync-Option -mp“ auf Seite 125](#)

Beispiel

Die folgende dbmlsync-Befehlszeile zeigt, wie Sie diese Option beim Start von dbmlsync verwenden können.

```
dbmlsync -e "mp=password"
```

Die folgende SQL-Anweisung zeigt, wie Sie diese Option in der Datenbank speichern können.

```
CREATE SYNCHRONIZATION SUBSCRIPTION  
  TO sales_publication  
  FOR ml_user1  
  OPTION mp='password';
```

Erweiterte Option NewMobiLinkPwd (mn)

Legt ein neues MobiLink-Kennwort fest.

Syntax

mn=*new-password*; ...

NewMobiLinkPwd=*new-password*; ...

Bemerkungen

Legt ein neues Kennwort für den MobiLink-Benutzer an, dessen Subskriptionen synchronisiert werden. Benutzen Sie diese Option, wenn Sie ein bestehendes Kennwort ändern wollen. In der Standardeinstellung wird das Kennwort nicht geändert.

Siehe auch

- [Übersicht über erweiterte dbmlsync-Optionen auf Seite 144](#)
- [„Erweiterte Option MobiLinkPwd \(mp\)“ auf Seite 162](#)
- [„dbmlsync-Option -mn“ auf Seite 124](#)
- [„dbmlsync-Option -mp“ auf Seite 125](#)

Beispiel

Die folgende dbmlsync-Befehlszeile zeigt, wie Sie diese Option beim Start von dbmlsync verwenden können.

```
dbmlsync -e "mp=oldpassword;mn=newpassword"
```

Die folgende SQL-Anweisung zeigt, wie Sie diese Option in der Datenbank speichern können.

```
CREATE SYNCHRONIZATION SUBSCRIPTION  
  TO sales_publication  
  FOR ml_user1  
  OPTION mp='oldpassword';mn='newpassword'
```

Erweiterte Option NoSyncOnStartup (nss)

Verhindert eine Synchronisation durch dbmlsync beim Start, wenn dies normalerweise durch eine Abfolgeplanungsoption ausgelöst würde.

Syntax

nss={ **on** | **off** }; ...

NoSyncOnStartup={ **on** | **off** }; ...

Bemerkungen

Diese Option wird nur bei Verwendung der erweiterten Option für die Abfolgeplanungen mit den Klauseln EVERY oder INFINITE aktiv. Diese Abfolgeplanungsoptionen lösen eine automatische Synchronisation durch dbmlsync beim Start aus.

Die Option ist standardmäßig deaktiviert.

Wenn Sie NoSyncOnStartup auf "on" setzen und eine Abfolgeplanung mit der Klausel INFINITE verwenden, wird die Synchronisation erst ausgelöst, wenn eine Fensternachricht eingeht.

Wenn Sie NoSyncOnStartup auf "on" setzen und eine Abfolgeplanung mit der Klausel EVERY verwenden, wird die Synchronisation nach der Zeitspanne ausgelöst, die in der Klausel EVERY festgelegt wurde.

Diese Einstellung beeinflusst die Abfolgeplanung nur beim Start mit dbmlsync.

Siehe auch

- [Übersicht über erweiterte dbmlsync-Optionen auf Seite 144](#)
- [„Erweiterte Option Schedule \(sch\)“ auf Seite 166](#)
- [„Synchronisationszeitpläne“ auf Seite 100](#)

Beispiel

Der folgende Ausschnitt aus einer dbmlsync-Befehlszeile zeigt, wie Sie diese Option beim Start von dbmlsync verwenden können.

```
dbmlsync -e "schedule=EVERY:01:00;nss=off"...
```

Die folgende SQL-Anweisung zeigt, wie Sie diese Option in der Datenbank speichern können.

```
CREATE SYNCHRONIZATION SUBSCRIPTION  
  TO sales_publication  
  FOR ml_user1  
  OPTION nss='off', schedule='EVERY:01:00';
```

Erweiterte Option OfflineDirectory (dir)

Legt den Pfad fest, der Offline-Transaktionslogs enthält.

Syntax

`dir=path; ...`

`OfflineDirectory=path; ...`

Bemerkungen

Standardmäßig sucht dbmlsync nach umbenannten Logs in dem Verzeichnis, in dem sich das Online-Transaktionslog befindet. Diese Option muss nur angegeben werden, wenn sich die umbenannten Offline-Transaktionslogs in einem anderen Verzeichnis befinden.

Siehe auch

- [Übersicht über erweiterte dbmlsync-Optionen auf Seite 144](#)

Beispiel

Die folgende dbmlsync-Befehlszeile zeigt, wie Sie diese Option beim Start von dbmlsync verwenden können.

```
dbmlsync -e "dir=c:\db\logs"
```

Die folgende SQL-Anweisung zeigt, wie Sie diese Option in der Datenbank speichern können.

```
CREATE SYNCHRONIZATION SUBSCRIPTION  
  TO sales_publication  
  FOR ml_user1  
  OPTION dir='c:\db\logs';
```

Erweiterte Option PollingPeriod (pp)

Legt die Log-Scan-Abrufsperiode fest.

Syntax

pp=*number*[**S** | **M** | **H** | **D**]; ...

PollingPeriod=*number*[**S** | **M** | **H** | **D**]; ...

Bemerkungen

Diese Option legt das Intervall zwischen Logscans fest. Benutzen Sie die Suffixe "s", "m", "h" oder "d", um Sekunden, Minuten, Stunden oder Tage anzugeben. Standardwert ist **1** Minute. Wenn Sie kein Suffix eingeben, wird als Standardzeiteinheit die Minute verwendet.

Logscan-Abruf erfolgt nur, wenn Sie Synchronisationen planen oder den Hook `sp_hook_dbmlsync_delay` verwenden.

Eine Erklärung des Logscan-Abrufs finden Sie unter „[Erweiterte Option DisablePolling \(p\)](#)“ auf Seite 152.

Diese Option ist identisch mit **dbmlsync -pp**.

Siehe auch

- [Übersicht über erweiterte dbmlsync-Optionen auf Seite 144](#)
- „[Erweiterte Option DisablePolling \(p\)](#)“ auf Seite 152
- „[dbmlsync-Option -pp](#)“ auf Seite 130
- „[dbmlsync-Option -p](#)“ auf Seite 127
- „[sp_hook_dbmlsync_delay](#)“ auf Seite 218
- „[Erweiterte Option Schedule \(sch\)](#)“ auf Seite 166

Beispiel

Die folgende dbmlsync-Befehlszeile zeigt, wie Sie diese Option beim Start von dbmlsync verwenden können.

```
dbmlsync -e "pp=5"
```

Die folgende SQL-Anweisung zeigt, wie Sie diese Option in der Datenbank speichern können.

```
CREATE SYNCHRONIZATION SUBSCRIPTION  
  TO sales_publication  
  FOR ml_user1  
  OPTION pp='5';
```

Erweiterte Option Schedule (sch)

Legt eine Zeitplanung für die Synchronisation fest.

Syntax

sch=*schedule*; ...

Schedule=*schedule*; ...

schedule : { **EVERY**:*hhhh:mm* | **INFINITE** | *singleSchedule* }

hhhh : **00** ... **9999**

mm : **00** ... **59**

singleSchedule : *day* @*hh:mm* [**AM** | **PM**] [-*hh:mm* [**AM** | **PM**]] ,...

hh : **00** ... **24**

mm : **00** ... **59**

day :

EVERYDAY | **WEEKDAY** | **MON** | **TUE** | **WED** | **THU** | **FRI** | **SAT** | **SUN** | *dayOfMonth*

dayOfMonth : **0**... **31**

Bemerkungen

EVERY Mit dem Schlüsselwort **EVERY** findet die Synchronisation beim Start statt, und die Synchronisation wird nach Ablauf des angegebenen Zeitraums ohne Begrenzung wiederholt. Wenn der Synchronisationsprozess länger dauert als der angegebene Zeitraum, beginnt die Synchronisation sofort von Neuem.

Mit der Option **NoSyncOnStartup** können Sie eine Synchronisation beim Start von **dbmlsync** vermeiden.

singleSchedule Wenn ein oder mehrere einzelne Abfolgepläne angegeben werden, findet die Synchronisation nur am angegebenen Tag und zur angegebenen Uhrzeit statt.

Ein Intervall wird in der folgenden Form angegeben: @*hh:mm-hh:mm* (mit optionaler Angabe von **AM** oder **PM**). Wenn weder **AM** noch **PM** angegeben wird, geht das System von einer 24-Stunden-Angabe aus. 24:00 wird als 00:00 am nächsten Tag interpretiert. Wenn ein Intervall angegeben wird, findet die Synchronisation zu einem beliebigen Zeitpunkt innerhalb des Intervalls statt. Das Intervall stellt ein Zeitfenster für die Synchronisation bereit, sodass mehrere entfernte Datenbanken mit demselben Zeitplan nicht durch Synchronisation zur selben Zeit Datenstau auf dem MobiLink-Server verursachen.

Die Endzeit des Intervalls wird immer so interpretiert, dass sie nach der Startzeit liegt. Wenn das Zeitintervall Mitternacht einschließt, endet es am nächsten Tag. Wenn dbmlsync in der Mitte des Intervalls gestartet wird, findet die Synchronisation zu einem beliebigen Zeitpunkt vor der Endzeit statt.

EVERYDAY EVERYDAY bezeichnet alle sieben Wochentage.

WEEKDAY WEEKDAY ist Montag bis Freitag. Sie können die Tage abgekürzt oder in Langform eingeben. Die Langform müssen Sie jedoch verwenden, wenn die verwendete Sprache nicht Englisch ist, nicht die vom Client in der Verbindungszeichenfolge verlangte Sprache ist, und nicht die Sprache ist, die im Fenster "Servermeldungen" erscheint.

dayOfMonth Um den letzten Tag des Monats unabhängig von der Länge des Monats einzugeben, setzen Sie *dayOfMonth* auf 0.

INFINITE Mit dem Schlüsselwort INFINITE führt dbmlsync beim Start eine Synchronisation durch. Die nächste Synchronisation muss dann durch ein anderes Programm ausgelöst werden, dass eine Fensternachricht an dbmlsync sendet. Mit der erweiterten Option NoSyncOnStartup können Sie die Synchronisation beim Start verhindern.

Sie können diese Option in Verbindung mit der dbmlsync-Option -wc verwenden, um dbmlsync zu aktivieren und eine Synchronisation auszuführen.

Wenn eine frühere Synchronisation zu einem vorgesehenen Zeitpunkt nicht abgeschlossen ist, beginnt die eingeplante Synchronisation, sobald die frühere Synchronisation abgeschlossen ist.

Der Standardwert ist "Kein Zeitplan".

Die Option für Abfolgepläne wird ignoriert, wenn die Dbmlsync-API benutzt oder die SQL SYNCHRONIZE-Anweisung verwendet wird.

Die erweiterte Option IgnoreScheduling und die dbmlsync-Option -is weisen dbmlsync an, die Abfolgeplanung zu ignorieren, was zu einer sofortigen Synchronisation führt.

Siehe auch

- [Übersicht über erweiterte dbmlsync-Optionen auf Seite 144](#)
- [„Erweiterte Option NoSyncOnStartup \(nss\)“ auf Seite 163](#)
- [„dbmlsync-Option -wc“ auf Seite 142](#)
- [„Synchronisationszeitpläne“ auf Seite 100](#)
- [„Erweiterte Option IgnoreScheduling \(isc\)“ auf Seite 158](#)
- [„dbmlsync-Option -is“ auf Seite 123](#)

Beispiel

Die folgende dbmlsync-Befehlszeile zeigt, wie Sie diese Option beim Start von dbmlsync verwenden können.

```
dbmlsync -e "sch=WEEKDAY@8:00am,SUN@9:00pm"
```

Die folgende SQL-Anweisung zeigt, wie Sie diese Option in der Datenbank speichern können.

```
CREATE SYNCHRONIZATION SUBSCRIPTION  
TO sales_publication
```

```
FOR ml_user1  
OPTION sch='WEEKDAY@8:00am,SUN@9:00pm';
```

Erweiterte Option ScriptVersion (sv)

Legt eine Skriptversion fest.

Vorsicht

Es wird dringend empfohlen, dass Sie die Skriptversion mit der SCRIPT VERSION-Klausel in den CREATE SYNCHRONIZATION SUBSCRIPTION- und ALTER SYNCHRONIZATION SUBSCRIPTION-Anweisungen festlegen und nicht mit der erweiterten ScriptVersion-Option, weil die SCRIPT VERSION-Klausel Schema-Upgrades wesentlich vereinfacht.

Die erweiterte Option ScriptVersion (sv) überschreibt den mithilfe der SCRIPT VERSION-Klausel gespeicherten Wert und sollte nur aus Gründen der Abwärtskompatibilität oder für den seltenen Fall verwendet werden, in dem Sie die für eine Synchronisation verwendete Skriptversion explizit angeben müssen.

Siehe „CREATE SYNCHRONIZATION SUBSCRIPTION-Anweisung [MobiLink]“ [*SQL Anywhere Server - SQL-Referenzhandbuch*] und „ALTER SYNCHRONIZATION SUBSCRIPTION-Anweisung [MobiLink]“ [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Syntax

sv=*version-name*; ...

ScriptVersion=*version-name*; ...

Bemerkungen

Die Skriptversion legt fest, welche Skripten von MobiLink in der konsolidierten Datenbank während der Synchronisation ausgeführt werden. Der Standard-Skriptversionsname ist **default**.

Siehe auch

- [Übersicht über erweiterte dbmlsync-Optionen auf Seite 144](#)

Beispiel

Die folgende dbmlsync-Befehlszeile zeigt, wie Sie diese Option beim Start von dbmlsync verwenden können.

```
dbmlsync -e "sv=SysAd001"
```

Die folgende SQL-Anweisung zeigt, wie Sie diese Option in der Datenbank speichern können.

```
CREATE SYNCHRONIZATION SUBSCRIPTION  
TO sales_publication  
FOR ml_user1  
OPTION sv='SysAd001';
```

Erweiterte Option SendDownloadAck (sa)

Legt fest, dass eine Downloadbestätigung vom Client an den Server gesendet werden soll.

Syntax

sa={ ON | OFF }; ...

SendDownloadAck={ ON | OFF }; ...

Bemerkungen

Eine Downloadbestätigung teilt dem MobiLink-Server mit, dass ein Download in eine entfernte Datenbank übernommen wurde. Sie können Synchronisationsskripte in Ihrer konsolidierten Datenbank schreiben, um die Bestätigung zu verarbeiten und zu diesem Zeitpunkt Geschäftslogik auszuführen. Eine Downloadbestätigung kann nicht gesendet werden, wenn die Netzwerksitzung gelöscht wird, nachdem der Client den Download übernommen hat. Ihre Skripte sollten diese Möglichkeit also berücksichtigen. Siehe „[nonblocking_download_ack \(Verbindungsereignis\)](#)“ [*MobiLink - Serveradministration*].

Hinweis: Wenn SendDownloadAck aktiviert ist und Sie den Modus der ausführlichen Protokollierung eingeschaltet haben, wird eine Bestätigungszeile in das Clientlog geschrieben.

Der Standardwert ist **OFF**.

Siehe auch

- [Übersicht über erweiterte dbmlsync-Optionen auf Seite 144](#)

Beispiel

Die folgende dbmlsync-Befehlszeile zeigt, wie Sie diese Option beim Start von dbmlsync verwenden können.

```
dbmlsync -e "sa=on"
```

Die folgende SQL-Anweisung zeigt, wie Sie diese Option in der Datenbank speichern können.

```
CREATE SYNCHRONIZATION SUBSCRIPTION  
  TO sales_publication  
  FOR ml_user1  
  OPTION sa='on';
```

Erweiterte Option SendTriggers (st)

Legt fest, dass Triggeraktionen beim Upload gesendet werden sollen.

Syntax

st={ ON | OFF }; ...

SendTriggers={ ON | OFF }; ...

Bemerkungen

Kaskadierendes Löschen wird ebenfalls als Triggeraktion angesehen.

Der Standardwert ist **OFF**.

Wenn zwei Subskriptionen dieselbe(n) Tabelle(n) enthalten, müssen beide Subskriptionen mit derselben Einstellungen für die SendTriggers-Option synchronisiert werden.

Hinweis

Wenn in der Downloadphase der Synchronisation Änderungen an der Datenbank vorgenommen werden und diese zu Triggeraktionen führen, werden diese nie synchronisiert, unabhängig vom Wert der SendTriggers-Option.

Damit ein Vorgang in einem Trigger synchronisiert wird, wenn die SendTrigger-Option auf **ON** gesetzt ist, muss der Vorgang im Trigger auf eine zu synchronisierende Tabelle in einer Publikation angewendet werden. Es ist nicht notwendig, dass der zugrunde liegende Vorgang, der den Trigger ausgelöst hat, sich ebenfalls in der Publikation befindet.

Dbmlsync fügt Vorgänge aus einer einzelnen Zeile zusammen, aber wenn die SendTrigger-Option auf **ON** gesetzt ist, werden alle im Trigger ausgelösten Vorgänge ebenfalls gesendet, auch wenn der zugrunde liegende Vorgang, der den Trigger ausgelöst hat, mit anderen Vorgängen zusammengefügt wurde.

Integrierte Aktionen für die referenzielle Integrität von Fremdschlüsseln (ON DELETE CASCADE und ON UPDATE CASCADE) gelten als Triggeraktionen und werden nur synchronisiert, wenn die SendTrigger-Option auf **ON** gesetzt ist. Die einzige Ausnahme ist der Fall, dass eine Aktion für die referenzielle Integrität in einer Zeile ausgeführt wird, die sich bereits im Upload-Datenstrom befindet. In diesem Fall wird die Aktion für die referenzielle Integrität im systemgenerierten Trigger mit dem Zustand der Zeile zusammengefügt, die sich bereits im Upload-Datenstrom befindet, wie im folgenden Beispiel dargestellt.

In diesem Beispiel gibt es einen Fremdschlüssel zwischen der **Parent**-Tabelle und der **Child**-Tabelle mit einer ON DELETE CASCADE-Aktion für die referenzielle Integrität.

```
INSERT INTO Parent (pid, pname) values ( 100, 'Amy' );
INSERT INTO Child (cid, pid, cname) values ( 2000, 100, 'Alex' );
COMMIT;
DELETE FROM Parent WHERE pid = 100;
COMMIT;
```

Es ist wichtig, zu beachten, dass die RI-Aktion auch Zeile 2000 aus der Tabelle **Child** löscht, wenn Sie Zeile 100 aus der Tabelle **Parent** löschen. Wenn dbmlsync zu diesem Zeitpunkt ausgeführt wird, führt der INSERT-Vorgang mit anschließendem DELETE in Zeile 100 der **Parent**-Tabelle dazu, dass die Zeile nicht synchronisiert wird. Die Einfügung in die **Child**-Tabelle wird jedoch gesendet, wenn die SendTrigger-Option auf **OFF** gesetzt ist. Weil dbmlsync in diesem Fall bereits Zeile 2000 der **Child**-Tabelle zum Upload-Datenstrom hinzugefügt hat, wird die Aktion für die referenzielle Integrität beim Löschen von Zeile 100 der **Parent**-Tabelle, die Zeile 2000 der **Child**-Tabelle löscht, mit der vorhandenen Zeile im Upload-Datenstrom zusammengefügt, was dazu führt, dass keine der Zeilen synchronisiert wird und die Konsistenz der Daten auf beiden Seiten erhalten bleibt.

Siehe auch

- [Übersicht über erweiterte dbmlsync-Optionen auf Seite 144](#)

Beispiel

Die folgende dbmlsync-Befehlszeile zeigt, wie Sie diese Option beim Start von dbmlsync verwenden können.

```
dbmlsync -e "st=on"
```

Die folgende SQL-Anweisung zeigt, wie Sie diese Option in der Datenbank speichern können.

```
CREATE SYNCHRONIZATION SUBSCRIPTION  
  TO sales_publication  
  FOR ml_user1  
  OPTION st='on';
```

Erweiterte Option TableOrder (tor)

Legt die Reihenfolge der Tabellen im Upload fest.

Syntax

tor=*tables*; ...

TableOrder=*tables*; ...

tables = *table-name* [, *table-name*], ...

Bemerkungen

Mit dieser Option legen Sie die Reihenfolge fest, in der die Tabellen hochgeladen werden. Sie müssen alle Tabellen angeben, für die ein Upload erfolgen soll. Wenn Sie Tabellen einbeziehen, die in der Synchronisation nicht enthalten sind, werden diese ignoriert.

Die festgelegte Tabellenfolge muss die referenzielle Integrität bewahren. Wenn Tabelle1 eine Fremdschlüsselreferenz auf Tabelle2 enthält, muss Tabelle2 vor Tabelle1 hochgeladen werden. Wenn Sie die Tabellen nicht in der richtigen Reihenfolge angeben, wird ein Fehler erzeugt. Ausnahmen sind die folgenden beiden Fälle:

- Sie setzen TableOrderChecking=OFF.
- Ihre Tabellen verfügen über eine zyklische Fremdschlüsselbeziehung. (In diesem Fall gibt es keine Reihenfolge, die der Regel entspricht. Daher können die einbezogenen Tabellen des Zyklus in jeder beliebigen Reihenfolge heraufgeladen werden.)

Wenn Sie keine Tabellenfolge festlegen, wählt dbmlsync eine Reihenfolge, die der referenziellen Integrität entspricht.

Die Tabellenfolge des Downloads entspricht der des Uploads. Die Kontrolle der Tabellenfolge in Uploads erleichtert das Erstellen serverseitiger Skripten, vor allem wenn entfernte und konsolidierte Datenbanken unterschiedliche Fremdschlüsselintegritätsregeln aufweisen.

Siehe auch

- [Übersicht über erweiterte dbmsync-Optionen auf Seite 144](#)
- [„Erweiterte Option TableOrderChecking \(toc\)“ auf Seite 172](#)
- [„Verarbeitung des Uploads“ \[*MobiLink - Erste Orientierung*\]](#)
- [„Referenzielle Integrität und Synchronisation“ \[*MobiLink - Erste Orientierung*\]](#)

Beispiel

Die folgende dbmsync-Befehlszeile zeigt, wie Sie diese Option beim Start von dbmsync verwenden können.

```
dbmsync -e "tor=admin,primary,foreign"
```

Die folgende SQL-Anweisung zeigt, wie Sie diese Option in der Datenbank speichern können.

```
CREATE SYNCHRONIZATION SUBSCRIPTION  
TO sales_publication  
FOR ml_user1  
OPTION tor='admin,primary,foreign';
```

Erweiterte Option TableOrderChecking (toc)

Hiermit können Sie die Prüfung der referenziellen Integrität für die Tabellenfolge deaktivieren, die durch die erweiterte TableOrder-Option festgelegt wird.

Syntax

```
toc={ OFF | ON }; ...
```

```
TableOrderChecking={ OFF | ON }; ...
```

Bemerkungen

Bei den meisten Anwendungen haben Tabellen in den entfernten Datenbanken und der konsolidierten Datenbank dieselben Fremdschlüsselbeziehungen. In diesen Fällen sollten Sie die Standardeinstellung ON für TableOrderChecking verwenden. Dbmsync stellt dann sicher, dass keine Tabelle vor einer anderen Tabelle übertragen wird, für die ein Fremdschlüssel existiert. Dies garantiert referenzielle Integrität.

Diese Option ist hilfreich, wenn konsolidierte und entfernte Datenbanken unterschiedliche Fremdschlüsselbeziehungen aufweisen. Verwenden Sie sie mit der erweiterten TableOrder-Option, um eine Tabellenfolge zu definieren, die die Integritätsregeln zur Erhaltung der referenziellen Integrität auf dem Server auch dann erfüllt, wenn sie diejenigen der entfernten Datenbank verletzt.

Diese Option ist nur hilfreich, wenn die erweiterte Option TableOrder festgelegt wurde.

Der Standardwert ist **ON**.

Siehe auch

- [Übersicht über erweiterte dbmlsync-Optionen auf Seite 144](#)
- [„Erweiterte Option TableOrder \(tor\)“ auf Seite 171](#)
- [„Verarbeitung des Uploads“ \[*MobiLink - Erste Orientierung*\]](#)
- [„Referenzielle Integrität und Synchronisation“ \[*MobiLink - Erste Orientierung*\]](#)

Beispiel

Die folgende dbmlsync-Befehlszeile zeigt, wie Sie diese Option beim Start von dbmlsync verwenden können.

```
dbmlsync -e "toc=OFF"
```

Die folgende SQL-Anweisung zeigt, wie Sie diese Option in der Datenbank speichern können.

```
CREATE SYNCHRONIZATION SUBSCRIPTION  
  TO sales_publication  
  FOR ml_user1  
  OPTION toc='Off';
```

Erweiterte Option UploadOnly (uo)

Legt fest, dass die Synchronisation nur einen Uploadvorgang enthalten sollte.

Syntax

uo={ ON | OFF }; ...

UploadOnly={ ON | OFF }; ...

Bemerkungen

Während einer reinen Upload-Synchronisation bereitet dbmlsync einen Upload für den MobiLink-Server vor und versendet ihn genau so wie bei einer vollständigen Synchronisation. Anstatt aber einen Download zurückzusenden, sendet der MobiLink-Server nur eine Bestätigung, die angibt, ob der Upload erfolgreich festgeschrieben wurde.

Eine Liste aller Skripten, die für reine Upload-Synchronisationen definiert werden müssen, finden Sie unter [„Erforderliche Skripten“ \[*MobiLink - Serveradministration*\]](#).

Der Standardwert ist **OFF**.

Siehe auch

- [Übersicht über erweiterte dbmlsync-Optionen auf Seite 144](#)
- [„Reine Upload- und reine Download-Synchronisation“ \[*MobiLink - Serveradministration*\]](#)
- [„Erweiterte Option DownloadOnly \(ds\)“ auf Seite 153](#)

Beispiel

Die folgende dbmlsync-Befehlszeile zeigt, wie Sie diese Option beim Start von dbmlsync verwenden können.

```
dbmlsync -e "uo=on"
```

Die folgende SQL-Anweisung zeigt, wie Sie diese Option in der Datenbank speichern können.

```
CREATE SYNCHRONIZATION SUBSCRIPTION  
  TO sales_publication  
  FOR ml_user1  
  OPTION uo='on';
```

Erweiterte Option Verbose (v)

Legt fest, dass in aller Ausführlichkeit protokolliert wird.

Syntax

v={ ON | OFF }; ...

Verbose={ ON | OFF }; ...

Bemerkungen

Diese Option legt eine hohe Stufe der Ausführlichkeit fest, was die Performance beeinflussen kann und normalerweise nur in der Entwicklungsphase verwendet werden sollte.

Diese Option ist identisch mit **dbmlsync -v+**. Wenn Sie sowohl -v als auch die erweiterten Optionen angeben und Konflikte vorliegen, hebt die Option -v die erweiterte Option auf. Wenn kein Konflikt vorliegt, gelten die Ausführlichkeitsoptionen zusätzlich zu allen von Ihnen verwendeten Optionen. Wenn die Protokollausführlichkeit mithilfe von erweiterten Optionen festgelegt wird, findet die Protokollierung nicht sofort statt, sodass Startinformationen nicht protokolliert werden. Zum Zeitpunkt der ersten Synchronisation ist das Protokollierungsverhalten zwischen den Optionen -v und den erweiterten Optionen identisch.

Der Standardwert ist **OFF**.

Siehe auch

- [Übersicht über erweiterte dbmlsync-Optionen auf Seite 144](#)
- [„dbmlsync-Option -v“ auf Seite 141](#)
- [„Erweiterte Option VerboseHooks \(vs\)“ auf Seite 175](#)
- [„Erweiterte Option VerboseMin \(vm\)“ auf Seite 176](#)
- [„Erweiterte Option VerboseOptions \(vo\)“ auf Seite 176](#)
- [„Erweiterte Option VerboseRowCounts \(vn\)“ auf Seite 177](#)
- [„Erweiterte Option VerboseRowValues \(vr\)“ auf Seite 178](#)
- [„Erweiterte Option VerboseUpload \(vu\)“ auf Seite 179](#)

Beispiel

Die folgende dbmlsync-Befehlszeile zeigt, wie Sie diese Option beim Start von dbmlsync verwenden können.

```
dbmlsync -e "v=on"
```

Die folgende SQL-Anweisung zeigt, wie Sie diese Option in der Datenbank speichern können.

```
CREATE SYNCHRONIZATION SUBSCRIPTION  
  TO sales_publication
```

```
FOR ml_user1  
OPTION v='on';
```

Erweiterte Option VerboseHooks (vs)

Legt fest, dass Meldungen im Zusammenhang mit Hook-Prozedurskripten protokolliert werden sollen.

Syntax

vs={ ON | OFF }; ...

VerboseHooks={ ON | OFF }; ...

Bemerkungen

Diese Option ist identisch mit **dbmlsync -vs**. Wenn Sie sowohl -v als auch die erweiterten Optionen angeben und Konflikte vorliegen, hebt die Option -v die erweiterte Option auf. Wenn kein Konflikt vorliegt, gelten die Ausführlichkeitsoptionen zusätzlich zu allen von Ihnen verwendeten Optionen. Wenn die Protokollausführlichkeit mithilfe von erweiterten Optionen festgelegt wird, findet die Protokollierung nicht sofort statt, sodass Startinformationen nicht protokolliert werden. Zum Zeitpunkt der ersten Synchronisation ist das Protokollierungsverhalten zwischen den Optionen -v und den erweiterten Optionen identisch.

Der Standardwert ist **OFF**.

Siehe auch

- [Übersicht über erweiterte dbmlsync-Optionen auf Seite 144](#)
- [„dbmlsync-Option -v“ auf Seite 141](#)
- [„Ereignis-Hooks für SQL Anywhere-Clients“ auf Seite 202](#)
- [„Erweiterte Option Verbose \(v\)“ auf Seite 174](#)
- [„Erweiterte Option VerboseMin \(vm\)“ auf Seite 176](#)
- [„Erweiterte Option VerboseOptions \(vo\)“ auf Seite 176](#)
- [„Erweiterte Option VerboseRowCounts \(vn\)“ auf Seite 177](#)
- [„Erweiterte Option VerboseRowValues \(vr\)“ auf Seite 178](#)
- [„Erweiterte Option VerboseUpload \(vu\)“ auf Seite 179](#)

Beispiel

Die folgende dbmlsync-Befehlszeile zeigt, wie Sie diese Option beim Start von dbmlsync verwenden können.

```
dbmlsync -e "vs=on"
```

Die folgende SQL-Anweisung zeigt, wie Sie diese Option in der Datenbank speichern können.

```
CREATE SYNCHRONIZATION SUBSCRIPTION  
TO sales_publication  
FOR ml_user1  
OPTION vs='on';
```

Erweiterte Option VerboseMin (vm)

Legt fest, dass eine geringe Menge von Informationen protokolliert werden soll.

Syntax

vm={ ON | OFF }; ...

VerboseMin={ ON | OFF }; ...

Bemerkungen

Diese Option ist identisch mit **dbmlsync -v**. Wenn Sie sowohl -v als auch die erweiterten Optionen angeben und Konflikte vorliegen, hebt die Option -v die erweiterte Option auf. Wenn kein Konflikt vorliegt, gelten die Ausführlichkeitsoptionen zusätzlich zu allen von Ihnen verwendeten Optionen. Wenn die Protokollausführlichkeit mithilfe von erweiterten Optionen festgelegt wird, findet die Protokollierung nicht sofort statt, sodass Startinformationen nicht protokolliert werden. Zum Zeitpunkt der ersten Synchronisation ist das Protokollierungsverhalten zwischen den Optionen -v und den erweiterten Optionen identisch.

Der Standardwert ist **OFF**.

Siehe auch

- [Übersicht über erweiterte dbmlsync-Optionen auf Seite 144](#)
- [„dbmlsync-Option -v“ auf Seite 141](#)
- [„Erweiterte Option Verbose \(v\)“ auf Seite 174](#)
- [„Erweiterte Option VerboseOptions \(vo\)“ auf Seite 176](#)
- [„Erweiterte Option VerboseRowCounts \(vn\)“ auf Seite 177](#)
- [„Erweiterte Option VerboseRowValues \(vr\)“ auf Seite 178](#)
- [„Erweiterte Option VerboseUpload \(vu\)“ auf Seite 179](#)

Beispiel

Die folgende dbmlsync-Befehlszeile zeigt, wie Sie diese Option beim Start von dbmlsync verwenden können.

```
dbmlsync -e "vm=on"
```

Die folgende SQL-Anweisung zeigt, wie Sie diese Option in der Datenbank speichern können.

```
CREATE SYNCHRONIZATION SUBSCRIPTION  
TO sales_publication  
FOR ml_user1  
OPTION vm='on';
```

Erweiterte Option VerboseOptions (vo)

Legt fest, dass Informationen über die von Ihnen festgelegten Befehlszeilenoptionen (einschließlich erweiterten Optionen) protokolliert werden sollen.

Syntax

vo={ ON | OFF }; ...

VerboseOptions={ ON | OFF }; ...

Bemerkungen

Diese Option ist identisch mit **dbmlsync -vo**. Wenn Sie sowohl -v als auch die erweiterten Optionen angeben und Konflikte vorliegen, hebt die Option -v die erweiterte Option auf. Wenn kein Konflikt vorliegt, gelten die Ausführlichkeitsoptionen zusätzlich zu allen von Ihnen verwendeten Optionen. Wenn die Protokollausführlichkeit mithilfe von erweiterten Optionen festgelegt wird, findet die Protokollierung nicht sofort statt, sodass Startinformationen nicht protokolliert werden. Zum Zeitpunkt der ersten Synchronisation ist das Protokollierungsverhalten zwischen den Optionen -v und den erweiterten Optionen identisch.

Der Standardwert ist **OFF**.

Siehe auch

- [Übersicht über erweiterte dbmlsync-Optionen auf Seite 144](#)
- [„dbmlsync-Option -v“ auf Seite 141](#)
- [„Erweiterte Option Verbose \(v\)“ auf Seite 174](#)
- [„Erweiterte Option VerboseMin \(vm\)“ auf Seite 176](#)
- [„Erweiterte Option VerboseRowCounts \(vn\)“ auf Seite 177](#)
- [„Erweiterte Option VerboseRowValues \(vr\)“ auf Seite 178](#)
- [„Erweiterte Option VerboseUpload \(vu\)“ auf Seite 179](#)

Beispiel

Die folgende dbmlsync-Befehlszeile zeigt, wie Sie diese Option beim Start von dbmlsync verwenden können.

```
dbmlsync -e "vo=on"
```

Die folgende SQL-Anweisung zeigt, wie Sie diese Option in der Datenbank speichern können.

```
CREATE SYNCHRONIZATION SUBSCRIPTION  
TO sales_publication  
FOR ml_user1  
OPTION vo='on' ;
```

Erweiterte Option VerboseRowCounts (vn)

Legt fest, dass die Anzahl von Zeilen für den Upload und den Download protokolliert werden sollen.

Syntax

vn={ ON | OFF }; ...

VerboseRowCounts={ ON | OFF }; ...

Bemerkungen

Diese Option ist identisch mit **dbmlsync -vn**. Wenn Sie sowohl -v als auch die erweiterten Optionen angeben und Konflikte vorliegen, hebt die Option -v die erweiterte Option auf. Wenn kein Konflikt vorliegt, gelten die Ausführlichkeitsoptionen zusätzlich zu allen von Ihnen verwendeten Optionen. Wenn die Protokollausführlichkeit mithilfe von erweiterten Optionen festgelegt wird, findet die Protokollierung nicht sofort statt, sodass Startinformationen nicht protokolliert werden. Zum Zeitpunkt der ersten Synchronisation ist das Protokollierungsverhalten zwischen den Optionen -v und den erweiterten Optionen identisch.

nicht sofort statt, sodass Startinformationen nicht protokolliert werden. Zum Zeitpunkt der ersten Synchronisation ist das Protokollierungsverhalten zwischen den Optionen -v und den erweiterten Optionen identisch.

Der Standardwert ist **OFF**.

Siehe auch

- [Übersicht über erweiterte dbmlsync-Optionen auf Seite 144](#)
- [„dbmlsync-Option -v“ auf Seite 141](#)
- [„Erweiterte Option Verbose \(v\)“ auf Seite 174](#)
- [„Erweiterte Option Verbose \(v\)“ auf Seite 174](#)
- [„Erweiterte Option VerboseMin \(vm\)“ auf Seite 176](#)
- [„Erweiterte Option VerboseOptions \(vo\)“ auf Seite 176](#)
- [„Erweiterte Option VerboseRowValues \(vr\)“ auf Seite 178](#)
- [„Erweiterte Option VerboseUpload \(vu\)“ auf Seite 179](#)

Beispiel

Die folgende dbmlsync-Befehlszeile zeigt, wie Sie diese Option beim Start von dbmlsync verwenden können.

```
dbmlsync -e "vn=on"
```

Die folgende SQL-Anweisung zeigt, wie Sie diese Option in der Datenbank speichern können.

```
CREATE SYNCHRONIZATION SUBSCRIPTION  
  TO sales_publication  
  FOR ml_user1  
  OPTION vn='on';
```

Erweiterte Option VerboseRowValues (vr)

Legt fest, dass die Werte von Zeilen für den Upload und den Download protokolliert werden sollen.

Syntax

vr={ ON | OFF }; ...

VerboseRowValues={ ON | OFF }; ...

Bemerkungen

Diese Option ist identisch mit **dbmlsync -vr**. Wenn Sie sowohl -v als auch die erweiterten Optionen angeben und Konflikte vorliegen, hebt die Option -v die erweiterte Option auf. Wenn kein Konflikt vorliegt, gelten die Ausführlichkeitsoptionen zusätzlich zu allen von Ihnen verwendeten Optionen. Wenn die Protokollausführlichkeit mithilfe von erweiterten Optionen festgelegt wird, findet die Protokollierung nicht sofort statt, sodass Startinformationen nicht protokolliert werden. Zum Zeitpunkt der ersten Synchronisation ist das Protokollierungsverhalten zwischen den Optionen -v und den erweiterten Optionen identisch.

Der Standardwert ist **OFF**.

Siehe auch

- [Übersicht über erweiterte dbmlsync-Optionen auf Seite 144](#)
- [„dbmlsync-Option -v“ auf Seite 141](#)
- [„Erweiterte Option Verbose \(v\)“ auf Seite 174](#)
- [„Erweiterte Option Verbose \(v\)“ auf Seite 174](#)
- [„Erweiterte Option VerboseMin \(vm\)“ auf Seite 176](#)
- [„Erweiterte Option VerboseOptions \(vo\)“ auf Seite 176](#)
- [„Erweiterte Option VerboseRowCounts \(vn\)“ auf Seite 177](#)
- [„Erweiterte Option VerboseUpload \(vu\)“ auf Seite 179](#)

Beispiel

Die folgende dbmlsync-Befehlszeile zeigt, wie Sie diese Option beim Start von dbmlsync verwenden können.

```
dbmlsync -e "vr=on"
```

Die folgende SQL-Anweisung zeigt, wie Sie diese Option in der Datenbank speichern können.

```
CREATE SYNCHRONIZATION SUBSCRIPTION  
  TO sales_publication  
  FOR ml_user1  
  OPTION vr='on' ;
```

Erweiterte Option VerboseUpload (vu)

Legt fest, dass Informationen über den Upload-Datenstrom protokolliert werden sollen.

Syntax

vu={ ON | OFF }; ...

VerboseUpload={ ON | OFF }; ...

Bemerkungen

Diese Option ist identisch mit **dbmlsync -vu**. Wenn Sie sowohl -v als auch die erweiterten Optionen angeben und Konflikte vorliegen, hebt die Option -v die erweiterte Option auf. Wenn kein Konflikt vorliegt, gelten die Ausführlichkeitsoptionen zusätzlich zu allen von Ihnen verwendeten Optionen. Wenn die Protokollausführlichkeit mithilfe von erweiterten Optionen festgelegt wird, findet die Protokollierung nicht sofort statt, sodass Startinformationen nicht protokolliert werden. Zum Zeitpunkt der ersten Synchronisation ist das Protokollierungsverhalten zwischen den Optionen -v und den erweiterten Optionen identisch.

Der Standardwert ist **OFF**.

Siehe auch

- [Übersicht über erweiterte dbmlsync-Optionen auf Seite 144](#)
- [„dbmlsync-Option -v“ auf Seite 141](#)
- [„Erweiterte Option Verbose \(v\)“ auf Seite 174](#)
- [„Erweiterte Option Verbose \(v\)“ auf Seite 174](#)
- [„Erweiterte Option VerboseMin \(vm\)“ auf Seite 176](#)
- [„Erweiterte Option VerboseOptions \(vo\)“ auf Seite 176](#)
- [„Erweiterte Option VerboseRowCounts \(vn\)“ auf Seite 177](#)
- [„Erweiterte Option VerboseRowValues \(vr\)“ auf Seite 178](#)

Beispiel

Die folgende dbmlsync-Befehlszeile zeigt, wie Sie diese Option beim Start von dbmlsync verwenden können.

```
dbmlsync -e "vu=on"
```

Die folgende SQL-Anweisung zeigt, wie Sie diese Option in der Datenbank speichern können.

```
CREATE SYNCHRONIZATION SUBSCRIPTION  
  TO sales_publication  
  FOR ml_user1  
  OPTION vu='on';
```

MobiLink SQL-Anweisungen

Im Folgenden werden die SQL-Anweisungen für die Konfiguration und Ausführung von MobiLink SQL Anywhere-Clients aufgelistet:

- „ALTER PUBLICATION-Anweisung [MobiLink] [SQL Remote]“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- „ALTER SYNCHRONIZATION PROFILE-Anweisung [MobiLink]“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- „ALTER SYNCHRONIZATION SUBSCRIPTION-Anweisung [MobiLink]“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- „ALTER SYNCHRONIZATION USER-Anweisung [MobiLink]“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- „CREATE PUBLICATION-Anweisung [MobiLink] [SQL Remote]“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- „CREATE SYNCHRONIZATION PROFILE-Anweisung [MobiLink]“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- „CREATE SYNCHRONIZATION SUBSCRIPTION-Anweisung [MobiLink]“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- „CREATE SYNCHRONIZATION USER-Anweisung [MobiLink]“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- „DROP PUBLICATION-Anweisung [MobiLink] [SQL Remote]“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- „DROP SYNCHRONIZATION PROFILE-Anweisung [MobiLink]“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- „DROP SYNCHRONIZATION SUBSCRIPTION-Anweisung [MobiLink]“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- „DROP SYNCHRONIZATION USER-Anweisung [MobiLink]“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- „GRANT ROLE SYS_REPLICATION_ADMIN_ROLE-Anweisung [MobiLink] [SQL Remote]“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- „GRANT ROLE SYS_RUN_REPLICATION_ROLE-Anweisung [MobiLink] [SQL Remote]“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- „START SYNCHRONIZATION DELETE-Anweisung [MobiLink]“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- „START SYNCHRONIZATION SCHEMA CHANGE-Anweisung [MobiLink]“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- „STOP SYNCHRONIZATION DELETE-Anweisung [MobiLink]“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- „STOP SYNCHRONIZATION SCHEMA CHANGE-Anweisung [MobiLink]“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- „SYNCHRONIZE-Anweisung [MobiLink]“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]

UltraLite-Clients

Siehe „UltraLite-SQL-Anweisungen“ [*UltraLite - Datenbankverwaltung*].

MobiLink-Synchronisationsprofile

Mithilfe von Synchronisationsprofilen können Sie einige dbmlsync-Optionen in die Datenbank schreiben. Das erstellte Synchronisationsprofil kann eine Reihe von Synchronisationsoptionen enthalten.

Sie können Synchronisationsprofile mit folgenden Anweisungen erstellen, ändern und löschen:

- „CREATE SYNCHRONIZATION PROFILE-Anweisung [MobiLink]“ [\[SQL Anywhere Server - SQL-Referenzhandbuch\]](#)
- „ALTER SYNCHRONIZATION PROFILE-Anweisung [MobiLink]“ [\[SQL Anywhere Server - SQL-Referenzhandbuch\]](#)
- „DROP SYNCHRONIZATION PROFILE-Anweisung [MobiLink]“ [\[SQL Anywhere Server - SQL-Referenzhandbuch\]](#)

Synchronisationsprofile können mit der dbmlsync-Option -sp, der Sync-Methode in der Dbmlsync-API, der SQL SYNCHRONIZE-Anweisung und mit den von der zentralen Administration für entfernte Datenbanken erstellten entfernten Aufgaben aufgerufen werden. In allen Fällen gibt es die Möglichkeit, zusätzliche Optionen anzugeben, die mit den Optionen in den Synchronisationsprofilen zusammengeführt werden. Wenn eine der zusätzlich angegebenen Optionen mit einer im Synchronisationsprofil angegebenen Option in Konflikt steht, wird der Wert der zusätzlichen Optionen verwendet.

Bei Verwendung der dbmlsync-Optionen -sp werden alle anderen Optionen in der Befehlszeile als zusätzliche Optionen behandelt. Die anderen Schnittstellen bieten einen bestimmten Parameter oder eine Klausel für die Angabe von zusätzlichen Optionen.

Hinweise zur Verwendung von Synchronisationsprofilen in UltraLite finden Sie unter „Synchronisationsprofiloptionen“ [\[UltraLite - Datenbankverwaltung\]](#).

Die folgenden Optionen können in einem Synchronisationsprofil angegeben werden:

Langform des Optionsnamens	Kurzform des Namens	Zulässige Werte	Beschreibung
AuthParms	ap	Zeichenfolge	Übergibt Parameter an das Skript authenticate_parameters und an Authentifizierungsparameter. Siehe „AuthParms-Synchronisationsprofiloption“ auf Seite 186.
ApplyDnldFile	ba	Zeichenfolge	Übernimmt eine Download-Datei. Siehe „ApplyDnldFile-Synchronisationsprofiloption“ auf Seite 186.
Background	bk	Zeichenfolge	Aktiviert die Hintergrundsynchroisation, wenn auf TRUE gesetzt. Siehe „Background-Synchronisationsprofiloption“ auf Seite 186.

Langform des Optionsnamens	Kurzform des Namens	Zulässige Werte	Beschreibung
BackgroundRetry	bkr	Ganzzahl	Steuert wie dbmlsync sich nach dem Unterbrechen einer Hintergrundsynchroisation verhält. Siehe „BackgroundRetry-Synchronisationsprofiloption“ auf Seite 187.
ContinueDownload	dc	Boolescher Wert	Startet einen fehlgeschlagenen Download erneut. Siehe „ContinueDownload-Synchronisationsprofiloption“ auf Seite 188.
CreateDnldFile	bc	Zeichenfolge	Erstellt eine Download-Datei. Siehe „CreateDnldFile-Synchronisationsprofiloption“ auf Seite 188.
DnldFileExtra	be	Zeichenfolge	Beim Erstellen einer Download-Datei gibt diese Option eine zusätzliche Zeichenfolge an, die in die Datei aufgenommen werden soll. Siehe „DnldFileExtra-Synchronisationsprofiloption“ auf Seite 189.
DownloadOnly	ds	Boolescher Wert	Führt eine reine Download-Synchronisation durch. Siehe „DownloadOnly-Synchronisationsprofiloption“ auf Seite 190.
DownloadReadSize	drs	Ganzzahl	Bei neu startbaren Downloads legt diese Option den maximalen Umfang der Daten fest, die nach einem Kommunikationsfehler erneut gesendet werden müssen. Siehe „DownloadReadSize-Synchronisationsprofiloption“ auf Seite 190.
ExtOpt	e	Zeichenfolge	Legt erweiterte Optionen fest. Siehe „ExtOpt-Synchronisationsprofiloption“ auf Seite 191.
IgnoreHookErrors	eh	Boolescher Wert	Ignoriert alle Fehler, die in Hook-Funktionen auftreten. Siehe „IgnoreHookErrors-Synchronisationsprofiloption“ auf Seite 192.
IgnoreScheduling	is	Boolescher Wert	Ignoriert Anweisungen zur Abfolgeplanung, sodass die Synchronisation sofort ausgeführt wird. Siehe „IgnoreScheduling-Synchronisationsprofiloption“ auf Seite 192.
KillConnections	d	Boolescher Wert	Löscht kollidierende Sperren in der entfernten Datenbank. Siehe „KillConnections-Synchronisationsprofiloption“ auf Seite 192.

Langform des Optionsnamens	Kurzform des Namens	Zulässige Werte	Beschreibung
LogRenameSize	x	Eine optional durch K oder M ergänzte Ganzzahl.	Benennt das Transaktionslog um und startet es neu, nachdem es nach Upload-Daten durchsucht wurde. Siehe „ LogRenameSize-Synchronisationsprofiloption “ auf Seite 193.
MobiLinkPwd	mp	Zeichenfolge	Gibt das Kennwort des MobiLink-Benutzers an. Siehe „ MobiLinkPwd-Synchronisationsprofiloption “ auf Seite 194.
MLUser	u	Zeichenfolge	Gibt den MobiLink-Benutzernamen an. Siehe „ MLUser-Synchronisationsprofiloption (nicht mehr empfohlen) “ auf Seite 194.
NewMobi-LinkPwd	mn	Zeichenfolge	Übergibt ein neues Kennwort für den MobiLink-Benutzer. Benutzen Sie diese Option, wenn Sie ein bestehendes Kennwort ändern wollen. Siehe „ NewMobi-LinkPwd-Synchronisationsprofiloption “ auf Seite 195.
Ping	pi	Boolescher Wert	Pingt einen MobiLink-Server, um die Kommunikation zwischen dem Client und MobiLink zu bestätigen. Siehe „ Ping-Synchronisationsprofiloption “ auf Seite 195.
Publication	n	Zeichenfolge	Diese Option wird nicht mehr empfohlen. Gibt die zu synchronisierenden Publikationen an. Publication kann nur einmal in einem Synchronisationsprofil angegeben werden, aber die Befehlszeilenoption kann mehrmals angegeben werden. Siehe „ Publication-Synchronisationsprofiloption (nicht mehr empfohlen) “ auf Seite 196.
RemoteProgressGreater	ra	Boolescher Wert	Legt fest, dass der Offset der entfernten Datenbank verwendet werden soll, wenn er höher ist als der der konsolidierten Datenbank. Dies ist das Äquivalent zur dbmlsync-Option -ra. Siehe „ RemoteProgressGreater-Synchronisationsprofiloption “ auf Seite 197.
RemoteProgressLess	rb	Boolescher Wert	Legt fest, dass der Offset der entfernten Datenbank benutzt werden soll, wenn er niedriger ist als der der konsolidierten Datenbank (z.B. wenn die entfernte Datenbank aus einer Sicherung wiederhergestellt wurde). Dies ist das Äquivalent zur dbmlsync-Option -rb. Siehe „ RemoteProgressLess-Synchronisationsprofiloption “ auf Seite 197.

Langform des Optionsnamens	Kurzform des Namens	Zulässige Werte	Beschreibung
Subscription	s	Zeichenfolge	Gibt die zu synchronisierende(n) Subskription(en) an. Subscription kann nur einmal in einem Synchronisationsprofil angegeben werden. Siehe „ Subscription-Synchronisationsprofiloption “ auf Seite 198.
TransactionalUpload	tu	Boolescher Wert	Legt fest, dass jede Transaktion in der entfernten Datenbank als eigene Transaktion in einer eigenen Synchronisation ausgelesen werden soll. Siehe „ TransactionalUpload-Synchronisationsprofiloption “ auf Seite 198.
UpdateGenNum	bg	Boolescher Wert	Beim Erstellen einer Download-Datei erstellt diese Option eine Datei, die mit noch nicht synchronisierten entfernten Datenbanken verwendet werden kann. Siehe „ UpdateGenNum-Synchronisationsprofiloption “ auf Seite 199.
UploadOnly	uo	Boolescher Wert	Gibt an, dass die Synchronisation nur einen Upload umfasst und keine Downloads ausgeführt werden. Siehe „ UploadOnly-Synchronisationsprofiloption “ auf Seite 200.
UploadRowCnt	urc	Ganzzahl	Gibt eine Schätzung der Anzahl der Zeilen an, für die bei einer Synchronisation ein Upload durchgeführt werden soll. Siehe „ UploadRowCnt-Synchronisationsprofiloption “ auf Seite 200.
Verbosity		Zeichenfolge (eine durch Kommas getrennte Liste mit Optionen)	<p>Steuert die Ausführlichkeitsstufe von dbmlsync. Ähnlich wie „Verbosity-Synchronisationsprofiloption“ auf Seite 201.</p> <p>Die Werte müssen als durch Kommas getrennte Liste von mindestens einer Option eingegeben werden, die jeweils einer bestehenden -v-Option in der nachstehend beschriebenen Form entsprechen:</p> <ul style="list-style-type: none"> • BASIC - entspricht -v • HIGH - entspricht -v+ • CONNECT_STR - entspricht -vc • ROW_CNT - entspricht -vn • OPTIONS - entspricht -vo • ML_PASSWORD - entspricht -vp • ROW_DATA - entspricht -vr

AuthParms-Synchronisationsprofiloption

Übergibt Parameter an das Skript `authenticate_parameters` und an Authentifizierungsparameter.

Syntax

ap=*parameters*

Authparms=*parameters*

Bemerkungen

Benutzen Sie diese Option mit dem Verbindungsskript `authenticate_parameters` oder Authentifizierungsparametern.

Die Parameter werden an den MobiLink-Server geschickt und an das `authenticate_parameters`-Skript oder andere Ereignisse in der konsolidierten Datenbank übergeben.

Siehe auch

- „dbmlsync-Option -ap“ auf Seite 112

Beispiel

```
AuthParms=p1,p2,p3
```

ApplyDnldFile-Synchronisationsprofiloption

Übernimmt eine Download-Datei.

Syntax

ba=*filename*

ApplyDnldFile=*filename*

Bemerkungen

Geben Sie den Namen einer bestehenden Download-Datei an, die in die entfernte Datenbank übernommen werden soll.

Siehe auch

- „dbmlsync-Option -ba“ auf Seite 112

Beispiel

```
ApplyDnldFile=filename
```

Background-Synchronisationsprofiloption

Aktiviert die Hintergrundsynchronisation, wenn diese Option auf ON gesetzt ist.

Syntax**bk**={ON|OFF}**Background**={ON|OFF}**Bemerkungen**

Während einer Hintergrundsynchroisation trennt die Datenbank-Engine die dbmlsync-Verbindung mit der entfernten Datenbank und setzt alle noch nicht festgeschriebenen dbmlsync-Vorgänge zurück, wenn eine andere Verbindung darauf wartet, auf eine von dbmlsync gesperrte Datenbankressource zugreifen zu können. Dadurch können andere Verbindungen genutzt werden, ohne auf den Abschluss der Synchronisation zu warten. Abhängig von den Vorgängen, die für dbmlsync ausstanden, als die Verbindung getrennt wurde, kann es trotzdem zu einer längeren Verzögerung für die wartende Verbindung kommen, da die Datenbank die nicht festgeschriebenen dbmlsync-Änderungen zurücksetzt.

Siehe auch

- „dbmlsync-Option -bk“ auf Seite 115

Beispiel

Das folgende Beispiel zeigt die Verwendung der Background-Synchronisationsprofiloption:

```
Background=on
```

BackgroundRetry-Synchronisationsprofiloption

Ganzzahl. Steuert wie dbmlsync sich nach dem Unterbrechen einer Hintergrundsynchroisation verhält.

Syntax**bkr**=*integer***BackgroundRetry**=*integer***Bemerkungen**

Diese Option hat eine Kurz- und eine Langform: Sie können bkr oder BackgroundRetry verwenden.

Setzen Sie diesen Wert als Ganzzahl größer oder gleich -1. Wenn der Wert -1 ist, wiederholt dbmlsync eine unterbrochen Synchronisation, bis sie ohne Unterbrechung mit oder ohne Erfolg abgeschlossen werden kann. Wenn der Wert 0 ist, wiederholt dbmlsync die unterbrochene Synchronisation nicht. Wenn der Wert größer als 0 ist, wiederholt dbmlsync die Synchronisation so oft wie angegeben. Wenn die Synchronisation nach der angegebenen Anzahl von Versuchen noch nicht abgeschlossen wurde, wird sie als Vordergrundsynchroisation ausgeführt, damit sie ohne Unterbrechung abgeschlossen werden kann.

Standardmäßig ist BackgroundRetry auf 0 gesetzt.

Es wird ein Fehler erzeugt, wenn BackgroundRetry auf einen anderen Wert als null gesetzt wird, ohne die Background-Option auf ON zu setzen.

Diese Option wird ignoriert, wenn die Synchronisation mit der Dbmlsync-API oder der SQL SYNCHRONIZE-Anweisung initiiert wird.

Siehe auch

- „dbmlsync-Option -bkr“ auf Seite 115

Beispiel

Das folgende Beispiel zeigt, wie Sie die BackgroundRetry-Synchronisationsprofiloption verwenden:

```
BackgroundRetry=4
```

ContinueDownload-Synchronisationsprofiloption

Startet einen fehlgeschlagenen Download erneut.

Syntax

```
dc={ON|OFF}
```

```
ContinueDownload={ON|OFF}
```

Bemerkungen

Falls dbmlsync während eines Downloads einen Fehler feststellt, wird kein Teil des Downloads in die entfernte Datenbank übernommen. Der empfangene Teil des Downloads wird jedoch in einer temporären Datei auf dem entfernten Gerät gespeichert, sodass Sie den Download schneller abschließen können, wenn Sie bei der nächsten Synchronisation ContinueDownload=on angeben.

Bei der Verwendung von ContinueDownload=on versucht dbmlsync den Teil der Daten einzulesen, der beim letzten Downloadversuch nicht empfangen wurde. Wenn die verbliebenden Daten eingelesen werden können, wird der gesamte Download in Ihre entfernte Datenbank übernommen. Andernfalls schlägt die Synchronisation fehl.

Falls Sie ContinueDownload=on verwenden und neue Daten für einen Upload anstehen, schlägt der neu startbare Download fehl. Sie können auch einen fehlgeschlagenen Netzwerk-Download mit der erweiterten Option ContinueDownload oder dem Hook sp_hook_dbmlsync_end neu starten.

Siehe auch

- „dbmlsync-Option -dc“ auf Seite 118
- „sp_hook_dbmlsync_end“ auf Seite 232
- „Erweiterte Option ContinueDownload (cd)“ auf Seite 151

Beispiel

Das folgende Beispiel zeigt die Verwendung der ContinueDownload-Synchronisationsprofiloption:

```
ContinueDownload=on
```

CreateDnldFile-Synchronisationsprofiloption

Erstellt eine Download-Datei mit dem angegebenen Namen.

Syntax

bc=*filename*

CreateDnldFile=*filename*

Bemerkungen

Für Download-Dateien müssen Sie die Erweiterung *.df* verwenden.

Optional können Sie auch einen Pfad eingeben. Wenn Sie keinen Pfad eingeben, ist der Standardort das aktuelle Arbeitsverzeichnis von dbmlsync (das Verzeichnis, in dem dbmlsync gestartet wurde).

Sie können auch die Option *-be* verwenden, um eine Zeichenfolge anzugeben, die in der entfernten Datenbank validiert wird, sowie die Option *-bg*, um eine Download-Datei für eine neue entfernte Datenbank anzulegen.

Siehe auch

- „dbmlsync-Option *-bc*“ auf Seite 113
- „MobiLink - dateibasierte Downloads“ [*MobiLink - Serveradministration*]

Beispiel

```
CreateDnldFile=dnld1.df
```

DnldFileExtra-Synchronisationsprofiloption

Beim Erstellen einer Download-Datei gibt diese Option eine zusätzliche Zeichenfolge an, die in die Datei aufgenommen werden soll.

Syntax

be=*string*

DnldFileExtra=*string*

Bemerkungen

Die Zeichenfolge kann für die Authentifizierung oder andere Zwecke verwendet werden. Sie wird an die gespeicherte Prozedur `sp_hook_dbmlsync_validate_download_file` in der entfernten Datenbank übergeben, wenn die Download-Datei übernommen wird.

Die Zeichenfolge darf keine Semikolons enthalten.

Siehe auch

- „dbmlsync-Option *-be*“ auf Seite 113
- „`sp_hook_dbmlsync_validate_download_file`“ auf Seite 264

Beispiel

Das folgende Beispiel zeigt die Verwendung der DnldFileExtra-Synchronisationsprofiloption:

```
DnldFileExtra=val1,val2,val3
```

DownloadOnly-Synchronisationsprofiloption

Diese Option führt eine Synchronisation mit reinem Download durch, wenn sie auf On gesetzt wird.

Syntax

ds={ON|OFF}

DownloadOnly={ON|OFF}

Bemerkungen

Wenn eine reine Download-Synchronisation vorgenommen wird, führt dbmlsync keinen Upload von Datenbankänderungen durch. Allerdings wird ein Upload von Informationen über das Schema und das Offset für den Verarbeitungsfortschritt vorgenommen.

Außerdem sorgt dbmlsync dafür, dass Änderungen in der entfernten Datenbank während einer reinen Download-Synchronisation nicht überschrieben werden. Dazu wird das Log gescannt, um Zeilen zu ermitteln, für die Vorgänge auf einen Upload warten. Wenn eine dieser Zeilen durch den Download geändert würde, wird der Download zurückgesetzt und die Synchronisation schlägt fehl. Wenn die Synchronisation aus diesem Grund fehlschlägt, müssen Sie eine vollständige Synchronisation vornehmen, um das Problem zu beheben.

Wenn Sie mit entfernten Datenbanken arbeiten, die durch reine Download-Synchronisation synchronisiert werden, sollten Sie regelmäßig eine vollständige bidirektionale Synchronisation durchführen, um das Volumen des Logs zu verkleinern, das von der reinen Download-Synchronisation gescannt werden muss. Sonst würden die reinen Download-Synchronisationen zunehmend mehr Zeit brauchen.

Siehe auch

- „dbmlsync-Option -ds“ auf Seite 120
- „Erweiterte Option DownloadOnly (ds)“ auf Seite 153

Beispiel

Das folgende Beispiel zeigt die Verwendung der DownloadOnly-Synchronisationsprofiloption:

```
DownloadOnly=on
```

DownloadReadSize-Synchronisationsprofiloption

Bei neu startbaren Downloads legt diese Option die maximale Anzahl von Byte fest, die nach einem Kommunikationsfehler erneut gesendet werden müssen.

Syntax

drs=size

DownloadReadSize=size

Bemerkungen

Dbmlsync liest den Download in Abschnitten. DownloadReadSize (Download-Lesegröße) legt die Größe dieser Abschnitte fest. Wenn ein Kommunikationsfehler auftritt, verliert dbmlsync den gesamten bearbeiteten Abschnitt. Abhängig davon, wann der Fehler auftritt, liegt die Anzahl der verlorenen Byte zwischen 0 und der Größe des Downloadlesevorgangs (DownloadReadSize) -1. Beispiel: Wenn DownloadReadSize 100 Byte beträgt und nach dem Lesen von 497 Byte ein Fehler auftritt, gehen die letzten 97 gelesenen Byte verloren. Auf diese Weise verlorene Byte werden erneut gesendet, wenn der Download erneut gestartet wird.

Im Allgemeinen führt eine größere Download-Lesegröße zu einer besseren Performance bei erfolgreichen Synchronisationen, aber zu einer Übertragung von mehr Daten, wenn ein Fehler auftritt. Diese Option wird typischerweise verwendet, wenn die Standardgröße bei einer unzuverlässigen Verbindung verkleinert werden soll.

Standardwert ist "32767". Wenn der Wert dieser Option über 32767 liegt, wird der Wert 32767 verwendet.

Sie können auch die Download-Lesegröße mit der erweiterten Option DownloadReadSize festlegen.

Siehe auch

- [„dbmlsync-Option -drs“ auf Seite 120](#)
- [„Erweiterte Option DownloadReadSize \(drs\)“ auf Seite 154](#)

Beispiel

Das folgende Beispiel zeigt die Verwendung der DownloadReadSize-Synchronisationsprofiloption:

```
DownloadReadSize=100
```

ExtOpt-Synchronisationsprofiloption

Legt erweiterte Optionen fest.

Syntax

```
e={option=value; ...}
```

```
ExtOpt={option=value; ...}
```

Bemerkungen

Erweiterte Optionen können in ihrer Langform oder ihrer Kurzform angegeben werden. Siehe [„Erweiterte Optionen von MobiLink SQL Anywhere-Clients“ auf Seite 143](#).

Erweiterte Optionen werden paarweise in der Form Option=Wert angegeben. Die Liste der gesetzten erweiterten Optionen muss in geschweifte Klammern { } eingeschlossen sein.

Siehe auch

- [„dbmlsync-Option -e“ auf Seite 121](#)

Beispiel

Das folgende Beispiel zeigt die Verwendung der ExtOpt-Synchronisationsprofiloption:

```
ExtOpt={lt=exclusive;tableorder=t1,t2,t3}
```

IgnoreHookErrors-Synchronisationsprofiloption

Ignoriert alle Fehler in Hook-Funktionen, wenn auf On gesetzt.

Syntax

```
eh={ON|OFF}
```

```
IgnoreHookErrors={ON|OFF}
```

Siehe auch

- [„dbmlsync-Option -eh“ auf Seite 122](#)

Beispiel

Das folgende Beispiel zeigt die Verwendung der IgnoreHookErrors-Synchronisationsprofiloption:

```
IgnoreHookErrors=on
```

IgnoreScheduling-Synchronisationsprofiloption

Ignoriert die erweiterte Option für die Abfolgeplanung, damit die Synchronisation sofort erfolgt.

Syntax

```
is={ON|OFF}
```

```
IgnoreScheduling={ON|OFF}
```

Bemerkungen

Diese Option hat eine Kurz- und eine Langform: Sie können is oder IgnoreScheduling verwenden.

Siehe auch

- [„dbmlsync-Option -is“ auf Seite 123](#)
- [„Synchronisationszeitpläne“ auf Seite 100](#)

Beispiel

Das folgende Beispiel zeigt die Verwendung der IgnoreScheduling-Synchronisationsprofiloption:

```
IgnoreScheduling=on
```

KillConnections-Synchronisationsprofiloption

Löscht kollidierende Sperren in der entfernten Datenbank.

Syntax

d={ON|OFF}

KillConnections={ON|OFF}

Bemerkungen

In Fällen, in denen dbmlsync zu synchronisierende Tabellen sperren muss, die bereits von einer anderen Verbindung gesperrt werden, kann die Synchronisation fehlschlagen oder verzögert werden. Wenn Sie diese Option angeben, zwingen Sie SQL Anywhere, alle anderen Verbindungen zur entfernten Datenbank zu löschen, die Konfliktsperren beinhalten, damit die Synchronisation sofort fortgesetzt werden kann.

Siehe auch

- „dbmlsync-Option -d“ auf Seite 118
- „Parallelität während der Synchronisation“ auf Seite 94

Beispiel

Das folgende Beispiel zeigt die Verwendung der KillConnections-Synchronisationsprofiloption:

```
KillConnections=on
```

LogRenameSize-Synchronisationsprofiloption

Damit wird das Transaktionslog umbenannt und neu gestartet.

Syntax

x=size[K | M]

LogRenameSize=size[K | M]

Bemerkungen

Wenn diese Option eingestellt ist, wird das Transaktionslog während der Synchronisation umbenannt und neu gestartet, sobald es die angegebene Größe in Byte überschreitet. Stellen Sie ein "K" oder "M" dahinter, um die Einheit in kB oder MB anzugeben.

Setzen Sie die Größe auf 0, um das Transaktionslog unabhängig von seiner Größe umzubenennen.

Siehe auch

- „dbmlsync-Option -x“ auf Seite 143

Beispiel

Das folgende Beispiel zeigt die Verwendung der LogRenameSize-Synchronisationsprofiloption:

```
LogRenameSize=512k
```

MobiLinkPwd-Synchronisationsprofiloption

Gibt das Kennwort des MobiLink-Benutzers an.

Syntax

mp=*password*

MobiLinkPwd=*password*

Siehe auch

- „dbmlsync-Option -mp“ auf Seite 125
- „Erweiterte Option MobiLinkPwd (mp)“ auf Seite 162
- „Erweiterte Option NewMobiLinkPwd (mn)“ auf Seite 163
- „dbmlsync-Option -mn“ auf Seite 124

Beispiel

Das folgende Beispiel zeigt die Verwendung der MobiLinkPwd-Synchronisationsprofiloption:

```
MobiLinkPwd=mypassword
```

MLUser-Synchronisationsprofiloption (nicht mehr empfohlen)

Gibt den MobiLink-Benutzernamen an.

Syntax

u=*username*

MLUser=*username*

Bemerkungen

Diese Option wird nicht mehr empfohlen. Verwenden Sie stattdessen die Subscription-Synchronisationsprofiloption.

Siehe auch

- „Subscription-Synchronisationsprofiloption“ auf Seite 198
- „MobiLink-Benutzer“ auf Seite 4
- „dbmlsync-Option -u (nicht mehr empfohlen)“ auf Seite 138

Beispiel

Das folgende Beispiel zeigt die Verwendung der MLUser-Synchronisationsprofiloption:

```
MLUser=my_user_name
```

NewMobiLinkPwd-Synchronisationsprofiloption

Übergibt ein neues Kennwort für den MobiLink-Benutzer. Benutzen Sie diese Option, wenn Sie das bestehende Kennwort ändern wollen.

Syntax

`mn=new_password`

`NewMobiLinkPwd=new_password`

Siehe auch

- „dbmlsync-Option -mp“ auf Seite 125
- „dbmlsync-Option -mn“ auf Seite 124
- „MobiLink-Benutzer“ auf Seite 4

Beispiel

Das folgende Beispiel zeigt die Verwendung der NewMobiLinkPwd-Synchronisationsprofiloption:

```
NewMobiLinkPwd=new_password
```

Ping-Synchronisationsprofiloption

Pingt einen MobiLink-Server

Syntax

`pi={ON|OFF}`

`Ping={ON|OFF}`

Bemerkungen

Wenn Sie die Ping-Synchronisationsprofiloption verwenden, verbindet sich dbmlsync mit der entfernten Datenbank, ruft die erforderlichen Informationen ab, um sich mit dem MobiLink-Server zu verbinden, verbindet sich mit dem Server und authentifiziert den angegebenen MobiLink-Benutzer.

Wenn der MobiLink-Server eine ping-Anfrage erhält, verbindet er sich mit der konsolidierten Datenbank, authentifiziert den Benutzer und sendet dann den Benutzerauthentifizierungs-Status und -Wert an den Client zurück. Wenn der MobiLink-Benutzername nicht in der Systemtabelle ml_user gefunden werden kann und der MobiLink-Server mit der Befehlszeilenoption -zu+ ausgeführt wird, fügt er den Benutzer der MobiLink-Systemtabelle ml_user hinzu.

Um die Verbindung adäquat zu testen, sollten Sie die Ping-Synchronisationsprofiloption mit allen Synchronisationsoptionen nutzen, die Sie für die Synchronisation mit dbmlsync verwenden möchten. Wenn die Ping-Synchronisationsprofiloption enthalten ist, führt dbmlsync keine Synchronisation durch.

Wird der ping-Parameter erfolgreich ausgeführt, gibt der MobiLink-Server eine Informationsmeldung aus. Wird der ping-Parameter erfolglos ausgeführt, gibt der Server eine Fehlermeldung aus.

Siehe auch

- [„dbmlsync-Option -pi“ auf Seite 129](#)

Beispiel

Das folgende Beispiel zeigt die Verwendung der Ping-Synchronisationsprofiloption:

```
Ping=on
```

Publication-Synchronisationsprofiloption (nicht mehr empfohlen)

Gibt die zu synchronisierenden Publikationen an.

Syntax

```
n=pubname, ...
```

```
Publication=pubname, ...
```

Bemerkungen

Die Publication-Synchronisationsprofiloption kann nur einmal in einem Synchronisationsprofil angegeben werden.

Hinweis

Diese Option ist nicht mehr empfohlen. Es wird empfohlen, dass Sie entweder die Subscription-Synchronisationsprofiloption oder stattdessen den dbmlsync-Parameter -s verwenden. Siehe [„Subscription-Synchronisationsprofiloption“ auf Seite 198](#) und [„dbmlsync-Option -s“ auf Seite 133](#).

Um die dbmlsync-Option -s zu verwenden, sollten Sie den Namen der Subskription, die Sie synchronisieren wollen, überprüfen. Sie können den Subskriptionsnamen mit der folgenden Abfrage ermitteln:

```
SELECT subscription_name  
FROM syssync JOIN sys.syspublication  
WHERE site_name = <ml_user> AND publication_name = <pub_name>;
```

Ersetzen Sie <ml_user> durch den MobiLink-Benutzer, den Sie synchronisieren. Hierbei handelt es sich um den Wert, der durch die Option -u der dbmlsync-Befehlszeile festgelegt wird. Siehe [„dbmlsync-Option -u \(nicht mehr empfohlen\)“ auf Seite 138](#).

Ersetzen Sie <pub_name> durch den Namen der zu synchronisierenden Publikation. Hierbei handelt es sich um den Wert, der mit der Option -n in der dbmlsync-Befehlszeile angegeben wird. Siehe [„dbmlsync-Option -n \(nicht mehr empfohlen\)“ auf Seite 125](#).

Siehe auch

- [„dbmlsync-Option -n \(nicht mehr empfohlen\)“ auf Seite 125](#)

Beispiel

```
Publication=overnight
```

RemoteProgressGreater-Synchronisationsprofiloption

Legt fest, dass der Offset der entfernten Datenbank verwendet werden soll, wenn er höher ist als der der konsolidierten Datenbank. Dies ist das Äquivalent zur dbmlsync-Option -ra.

Syntax

ra={ON|OFF}

RemoteProgressGreater={ON|OFF}

Bemerkungen

Diese Option sollte nur in seltenen Fällen verwendet werden. Wenn Sie diese Option verwenden, wird der Upload ab dem Offset nochmals versucht, der aus der entfernten Datenbank bezogen wurde, wenn der Offset der entfernten Datenbank größer ist als der aus der konsolidierten Datenbank. Wenn Sie diese Option verwenden und der Offset in der entfernten Datenbank nicht größer ist als der Offset aus der konsolidierten Datenbank, wird ein Fehler gemeldet und die Synchronisation wird abgebrochen.

Diese Option muss mit Umsicht eingesetzt werden. Wenn die Offsets aufgrund einer Wiederherstellung der konsolidierten Datenbank nicht übereinstimmen, gehen die Änderungen verloren, die in der entfernten Datenbank in der Lücke zwischen den beiden erfassten Offsets vorgenommen wurden. Diese Option kann sinnvoll sein, wenn die konsolidierte Datenbank aus einer Sicherung wiederhergestellt wurde und das Transaktionslog der entfernten Datenbank an derselben Stelle gekürzt wurde wie der Offset der entfernten Datenbank. In diesem Fall gehen alle Daten, für die ein Upload aus der entfernten Datenbank bereits erfolgt ist, ab dem Punkt in der konsolidierten Datenbank bis zu dem Punkt des Offsets der entfernten Datenbank verloren.

Siehe auch

- „dbmlsync-Option -r“ auf Seite 132

Beispiel

`RemoteProgressGreater=on`

RemoteProgressLess-Synchronisationsprofiloption

Legt fest, dass der Offset der entfernten Datenbank benutzt werden soll, wenn er niedriger ist als der der konsolidierten Datenbank (z.B. wenn die entfernte Datenbank aus einer Sicherung wiederhergestellt wurde). Dies ist das Äquivalent zur dbmlsync-Option -rb.

Syntax

rb={ON|OFF}

RemoteProgressLess={ON|OFF}

Bemerkungen

Wenn die entfernte Datenbank aus einer Sicherung wiederhergestellt wird, kann das Standardverhalten zu Datenverlust führen. Wenn Sie diese Option verwenden, wird der Upload ab dem Offset fortgesetzt, der in der entfernten Datenbank erfasst wurde, sofern dieser Offset kleiner ist als derjenige, der aus der

konsolidierten Datenbank abgerufen wurde. Wenn Sie diese Option verwenden und der Offset in der entfernten Datenbank nicht kleiner ist als der Offset aus der konsolidierten Datenbank, wird ein Fehler gemeldet und die Synchronisation wird abgebrochen.

Diese Option kann dazu führen, dass ein Upload von Daten erfolgt, die bereits versendet wurden. Daraus können sich Konflikte in der konsolidierten Datenbank ergeben, die mit entsprechenden Skripten zur Konfliktlösung verarbeitet werden sollten.

Siehe auch

- „dbmlsync-Option -r“ auf Seite 132

Beispiel

```
RemoteProgressLess=on
```

Subscription-Synchronisationsprofiloption

Gibt die zu synchronisierende(n) Subskription(en) an.

Syntax

```
s=pubname, ...
```

```
Subscription=subname, ...
```

Bemerkungen

Eine Subskription kann nur einmal in einem Synchronisationsprofil angegeben werden. Der Befehlszeilenparameter kann hingegen mehrfach angegeben werden.

Siehe auch

- „dbmlsync-Option -s“ auf Seite 133

Beispiel

```
Subscription=mySubscription
```

TransactionalUpload-Synchronisationsprofiloption

Boolescher Wert. Legt fest, dass jede Transaktion in der entfernten Datenbank als eigene Transaktion in einer eigenen Synchronisation ausgelesen werden soll.

Syntax

```
tu={ON|OFF}
```

```
TransactionalUpload={ON|OFF}
```

Bemerkungen

Wenn Sie die TransactionalUpload-Synchronisationsprofiloption verwenden, erstellen Sie einen transaktionalen Upload: dbmlsync liest jede Transaktion in die entfernte Datenbank als einzelne

Transaktion ein. Im MobiLink-Server werden alle Transaktionen beim Empfang einzeln angewendet und festgeschrieben.

Siehe auch

- „dbmlsync-Option -tu“ auf Seite 136

Beispiel

```
TransactionalUpload=on
```

UpdateGenNum-Synchronisationsprofiloption

Beim Erstellen einer Download-Datei legt diese Option eine Datei an, die mit noch nicht synchronisierten entfernten Datenbanken verwendet werden kann. Sonst müssen Sie eine Synchronisation vornehmen, bevor Sie eine Download-Datei übernehmen.

Syntax

```
bg={ON|OFF}
```

```
UpdateGenNum={ON|OFF}
```

Bemerkungen

Diese Option sorgt dafür, dass die Download-Datei die Generationsnummern in der entfernten Datenbank aktualisiert.

Download-Dateien, die mit dieser Option erstellt wurden, sollten Snapshot-Downloads sein. Zeitstempelbasierte Downloads funktionieren nicht mit entfernten Datenbanken, die nicht synchronisiert wurden. Dies liegt daran, dass der Zeitstempel des letzten Downloads bei einer neuen entfernten Datenbank standardmäßig auf den 1. Januar 1900 gesetzt wird und damit vor dem Zeitstempel des letzten Downloads in der Download-Datei liegt. Für zeitstempelbasierte, dateibasierte Downloads muss der Zeitstempel des letzten Downloads in der Download-Datei gleich oder früher sein als in der entfernten Datenbank.

Übernehmen Sie keine mit UpdateGenNum erstellten Download-Dateien in entfernte Datenbanken, die bereits synchronisiert wurden, wenn Ihr System von Funktionalitäten abhängt, die durch Generierungsnummern gesteuert werden, da diese Option diese Funktionalität umgeht.

Siehe auch

- „dbmlsync-Option -bg“ auf Seite 114
- „MobiLink-Generierungsnummern“ [*MobiLink - Serveradministration*]
- „Synchronisation neuer entfernter Datenbanken“ [*MobiLink - Serveradministration*]

Beispiel

```
UpdateGenNum=on
```

UploadOnly-Synchronisationsprofiloption

Gibt an, dass die Synchronisation nur einen Upload umfasst und kein Download ausgeführt wird.

Syntax

`uo={ON|OFF}`

`UploadOnly={ON|OFF}`

Bemerkungen

Während einer reinen Upload-Synchronisation bereitet dbmlsync einen Upload für den MobiLink-Synchronisationsserver vor und versendet ihn genau so wie bei einer vollständigen Synchronisation. Anstatt aber einen Download zurückzusenden, sendet MobiLink nur eine Bestätigung, in der festgestellt wird, ob der Upload erfolgreich festgeschrieben wurde.

Eine Liste aller Skripten, die für reine Upload-Synchronisationen definiert werden müssen, finden Sie unter „Erforderliche Skripten“ [[MobiLink - Serveradministration](#)].

Siehe auch

- „dbmlsync-Option -uo“ auf Seite 139
- „Erweiterte Option UploadOnly (uo)“ auf Seite 173

Beispiel

`UploadOnly=on`

UploadRowCnt-Synchronisationsprofiloption

Ganzzahl. Gibt eine Schätzung der Anzahl der Zeilen an, für die bei einer Synchronisation ein Upload durchgeführt werden soll.

Syntax

`urc=rowcount`

`UploadRowCnt=rowcount`

Bemerkungen

Um die Performance zu steigern, können Sie eine Schätzung der Zeilenzahl angeben, die bei einer Synchronisation hochgeladen werden sollen. Diese Einstellung ist vor allem sinnvoll, wenn Sie viele Zeilen aktualisieren. Eine höhere Schätzung führt zu schnelleren Uploads, aber auch zu mehr Speicherbedarf.

Die Synchronisation wird einwandfrei ausgeführt, unabhängig von der Schätzung.

Siehe auch

- „dbmlsync-Option -urc“ auf Seite 140

Beispiel

UploadRowCnt=100

Verbosity-Synchronisationsprofiloption

Steuert die Ausführlichkeitsstufe von dbmlsync.

Syntax

v={BASIC|HIGH|CONNECT_STR|ROW_CNT|OPTIONS|ML_PASSWORD|ROW_DATA|HOOK}, ...

Verbosity={BASIC|HIGH|CONNECT_STR|ROW_CNT|OPTIONS|ML_PASSWORD|ROW_DATA|HOOK}, ...

Zulässige Werte

Der Wert muss eine durch Kommas getrennte Liste mit einer oder mehreren der folgenden Optionen sein, von denen jede einen anderen Typ der Ausführlichkeit ermöglicht:

- **BASIC** Begrenzte Ausführlichkeitsstufe
- **HIGH** Maximale Ausführlichkeitsstufe
- **CONNECT_STR** Die Verbindungszeichenfolge im Log aufzeichnen
- **ROW_CNT** Die Anzahl der Zeilen aus dem Upload bzw. Download protokollieren
- **OPTIONS** Die Optionen zur Angabe einer Synchronisation protokollieren
- **ML_PASSWORD** Das MobiLink-Kennwort im Log aufzeichnen
- **ROW_DATA** Zeilen aus dem Upload bzw. Download protokollieren
- **HOOK** Hook-Skript-bezogene Meldungen protokollieren

Bemerkungen

Wenn Sie eine erweiterte Ausführlichkeitsoption und eine Ausführlichkeits-Synchronisationsprofiloption angeben und Konflikte vorliegen, hebt die Ausführlichkeits-Synchronisationsprofiloption die erweiterte Ausführlichkeitsoption auf.

Wenn Sie sowohl -v als auch eine erweiterte Verbosity-Option angeben und Konflikte vorliegen, hebt die Option -v die erweiterte Option auf. Wenn kein Konflikt vorliegt, werden alle von Ihnen angegebenen Ausführlichkeitsoptionen verwendet. Wenn Sie die Befehlszeilenoption -v verwenden, werden die Ausführlichkeitsoptionen sofort wirksam. Wenn Sie die erweiterte Option verwenden, treten die Ausführlichkeitsoptionen erst dann in Kraft, wenn die erste Synchronisation beginnt, sodass Startinformationen nicht protokolliert werden. Nach der ersten Synchronisation sollte das Verhalten dasselbe sein, unabhängig davon, wie die Option angegeben wurde.

Siehe auch

- „dbmlsync-Option -v“ auf Seite 141

Beispiel

`Verbosity=OPTIONS, ML_PASSWORD`

Ereignis-Hooks für SQL Anywhere-Clients

Der SQL Anywhere-Synchronisationsclient dbmlsync bietet eine Reihe von Ereignis-Hooks, mit denen Sie den Synchronisationsprozess anpassen können. Wenn ein Hook implementiert wurde, wird er zu einem bestimmten Zeitpunkt während des Synchronisationsprozesses aufgerufen.

Sie implementieren einen Ereignis-Hook, indem Sie eine gespeicherte SQL-Prozedur mit einem bestimmten Namen erstellen. Die meisten gespeicherten Prozeduren eines Ereignis-Hooks werden auf derselben Verbindung wie die Synchronisation selbst ausgeführt.

Mit Ereignis-Hooks können Sie Synchronisationsereignisse protokollieren und verarbeiten. Sie können beispielsweise Synchronisationen basierend auf logischen Ereignissen planen, fehlgeschlagene Verbindungen erneut aufbauen oder Fehler und Verletzungen der referenziellen Integrität verarbeiten.

Außerdem können Sie die Ereignis-Hooks auch verwenden, um Teilmengen von Daten zu synchronisieren, die in einer Publikation nicht problemlos definiert werden können. Zum Beispiel können Sie Daten in einer temporären Tabelle synchronisieren, indem Sie eine Hook-Prozedur schreiben, die vor der Synchronisation Daten aus der temporären Tabelle in eine permanente Tabelle kopiert, und eine andere, die die Daten danach wieder zurückkopiert.

Vorsicht

Die Integrität des Synchronisationsprozesses basiert auf einer Folge von integrierten Transaktionen. Sie dürfen innerhalb Ihrer Hook-Prozeduren Transaktionen nicht implizit oder explizit festschreiben bzw. zurücksetzen.

Ändern Sie eine beliebige Verbindungseinstellung in einem Hook, müssen Sie den vorherigen Wert der Einstellung wiederherstellen, bevor der Hook endet. Andernfalls kann es zu unerwarteten Ergebnissen kommen.

Privilegien

Hook-Prozeduren können von jedem Benutzer mit dem MANAGE REPLICATION-Systemprivileg erstellt werden. Um zu gewährleisten, dass ein Hook auf die #hook_dict-Tabelle zugreifen kann, die verwendet wird, um Daten in und aus Hooks zu übertragen, müssen Hooks eine der folgenden Anforderungen erfüllen:

- Eigentümer muss ein Benutzer sein, der die Systemprivilegien SELECT ANY TABLE und UPDATE ANY TABLE hat.
- Sie müssen durch die SQL SECURITY INVOKER-Klausel der CREATE PROCEDURE-Anweisung definiert sein.

dbmlsync-Schnittstellen

Die Client-Ereignis-Hooks können mit dem Befehlszeilendienstprogramm dbmlsync oder jeder anderen Programmierschnittstelle zur Synchronisation von SQL Anywhere-Clients, z.B. der Dbmlsync-API und der DBTools-Schnittstelle für dbmlsync, verwendet werden.

Siehe „Anpassen der dbmlsync-Synchronisation“ auf Seite 102.

Hook-Sequenz bei der Synchronisation

Der folgende Pseudocode zeigt die verfügbaren Ereignisse und veranschaulicht den Punkt, an dem jedes einzelne Ereignis im Synchronisationsprozess aufgerufen wird. Zum Beispiel ist `sp_hook_dbmlsync_abort` der zuerst zu aktivierende Ereignis-Hook.

```

sp_hook_dbmlsync_abort //not called when Dbmlsync API or the SQL SYNCHRONIZE
STATEMENT is used
sp_hook_dbmlsync_set_extended_options
loop until return codes direct otherwise (
    sp_hook_dbmlsync_abort
    sp_hook_dbmlsync_delay
)
sp_hook_dbmlsync_abort
// start synchronization
sp_hook_dbmlsync_begin
// upload events
for each upload segment
    // a normal synchronization has one upload segment
    // a transactional upload has one segment per transaction
    // an incremental upload has one segment per upload piece
    sp_hook_dbmlsync_logscan_begin //not called for scripted upload
    sp_hook_dbmlsync_logscan_end //not called for scripted upload
    sp_hook_dbmlsync_set_ml_connect_info //only called during first upload
    sp_hook_dbmlsync_upload_begin
    sp_hook_dbmlsync_set_upload_end_progress //only called for scripted upload
    sp_hook_dbmlsync_upload_end
next upload segment
// download events
sp_hook_dbmlsync_validate_download_file (only called
    when -ba option is used)
sp_hook_dbmlsync_download_begin
for each table
    sp_hook_dbmlsync_download_table_begin
    sp_hook_dbmlsync_download_table_end
next table
sp_hook_dbmlsync_download_end

sp_hook_dbmlsync_schema_upgrade
// end synchronization
sp_hook_dbmlsync_end
sp_hook_dbmlsync_process_exit_code
sp_hook_dbmlsync_log_rescan

```

Ereignis-Hooks

Jeder Hook ist mit Parameterwerten versehen, die Sie beim Implementieren der Prozedur verwenden können. In einigen Fällen können Sie den Wert ändern, damit ein neuer Wert zurückgegeben wird. Andere Parameter sind schreibgeschützt. Diese Parameter sind keine Argumente gespeicherter

Prozeduren. An die gespeicherten Prozeduren des Hooks werden keine Argumente übergeben. Stattdessen werden die Argumente durch Lesen und Ändern von Zeilen in der Tabelle #hook_dict ausgetauscht.

Beispiel: Die Prozedur sp_hook_dbmlsync_begin hat einen Parameter für den MobiLink-Benutzer, also den MobiLink-Benutzer, der synchronisiert wird. Sie können diesen Wert aus der Tabelle #hook_dict abrufen.

Wenngleich die Sequenz Ähnlichkeiten mit der Hook-Sequenz auf dem MobiLink-Server hat, gibt es nur wenige Überschneidungen in der Art der Logik, die Sie den konsolidierten und entfernten Datenbanken hinzufügen. Die beiden Schnittstellen sind daher getrennt und unterschiedlich.

Wenn ein *_begin-Hook erfolgreich verarbeitet ist, wird der entsprechende *_end-Hook unbeschadet von Fehlern aufgerufen, die nach dem *_begin-Hook auftreten. Wenn der *_begin-Hook nicht definiert ist, Sie aber einen *_end-Hook definiert haben, wird *_end-Hook aufgerufen, sofern nicht ein Fehler vor dem Zeitpunkt auftritt, an dem der *_begin-Hook normalerweise aufgerufen würde.

Wenn der Hook Daten in Ihrer Datenbank ändert, werden alle Änderungen bis einschließlich sp_hook_dbmlsync_logscan_begin in der laufenden Synchronisationssitzung synchronisiert. Danach werden Änderungen in der nächsten Sitzung synchronisiert.

Hook-Prozeduren

Dieser Abschnitt enthält einige Hinweise zur Planung und Verwendung von Hook-Prozeduren.

Hinweise

- Führen Sie in Hook-Prozeduren keine COMMIT- oder ROLLBACK-Anweisungen aus. Die Prozeduren werden auf derselben Verbindung wie die Synchronisation selbst ausgeführt und eine COMMIT- oder ROLLBACK-Anweisung kann die Synchronisation beeinträchtigen.
- Ändern Sie eine beliebige Verbindungseinstellung in einem Hook, müssen Sie den vorherigen Wert der Einstellung wiederherstellen, bevor der Hook endet. Andernfalls kann es zu unerwarteten Ergebnissen kommen.
- Dbmlsync ruft die gespeicherte Prozeduren ohne Angabe des Eigentümers auf. Eigentümer der gespeicherten Prozeduren muss daher entweder der Benutzername sein, der bei der dbmlsync-Verbindung verwendet wird, oder eine Gruppe, deren Mitglied der Benutzer ist.
- Hook-Prozeduren können von jedem Benutzer mit dem MANAGE REPLICATION-Systemprivileg erstellt werden. Um jedoch zu gewährleisten, dass ein Hook auf die #hook_dict-Tabelle zugreifen kann, die verwendet wird, um Daten an und aus Hooks zu übergeben, müssen Hooks eine der folgenden Anforderungen erfüllen:
 - Eigentümer muss ein Benutzer sein, der die Systemprivilegien SELECT ANY TABLE und UPDATE ANY TABLE hat.
 - Sie müssen durch die SQL SECURITY INVOKER-Klausel der CREATE PROCEDURE-Anweisung definiert sein.

#hook_dict-Tabelle

Unmittelbar bevor ein Ereignis-Hook aufgerufen wird, erstellt dbmlsync die #hook_dict-Tabelle in der entfernten Datenbank mit der folgenden CREATE-Anweisung. Das # vor dem Tabellennamen bedeutet, dass die Tabelle eine temporäre Tabelle ist.

```
CREATE TABLE #hook_dict(
  name VARCHAR(128) NOT NULL UNIQUE,
  value VARCHAR(10240) NOT NULL)
```

Das Dienstprogramm dbmlsync verwendet die Tabelle #hook_dict, um Werte an Hook-Funktionen weiterzugeben. Hook-Funktionen verwenden die Tabelle #hook_dict, um Werte an dbmlsync zurückzugeben.

Jede Hook-Prozedur empfängt Parameterwerte. In einigen Fällen können Sie den Wert ändern, damit ein neuer Wert zurückgegeben wird. Andere Parameter sind schreibgeschützt. Jede einzelne Zeile in der Tabelle enthält den Wert für einen Parameter.

Zwei Subskriptionen können z.B. wie folgt definiert sein:

```
CREATE SYNCHRONIZATION SUBSCRIPTION sub1
TO pub1
FOR MyUser;
SCRIPT VERSION 'v1'

CREATE SYNCHRONIZATION SUBSCRIPTION sub2
TO pub2
FOR MyUser;
SCRIPT VERSION 'v1'
```

Wenn sp_hook_dbmlsync_begin für die folgenden Befehlszeile von dbmlsync aufgerufen wird:

```
dbmlsync -c 'DSN=MyDsn' -s sub1,sub2
```

enthält die #hook_dict-Tabelle die folgenden Zeilen:

Name	Wert
subscription_0	sub1
subscription_1	sub2
publication_0	pub1
publication_1	pub2
MobiLink user	MyUser
Script version	v1

Hinweis

Die publication_n-Zeilen werden nicht mehr empfohlen und in einer zukünftigen Version möglicherweise entfernt.

Ein Hook kann Werte aus der Tabelle #hook_dict abrufen und sie benutzen, um sein Verhalten anzupassen. Um beispielsweise den MobiLink-Benutzer abzurufen, verwenden Sie eine SELECT-Anweisung der folgenden Art:

```
SELECT value
FROM #hook_dict
WHERE name = 'MobiLink user'
```

I/O-Parameter können von Ihrem Hook aktualisiert werden, um das Verhalten von dbmlsync zu ändern. Im sp_hook_dbmlsync_abort-Hook können Sie beispielsweise dbmlsync anweisen, die Synchronisation abubrechen, indem die Synchronisationszeile 'abort' der Tabelle mit einer Anweisung der folgenden Art aktualisieren:

```
UPDATE #hook_dict
SET value='true'
WHERE name='abort synchronization'
```

Die Beschreibung der einzelnen Hooks listet die Zeilen in der Tabelle #hook_dict auf.

Privilegien

Hook-Prozeduren können von jedem Benutzer mit dem MANAGE REPLICATION-Systemprivileg erstellt werden. Um jedoch zu gewährleisten, dass ein Hook auf die #hook_dict-Tabelle zugreifen kann, die verwendet wird, um Daten an und aus Hooks zu übergeben, müssen Hooks eine der folgenden Anforderungen erfüllen:

- Eigentümer muss ein Benutzer sein, der die Systemprivilegien SELECT ANY TABLE und UPDATE ANY TABLE hat.
- Sie müssen durch die SQL SECURITY INVOKER-Klausel der CREATE PROCEDURE-Anweisung definiert sein.

Beispiele

Folgendes Beispiel einer sp_hook_dbmlsync_delay-Prozedur veranschaulicht die Verwendung von I/O-Parametern in der #hook_dict-Tabelle. Die Prozedur gestattet die Synchronisation nur außerhalb einer geplanten Auszeit des MobiLink-Systems zwischen 18:00 Uhr und 19:00 Uhr.

```
CREATE PROCEDURE sp_hook_dbmlsync_delay()
BEGIN
    DECLARE delay_val integer;
    SET delay_val=DATEDIFF(
        second, CURRENT TIME, '19:00');
    IF (delay_val>0 AND
        delay_val<3600)
    THEN
        UPDATE #hook_dict SET value=delay_val
            WHERE name='delay duration';
    END IF;
END
```

Die folgende Prozedur wird am Anfang der Synchronisation in der entfernten Datenbank ausgeführt. Sie ruft den aktuellen MobiLink-Benutzernamen auf (einen der Parameter, die für das Ereignis sp_hook_dbmlsync_begin verfügbar sind) und zeigt ihn im SQL Anywhere-Meldungsfenster an.

```
CREATE PROCEDURE sp_hook_dbmlsync_begin()
BEGIN
```

```
DECLARE MLuser VARCHAR(150);
SELECT '>>>MLuser = ' || value INTO MLuser
FROM #hook_dict
WHERE name = 'MobiLink user';
MESSAGE MLuser TYPE INFO TO CONSOLE;
END
```

Verbindungen für Hook-Prozeduren

Jede Hook-Prozedur wird auf derselben Verbindung wie die Synchronisation selbst ausgeführt. Dabei gelten folgende Ausnahmen:

- sp_hook_dbmlsync_all_error
- sp_hook_dbmlsync_communication_error
- sp_hook_dbmlsync_download_log_ri_violation
- sp_hook_dbmlsync_misc_error
- sp_hook_dbmlsync_sql_error

Diese Prozeduren werden aufgerufen, bevor eine Synchronisation fehlschlägt. Bei einem Fehler werden die Synchronisationsvorgänge zurückgesetzt. Wenn mit einer separaten Verbindung gearbeitet wird, können Sie diese Prozeduren verwenden, um Informationen über den Fehler zu protokollieren, ohne die Protokollierungsaktionen zusammen mit den Synchronisationsvorgängen zurückzusetzen.

Behandlung von Fehlern und Warnungen in Ereignis-Hook-Prozeduren

Sie können gespeicherte Ereignis-Hook-Prozeduren erstellen, um Synchronisationsfehler, MobiLink-Verbindungsfehler und Verletzungen der referenziellen Integrität zu behandeln. In diesem Abschnitt werden Hook-Prozeduren beschrieben, die zur Behandlung von Fehlern und Warnungen verwendet werden. Nach der Implementierung werden die einzelnen Prozeduren automatisch ausgeführt, sobald ein Fehler des betreffenden Typs auftritt.

Umgang mit RI-Verletzungen

Eine Verletzung der referenziellen Integrität tritt ein, wenn Zeilen im Download die Fremdschlüsselbeziehungen in der entfernten Datenbank verletzen. Mit den folgenden Ereignis-Hooks können Sie Verletzungen der referenziellen Integrität protokollieren und verarbeiten:

- „sp_hook_dbmlsync_download_log_ri_violation“
- „sp_hook_dbmlsync_download_ri_violation“

Umgang mit MobiLink-Verbindungsfehlern

Mit dem Ereignis-Hook sp_hook_dbmlsync_ml_connect_failed können Sie nach einer gescheiterten Verbindungsaufnahme mit dem MobiLink-Server einen neuen Versuch starten, indem Sie einen anderen Verbindungstyp oder eine andere Adresse benutzen. Wenn die Verbindung schließlich fehlschlägt, ruft dbmlsync die Hooks sp_hook_dbmlsync_communication_error und sp_hook_dbmlsync_all_error auf.

Siehe [„sp_hook_dbmlsync_ml_connect_failed“](#) auf Seite 244.

Umgang mit dbmlsync-Fehlern

Jedes Mal, wenn eine dbmlsync-Fehlermeldung generiert wird, werden die folgenden Hooks aufgerufen:

- Zunächst wird je nach Fehlertyp einer der folgenden Hooks aufgerufen:
sp_hook_dbmlsync_communication_error, sp_hook_dbmlsync_misc_error oder
sp_hook_dbmlsync_sql_error. Die Hooks enthalten Informationen, die sich auf den speziellen
Fehlertyp beziehen. 'Sqlcode' und 'sqlstate' beispielsweise weisen auf SQL-Fehler hin.
- Dann wird sp_hook_dbmlsync_all_error aufgerufen. Dieser Hook kann zur Protokollierung aller
auftretenden Fehler verwendet werden.

Siehe:

- [„sp_hook_dbmlsync_communication_error“](#)
- [„sp_hook_dbmlsync_sql_error“](#)
- [„sp_hook_dbmlsync_misc_error“](#)
- [„sp_hook_dbmlsync_all_error“](#)

Um als Reaktion auf einen Fehler eine Synchronisation neu zu starten, können Sie den Parameter für den
Benutzerzustand in sp_hook_dbmlsync_end verwenden.

Siehe [„sp_hook_dbmlsync_end“](#) auf Seite 232.

Fehler ignorieren

Standardmäßig stoppt die Synchronisation, wenn in einer Hook-Prozedur ein nicht behandelter Fehler
festgestellt wird. Sie können das Dienstprogramm dbmlsync anweisen, diese Fehler zu ignorieren, indem
Sie die Option -eh verwenden.

Siehe [„Erweiterte Option IgnoreHookErrors \(eh\)“](#) auf Seite 158.

sp_hook_dbmlsync_abort

Verwenden Sie diese gespeicherte Prozedur, um den Synchronisationsprozess abubrechen.

Zeilen in der Tabelle #hook_dict

Name	Wert	Beschreibung
abort synchronization (in out)	true false	Wenn Sie diese Zeile in der Tabelle #hook_dict auf true einstellen, wird die Syn- chronisation sofort nach dem Ereignis been- det.

Name	Wert	Beschreibung
publication_ <i>n</i> (in)	Publikation	Nicht mehr empfohlen. Verwenden Sie stattdessen subscription_ <i>n</i> . Die synchronisierten Publikationen, wobei <i>n</i> eine Ganzzahl ist. Es gibt einen publication_ <i>n</i> -Eintrag für jede synchronisierte Publikation. Die Nummerierung von <i>n</i> beginnt bei Null.
MobiLink user (in)	MobiLink-Benutzername	Der MobiLink-Benutzer, für den Sie synchronisieren
exit code (in out)	Zahl	Wenn 'abort synchronization' auf TRUE gesetzt ist, können Sie diesen Wert verwenden, um den Beendigungscode für die abgebrochene Synchronisation zu definieren. 0 gibt an, dass die Synchronisation erfolgreich war. Jede andere Zahl gibt an, dass die Synchronisation fehlgeschlagen ist.
script version (in out)	Skriptversionsname	Die MobiLink-Skriptversion, die für die Synchronisation verwendet werden soll
subscription_ <i>n</i> (in)	Subskriptionsname(<i>n</i>)	Die synchronisierten Subskriptionen, wobei <i>n</i> eine Ganzzahl ist. Es gibt einen subscription_ <i>n</i> -Eintrag für jede synchronisierte Subskription. Die Nummerierung von <i>n</i> beginnt bei Null.

Bemerkungen

Wenn eine Prozedur dieses Namens vorhanden ist, wird sie beim dbmlsync-Start und dann nach jeder Synchronisationsverzögerung aufgerufen, die von sp_hook_dbmlsync_delay verursacht wird.

Wenn dbmlsync von der Befehlszeile aus ausgeführt wird, bewirkt die Einstellung von TRUE für das Abbrechen der Synchronisation, dass alle restlichen Synchronisationen (einschließlich der geplanten Synchronisationen) abgebrochen werden. Wenn die dbmlsync-API oder die SQL SYNCHRONIZE-Anweisung verwendet wird, bewirkt die Einstellung von TRUE für das Abbrechen der Synchronisation, dass nur die aktuelle Synchronisation abgebrochen wird.

Die Aktionen dieser Prozedur werden sofort nach ihrer Ausführung festgeschrieben.

Privilegien

Hook-Prozeduren können von jedem Benutzer mit dem MANAGE REPLICATION-Systemprivileg erstellt werden. Um jedoch zu gewährleisten, dass ein Hook auf die #hook_dict-Tabelle zugreifen kann, die verwendet wird, um Daten an und aus Hooks zu übergeben, müssen Hooks eine der folgenden Anforderungen erfüllen:

- Eigentümer muss ein Benutzer sein, der die Systemprivilegien SELECT ANY TABLE und UPDATE ANY TABLE hat.
- Sie müssen durch die SQL SECURITY INVOKER-Klausel der CREATE PROCEDURE-Anweisung definiert sein.

Siehe auch

- „Hook-Sequenz bei der Synchronisation“ auf Seite 203
- „sp_hook_dbmlsync_process_exit_code“ auf Seite 247

Beispiele

Die folgende Prozedur verhindert die Synchronisation während der planmäßig täglich zwischen 19:00 und 20:00 Uhr durchgeführten Wartung.

```
CREATE PROCEDURE sp_hook_dbmlsync_abort()  
BEGIN  
    DECLARE down_time_start TIME;  
    DECLARE is_down_time VARCHAR(128);  
    SET down_time_start='19:00';  
    IF datediff( hour,down_time_start,now(*) ) < 1  
    THEN  
        set is_down_time='true';  
    ELSE  
        SET is_down_time='false';  
    END IF;  
    UPDATE #hook_dict  
    SET value = is_down_time  
    WHERE name = 'abort synchronization'  
END;
```

Sie haben einen abort-Hook, der die Synchronisation aus einem von zwei Gründen abbrechen kann. Einer der Gründe ist der normale Abschluss der Synchronisation, und Sie wollen, dass dbmlsync den Beendigungscode 0 hat. Der andere Grund gibt eine Fehlerursache an, und daher soll dbmlsync einen von Null verschiedenen Code ausgeben. Sie könnten dies mit einem sp_hook_dbmlsync_abort-Hook erreichen, der wie folgt definiert ist.

```
BEGIN  
    IF [condition that defines the normal abort case] THEN  
        UPDATE #hook_dict SET value = '0'  
        WHERE name = 'exit code';  
        UPDATE #hook_dict SET value = 'TRUE'  
        WHERE name = 'abort synchronization';  
    ELSEIF [condition that defines the error abort case] THEN  
        UPDATE #hook_dict SET value = '1'  
        WHERE name = 'exit code';  
        UPDATE #hook_dict SET value = 'TRUE'  
        WHERE name = 'abort synchronization';  
    END IF;  
END;
```

sp_hook_dbmlsync_all_error

Mit dieser gespeicherten Prozedur können Sie alle Fehlermeldungstypen von dbmlsync verarbeiten. Sie können beispielsweise den Hook `sp_hook_dbmlsync_all_error` einführen, um Fehler zu protokollieren oder beim Eintreten eines bestimmten Fehlers eine bestimmte Aktion durchzuführen.

Zeilen in der Tabelle #hook_dict

Name	Wert	Beschreibung
publication_ <i>n</i> (in)	Publikation	Nicht mehr empfohlen. Verwenden Sie stattdessen subscription_ <i>n</i> . Die synchronisierten Publikationen, wobei <i>n</i> eine Ganzzahl ist. Es gibt einen publication_ <i>n</i> -Eintrag für jede synchronisierte Publikation. Die Nummerierung von <i>n</i> beginnt bei Null.
MobiLink user (in)	MobiLink-Benutzername	Der MobiLink-Benutzer, für den Sie synchronisieren
script version (in)	Skriptversionsname	Die MobiLink-Skriptversion, die für die Synchronisation verwendet werden soll
error message (in)	Fehlermeldungstext	Dies ist der Text, der auch im dbmlsync-Log angezeigt wird.
error id (in)	Ganzzahl	Eine ID, die die Meldung eindeutig identifiziert. In dieser Zeile können Sie die Fehlermeldung identifizieren, da der Meldungstext abweichen kann.
error hook user state (in out)	Ganzzahl	Dieser Wert kann vom Hook eingestellt werden, um Statusinformationen in zukünftigen Aufrufen an die Hooks <code>sp_hook_dbmlsync_all_error</code> , <code>sp_hook_dbmlsync_communication_error</code> , <code>sp_hook_dbmlsync_misc_error</code> , <code>sp_hook_dbmlsync_sql_error</code> oder <code>sp_hook_dbmlsync_end</code> zu übergeben. Wenn einer dieser Hooks zum ersten Mal aufgerufen wird, ist der Wert der Zeile 0. Wenn ein Hook den Wert der Zeile ändert, wird der neue Wert im nächsten Hook-Aufruf verwendet.

Name	Wert	Beschreibung
subscription_ <i>n</i> (in)	Subskriptionsname(<i>n</i>)	Die synchronisierten Subskriptionen, wobei <i>n</i> eine Ganzzahl ist. Dies ist ein subscription_ <i>n</i> -Eintrag für jede synchronisierte Subskription. Die Nummerierung von <i>n</i> beginnt bei Null.

Bemerkungen

Jedes Mal, wenn eine dbmlsync-Fehlermeldung generiert wird, werden die folgenden Hooks aufgerufen:

- Zunächst wird je nach Fehlertyp einer der folgenden Hooks aufgerufen: sp_hook_dbmlsync_communication_error, sp_hook_dbmlsync_misc_error oder sp_hook_dbmlsync_sql_error. Die Hooks enthalten Informationen, die sich auf den speziellen Fehlertyp beziehen. 'Sqlcode' und 'sqlstate' beispielsweise weisen auf SQL-Fehler hin.
- Dann wird sp_hook_dbmlsync_all_error aufgerufen. Dieser Hook kann zur Protokollierung aller aufgetretenen Fehler verwendet werden.

Wenn beim Start ein Fehler auftritt, bevor eine Synchronisation initiiert werden konnte, werden die #hook_dict-Einträge für den MobiLink-Benutzer und die Skriptversion auf eine leere Zeichenfolge zurückgesetzt, und in der #hook_dict-Tabelle werden keine publication_ *n*- oder subscription_ *n*-Zeilen eingestellt.

Die Zeile "error hook user state" bietet ein nützliches Verfahren für die Übergabe von Informationen über die Art des Fehlers an den Hook sp_hook_dbmlsync_end. Anhand dieser Informationen können Sie entscheiden, ob die Synchronisation erneut ausgeführt werden soll.

Diese Prozedur wird in einer eigenen Verbindung ausgeführt, damit sichergestellt ist, dass die dadurch bewirkten Vorgänge nicht verloren gehen, wenn ein Zurücksetzen der Synchronisationsverbindung erfolgt. Wenn dbmlsync keine getrennte Verbindung herstellen kann, wird die Prozedur nicht aufgerufen.

Da dieser Hook in einer eigenen Verbindung ausgeführt wird, sollten Sie beim Zugriff auf Tabellen vorsichtig sein, die in Ihrer Hook-Prozedur synchronisiert werden, denn diese Tabellen werden von dbmlsync möglicherweise gesperrt. Diese Sperren können zum Fehlschlagen von Vorgängen im Hook oder zu unbegrenzten Wartezeiten führen.

Die Aktionen dieser Prozedur werden sofort nach Beenden des Hooks festgeschrieben.

Privilegien

Hook-Prozeduren können von jedem Benutzer mit dem MANAGE REPLICATION-Systemprivileg erstellt werden. Um jedoch zu gewährleisten, dass ein Hook auf die #hook_dict-Tabelle zugreifen kann, die verwendet wird, um Daten an und aus Hooks zu übergeben, müssen Hooks eine der folgenden Anforderungen erfüllen:

- Eigentümer muss ein Benutzer sein, der die Systemprivilegien SELECT ANY TABLE und UPDATE ANY TABLE hat.
- Sie müssen durch die SQL SECURITY INVOKER-Klausel der CREATE PROCEDURE-Anweisung definiert sein.

Siehe auch

- „Behandlung von Fehlern und Warnungen in Ereignis-Hook-Prozeduren“ auf Seite 207
- „sp_hook_dbmlsync_communication_error“ auf Seite 215
- „sp_hook_dbmlsync_misc_error“ auf Seite 240
- „sp_hook_dbmlsync_sql_error“ auf Seite 257

Beispiel

Nehmen wir an, Sie verwenden die folgende Tabelle, um Fehler in der entfernten Datenbank zu protokollieren.

```
CREATE TABLE error_log
(
  pk INTEGER DEFAULT AUTOINCREMENT PRIMARY KEY,
  err_id INTEGER,
  err_msg VARCHAR(10240),
);
```

Im folgenden Beispiel werden Fehler mithilfe von sp_hook_dbmlsync_all_error protokolliert.

```
CREATE PROCEDURE sp_hook_dbmlsync_all_error()
BEGIN
  DECLARE msg VARCHAR(10240);
  DECLARE id INTEGER;

  // get the error message text
  SELECT value INTO msg
  FROM #hook_dict
  WHERE name = 'error message';

  // get the error id
  SELECT value INTO id
  FROM #hook_dict
  WHERE name = 'error id';

  // log the error information
  INSERT INTO error_log(err_msg, err_id)
  VALUES (msg, id);
END;
```

Um mögliche Fehler-IDs zu ermitteln, sollten Sie einen Testdurchlauf von dbmlsync durchführen. Wenn dbmlsync beispielsweise den Fehler "Verbindung mit MobiLink-Server kann nicht hergestellt werden" zurückgibt, fügt sp_hook_dbmlsync_all_error folgende Zeile in error_log ein:

```
1,14173,
'Unable to connect to MobiLink server'
```

Nun können Sie den Fehler "Unable to connect to MobiLink server" (Verbindung mit MobiLink-Server kann nicht hergestellt werden) mit der Fehler-ID 14173 in Verbindung bringen.

Im folgenden Beispiel werden Hooks so eingerichtet, dass die Synchronisation neu gestartet wird, sobald der Fehler 14173 auftritt.

```
CREATE PROCEDURE sp_hook_dbmlsync_all_error()
BEGIN
  IF EXISTS( SELECT value FROM #hook_dict
            WHERE name = 'error id' AND value = '14173' )
  THEN
```

```
        UPDATE #hook_dict SET value = '1'
        WHERE name = 'error hook user state';
    END IF;
END;

CREATE PROCEDURE sp_hook_dbmlsync_end()
BEGIN
    IF EXISTS( SELECT value FROM #hook_dict
        WHERE name='error hook user state' AND value='1')
    THEN
        UPDATE #hook_dict SET value = 'sync'
        WHERE name='restart';
    END IF;
END;
```

Siehe „sp_hook_dbmlsync_end“ auf Seite 232.

sp_hook_dbmlsync_begin

Verwenden Sie diese gespeicherte Prozedur, um benutzerdefinierte Aktionen zu Beginn des Synchronisationsprozesses hinzuzufügen.

Zeilen in der Tabelle #hook_dict

Name	Wert	Beschreibung
publication_ <i>n</i> (in)	Publikation	Nicht mehr empfohlen. Verwenden Sie stattdessen subscription_ <i>n</i> . Die synchronisierten Publikationen, wobei <i>n</i> eine Ganzzahl ist. Es gibt einen publication_ <i>n</i> -Eintrag für jede synchronisierte Publikation. Die Nummerierung von <i>n</i> beginnt bei Null.
MobiLink user (in)	MobiLink-Benutzername	Der MobiLink-Benutzer, für den Sie synchronisieren
script version (in)	Skriptversionsname	Die MobiLink-Skriptversion, die für die Synchronisation verwendet werden soll
subscription_ <i>n</i> (in)	Subskriptionsname(<i>n</i>)	Die synchronisierten Subskriptionen, wobei <i>n</i> eine Ganzzahl ist. Dies ist ein subscription_ <i>n</i> -Eintrag für jede synchronisierte Subskription. Die Nummerierung von <i>n</i> beginnt bei Null.

Bemerkungen

Wenn eine Prozedur dieses Namens vorhanden ist, wird sie zu Beginn des Synchronisationsprozesses aufgerufen.

Die Aktionen dieser Prozedur werden sofort nach ihrer Ausführung festgeschrieben.

Privilegien

Hook-Prozeduren können von jedem Benutzer mit dem MANAGE REPLICATION-Systemprivileg erstellt werden. Um jedoch zu gewährleisten, dass ein Hook auf die #hook_dict-Tabelle zugreifen kann, die verwendet wird, um Daten an und aus Hooks zu übergeben, müssen Hooks eine der folgenden Anforderungen erfüllen:

- Eigentümer muss ein Benutzer sein, der die Systemprivilegien SELECT ANY TABLE und UPDATE ANY TABLE hat.
- Sie müssen durch die SQL SECURITY INVOKER-Klausel der CREATE PROCEDURE-Anweisung definiert sein.

Siehe auch

- [„Hook-Sequenz bei der Synchronisation“ auf Seite 203](#)

Beispiele

Nehmen wir an, Sie verwenden die folgende Tabelle, um Synchronisationsereignisse in der entfernten Datenbank zu protokollieren.

```
CREATE TABLE SyncLog
(
    "event_id"          integer NOT NULL DEFAULT AUTOINCREMENT ,
    "event_time"        timestamp NULL,
    "event_name"        varchar(128) NOT NULL ,
    "subs"              varchar(1024) NULL ,
    PRIMARY KEY ("event_id")
)
```

Folgendes zeichnet den Beginn jeder Synchronisation in der Tabelle auf.

```
CREATE PROCEDURE sp_hook_dbmlsync_begin ()
BEGIN

    DECLARE subs_list VARCHAR(1024);

    -- build a list of subscriptions being synchronized
    SELECT LIST(value) INTO subs_list
    FROM #hook_dict
    WHERE name LIKE 'subscription_%';

    -- log the event
    INSERT INTO SyncLog(event_time, event_name, subs)
    VALUES( CURRENT_TIMESTAMP, 'sp_hook_dbmlsync_begin', subs_list );
END
```

sp_hook_dbmlsync_communication_error

Verwenden Sie diese gespeicherte Prozedur, um Verbindungsfehler zu bearbeiten.

Zeilen in der Tabelle #hook_dict

Name	Wert	Beschreibung
publication_ <i>n</i> (in)	Publikation	Nicht mehr empfohlen. Verwenden Sie stattdessen subscription_ <i>n</i> . Die synchronisierten Publikationen, wobei <i>n</i> eine Ganzzahl ist. Es gibt einen publication_ <i>n</i> -Eintrag für jede synchronisierte Publikation. Die Nummerierung von <i>n</i> beginnt bei Null.
MobiLink user (in)	MobiLink-Benutzername	Der MobiLink-Benutzer, für den Sie synchronisieren
script version (in)	Skriptversionsname	Die MobiLink-Skriptversion, die für die Synchronisation verwendet werden soll
error message (in)	Fehlermeldungstext	Dies ist der Text, der auch im dbmlsync-Log angezeigt wird.
error id (in)	Nummerisch	Eine ID, die die Meldung eindeutig identifiziert. In dieser Zeile können Sie die Fehlermeldung identifizieren, da der Meldungstext abweichen kann.
error hook user state (in out)	Ganzzahl	Dieser Wert kann vom Hook eingestellt werden, um Statusinformationen in zukünftigen Aufrufen an die Hooks sp_hook_dbmlsync_all_error, sp_hook_dbmlsync_communication_error, sp_hook_dbmlsync_misc_error, sp_hook_dbmlsync_sql_error oder sp_hook_dbmlsync_end zu übergeben. Wenn einer dieser Hooks zum ersten Mal aufgerufen wird, ist der Wert der Zeile 0. Wenn ein Hook den Wert der Zeile ändert, wird der neue Wert im nächsten Hook-Aufruf verwendet.
stream error code (in)	Ganzzahl	Der Fehler, der vom Datenstrom gemeldet wurde
system error code (in)	Ganzzahl	Ein systemspezifischer Fehlercode.
subscription_ <i>n</i> (in)	Subskriptionsname(<i>n</i>)	Die synchronisierten Subskriptionen, wobei <i>n</i> eine Ganzzahl ist. Dies ist ein subscription_ <i>n</i> -Eintrag für jede synchronisierte Subskription. Die Nummerierung von <i>n</i> beginnt bei Null.

Bemerkungen

Wenn beim Start ein Fehler auftritt, bevor eine Synchronisation initiiert werden konnte, werden die #hook_dict-Einträge für den MobiLink-Benutzer und die Skriptversion auf eine leere Zeichenfolge zurückgesetzt, und in der #hook_dict-Tabelle werden keine publication_n- oder subscription_n-Zeilen eingestellt.

Wenn zwischen dbmlsync und dem MobiLink-Server Verbindungsfehler auftreten, können Sie mit diesem Hook auf datenstromspezifische Fehlerinformationen zugreifen.

Der Parameter **stream error code** ist eine Ganzzahl, die für den Fehlertyp des Verbindungsfehlers steht.

Die Zeile "error hook user state" bietet ein nützliches Verfahren für die Übergabe von Informationen über die Art des Fehlers an den Hook sp_hook_dbmlsync_end. Anhand dieser Informationen können Sie entscheiden, ob die Synchronisation erneut ausgeführt werden soll.

Diese Prozedur wird in einer eigenen Verbindung ausgeführt, damit sichergestellt ist, dass die dadurch bewirkten Vorgänge nicht verloren gehen, wenn ein Zurücksetzen der Synchronisationsverbindung erfolgt. Wenn dbmlsync keine getrennte Verbindung herstellen kann, wird die Prozedur nicht aufgerufen.

Da dieser Hook in einer eigenen Verbindung ausgeführt wird, sollten Sie beim Zugriff auf Tabellen vorsichtig sein, die in Ihrer Hook-Prozedur synchronisiert werden, denn diese Tabellen werden von dbmlsync möglicherweise gesperrt. Diese Sperren können zum Fehlschlagen von Vorgängen im Hook oder zu unbegrenzten Wartezeiten führen.

Die Aktionen dieser Prozedur werden sofort nach ihrer Ausführung festgeschrieben.

Privilegien

Hook-Prozeduren können von jedem Benutzer mit dem MANAGE REPLICATION-Systemprivileg erstellt werden. Um jedoch zu gewährleisten, dass ein Hook auf die #hook_dict-Tabelle zugreifen kann, die verwendet wird, um Daten an und aus Hooks zu übergeben, müssen Hooks eine der folgenden Anforderungen erfüllen:

- Eigentümer muss ein Benutzer sein, der die Systemprivilegien SELECT ANY TABLE und UPDATE ANY TABLE hat.
- Sie müssen durch die SQL SECURITY INVOKER-Klausel der CREATE PROCEDURE-Anweisung definiert sein.

Siehe auch

- „Fehlermeldungen zur MobiLink-Kommunikation“ [[Fehlermeldungen](#)]
- „Behandlung von Fehlern und Warnungen in Ereignis-Hook-Prozeduren“ auf Seite 207
- „sp_hook_dbmlsync_all_error“ auf Seite 211
- „sp_hook_dbmlsync_misc_error“ auf Seite 240
- „sp_hook_dbmlsync_sql_error“ auf Seite 257

Beispiel

Nehmen wir an, Sie verwenden die folgende Tabelle, um Verbindungsfehler in der entfernten Datenbank zu protokollieren.

```
CREATE TABLE communication_error_log
(
  error_msg VARCHAR(10240),
  error_code VARCHAR(128)
);
```

Im folgenden Beispiel werden Verbindungsfehler mithilfe von `sp_hook_dbmlsync_communication_error` protokolliert.

```
CREATE PROCEDURE sp_hook_dbmlsync_communication_error()
BEGIN
  DECLARE msg VARCHAR(255);
  DECLARE code INTEGER;

  // get the error message text
  SELECT value INTO msg
  FROM #hook_dict
  WHERE name = 'error message';

  // get the error code
  SELECT value INTO code
  FROM #hook_dict
  WHERE name = 'stream error code';

  // log the error information
  INSERT INTO communication_error_log(error_code,error_msg)
  VALUES (code,msg);
END
```

sp_hook_dbmlsync_delay

Verwenden Sie diese gespeicherte Prozedur, um zu steuern, wo die Synchronisation stattfindet.

Zeilen in der Tabelle #hook_dict

Name	Wert	Beschreibung
delay duration (in out)	Anzahl von Sekunden	Wenn die Prozedur den Wert der Verzögerungsdauer (Delay Duration) auf Null setzt, wird die dbmlsync-Synchronisation unmittelbar fortgesetzt. Wenn der Wert für die Verzögerungsdauer nicht Null ist, wird damit die Anzahl der Sekunden festgelegt, bevor der Verzögerungs-Hook ("delay hook") erneut aufgerufen wird.

Name	Wert	Beschreibung
maximum accumulated delay (in out)	Anzahl von Sekunden	Die maximale angesammelte Verzögerung gibt die maximale Anzahl von Sekunden Verzögerung vor jeder Synchronisation an. Dbmlsync registriert die gesamte Verzögerung, die von allen Aufrufen des Verzögerungs-Hooks seit der letzten Synchronisation akkumuliert wurden. Wenn seit dem Start von dbmlsync keine Synchronisation erfolgt ist, wird die Gesamtverzögerung aus der Zeit errechnet, die seit dem Start von dbmlsync verstrichen ist. Wenn die Gesamtverzögerung den Wert von "maximum accumulated delay" überschreitet, beginnt die Synchronisation ohne weiteren Aufruf des Verzögerungs-Hooks.
publication_ <i>n</i> (in)	Publikation	Nicht mehr empfohlen. Verwenden Sie stattdessen subscription_ <i>n</i> . Die synchronisierten Publikationen, wobei <i>n</i> eine Ganzzahl ist. Es gibt einen publication_ <i>n</i> -Eintrag für jede synchronisierte Publikation. Die Nummerierung von <i>n</i> beginnt bei Null.
MobiLink user (in)	MobiLink-Benutzername	Der MobiLink-Benutzer, für den Sie synchronisieren
script version (in)	Skriptversionsname	Die MobiLink-Skriptversion, die für die Synchronisation verwendet werden soll
subscription_ <i>n</i> (in)	Subskriptionsname(<i>n</i>)	Die synchronisierten Subskriptionen, wobei <i>n</i> eine Ganzzahl ist. Dies ist ein subscription_ <i>n</i> -Eintrag für jede synchronisierte Subskription. Die Nummerierung von <i>n</i> beginnt bei Null.

Bemerkungen

Wenn eine Prozedur dieses Namens vorhanden ist, wird sie zu Beginn des Synchronisationsprozesses vor **sp_hook_dbmlsync_begin** aufgerufen.

Dieser Hook wird nicht aufgerufen, wenn die Synchronisation mit der Dbmlsync-API oder der SQL SYNCHRONIZE-Anweisung initiiert wird.

Die Aktionen dieser Prozedur werden sofort nach ihrer Ausführung festgeschrieben.

Privilegien

Hook-Prozeduren können von jedem Benutzer mit dem MANAGE REPLICATION-Systemprivileg erstellt werden. Um jedoch zu gewährleisten, dass ein Hook auf die #hook_dict-Tabelle zugreifen kann, die verwendet wird, um Daten an und aus Hooks zu übergeben, müssen Hooks eine der folgenden Anforderungen erfüllen:

- Eigentümer muss ein Benutzer sein, der die Systemprivilegien SELECT ANY TABLE und UPDATE ANY TABLE hat.
- Sie müssen durch die SQL SECURITY INVOKER-Klausel der CREATE PROCEDURE-Anweisung definiert sein.

Siehe auch

- „Hook-Sequenz bei der Synchronisation“ auf Seite 203
- „Initiieren einer Synchronisation mit Ereignis-Hooks“ auf Seite 101
- „sp_hook_dbmlsync_download_end“ auf Seite 222

Beispiel

Nehmen wir an, Sie verwenden die folgende Tabelle, um Bestellungen in der entfernten Datenbank zu protokollieren.

```
CREATE TABLE OrdersTable(  
  "id" INTEGER PRIMARY KEY DEFAULT AUTOINCREMENT,  
  "priority" VARCHAR(128)  
);
```

Das folgende Beispiel verzögert die Synchronisation um eine maximale Gesamtverzögerung von einer Stunde. Alle zehn Sekunden wird der Hook erneut aufgerufen, der die Tabelle OrdersTable auf eine Zeile hoher Priorität überprüft. Wenn eine Zeile hoher Priorität existiert, wird "delay duration" auf Null gesetzt, um den Synchronisationsprozess zu starten.

```
CREATE PROCEDURE sp_hook_dbmlsync_delay()  
BEGIN  
  -- Set the maximum delay between synchronizations  
  -- or before the first synchronization starts to 1 hour  
  UPDATE #hook_dict SET value = '3600' // 3600 seconds  
  WHERE name = 'maximum accumulated delay';  
  
  -- check if a high priority order exists in OrdersTable  
  IF EXISTS (SELECT * FROM OrdersTable where priority='high') THEN  
    -- start the synchronization to process the high priority row  
    UPDATE #hook_dict  
      SET value = '0'  
      WHERE name='delay duration';  
  ELSE  
    -- set the delay duration to call this procedure again  
    -- following a 10 second delay  
    UPDATE #hook_dict  
      SET value = '10'  
      WHERE name='delay duration';  
  END IF;  
END;
```

Im sp_hook_dbmlsync_end-Hook können Sie die Zeile hoher Priorität als verarbeitet markieren.

```

CREATE PROCEDURE sp_hook_dbmlsync_upload_end()
BEGIN
    IF EXISTS( SELECT value FROM #hook_dict
        WHERE name = 'Upload status'
        AND value = 'committed' ) THEN
        UPDATE OrderTable SET priority = 'high-processed'
        WHERE priority = 'high';
    END IF;
END;

```

In diesem Beispiel wird vorausgesetzt, dass Sie die erweiterte LockTables-Option verwendet haben, um sicherzustellen, dass die Tabellen während der Synchronisation gesperrt sind. Wenn die Tabellen nicht gesperrt sind, ist es möglich, dass eine Zeile hoher Priorität eingefügt wurde, nachdem der Upload erstellt wurde, jedoch noch bevor der Hook `sp_hook_dbmlsync_end` ausgeführt wird. In diesem Fall wird die Priorität der Zeile in "high-processed" geändert, auch wenn sie nie hochgeladen wurde.

Siehe „`sp_hook_dbmlsync_end`“ auf Seite 232.

sp_hook_dbmlsync_download_begin

Verwenden Sie diese gespeicherte Prozedur, um benutzerdefinierte Aktionen am Anfang des Downloadvorgangs des Synchronisationsprozesses hinzuzufügen.

Zeilen in der Tabelle #hook_dict

Name	Wert	Beschreibung
publication_ <i>n</i> (in)	Publikation	Nicht mehr empfohlen. Verwenden Sie stattdessen subscription_ <i>n</i> . Die synchronisierten Publikationen, wobei <i>n</i> eine Ganzzahl ist. Es gibt einen publication_ <i>n</i> -Eintrag für jede synchronisierte Publikation. Die Nummerierung von <i>n</i> beginnt bei Null.
MobiLink user (in)	MobiLink-Benutzername	Der MobiLink-Benutzer, für den Sie synchronisieren
script version (in)	Skriptversionsname	Die MobiLink-Skriptversion, die für die Synchronisation verwendet werden soll
subscription_ <i>n</i> (in)	Subskriptionsname(<i>n</i>)	Die synchronisierten Subskriptionen, wobei <i>n</i> eine Ganzzahl ist. Dies ist ein subscription_ <i>n</i> -Eintrag für jede synchronisierte Subskription. Die Nummerierung von <i>n</i> beginnt bei Null.

Bemerkungen

Wenn eine Prozedur dieses Namens vorhanden ist, wird sie zu Beginn des Downloadvorgangs des Synchronisationsprozesses aufgerufen.

Die Aktionen dieser Prozedur werden festgeschrieben bzw. zurückgesetzt, wenn der Download festgeschrieben bzw. zurückgesetzt wird.

Privilegien

Hook-Prozeduren können von jedem Benutzer mit dem MANAGE REPLICATION-Systemprivileg erstellt werden. Um jedoch zu gewährleisten, dass ein Hook auf die #hook_dict-Tabelle zugreifen kann, die verwendet wird, um Daten an und aus Hooks zu übergeben, müssen Hooks eine der folgenden Anforderungen erfüllen:

- Eigentümer muss ein Benutzer sein, der die Systemprivilegien SELECT ANY TABLE und UPDATE ANY TABLE hat.
- Sie müssen durch die SQL SECURITY INVOKER-Klausel der CREATE PROCEDURE-Anweisung definiert sein.

Siehe auch

- [„Hook-Sequenz bei der Synchronisation“ auf Seite 203](#)

Beispiel

Nehmen wir an, Sie verwenden die folgende Tabelle, um Synchronisationsereignisse in der entfernten Datenbank zu protokollieren.

```
CREATE TABLE SyncLog
(
    "event_id"          integer NOT NULL DEFAULT AUTOINCREMENT ,
    "event_time"        timestamp NULL,
    "event_name"        varchar(128) NOT NULL ,
    "subs"              varchar(1024) NULL ,
    PRIMARY KEY ("event_id")
)
```

Folgendes zeichnet den Download-Beginn für jede Synchronisation auf.

```
CREATE PROCEDURE sp_hook_dbmlsync_download_begin ()
BEGIN

    DECLARE subs_list VARCHAR(1024);

    -- build a list of subscriptions being synchronized
    SELECT LIST(value) INTO subs_list
    FROM #hook_dict
    WHERE name LIKE 'subscription_%';

    -- log the event
    INSERT INTO SyncLog(event_time, event_name, subs)
    VALUES( CURRENT TIMESTAMP, 'sp_hook_dbmlsync_download_begin',
    subs_list );
END
```

sp_hook_dbmlsync_download_end

Verwenden Sie diese gespeicherte Prozedur, um benutzerdefinierte Aktionen am Ende des Downloadvorgangs des Synchronisationsprozesses hinzuzufügen.

Zeilen in der Tabelle #hook_dict

Name	Wert	Beschreibung
publication_ <i>n</i> (in)	Publikation	Nicht mehr empfohlen. Verwenden Sie stattdessen subscription_ <i>n</i> . Die synchronisierten Publikationen, wobei <i>n</i> eine Ganzzahl ist. Es gibt einen publication_ <i>n</i> -Eintrag für jede synchronisierte Publikation. Die Nummerierung von <i>n</i> beginnt bei Null.
MobiLink user (in)	MobiLink-Benutzername	Der MobiLink-Benutzer, für den Sie synchronisieren
script version (in)	Skriptversionsname	Die MobiLink-Skriptversion, die für die Synchronisation verwendet werden soll
subscription_ <i>n</i> (in)	Subskriptionsname(<i>n</i>)	Die synchronisierten Subskriptionen, wobei <i>n</i> eine Ganzzahl ist. Dies ist ein subscription_ <i>n</i> -Eintrag für jede synchronisierte Subskription. Die Nummerierung von <i>n</i> beginnt bei Null.

Bemerkungen

Wenn eine Prozedur dieses Namens vorhanden ist, wird sie zum Ende des Downloadvorgangs des Synchronisationsprozesses aufgerufen.

Die Aktionen dieser Prozedur werden festgeschrieben bzw. zurückgesetzt, wenn der Download festgeschrieben bzw. zurückgesetzt wird.

Privilegien

Hook-Prozeduren können von jedem Benutzer mit dem MANAGE REPLICATION-Systemprivileg erstellt werden. Um jedoch zu gewährleisten, dass ein Hook auf die #hook_dict-Tabelle zugreifen kann, die verwendet wird, um Daten an und aus Hooks zu übergeben, müssen Hooks eine der folgenden Anforderungen erfüllen:

- Eigentümer muss ein Benutzer sein, der die Systemprivilegien SELECT ANY TABLE und UPDATE ANY TABLE hat.
- Sie müssen durch die SQL SECURITY INVOKER-Klausel der CREATE PROCEDURE-Anweisung definiert sein.

Siehe auch

- „Hook-Sequenz bei der Synchronisation“ auf Seite 203
- „Initiieren einer Synchronisation mit Ereignis-Hooks“ auf Seite 101
- „sp_hook_dbmlsync_delay“ auf Seite 218

Beispiele

Nehmen wir an, Sie verwenden die folgende Tabelle, um Synchronisationsereignisse in der entfernten Datenbank zu protokollieren.

```
CREATE TABLE SyncLog
(
  "event_id"          integer NOT NULL DEFAULT autoincrement ,
  "event_time"        timestamp NULL,
  "event_name"        varchar(128) NOT NULL ,
  "subs"              varchar(1024) NULL ,
  PRIMARY KEY ("event_id")
)
```

Folgendes zeichnet das Download-Ende für jede Synchronisation auf.

```
CREATE PROCEDURE sp_hook_dbmlsync_download_end ()
BEGIN

  DECLARE subs_list VARCHAR(1024);

  -- build a list of subscriptions being synchronized
  SELECT LIST(value) INTO subs_list
  FROM #hook_dict
  WHERE name LIKE 'subscription_%';

  -- log the event
  INSERT INTO SyncLog(event_time, event_name, subs)
  VALUES( CURRENT_TIMESTAMP, 'sp_hook_dbmlsync_download_end', subs_list );
END
```

sp_hook_dbmlsync_download_log_ri_violation

Protokolliert referenzielle Integrität beim Downloadvorgang

Zeilen in der Tabelle #hook_dict

Name	Wert	Beschreibung
publication_n (in)	Publikation	Nicht mehr empfohlen. Verwenden Sie stattdessen subscription_n. Die synchronisierten Publikationen, wobei n eine Ganzzahl ist. Es gibt einen publication_n-Eintrag für jede synchronisierte Publikation. Die Nummerierung von n beginnt bei Null.
MobiLink user (in)	MobiLink-Benutzername	Der MobiLink-Benutzer, für den Sie synchronisieren
foreign key table (in)	Tabellenname	Die Tabelle, die die Fremdspalte enthält, für die der Hook aufgerufen wird

Name	Wert	Beschreibung
primary key table (in)	Tabellenname	Die Tabelle, die vom Fremdschlüssel referenziert wird, für den der Hook aufgerufen wurde
role name (in)	Rollenname	Der Rollenname des Fremdschlüssels, für den der Hook aufgerufen wird
script version (in)	Skriptversionsname	Die MobiLink-Skriptversion, die für die Synchronisation verwendet werden soll
subscription_ <i>n</i> (in)	Subskriptionsname(<i>n</i>)	Die synchronisierten Subskriptionen, wobei <i>n</i> eine Ganzzahl ist. Dies ist ein subscription_ <i>n</i> -Eintrag für jede synchronisierte Subskription. Die Nummerierung von <i>n</i> beginnt bei Null.

Bemerkungen

Eine Download-RI-Verletzung tritt ein, wenn Zeilen im Download die Fremdschlüsselbeziehungen in der entfernten Datenbank verletzen. Diese Hooks gestatten es Ihnen, auftretende RI-Verletzungen zu protokollieren, sodass Sie ihre Ursache später untersuchen können.

Nachdem der Download abgeschlossen ist, aber noch vor dem Festschreiben, prüft dbmlsync auf RI-Verletzungen. Wenn eine Verletzung gefunden wird, identifiziert das Dienstprogramm einen Fremdschlüssel, der eine RI-Verletzung enthält, und ruft sp_hook_dbmlsync_download_log_ri_violation auf (sofern implementiert). Danach ruft es sp_hook_dbmlsync_download_ri_violation auf (sofern implementiert). Wenn der Konflikt weiterhin besteht, löscht dbmlsync die Zeile, die die Fremdschlüssel-Integrität verletzt. Dieser Prozess wird für die restlichen Fremdschlüssel wiederholt, bei denen RI-Verletzungen vorkommen.

Dieser Hook wird nur aufgerufen, wenn RI-Verletzungen für Tabellen vorkommen, die derzeit synchronisiert werden. Wenn RI-Verletzungen für Tabellen vorhanden sind, die nicht synchronisiert werden, wird dieser Hook nicht aufgerufen, und die Synchronisation schlägt fehl.

Dieser Hook wird über eine andere als die von dbmlsync für den Download verwendete Verbindung aufgerufen. Die vom Hook verwendete Verbindung hat die Isolationsstufe 0, sodass der Hook die Zeilen sehen kann, die vom Download übernommen wurden und noch nicht festgeschrieben sind. Die Aktionen des Hooks werden sofort nach seiner Ausführung festgeschrieben, sodass die durchgeführten Änderungen beibehalten werden, und zwar unabhängig davon, ob der Download festgeschrieben oder zurückgesetzt wird.

Da dieser Hook in einer eigenen Verbindung ausgeführt wird, sollten Sie beim Zugriff auf Tabellen vorsichtig sein, die in Ihrer Hook-Prozedur synchronisiert werden, denn diese Tabellen werden von dbmlsync möglicherweise gesperrt. Diese Sperren können zum Fehlschlagen von Vorgängen im Hook oder zu unbegrenzten Wartezeiten führen.

Versuchen Sie nicht, diesen Hook zu verwenden, um RI-Verletzungen aufzulösen. Er darf ausschließlich für die Protokollierung verwendet werden. Für die Behebung von RI-Verletzungen steht sp_hook_dbmlsync_download_ri_violation zur Verfügung.

Privilegien

Hook-Prozeduren können von jedem Benutzer mit dem MANAGE REPLICATION-Systemprivileg erstellt werden. Um jedoch zu gewährleisten, dass ein Hook auf die #hook_dict-Tabelle zugreifen kann, die verwendet wird, um Daten an und aus Hooks zu übergeben, müssen Hooks eine der folgenden Anforderungen erfüllen:

- Eigentümer muss ein Benutzer sein, der die Systemprivilegien SELECT ANY TABLE und UPDATE ANY TABLE hat.
- Sie müssen durch die SQL SECURITY INVOKER-Klausel der CREATE PROCEDURE-Anweisung definiert sein.

Siehe auch

- „sp_hook_dbmlsync_download_ri_violation“ auf Seite 226
- „Hook-Sequenz bei der Synchronisation“ auf Seite 203

Beispiele

Nehmen wir an, Sie verwenden die folgende Tabelle, um Verletzungen der referenziellen Integrität zu protokollieren.

```
CREATE TABLE DBA.LogRIViolationTable
(
    entry_time    TIMESTAMP,
    pk_table      VARCHAR( 255 ),
    fk_table      VARCHAR( 255 ),
    role_name     VARCHAR( 255 )
);
```

Das folgende Beispiel protokolliert den Fremdschlüssel-Tabellennamen, den Primärschlüssel-Tabellennamen und den Rollennamen, wenn in der entfernten Datenbank eine Verletzung der referenziellen Integrität entdeckt wird. Die Informationen werden in der LogRIViolationTable-Tabelle in der entfernten Datenbank gespeichert.

```
CREATE PROCEDURE sp_hook_dbmlsync_download_log_ri_violation()
BEGIN
    INSERT INTO DBA.LogRIViolationTable VALUES(
        CURRENT_TIMESTAMP,
        (SELECT value FROM #hook_dict WHERE name = 'Primary key table'),
        (SELECT value FROM #hook_dict WHERE name = 'Foreign key table'),
        (SELECT value FROM #hook_dict WHERE name = 'Role name' ) );
END;
```

sp_hook_dbmlsync_download_ri_violation

Gestattet es Ihnen, Verletzungen der referenziellen Integrität beim Downloadvorgang zu beheben

Zeilen in der Tabelle #hook_dict

Name	Wert	Beschreibung
publication_ <i>n</i> (in)	Publikation	Nicht mehr empfohlen. Verwenden Sie stattdessen subscription_ <i>n</i> . Die synchronisierten Publikationen, wobei <i>n</i> eine Ganzzahl ist. Es gibt einen publication_ <i>n</i> -Eintrag für jede synchronisierte Publikation. Die Nummerierung von <i>n</i> beginnt bei Null.
MobiLink user (in)	MobiLink-Benutzername	Der MobiLink-Benutzer, für den Sie synchronisieren
foreign key table (in)	Tabellenname	Die Tabelle, die die Fremdspalte enthält, für die der Hook aufgerufen wird
primary key table (in)	Tabellenname	Die Tabelle, die vom Fremdschlüssel referenziert wird, für den der Hook aufgerufen wurde
role name (in)	Rollenname	Der Rollenname des Fremdschlüssels, für den der Hook aufgerufen wird
script version (in)	Skriptversionsname	Die MobiLink-Skriptversion, die für die Synchronisation verwendet werden soll
subscription_ <i>n</i> (in)	Subskriptionsname(<i>n</i>)	Die synchronisierten Subskriptionen, wobei <i>n</i> eine Ganzzahl ist. Dies ist ein subscription_ <i>n</i> -Eintrag für jede synchronisierte Subskription. Die Nummerierung von <i>n</i> beginnt bei Null.

Bemerkungen

Eine Download-RI-Verletzung tritt ein, wenn Zeilen im Download die Fremdschlüsselbeziehungen in der entfernten Datenbank verletzen. Mit diesem Hook können Sie versuchen, RI-Verletzungen zu beheben, bevor dbmlsync die Zeilen entdeckt, die den Konflikt verursachen.

Nachdem der Download abgeschlossen ist, aber noch vor dem Festschreiben, prüft dbmlsync auf RI-Verletzungen. Wenn eine Verletzung gefunden wird, identifiziert das Dienstprogramm einen Fremdschlüssel, der eine RI-Verletzung enthält, und ruft sp_hook_dbmlsync_download_log_ri_violation auf (sofern implementiert). Danach ruft es sp_hook_dbmlsync_download_ri_violation auf (sofern implementiert). Wenn weiterhin ein Konflikt besteht, löscht dbmlsync die Zeilen. Dieser Prozess wird für die restlichen Fremdschlüssel wiederholt, bei denen RI-Verletzungen vorkommen.

Dieser Hook wird nur aufgerufen, wenn RI-Verletzungen für Tabellen vorkommen, die derzeit synchronisiert werden. Wenn RI-Verletzungen für Tabellen vorhanden sind, die nicht synchronisiert werden, wird dieser Hook nicht aufgerufen, und die Synchronisation schlägt fehl.

Dieser Hook wird über die gleiche Verbindung aufgerufen, die auch dbmsync für den Download verwendet. Der Hook darf keine expliziten oder impliziten Festschreibungen enthalten, da diese zur Inkonsistenz der Daten in der Datenbank führen können. Die Aktionen dieses Hooks werden festgeschrieben bzw. zurückgesetzt, wenn der Download festgeschrieben bzw. zurückgesetzt wird.

Im Unterschied zu anderen Hook-Aktionen werden die Vorgänge, die während dieses Hooks ausgeführt werden, bei der nächsten Synchronisation nicht in den Upload einbezogen.

Privilegien

Hook-Prozeduren können von jedem Benutzer mit dem MANAGE REPLICATION-Systemprivileg erstellt werden. Um jedoch zu gewährleisten, dass ein Hook auf die #hook_dict-Tabelle zugreifen kann, die verwendet wird, um Daten an und aus Hooks zu übergeben, müssen Hooks eine der folgenden Anforderungen erfüllen:

- Eigentümer muss ein Benutzer sein, der die Systemprivilegien SELECT ANY TABLE und UPDATE ANY TABLE hat.
- Sie müssen durch die SQL SECURITY INVOKER-Klausel der CREATE PROCEDURE-Anweisung definiert sein.

Siehe auch

- [„sp_hook_dbmsync_download_log_ri_violation“ auf Seite 224](#)

Beispiel

Dieses Beispiel verwendet die unten stehenden Department- und Employee-Tabellen:

```
CREATE TABLE Department(  
  "department_id" INTEGER primary key  
);  
  
CREATE TABLE Employee(  
  "employee_id" INTEGER PRIMARY KEY,  
  "department_id" INTEGER,  
  FOREIGN KEY EMPLOYEE_FK1 (department_id) REFERENCES Department  
);
```

Die folgende sp_hook_dbmsync_download_ri_violation-Definition bereinigt Verletzungen der referenziellen Integrität zwischen den Department- und Employee-Tabellen. Sie überprüft den Rollennamen für den Fremdschlüssel und fügt fehlende department_id-Werte in die Department-Tabelle ein.

```
CREATE PROCEDURE sp_hook_dbmsync_download_ri_violation()  
BEGIN  
  
IF EXISTS (SELECT * FROM #hook_dict WHERE name = 'role name'  
  AND value = 'EMPLOYEE_FK1') THEN  
  
  -- update the Department table with missing department_id values  
  INSERT INTO Department  
    SELECT distinct department_id FROM Employee  
    WHERE department_id NOT IN (SELECT department_id FROM Department)  
  
END IF;  
END;
```

sp_hook_dbmlsync_download_table_begin

Verwenden Sie diese gespeicherte Prozedur, um benutzerdefinierte Aktionen direkt vor dem Download jeder Tabelle einzufügen.

Zeilen in der Tabelle #hook_dict

Name	Wert	Beschreibung
table name (in)	Tabellenname	Die Tabelle, in der Vorgänge übernommen werden
publication_ <i>n</i> (in)	Publikation	Nicht mehr empfohlen. Verwenden Sie stattdessen subscription_ <i>n</i> . Die synchronisierten Publikationen, wobei <i>n</i> eine Ganzzahl ist. Es gibt einen publication_ <i>n</i> -Eintrag für jede synchronisierte Publikation. Die Nummerierung von <i>n</i> beginnt bei Null.
MobiLink user (in)	MobiLink-Benutzername	Der MobiLink-Benutzer, für den Sie synchronisieren
script version (in)	Skriptversionsname	Die MobiLink-Skriptversion, die für die Synchronisation verwendet werden soll
subscription_ <i>n</i> (in)	Subskriptionsname(<i>n</i>)	Die synchronisierten Subskriptionen, wobei <i>n</i> eine Ganzzahl ist. Dies ist ein subscription_ <i>n</i> -Eintrag für jede synchronisierte Subskription. Die Nummerierung von <i>n</i> beginnt bei Null.

Bemerkungen

Wenn eine Prozedur mit diesem Namen existiert, wird sie für jede Tabelle unmittelbar vor dem Punkt aufgerufen, an dem die aus dem Download kommenden Vorgänge in dieser Tabelle übernommen werden. Die Aktionen dieser Prozedur werden festgeschrieben bzw. zurückgesetzt, wenn der Download festgeschrieben bzw. zurückgesetzt wird.

Privilegien

Hook-Prozeduren können von jedem Benutzer mit dem MANAGE REPLICATION-Systemprivileg erstellt werden. Um jedoch zu gewährleisten, dass ein Hook auf die #hook_dict-Tabelle zugreifen kann, die verwendet wird, um Daten an und aus Hooks zu übergeben, müssen Hooks eine der folgenden Anforderungen erfüllen:

- Eigentümer muss ein Benutzer sein, der die Systemprivilegien SELECT ANY TABLE und UPDATE ANY TABLE hat.
- Sie müssen durch die SQL SECURITY INVOKER-Klausel der CREATE PROCEDURE-Anweisung definiert sein.

Siehe auch

- [„Hook-Sequenz bei der Synchronisation“ auf Seite 203](#)

Beispiele

Nehmen wir an, Sie verwenden die folgende Tabelle, um Synchronisationsereignisse in der entfernten Datenbank zu protokollieren.

```
CREATE TABLE SyncLog
(
    "event_id"            integer NOT NULL DEFAULT autoincrement ,
    "event_time"          timestamp NULL,
    "event_name"          varchar(128) NOT NULL ,
    "subs"                varchar(1024) NULL ,
    PRIMARY KEY ("event_id")
)
```

Folgendes zeichnet den Download-Beginn der einzelnen Tabellen für jede Synchronisation auf.

```
CREATE PROCEDURE sp_hook_dbmlsync_download_table_begin ()
BEGIN

    DECLARE subs_list VARCHAR(1024);

    -- build a list of subscriptions being synchronized
    SELECT LIST(value) INTO subs_list
    FROM #hook_dict
    WHERE name LIKE 'subscription_%';

    -- log the event
    INSERT INTO SyncLog(event_time, event_name, subs)
    VALUES( CURRENT TIMESTAMP, 'sp_hook_dbmlsync_download_table_begin',
subs_list );
END
```

sp_hook_dbmlsync_download_table_end

Verwenden Sie diese gespeicherte Prozedur, um benutzerdefinierte Aktionen direkt nach dem Download jeder Tabelle einzufügen.

Zeilen in der Tabelle #hook_dict

Name	Wert	Beschreibung
table name (in)	Tabellenname	Die Tabelle, in der Vorgänge gerade übernommen wurden
delete count (in)	Anzahl von Zeilen	Die Anzahl der Zeilen in dieser Tabelle, die durch den Download gelöscht werden
upsert count (in)	Anzahl von Zeilen	Die Anzahl der Zeilen in dieser Tabelle, die durch den Download aktualisiert oder eingefügt werden

Name	Wert	Beschreibung
publication_ <i>n</i> (in)	Publikation	Nicht mehr empfohlen. Verwenden Sie stattdessen subscription_ <i>n</i> . Die synchronisierten Publikationen, wobei <i>n</i> eine Ganzzahl ist. Es gibt einen publication_ <i>n</i> -Eintrag für jede Publikation der Synchronisation. Die Nummerierung von <i>n</i> beginnt bei Null.
MobiLink user (in)	MobiLink-Benutzername	Der MobiLink-Benutzer, für den Sie synchronisieren
script version (in)	Skriptversionsname	Die MobiLink-Skriptversion, die für die Synchronisation verwendet werden soll
subscription_ <i>n</i> (in)	Subskriptionsname(<i>n</i>)	Die synchronisierten Subskriptionen, wobei <i>n</i> eine Ganzzahl ist. Dies ist ein subscription_ <i>n</i> -Eintrag für jede synchronisierte Subskription. Die Nummerierung von <i>n</i> beginnt bei Null.

Bemerkungen

Wenn eine Prozedur mit diesem Namen besteht, wird sie unmittelbar nach dem Punkt aufgerufen, an dem alle Vorgänge im Download für eine Tabelle übernommen wurden.

Die Aktionen dieser Prozedur werden festgeschrieben bzw. zurückgesetzt, wenn der Download festgeschrieben bzw. zurückgesetzt wird.

Privilegien

Hook-Prozeduren können von jedem Benutzer mit dem MANAGE REPLICATION-Systemprivileg erstellt werden. Um jedoch zu gewährleisten, dass ein Hook auf die #hook_dict-Tabelle zugreifen kann, die verwendet wird, um Daten an und aus Hooks zu übergeben, müssen Hooks eine der folgenden Anforderungen erfüllen:

- Eigentümer muss ein Benutzer sein, der die Systemprivilegien SELECT ANY TABLE und UPDATE ANY TABLE hat.
- Sie müssen durch die SQL SECURITY INVOKER-Klausel der CREATE PROCEDURE-Anweisung definiert sein.

Siehe auch

- [„Hook-Sequenz bei der Synchronisation“ auf Seite 203](#)

Beispiele

Nehmen wir an, Sie verwenden die folgende Tabelle, um Synchronisationsereignisse in der entfernten Datenbank zu protokollieren.

```
CREATE TABLE SyncLog
(
```

```
"event_id"          integer NOT NULL DEFAULT autoincrement ,
"event_time"        timestamp NULL,
"event_name"        varchar(128) NOT NULL ,
"subs"              varchar(1024) NULL ,
PRIMARY KEY ("event_id")
)
```

Folgendes zeichnet das Download-Ende der einzelnen Tabellen für jede Synchronisation auf.

```
CREATE PROCEDURE sp_hook_dbmlsync_download_table_end ()
BEGIN

    DECLARE subs_list VARCHAR(1024);

    -- build a list of subscriptions being synchronized
    SELECT LIST(value) INTO subs_list
    FROM #hook_dict
    WHERE name LIKE 'subscription_%';

    -- log the event
    INSERT INTO SyncLog(event_time, event_name, subs)
    VALUES( CURRENT_TIMESTAMP, 'sp_hook_dbmlsync_download_table_end',
subs_list );
END
```

sp_hook_dbmlsync_end

Verwenden Sie diese gespeicherte Prozedur, um benutzerdefinierte Aktionen direkt vor dem Abschluss der Synchronisation einzufügen.

Zeilen in der Tabelle #hook_dict

Name	Wert	Beschreibung
restart (out)	sync download none	<p>Beim Wert sync versucht dbmlsync, die soeben abgeschlossene Synchronisation erneut auszuführen. Der Wert sync ersetzt true, der identisch ist, aber nicht mehr empfohlen wird.</p> <p>Beim Wert none (Standardwert) fährt dbmlsync herunter oder führt einen Neustart durch, je nach seinen Befehlszeilenargumenten. Der Wert none ersetzt false, der identisch ist, aber nicht mehr empfohlen wird.</p> <p>Wenn der Wert download und der Parameter für den neu startbaren Download true ist, versucht dbmlsync, den soeben fehlgeschlagenen Download erneut auszuführen.</p>

Name	Wert	Beschreibung
exit code (in)	Zahl	Der Beendigungscode für die gerade abgeschlossene Synchronisation. Ein anderer Wert als Null steht für einen Synchronisationsfehler.
publication_ <i>n</i> (in)	Publikation	Nicht mehr empfohlen. Verwenden Sie stattdessen subscription_ <i>n</i> . Die synchronisierten Publikationen, wobei <i>n</i> eine Ganzzahl ist. Es gibt einen publication_ <i>n</i> -Eintrag für jede synchronisierte Publikation. Die Nummerierung von <i>n</i> beginnt bei Null.
MobiLink user (in)	MobiLink-Benutzername	Der MobiLink-Benutzer, für den Sie synchronisieren
upload status (in)	not sent committed failed unknown	<p>Gibt den Status an, der vom MobiLink-Server zurückgegeben wird, wenn dbmlsync versucht, den Empfang des Uploads zu prüfen. Der Status kann sein:</p> <ul style="list-style-type: none"> • not sent - Es wurde kein Upload an den MobiLink-Server gesendet, da dies entweder durch einen Fehler verhindert wurde oder für die angeforderte Synchronisation nicht nötig war. Dies kann während einer Synchronisation mit reinem Download, einem erneut gestarteten Download oder einem dateibasierten Download auftreten. • committed - Der Upload wurde vom MobiLink-Server empfangen und festgeschrieben. • failed - Der MobiLink-Server hat den Upload nicht festgeschrieben. Bei einem transaktionalen Upload ist der Upload-Status 'failed', wenn nur einige der Transaktionen erfolgreich hochgeladen und vom Server akzeptiert wurden. • unknown - Der Upload wurde vom MobiLink-Server nicht bestätigt. Es gibt keine Möglichkeit zu erkennen, ob der Upload festgeschrieben wurde oder nicht.
script version (in)	Skriptversionsname	Die MobiLink-Skriptversion, die für die Synchronisation verwendet werden soll

Name	Wert	Beschreibung
restartable download (in)	true false	Wenn der Wert true lautet, ist der Download für die aktuelle Synchronisation fehlgeschlagen und kann neu gestartet werden. Wenn der Wert false ist, war der Download erfolgreich und kann nicht neu gestartet werden.
restartable download size (in)	Ganzzahl	Wenn der Parameter für einen neu startbaren Download true ist, zeigt dieser Parameter die Anzahl der Byte an, die empfangen wurden, bevor der Download fehlgeschlagen ist. Ist der neu startbare Download false , dann ist dieser Wert ohne Bedeutung.
error hook user state (in)	Ganzzahl	Dieser Wert enthält Informationen über Fehler und kann von folgenden Hooks gesendet werden: sp_hook_dbmlsync_all_error, sp_hook_dbmlsync_communication_error, sp_hook_dbmlsync_misc_error und sp_hook_dbmlsync_sql_error.
subscription_ <i>n</i> (in)	Subskriptionsname(<i>n</i>)	Die synchronisierten Subskriptionen, wobei <i>n</i> eine Ganzzahl ist. Dies ist ein subscription_ <i>n</i> -Eintrag für jede synchronisierte Subskription. Die Nummerierung von <i>n</i> beginnt bei Null.

Bemerkungen

Wenn eine Prozedur dieses Namens vorhanden ist, wird sie am Ende jeder Synchronisation aufgerufen.

Wenn ein sp_hook_dbmlsync_end-Hook so festgelegt ist, dass der Hook den Restart-Parameter immer auf **sync** setzt und Sie mehrfache Subskriptionen in der dbmlsync-Befehlszeile in der Form '-s sub1, -s sub2' usw. eingeben, dann synchronisiert dbmlsync die erste Publikation wiederholt und die zweite nie.

Die Aktionen dieser Prozedur werden sofort nach ihrer Ausführung festgeschrieben.

Privilegien

Hook-Prozeduren können von jedem Benutzer mit dem MANAGE REPLICATION-Systemprivileg erstellt werden. Um jedoch zu gewährleisten, dass ein Hook auf die #hook_dict-Tabelle zugreifen kann, die verwendet wird, um Daten an und aus Hooks zu übergeben, müssen Hooks eine der folgenden Anforderungen erfüllen:

- Eigentümer muss ein Benutzer sein, der die Systemprivilegien SELECT ANY TABLE und UPDATE ANY TABLE hat.
- Sie müssen durch die SQL SECURITY INVOKER-Klausel der CREATE PROCEDURE-Anweisung definiert sein.

Siehe auch

- „Ereignis-Hooks für SQL Anywhere-Clients“ auf Seite 202
- „Hook-Sequenz bei der Synchronisation“ auf Seite 203
- „Wiederaufnahme fehlgeschlagener Downloads“ [*MobiLink - Serveradministration*]
- „Behandlung von Fehlern und Warnungen in Ereignis-Hook-Prozeduren“ auf Seite 207

Beispiele

Im folgenden Beispiel wird der Download manuell neu gestartet, wenn er für die aktuelle Synchronisation fehlgeschlagen ist und neu gestartet werden kann.

```
CREATE PROCEDURE sp_hook_dbmlsync_end()
BEGIN
  -- Restart the download if the download for the current sync
  -- failed and can be restarted
  IF EXISTS (SELECT * FROM #hook_dict
    WHERE name = 'restartable download' AND value='true')
    THEN
      UPDATE #hook_dict SET value ='download' WHERE name='restart';
    END IF;
END;
```

sp_hook_dbmlsync_log_rescan

Verwenden Sie diese gespeicherte Prozedur, um programmgesteuert zu entscheiden, wann ein Rescan erforderlich ist.

Zeilen in der Tabelle #hook_dict

Name	Wert	Beschreibung
publication_ <i>n</i> (in)	Publikation	Nicht mehr empfohlen. Verwenden Sie stattdessen subscription_ <i>n</i> . Die synchronisierten Publikationen, wobei <i>n</i> eine Ganzzahl ist. Es gibt einen publication_ <i>n</i> -Eintrag für jede synchronisierte Publikation. Die Nummerierung von <i>n</i> beginnt bei Null.
MobiLink user (in)	MobiLink-Benutzername	Der MobiLink-Benutzer, für den Sie synchronisieren
discarded storage (in)	Zahl	Die Anzahl an Byte von verworfenem Speicher nach der letzten Synchronisation
rescan (in out)	true false	Wenn vom Hook auf TRUE gesetzt, führt dbmlsync einen kompletten Rescan vor der nächsten Synchronisation durch. Beim Eintritt ist dieser Wert auf FALSE gesetzt.

Name	Wert	Beschreibung
script version (in)	Skriptversionsname	Die MobiLink-Skriptversion, die für die Synchronisation verwendet werden soll
subscription_ <i>n</i> (in)	Subskriptionsname(<i>n</i>)	Die synchronisierten Subskriptionen, wobei <i>n</i> eine Ganzzahl ist. Dies ist ein subscription_ <i>n</i> -Eintrag für jede synchronisierte Subskription. Die Nummerierung von <i>n</i> beginnt bei Null.

Bemerkungen

Wenn mehr als eine Option -n oder -s in der Befehlszeile angegeben wird, können in dbmsync Fragmentierungen auftreten, sodass Speicherinhalt gelöscht wird. Der gelöschte Speicherinhalt kann durch erneutes Scannen des Datenbank-Transaktionslogs wiederhergestellt werden. Mit diesem Hook können Sie festlegen, ob dbmsync das Datenbank-Transaktionslog neu scannen soll, um den Speicher wiederherzustellen.

Wenn keine andere Bedingung erfüllt ist, die einen Rescan erzwingt, wird dieser Hook sofort nach dem sp_hook_dbmsync_process_exit_code-Hook aufgerufen.

Privilegien

Hook-Prozeduren können von jedem Benutzer mit dem MANAGE REPLICATION-Systemprivileg erstellt werden. Um jedoch zu gewährleisten, dass ein Hook auf die #hook_dict-Tabelle zugreifen kann, die verwendet wird, um Daten an und aus Hooks zu übergeben, müssen Hooks eine der folgenden Anforderungen erfüllen:

- Eigentümer muss ein Benutzer sein, der die Systemprivilegien SELECT ANY TABLE und UPDATE ANY TABLE hat.
- Sie müssen durch die SQL SECURITY INVOKER-Klausel der CREATE PROCEDURE-Anweisung definiert sein.

Siehe auch

- [„Erweiterte Option HoverRescanThreshold \(hrt\)“ auf Seite 157](#)

Beispiele

Das folgende Beispiel bewirkt einen Log-Scan, wenn der verworfene Speicher größer als 1000 Byte ist.

```
CREATE PROCEDURE sp_hook_dbmsync_log_rescan ()
BEGIN
  IF EXISTS(SELECT * FROM #hook_dict
    WHERE name = 'Discarded storage' AND value>1000)
  THEN
    UPDATE #hook_dict SET value = 'true' WHERE name='Rescan';
  END IF;
END;
```

sp_hook_dbmlsync_logscan_begin

Verwenden Sie diese gespeicherte Prozedur, um benutzerdefinierte Aktionen direkt vor dem Durchsuchen des Transaktionslogs für den Upload einzufügen.

Zeilen in der Tabelle #hook_dict

Name	Wert	Beschreibung
starting log offset_ <i>n</i> (in)	Zahl	Der Wert des Verarbeitungsfortschritts für jede synchronisierte Subskription. Der Wert des Verarbeitungsfortschritts ist der Offset im Transaktionslog, bis zu dem alle Daten für die Subskription per Upload übertragen wurden. Es gibt einen Wert für jede Subskription, die synchronisiert wird. Die Nummerierung von <i>n</i> beginnt bei Null. Dieser Wert entspricht subscription_ <i>n</i> . Beispiel: "log offset_0" ist der Offset für 'subscription_0'.
log scan retry (in)	true false	Wenn das Transaktionslog zum ersten Mal für diese Synchronisation durchsucht wurde, ist der Wert FALSE, andernfalls ist er TRUE. Das Log wird zweimal durchsucht, wenn der MobiLink-Server und dbmlsync unterschiedliche Informationen darüber haben, wo der Durchsuchungsvorgang beginnen soll.
publication_ <i>n</i> (in)	Publikation	Nicht mehr empfohlen. Verwenden Sie stattdessen subscription_ <i>n</i> . Die synchronisierten Publikationen, wobei <i>n</i> eine Ganzzahl ist. Es gibt einen publication_ <i>n</i> -Eintrag für jede synchronisierte Publikation. Die Nummerierung von <i>n</i> beginnt bei Null.
MobiLink user (in)	MobiLink-Benutzername	Der MobiLink-Benutzer, für den Sie synchronisieren
script version (in)	Skriptversionsname	Die MobiLink-Skriptversion, die für die Synchronisation verwendet werden soll
subscription_ <i>n</i> (in)	Subskriptionsname(<i>n</i>)	Die synchronisierten Subskriptionen, wobei <i>n</i> eine Ganzzahl ist. Dies ist ein subscription_ <i>n</i> -Eintrag für jede synchronisierte Subskription. Die Nummerierung von <i>n</i> beginnt bei Null.

Bemerkungen

Wenn eine Prozedur dieses Namens vorhanden ist, wird sie aufgerufen, bevor dbmlsync das Transaktionslog durchsucht, um den Upload zusammenzustellen.

Dieser Hook ist die ideale Lösung, um Änderungen in letzter Minute an den zu synchronisierenden Tabellen vorzunehmen, die in den Upload einbezogen werden sollen.

Die Aktionen dieser Prozedur werden sofort nach ihrer Ausführung festgeschrieben.

Privilegien

Hook-Prozeduren können von jedem Benutzer mit dem MANAGE REPLICATION-Systemprivileg erstellt werden. Um jedoch zu gewährleisten, dass ein Hook auf die #hook_dict-Tabelle zugreifen kann, die verwendet wird, um Daten an und aus Hooks zu übergeben, müssen Hooks eine der folgenden Anforderungen erfüllen:

- Eigentümer muss ein Benutzer sein, der die Systemprivilegien SELECT ANY TABLE und UPDATE ANY TABLE hat.
- Sie müssen durch die SQL SECURITY INVOKER-Klausel der CREATE PROCEDURE-Anweisung definiert sein.

Siehe auch

- [„Hook-Sequenz bei der Synchronisation“ auf Seite 203](#)

Beispiele

Nehmen wir an, Sie verwenden die folgende Tabelle, um Synchronisationsereignisse in der entfernten Datenbank zu protokollieren.

```
CREATE TABLE SyncLog
(
    "event_id"          integer NOT NULL DEFAULT autoincrement ,
    "event_time"        timestamp NULL,
    "event_name"        varchar(128) NOT NULL ,
    "subs"              varchar(1024) NULL ,
    PRIMARY KEY ("event_id")
)
```

Folgendes zeichnet den Beginn des Log-Scans für jede Synchronisation auf.

```
CREATE PROCEDURE sp_hook_dbmlsync_logscan_begin ()
BEGIN

    DECLARE subs_list VARCHAR(1024);

    -- build a list of subscriptions being synchronized
    SELECT LIST(value) INTO subs_list
    FROM #hook_dict
    WHERE name LIKE 'subscription_%';

    -- log the event
    INSERT INTO SyncLog(event_time, event_name, subs)
    VALUES( CURRENT_TIMESTAMP, 'sp_hook_dbmlsync_logscan_begin', subs_list );
END
```

sp_hook_dbmlsync_logscan_end

Verwenden Sie diese gespeicherte Prozedur, um benutzerdefinierte Aktionen direkt nach dem Durchsuchen des Transaktionslogs einzufügen.

Zeilen in der Tabelle #hook_dict

Name	Wert	Beschreibung
ending log offset (in)	Zahl	Der Log-Offset-Wert, bei dem der Scan beendet wurde
starting log offset_ <i>n</i> (in)	Zahl	Der Ausgangswert des Verarbeitungsfortschritts für jede Subskription, die Sie synchronisieren. Die <i>n</i> -Werte entsprechen jenen in publication_ <i>n</i> . Beispiel: "Starting log offset_1" ist der Offset für 'publication_1'.
log scan retry (in)	true false	Wenn das Transaktionslog zum ersten Mal für diese Synchronisation durchsucht wurde, ist der Wert FALSE, andernfalls ist er TRUE. Das Log wird zweimal durchsucht, wenn der MobiLink-Server und dbmlsync unterschiedliche Informationen darüber haben, wo der Durchsuchungsvorgang beginnen soll.
publication_ <i>n</i> (in)	Publikation	Nicht mehr empfohlen. Verwenden Sie stattdessen subscription_ <i>n</i> . Die synchronisierten Publikationen, wobei <i>n</i> eine Ganzzahl ist. Es gibt einen publication_ <i>n</i> -Eintrag für jede synchronisierte Publikation. Die Nummerierung von <i>n</i> beginnt bei Null.
MobiLink user (in)	MobiLink-Benutzername	Der MobiLink-Benutzer, für den Sie synchronisieren
script version (in)	Skriptversionsname	Die MobiLink-Skriptversion, die für die Synchronisation verwendet werden soll
subscription_ <i>n</i> (in)	Subskriptionsname(<i>n</i>)	Die synchronisierten Subskriptionen, wobei <i>n</i> eine Ganzzahl ist. Dies ist ein subscription_ <i>n</i> -Eintrag für jede synchronisierte Subskription. Die Nummerierung von <i>n</i> beginnt bei Null.

Bemerkungen

Wenn eine Prozedur dieses Namens vorhanden ist, wird sie sofort aufgerufen, nachdem dbmlsync das Transaktionslog durchsucht hat.

Die Aktionen dieser Prozedur werden sofort nach ihrer Ausführung festgeschrieben.

Privilegien

Hook-Prozeduren können von jedem Benutzer mit dem MANAGE REPLICATION-Systemprivileg erstellt werden. Um jedoch zu gewährleisten, dass ein Hook auf die #hook_dict-Tabelle zugreifen kann, die verwendet wird, um Daten an und aus Hooks zu übergeben, müssen Hooks eine der folgenden Anforderungen erfüllen:

- Eigentümer muss ein Benutzer sein, der die Systemprivilegien SELECT ANY TABLE und UPDATE ANY TABLE hat.
- Sie müssen durch die SQL SECURITY INVOKER-Klausel der CREATE PROCEDURE-Anweisung definiert sein.

Siehe auch

- [„Hook-Sequenz bei der Synchronisation“ auf Seite 203](#)

Beispiele

Nehmen wir an, Sie verwenden die folgende Tabelle, um Synchronisationsereignisse in der entfernten Datenbank zu protokollieren.

```
CREATE TABLE SyncLog
(
    "event_id"          integer NOT NULL DEFAULT autoincrement ,
    "event_time"        timestamp NULL,
    "event_name"        varchar(128) NOT NULL ,
    "subs"              varchar(1024) NULL ,
    PRIMARY KEY ("event_id")
)
```

Folgendes zeichnet das Ende des Log-Scans für jede Synchronisation auf.

```
CREATE PROCEDURE sp_hook_dbmlsync_logscan_end ()
BEGIN

    DECLARE subs_list VARCHAR(1024);

    -- build a list of subscriptions being synchronized
    SELECT LIST(value) INTO subs_list
    FROM #hook_dict
    WHERE name LIKE 'subscription_%';

    -- log the event
    INSERT INTO SyncLog(event_time, event_name, subs)
    VALUES( CURRENT TIMESTAMP, 'sp_hook_dbmlsync_logscan_end', subs_list );
END
```

sp_hook_dbmlsync_misc_error

Mit dieser gespeicherten Prozedur können dbmlsync-Fehler verarbeitet werden, die nicht als Datenbank- oder Verbindungsfehler erkannt werden. Sie können beispielsweise den Hook sp_hook_dbmlsync_misc_error einführen, um Fehler zu protokollieren oder beim Eintreten eines bestimmten Fehlers eine bestimmte Aktion durchzuführen.

Zeilen in der Tabelle #hook_dict

Name	Wert	Beschreibung
publication_ <i>n</i> (in)	Publikation	Nicht mehr empfohlen. Verwenden Sie stattdessen subscription_ <i>n</i> . Die synchronisierten Publikationen, wobei <i>n</i> eine Ganzzahl ist. Es gibt einen publication_ <i>n</i> -Eintrag für jede synchronisierte Publikation. Die Nummerierung von <i>n</i> beginnt bei Null.
MobiLink user (in)	MobiLink-Benutzername	Der MobiLink-Benutzer, für den Sie synchronisieren
script version (in)	Skriptversionsname	Die MobiLink-Skriptversion, die für die Synchronisation verwendet werden soll
error message (in)	Fehlermeldungstext	Dies ist der Text, der auch im dbmlsync-Log angezeigt wird.
error id (in)	Ganzzahl	Eine ID, die die Meldung eindeutig identifiziert. In dieser Zeile können Sie die Fehlermeldung identifizieren, da der Meldungstext abweichen kann.
error hook user state (in out)	Ganzzahl	Dieser Wert kann vom Hook eingestellt werden, um Statusinformationen in zukünftigen Aufrufen an die Hooks sp_hook_dbmlsync_all_error, sp_hook_dbmlsync_communication_error, sp_hook_dbmlsync_misc_error, sp_hook_dbmlsync_sql_error oder sp_hook_dbmlsync_end zu übergeben. Wenn einer dieser Hooks zum ersten Mal aufgerufen wird, ist der Wert der Zeile 0. Wenn ein Hook den Wert der Zeile ändert, wird der neue Wert im nächsten Hook-Aufruf verwendet.
subscription_ <i>n</i> (in)	Subskriptionsname(<i>n</i>)	Die synchronisierten Subskriptionen, wobei <i>n</i> eine Ganzzahl ist. Dies ist ein subscription_ <i>n</i> -Eintrag für jede synchronisierte Subskription. Die Nummerierung von <i>n</i> beginnt bei Null.

Bemerkungen

Wenn beim Start ein Fehler auftritt, bevor eine Synchronisation initiiert werden konnte, werden die #hook_dict-Einträge für den MobiLink-Benutzer und die Skriptversion auf eine leere Zeichenfolge

zurückgesetzt, und in der #hook_dict-Tabelle werden keine publication_*n*- oder subscription_*n*-Zeilen eingestellt.

Die Zeile "error hook user state" bietet ein nützliches Verfahren für die Übergabe der Informationen über die Art des Fehlers an den Hook sp_hook_dbmlsync_end. Anhand dieser Informationen können Sie entscheiden, ob die Synchronisation erneut ausgeführt werden soll.

Diese Prozedur wird in einer eigenen Verbindung ausgeführt, damit sichergestellt ist, dass die dadurch bewirkten Vorgänge nicht verloren gehen, wenn ein Zurücksetzen der Synchronisationsverbindung erfolgt. Wenn dbmlsync keine getrennte Verbindung herstellen kann, wird die Prozedur nicht aufgerufen.

Da dieser Hook in einer eigenen Verbindung ausgeführt wird, sollten Sie beim Zugriff auf Tabellen vorsichtig sein, die in Ihrer Hook-Prozedur synchronisiert werden, denn diese Tabellen werden von dbmlsync möglicherweise gesperrt. Diese Sperren können zum Fehlschlagen von Vorgängen im Hook oder zu unbegrenzten Wartezeiten führen.

Die Aktionen dieser Prozedur werden sofort nach ihrer Ausführung festgeschrieben.

Privilegien

Hook-Prozeduren können von jedem Benutzer mit dem MANAGE REPLICATION-Systemprivileg erstellt werden. Um jedoch zu gewährleisten, dass ein Hook auf die #hook_dict-Tabelle zugreifen kann, die verwendet wird, um Daten an und aus Hooks zu übergeben, müssen Hooks eine der folgenden Anforderungen erfüllen:

- Eigentümer muss ein Benutzer sein, der die Systemprivilegien SELECT ANY TABLE und UPDATE ANY TABLE hat.
- Sie müssen durch die SQL SECURITY INVOKER-Klausel der CREATE PROCEDURE-Anweisung definiert sein.

Siehe auch

- „Behandlung von Fehlern und Warnungen in Ereignis-Hook-Prozeduren“ auf Seite 207
- „sp_hook_dbmlsync_communication_error“ auf Seite 215
- „sp_hook_dbmlsync_all_error“ auf Seite 211
- „sp_hook_dbmlsync_sql_error“ auf Seite 257

Beispiele

Nehmen wir an, Sie verwenden die folgende Tabelle, um Fehler in der entfernten Datenbank zu protokollieren.

```
CREATE TABLE error_log
(
  pk INTEGER DEFAULT AUTOINCREMENT PRIMARY KEY,
  err_id INTEGER,
  err_msg VARCHAR(10240),
);
```

Im folgenden Beispiel werden alle Fehlermeldungen mithilfe von sp_hook_dbmlsync_misc_error protokolliert.

```
CREATE PROCEDURE sp_hook_dbmlsync_misc_error()
BEGIN
```

```

DECLARE msg VARCHAR(10240);
DECLARE id INTEGER;

// get the error message text
SELECT value INTO msg
  FROM #hook_dict
 WHERE name = 'error message';

// get the error id
SELECT value INTO id
  FROM #hook_dict
 WHERE name = 'error id';

// log the error information
INSERT INTO error_log(err_msg,err_id)
  VALUES (msg,id);
END;

```

Um mögliche Fehler-IDs zu ermitteln, sollten Sie einen Testdurchlauf von dbmlsync durchführen. Die folgende dbmlsync-Befehlszeile bezieht sich beispielsweise auf eine ungültige Subskription:

```
dbmlsync -c SERVER=custdb;UID=DBA;PWD=sql -s test
```

Nun enthält die error_log-Tabelle die folgende Zeile, die sich auf den Fehler mit der Fehler-ID 9931 bezieht:

```

1,19912,
'Subscription ''test'' not found.'

```

Um eine benutzerspezifische Fehlerbehandlung zu ermöglichen, überprüfen Sie die Fehler-ID 19912 in sp_hook_dbmlsync_misc_error.

```

ALTER PROCEDURE sp_hook_dbmlsync_misc_error()
BEGIN
  DECLARE msg VARCHAR(10240);
  DECLARE id INTEGER;

  // get the error message text
  SELECT value INTO msg
    FROM #hook_dict
   WHERE name = 'error message';

  // get the error id
  SELECT value INTO id
    FROM #hook_dict
   WHERE name = 'error id';

  // log the error information
  INSERT INTO error_log(err_msg,err_id)
    VALUES (msg,id);

  IF id = 19912 THEN
    // handle invalid subscription
  END IF;
END;

```

sp_hook_dbmlsync_ml_connect_failed

Verwenden Sie diese gespeicherte Prozedur, um nach einer gescheiterten Verbindungsaufnahme mit dem MobiLink-Server einen neuen Versuch zu starten, indem Sie einen anderen Verbindungstyp oder eine andere Adresse benutzen.

Zeilen in der Tabelle #hook_dict

Name	Wert	Beschreibung
publication_ <i>n</i> (in)	Publikation	Nicht mehr empfohlen. Verwenden Sie stattdessen subscription_ <i>n</i> . Die synchronisierten Publikationen, wobei <i>n</i> eine Ganzzahl ist. Es gibt einen publication_ <i>n</i> -Eintrag für jede synchronisierte Publikation. Die Nummerierung von <i>n</i> beginnt bei Null.
MobiLink user (in)	MobiLink-Benutzername	Der MobiLink-Benutzer, für den Sie synchronisieren
script version (in)	Skriptversionsname	Die MobiLink-Skriptversion, die für die Synchronisation verwendet werden soll
connection address (in out)	Verbindungsadresse	Wenn der Hook aufgerufen wird, ist dies die Adresse, die beim letzten gescheiterten Verbindungsversuch verwendet wurde. Sie können diesen Wert auf eine neue Verbindungsadresse einstellen. Wenn "retry" auf TRUE gesetzt ist, wird dieser Wert beim nächsten Verbindungsversuch verwendet. Eine Liste der Protokolloptionen finden Sie unter „ Netzwerkprotokolloptionen des MobiLink-Clients “ auf Seite 25.
connection type (in out)	Netzwerkprotokoll	Wenn der Hook aufgerufen wird, ist dies das Netzwerkprotokoll (z.B. TCPIP), das beim letzten gescheiterten Verbindungsversuch verwendet wurde. Sie können diesen Wert auf ein neues Netzwerkprotokoll einstellen. Wenn "retry" auf TRUE gesetzt ist, wird dieser Wert beim nächsten Verbindungsversuch verwendet. Eine Liste der Netzwerkprotokolle finden Sie unter „ Erweiterte Option CommunicationType (ctp) “ auf Seite 150.

Name	Wert	Beschreibung
user data (in out)	Benutzerdefinierte Daten	Statusinformationen, die verwendet werden, falls der nächste Verbindungsversuch scheitert. Es könnte zum Beispiel nützlich sein, die Anzahl der aufgetretenen Versuche zu speichern. Standardwert ist eine leere Zeichenfolge.
allow remote ahead (in out)	true false	Dies ist nur TRUE, wenn die dbmlsync-Option -ra oder die Synchronisationsprofiloption RemoteProgressGreater=on für diese Synchronisation angegeben wurde. Durch Ändern des Werts dieser Zeile können Sie den Wert der Option nur für die aktuelle Synchronisation ändern. Siehe „dbmlsync-Option -r“ auf Seite 132.
allow remote behind (in out)	true false	Dies ist nur TRUE, wenn die dbmlsync-Option -ra oder die Synchronisationsprofiloption RemoteProgressLess=on für diese Synchronisation angegeben wurde. Durch Ändern des Werts dieser Zeile können Sie den Wert der Option nur für die aktuelle Synchronisation ändern. Siehe „dbmlsync-Option -r“ auf Seite 132.
retry (in out)	true false	Setzen Sie diesen Wert auf TRUE, wenn Sie einen gescheiterten Verbindungsversuch wiederholen möchten. Standardwert ist FALSE.
subscription_ <i>n</i> (in)	Subskriptionsname(<i>n</i>)	Die synchronisierten Subskriptionen, wobei <i>n</i> eine Ganzzahl ist. Dies ist ein subscription_ <i>n</i> -Eintrag für jede synchronisierte Subskription. Die Nummerierung von <i>n</i> beginnt bei Null.

Bemerkungen

Wenn eine Prozedur dieses Namens existiert, wird sie aufgerufen, sobald dbmlsync beim Versuch scheitert, eine Verbindung mit dem MobiLink-Server herzustellen.

Dieser Hook ist nur bei Verbindungsversuchen mit dem MobiLink-Server, und nicht mit der Datenbank anwendbar.

Wenn Offsets für den Verarbeitungsfortschritt nicht übereinstimmen, trennt dbmlsync die Verbindung mit dem MobiLink-Server, um sie später wieder herzustellen. Bei dieser Art der erneuten Verbindungsaufnahme wird dieser Hook nicht aufgerufen, und ein Scheitern der Verbindung bewirkt, dass die Synchronisation fehlschlägt.

Die Aktionen dieser Prozedur werden sofort nach ihrer Ausführung festgeschrieben.

Privilegien

Hook-Prozeduren können von jedem Benutzer mit dem MANAGE REPLICATION-Systemprivileg erstellt werden. Um jedoch zu gewährleisten, dass ein Hook auf die #hook_dict-Tabelle zugreifen kann, die verwendet wird, um Daten an und aus Hooks zu übergeben, müssen Hooks eine der folgenden Anforderungen erfüllen:

- Eigentümer muss ein Benutzer sein, der die Systemprivilegien SELECT ANY TABLE und UPDATE ANY TABLE hat.
- Sie müssen durch die SQL SECURITY INVOKER-Klausel der CREATE PROCEDURE-Anweisung definiert sein.

Beispiele

Dieses Beispiel verwendet den sp_hook_dbmlsync_ml_connect_failed-Hook, um die Verbindungsaufnahme bis zu fünf Mal zu versuchen.

```
CREATE PROCEDURE sp_hook_dbmlsync_ml_connect_failed ()
BEGIN
    DECLARE idx integer;

    SELECT IF value = '' then 0 else cast(value as integer)endif
    INTO idx
    FROM #hook_dict
    WHERE name = 'user data';

    IF idx < 5 THEN
        UPDATE #hook_dict
        SET value = idx +1
        WHERE name = 'user data';

        UPDATE #hook_dict
        SET value = 'TRUE'
        WHERE name = 'retry';
    END IF;
END;
```

Das nächste Beispiel verwendet eine Tabelle mit Verbindungsinformationen. Wenn ein Verbindungsversuch fehlschlägt, probiert der Hook den nächsten Server auf der Liste.

```
CREATE TABLE conn_list (
    label    INTEGER PRIMARY KEY,
    addr    VARCHAR( 128 ),
    type    VARCHAR( 64 )
);
INSERT INTO conn_list
VALUES ( 1, 'host=server1;port=91', 'tcpip' );
INSERT INTO conn_list
VALUES ( 2, 'host=server2;port=92', 'http' );
INSERT INTO conn_list
VALUES ( 3, 'host=server3;port=93', 'tcpip' );
COMMIT;

CREATE PROCEDURE sp_hook_dbmlsync_ml_connect_failed ()
BEGIN
    DECLARE idx INTEGER;
```

```

DECLARE cnt INTEGER;

SELECT if value = ''then | else cast(value as integer)endif
  INTO idx
  FROM #hook_dict
  WHERE name = 'user data';

SELECT COUNT( label ) INTO cnt FROM conn_list;

IF idx <= cnt THEN
  UPDATE #hook_dict
    SET value = ( SELECT addr FROM conn_list WHERE label = idx )
    WHERE name = 'connection address';
  UPDATE #hook_dict
    SET value = (SELECT type FROM conn_list WHERE label=idx)
    WHERE name = 'connection type';

  UPDATE #hook_dict
    SET value = idx +1
    WHERE name = 'user data';

  UPDATE #hook_dict
    SET value = 'TRUE'
    WHERE name = 'retry';
END IF;
END;

```

sp_hook_dbmlsync_process_exit_code

Verwenden Sie diese gespeicherte Prozedur, um Beendigungscodes zu verwalten.

Zeilen in der Tabelle #hook_dict

Name	Wert	Beschreibung
publication_n (in)	Publikation	Nicht mehr empfohlen. Verwenden Sie stattdessen subscription_n. Die synchronisierten Publikationen, wobei n eine Ganzzahl ist. Es gibt einen publication_n-Eintrag für jede synchronisierte Publikation. Die Nummerierung von n beginnt bei Null.
MobiLink user (in)	MobiLink-Benutzername	Der MobiLink-Benutzer, für den Sie synchronisieren
fatal error (in)	true false	TRUE, wenn der Hook aufgrund eines Fehlers aufgerufen wird, der dbmlsync zum Abbruch bringt.
aborted synchronization (in)	true false	TRUE, wenn der Hook aufgrund einer Abbruchanforderung vom sp_hook_dbmlsync_abort-Hook aufgerufen wird

Name	Wert	Beschreibung
exit code (in)	Zahl	Der Beendigungscode des jüngsten Synchronisationsversuchs. 0 gibt an, dass die Synchronisation erfolgreich war. Jeder andere Wert gibt an, dass die Synchronisation fehlgeschlagen ist. Dieser Wert kann von <code>sp_hook_dbmlsync_abort</code> gesetzt werden, wenn dieser Hook benutzt wird, um die Synchronisation abubrechen.
last exit code (in)	Zahl	Entspricht dem Wert, der in der Zeile new exit code der Tabelle <code>#hook_dict</code> gespeichert ist, als der Hook zum letzten Mal aufgerufen wurde, oder "0", wenn dies der erste Aufruf des Hooks ist.
new exit code (in out)	Zahl	Der Beendigungscode, den Sie für den Prozess wählen. Wenn <code>dbmlsync</code> existiert, ist exit code der Wert, der in dieser Zeile durch den letzten Aufruf des Hooks gespeichert wurde. Der Wert muss -32768 bis 32767 sein.
script version (in)	Skriptversionsname	Die MobiLink-Skriptversion, die für die Synchronisation verwendet werden soll
subscription_ <i>n</i> (in)	Subskriptionsname(<i>n</i>)	Die synchronisierten Subskriptionen, wobei <i>n</i> eine Ganzzahl ist. Dies ist ein <code>subscription_ <i>n</i></code> -Eintrag für jede synchronisierte Subskription. Die Nummerierung von <i>n</i> beginnt bei Null.

Bemerkungen

Eine `dbmlsync`-Sitzung kann mehrere Synchronisationen durchführen, wenn Sie die Option `-n` oder `-s` mehr als einmal in der Befehlszeile angeben, die Zeitplanung benutzen oder den `restart`-Parameter in `sp_hook_dbmlsync_end` einsetzen. In diesen Fällen gilt: Wenn eine oder mehrere Synchronisationen fehlschlagen, zeigt der Standardbeendigungscode nicht an, welche fehlgeschlagen ist. Verwenden Sie diesen Hook, um den Beendigungscode für den `dbmlsync`-Prozess basierend auf den Beendigungscode aus den Synchronisationen zu definieren. Dieser Hook kann auch verwendet werden, um Beendigungscode zurückzugeben.

Wenn beim Start ein Fehler auftritt, bevor eine Synchronisation initiiert werden konnte, werden die `#hook_dict`-Einträge für den MobiLink-Benutzer und die Skriptversion auf eine leere Zeichenfolge zurückgesetzt, und in der `#hook_dict`-Tabelle werden keine `publication_ n`- oder `subscription_ n`-Zeilen eingestellt.

Privilegien

Hook-Prozeduren können von jedem Benutzer mit dem MANAGE REPLICATION-Systemprivileg erstellt werden. Um jedoch zu gewährleisten, dass ein Hook auf die #hook_dict-Tabelle zugreifen kann, die verwendet wird, um Daten an und aus Hooks zu übergeben, müssen Hooks eine der folgenden Anforderungen erfüllen:

- Eigentümer muss ein Benutzer sein, der die Systemprivilegien SELECT ANY TABLE und UPDATE ANY TABLE hat.
- Sie müssen durch die SQL SECURITY INVOKER-Klausel der CREATE PROCEDURE-Anweisung definiert sein.

Beispiel

Sie führen dbmlsync aus, um fünf Synchronisationen durchzuführen, und der Beendigungscode soll angeben, wie viele von diesen Synchronisationen fehlschlagen, wobei ein Beendigungscode "0" anzeigt, dass keine Fehlschläge vorlagen, und ein Code "1" anzeigt, dass eine Synchronisation fehlschlug, usw. Sie können dies erreichen, indem Sie den Hook sp_hook_dbmlsync_process_exit_code wie folgt definieren: In diesem Fall gilt: Wenn drei Synchronisationen fehlschlagen, lautet der Beendigungscode "3".

```
CREATE PROCEDURE sp_hook_dbmlsync_process_exit_code()
BEGIN
    DECLARE rc INTEGER;

    SELECT value INTO rc FROM #hook_dict WHERE name = 'exit code';
    IF rc <> 0 THEN
        SELECT value INTO rc FROM #hook_dict WHERE name = 'last exit code';
        UPDATE #hook_dict SET value = rc + 1 WHERE name = 'new exit code';
    END IF;
END;
```

Siehe auch

- „Hook-Sequenz bei der Synchronisation“ auf Seite 203
- „sp_hook_dbmlsync_abort“ auf Seite 208

sp_hook_dbmlsync_schema_upgrade

Verwenden Sie diese gespeicherte Prozedur, um ein SQL-Skript auszuführen, mit dem Sie Ihr Schema aktualisieren.

Zeilen in der Tabelle #hook_dict

Name	Wert	Beschreibung
publication_ <i>n</i> (in)	Publikation	Nicht mehr empfohlen. Verwenden Sie stattdessen subscription_ <i>n</i> . Die synchronisierten Publikationen, wobei <i>n</i> eine Ganzzahl ist. Es gibt einen publication_ <i>n</i> -Eintrag für jede synchronisierte Publikation. Die Nummerierung von <i>n</i> beginnt bei Null.
MobiLink user (in)	MobiLink-Benutzername	Der MobiLink-Benutzer, für den Sie synchronisieren
script version (in)	Skriptversionsname	Die Version des für die Synchronisation verwendeten Skripts.
drop hook (out)	never always on success	<p>Die Werte können sein:</p> <p>never - (Standard) sp_hook_dbmlsync_schema_upgrade-Hook nicht aus der Datenbank löschen</p> <p>always - Nach dem Versuch, den Hook auszuführen, den sp_hook_dbmlsync_schema_upgrade-Hook aus der Datenbank löschen</p> <p>on success - Wenn der Hook erfolgreich abläuft, sp_hook_dbmlsync_schema_upgrade-Hook aus der Datenbank löschen. "On success" ist mit "always" identisch, wenn die dbmlsync-Option -eh verwendet wird, die erweiterte dbmlsync-Option IgnoreHookErrors auf TRUE gesetzt ist oder die IgnoreHookErrors-Synchronisationsprofiloption auf "on" gesetzt ist.</p>
subscription_ <i>n</i> (in)	Subskriptionsname(<i>n</i>)	Die synchronisierten Subskriptionen, wobei <i>n</i> eine Ganzzahl ist. Dies ist ein subscription_ <i>n</i> -Eintrag für jede synchronisierte Subskription. Die Nummerierung von <i>n</i> beginnt bei Null.

Bemerkungen

Dieser Hook wird in erster Linie für die Rückwärtskompatibilität bereitgestellt. Außer wenn Sie die erweiterte ScriptVersion-Option verwenden, können Sie Schemaänderungen problemlos ohne diesen Hook durchführen, indem Sie die START SYNCHRONIZATION SCHEMA CHANGE-Anweisung verwenden.

Wenn dieser Hook implementiert ist, sperrt dbmlsync standardmäßig die zu synchronisierenden Tabellen.

Diese gespeicherte Prozedur wird verwendet, um das Deployment von Schemaänderungen in verteilten entfernten Datenbanken durchzuführen. Wenn Sie diesen Hook für Schemaänderungen verwenden, stellen Sie sicher, dass alle Änderungen in den entfernten Datenbanken synchronisiert sind, bevor die Änderung des Schemas erfolgt. Damit wird gewährleistet, dass die Datenbanken auch danach noch synchronisiert werden können. Wenn dieser Hook verwendet wird, dürfen Sie die erweiterte dbmsync-Option LockTables nicht auf "off" setzen.

Während einer Synchronisation, bei der der Upload erfolgreich übernommen und von MobiLink bestätigt wurde, wird dieser Hook nach `sp_hook_dbmsync_download_end` und vor `sp_hook_dbmsync_end` aufgerufen. Dieser Hook wird nicht während einer reinen Download-Synchronisation aufgerufen oder wenn ein dateibasierter Download erstellt bzw. übernommen wird.

Die in diesem Hook durchgeführten Änderungen werden sofort nach seinem Abschluss festgeschrieben. Festschreiben oder Zurücksetzen können in diesem Hook sicher durchgeführt werden.

Privilegien

Hook-Prozeduren können von jedem Benutzer mit dem `MANAGE REPLICATION`-Systemprivileg erstellt werden. Um jedoch zu gewährleisten, dass ein Hook auf die `#hook_dict`-Tabelle zugreifen kann, die verwendet wird, um Daten an und aus Hooks zu übergeben, müssen Hooks eine der folgenden Anforderungen erfüllen:

- Eigentümer muss ein Benutzer sein, der die Systemprivilegien `SELECT ANY TABLE` und `UPDATE ANY TABLE` hat.
- Sie müssen durch die `SQL SECURITY INVOKER`-Klausel der `CREATE PROCEDURE`-Anweisung definiert sein.

Siehe auch

- „[START SYNCHRONIZATION SCHEMA CHANGE-Anweisung \[MobiLink\]](#)“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- „[Schemaänderungen in entfernten MobiLink-Clients](#)“ auf Seite 62

Beispiele

Das folgende Beispiel verwendet die `sp_hook_dbmsync_schema_upgrade`-Prozedur, um eine Spalte in die Dealer-Tabelle in der entfernten Datenbank hinzuzufügen. Wenn das Upgrade erfolgreich ist, wird der `sp_hook_dbmsync_schema_upgrade`-Hook gelöscht.

```
CREATE PROCEDURE sp_hook_dbmsync_schema_upgrade( )
BEGIN
  -- Upgrade the schema of the Dealer table. Add a column:
  ALTER TABLE Dealer
    ADD dealer_description VARCHAR(128);

  -- Change the script version used to synchronize
  ALTER SYNCHRONIZATION SUBSCRIPTION sub1
    SET SCRIPT VERSION='v2';

  -- If the schema upgrade is successful, drop this hook:
  UPDATE #hook_dict
    SET value = 'on success'
```

```
WHERE name = 'drop hook';  
END;
```

sp_hook_dbmlsync_set_extended_options

Mit dieser gespeicherten Prozedur können Sie das Verhalten einer anstehenden Synchronisation programmiertechnisch anpassen, indem Sie erweiterte Optionen festlegen, die für die betreffende Synchronisation übernommen werden.

Zeilen in der Tabelle #hook_dict

Name	Wert	Beschreibung
publication_ <i>n</i> (in)	Publikation	Nicht mehr empfohlen. Verwenden Sie stattdessen subscription_ <i>n</i> . Die synchronisierten Publikationen, wobei <i>n</i> eine Ganzzahl ist. Es gibt einen publication_ <i>n</i> -Eintrag für jede synchronisierte Publikation. Die Nummerierung von <i>n</i> beginnt bei Null.
MobiLink user (in)	MobiLink-Benutzername	Der MobiLink-Benutzer, für den Sie synchronisieren
extended options (out)	<i>opt</i> =Wert;...	Bei der nächsten Synchronisation hinzuzufügende erweiterte Optionen
subscription_ <i>n</i> (in)	Subskriptionsname(<i>n</i>)	Die synchronisierten Subskriptionen, wobei <i>n</i> eine Ganzzahl ist. Dies ist ein subscription_ <i>n</i> -Eintrag für jede synchronisierte Subskription. Die Nummerierung von <i>n</i> beginnt bei Null.

Bemerkungen

Wenn eine Prozedur dieses Namens vorhanden ist, wird sie ein- oder mehrmals vor jeder Synchronisation aufgerufen.

Die von dieser Hook-Prozedur angegebenen erweiterten Optionen gelten nur für die von der Subskription und den MobiLink-Benutzereinträgen identifizierte Synchronisation und werden nur angewendet, bis die Hook-Prozedur das nächste Mal für die betreffende Synchronisation aufgerufen wird.

Die erweiterte Option für die Abfolgeplanung kann in diesem Hook nicht angegeben werden.

Die Aktionen dieser Prozedur werden sofort nach ihrer Ausführung festgeschrieben.

Privilegien

Hook-Prozeduren können von jedem Benutzer mit dem MANAGE REPLICATION-Systemprivileg erstellt werden. Um jedoch zu gewährleisten, dass ein Hook auf die #hook_dict-Tabelle zugreifen kann, die verwendet wird, um Daten an und aus Hooks zu übergeben, müssen Hooks eine der folgenden Anforderungen erfüllen:

- Eigentümer muss ein Benutzer sein, der die Systemprivilegien SELECT ANY TABLE und UPDATE ANY TABLE hat.
- Sie müssen durch die SQL SECURITY INVOKER-Klausel der CREATE PROCEDURE-Anweisung definiert sein.

Siehe auch

- [„Hook-Sequenz bei der Synchronisation“ auf Seite 203](#)
- [„Erweiterte Optionen von MobiLink SQL Anywhere-Clients“ auf Seite 143](#)
- [Prioritätenfolge auf Seite 144](#)

Beispiele

Im folgenden Beispiel wird `sp_hook_dbmlsync_set_extended_options` verwendet, um die erweiterte Option Increment festzulegen. Die erweiterte Option wird nur angewendet, wenn "sub1" synchronisiert wird.

```
CREATE PROCEDURE sp_hook_dbmlsync_set_extended_options ( )
BEGIN
  IF exists(SELECT * FROM #hook_dict
    WHERE name LIKE 'subscription_%' AND value='sub1')
  THEN
    -- specify the Increment extended option
    UPDATE #hook_dict
      SET value = 'Increment=10K'
    WHERE name = 'extended options';
  END IF;
END;
```

sp_hook_dbmlsync_set_ml_connect_info

Mit dieser gespeicherten Prozedur stellen Sie das Netzwerkprotokoll und die Netzwerkprotokolloption für die Verbindung zum MobiLink-Server ein.

Name	Wert	Beschreibung
publication_ <i>n</i> (in)	Publikation	Nicht mehr empfohlen. Verwenden Sie stattdessen subscription_ <i>n</i> . Die synchronisierten Publikationen, wobei <i>n</i> eine Ganzzahl ist. Es gibt einen publication_ <i>n</i> -Eintrag für jede synchronisierte Publikation. Die Nummerierung von <i>n</i> beginnt bei Null.
MobiLink user (in)	MobiLink-Benutzername	Der MobiLink-Benutzer, für den Sie synchronisieren
script version (in)	Skriptversionsname	Die MobiLink-Skriptversion, die für die Synchronisation verwendet werden soll

Name	Wert	Beschreibung
connection type (in/out)	tcpip, tls, http oder https	Das Netzwerkprotokoll, das für die Verbindung zum MobiLink-Server verwendet wird.
connection address (in/out)	Protokolloptionen	Die Kommunikationsadresse, die für die Verbindung zum MobiLink-Server benutzt wird. Siehe „ Netzwerkprotokolloptionen des MobiLink-Clients “ auf Seite 25.
subscription_ <i>n</i> (in)	Subskriptionsname(<i>n</i>)	Die synchronisierten Subskriptionen, wobei <i>n</i> eine Ganzzahl ist. Dies ist ein subscription_ <i>n</i> -Eintrag für jede synchronisierte Subskription. Die Nummerierung von <i>n</i> beginnt bei Null.

Bemerkungen

Mit diesem Hook können Sie das Netzwerkprotokoll und dessen Optionen für die Verbindung zum MobiLink-Server einstellen, indem Sie den Wert in der Verbindungstyp- und/oder den Verbindungsadresszeilen ändern.

Das Protokoll und die Optionen können auch in "sp_hook_dbmlsync_set_extended_options" festgelegt werden, wobei es sich um einen Hook handelt, der zu Beginn einer Synchronisation aufgerufen wird. "sp_hook_dbmlsync_set_ml_connect_info" wird unmittelbar vor dem Versuch von dbmlsync aufgerufen, eine Verbindung zum MobiLink-Server herzustellen.

Dieser Hook ist dann nützlich, wenn Sie die Optionen in einem Hook einstellen wollen, dies aber später im Synchronisationsprozess durchführen möchten als "sp_hook_dbmlsync_set_extended_options". Dies könnte beispielsweise der Fall sein, wenn die Optionen aufgrund der verfügbaren Signalstärke im benutzten Netzwerk eingestellt werden sollen.

Privilegien

Hook-Prozeduren können von jedem Benutzer mit dem MANAGE REPLICATION-Systemprivileg erstellt werden. Um jedoch zu gewährleisten, dass ein Hook auf die #hook_dict-Tabelle zugreifen kann, die verwendet wird, um Daten an und aus Hooks zu übergeben, müssen Hooks eine der folgenden Anforderungen erfüllen:

- Eigentümer muss ein Benutzer sein, der die Systemprivilegien SELECT ANY TABLE und UPDATE ANY TABLE hat.
- Sie müssen durch die SQL SECURITY INVOKER-Klausel der CREATE PROCEDURE-Anweisung definiert sein.

Siehe auch

- „Ereignis-Hooks für SQL Anywhere-Clients“ auf Seite 202
- „Hook-Sequenz bei der Synchronisation“ auf Seite 203
- „Erweiterte Option CommunicationType (ctp)“ auf Seite 150
- „Netzwerkprotokolloptionen des MobiLink-Clients“ auf Seite 25
- „sp_hook_dbmlsync_set_extended_options“ auf Seite 252

Beispiel

```
CREATE PROCEDURE sp_hook_dbmlsync_set_ml_connect_info()
begin
    UPDATE #hook_dict
    SET VALUE = 'tcpip'
    WHERE name = 'connection type';

    UPDATE #hook_dict
    SET VALUE = 'host=localhost'
    WHERE name = 'connection address';
end
```

sp_hook_dbmlsync_set_upload_end_progress

Mit dieser gespeicherten Prozedur können Sie einen Abschluss-Verarbeitungsfortschritt definieren, wenn eine skriptgesteuerte Upload-Subskription synchronisiert wird. Diese Prozedur wird nur aufgerufen, wenn eine skriptgesteuerte Upload-Publikation synchronisiert wird.

Zeilen in der Tabelle #hook_dict

Name	Wert	Beschreibung
generating download exclusion list (in)	TRUE FALSE	TRUE, wenn während der Synchronisation kein Upload erfolgt (zum Beispiel bei reinen Download-Synchronisationen oder bei einem dateibasierten Download). In diesen Fällen werden die Upload-Skripten immer noch aufgerufen und die generierten Vorgänge werden zum Erkennen der Downloadvorgänge verwendet, bei denen Zeilen verändert werden, die dann übertragen werden müssen. Wird ein solcher Vorgang ermittelt, wird kein Download durchgeführt.
publication_n (in)	Publikation	Nicht mehr empfohlen. Verwenden Sie stattdessen subscription_n. Die synchronisierten Publikationen, wobei n eine Ganzzahl ist. Es gibt einen publication_n-Eintrag für jede synchronisierte Publikation. Die Nummerierung von n beginnt bei Null.

Name	Wert	Beschreibung
Start-Verarbeitungsfortschritt als timestamp_ <i>n</i> (in)	Fortschritt als Zeitstempel	Der Start-Verarbeitungsfortschritt für jede zu synchronisierende Subskription wird als Zeitstempel ausgedrückt, wobei <i>n</i> der Ganzzahl entspricht, die zur Identifizierung der Subskription verwendet wird.
Start-Verarbeitungsfortschritt als bigint_ <i>n</i> (in)	Fortschritt als "bigint"	Der Start-Verarbeitungsfortschritt für jede zu synchronisierende Subskription wird als "bigint" ausgedrückt, wobei <i>n</i> der Ganzzahl entspricht, die zur Identifizierung der Subskription verwendet wird.
script version (in)	Skriptversionsname	Die MobiLink-Skriptversion, die für die Synchronisation verwendet werden soll
MobiLink user (in)	MobiLink-Benutzername	Der MobiLink-Benutzer, für den Sie synchronisieren
end progress is bigint (in out)	TRUE FALSE	<p>Wenn diese Zeile TRUE ist, ist der Abschluss-Verarbeitungsfortschritt ein "bigint" ohne Vorzeichen, dargestellt als Zeichenfolge (zum Beispiel '12345')</p> <p>Wenn diese Zeile auf FALSE gesetzt ist, wird für den Abschluss-Verarbeitungsfortschrittswert ein als Zeichenfolge dargestellter TIME-STAMP-Wert angenommen (z.B. "1900/01/01 12:00:00.000").</p> <p>Standardwert ist FALSE.</p>
end progress (in out)	Zeitstempel	<p>Der Hook kann diese Zeile ändern, um die Werte "end progress " und "raw end progress" zu ändern, die an die Upload-Skripten übergeben wurden. Diese Werte definieren den Zeitpunkt, bis zu dem alle Vorgänge im zu generierenden Upload berücksichtigt werden.</p> <p>Der Wert dieser Zeile kann ein "bigint" ohne Vorzeichen oder ein Zeitstempel sein, abhängig von der Einstellung in der Zeile "end progress is bigint". Der Standardwert dieser Zeile ist der aktuelle Zeitstempel.</p>

Name	Wert	Beschreibung
subscription_ <i>n</i> (in)	Subskriptionsname(<i>n</i>)	Die synchronisierten Subskriptionen, wobei <i>n</i> eine Ganzzahl ist. Dies ist ein subscription_ <i>n</i> -Eintrag für jede synchronisierte Subskription. Die Nummerierung von <i>n</i> beginnt bei Null.

Bemerkungen

Jedes Mal, wenn bei einem skriptgesteuerten Upload eine Prozedur aufgerufen wird, wird ein Start-Verarbeitungsfortschrittswert und ein Abschluss-Verarbeitungsfortschrittswert übergeben. Die Prozedur muss alle entsprechenden Vorgänge, die innerhalb des durch diese Werte definierten Zeitraums auftreten, zurückgeben. Der Start-Verarbeitungsfortschrittswert entspricht immer dem Abschluss-Verarbeitungsfortschrittswert der letzten erfolgreichen Synchronisation. Bei der ersten Synchronisation beginnt der Fortschritt mit dem Wert January 1, 1900, 00:00:00.000. Standardmäßig entspricht der Abschluss-Verarbeitungsfortschrittswert dem Zeitpunkt, an dem dbmlsync mit dem Upload beginnt.

Mit diesem Hook können Sie den standardmäßigen Abschluss-Verarbeitungsfortschrittswert überschreiben. Sie können einen kürzeren Zeitraum für den Upload definieren oder ein Fortschrittsprotokollschema anwenden, das nicht auf Zeitstempeln basiert, sondern beispielsweise auf Generierungsnummern.

Wenn "end progress is bigint" TRUE ist, muss der Abschluss-Verarbeitungsfortschritt eine Ganzzahl kleiner oder gleich der Millisekunden zwischen 1900-01-01 00:00:00 und 9999-12-31 23:59:59.999 sein, also 255.611.203.259.999.

Privilegien

Hook-Prozeduren können von jedem Benutzer mit dem MANAGE REPLICATION-Systemprivileg erstellt werden. Um jedoch zu gewährleisten, dass ein Hook auf die #hook_dict-Tabelle zugreifen kann, die verwendet wird, um Daten an und aus Hooks zu übergeben, müssen Hooks eine der folgenden Anforderungen erfüllen:

- Eigentümer muss ein Benutzer sein, der die Systemprivilegien SELECT ANY TABLE und UPDATE ANY TABLE hat.
- Sie müssen durch die SQL SECURITY INVOKER-Klausel der CREATE PROCEDURE-Anweisung definiert sein.

Siehe auch

- „Benutzerdefinierte Fortschrittswerte in skriptgesteuerten Uploads“ auf Seite 334
- „Hook-Sequenz bei der Synchronisation“ auf Seite 203
- „Skriptgesteuerter Upload“ auf Seite 323

sp_hook_dbmlsync_sql_error

Benutzen Sie diese gespeicherte Prozedur, um Datenbankfehler zu verarbeiten, die während der Synchronisation auftreten. Sie können beispielsweise den Hook sp_hook_dbmlsync_sql_error implementieren, um beim Eintreten eines bestimmten SQL-Fehlers eine bestimmte Aktion durchzuführen.

Zeilen in der Tabelle #hook_dict

Name	Wert	Beschreibung
publication_ <i>n</i> (in)	Publikation	Nicht mehr empfohlen. Verwenden Sie stattdessen subscription_ <i>n</i> . Die synchronisierten Publikationen, wobei <i>n</i> eine Ganzzahl ist. Es gibt einen publication_ <i>n</i> -Eintrag für jede synchronisierte Publikation. Die Nummerierung von <i>n</i> beginnt bei Null.
MobiLink user (in)	MobiLink-Benutzername	Der MobiLink-Benutzer, für den Sie synchronisieren
script version (in)	Skriptversionsname	Die MobiLink-Skriptversion, die für die Synchronisation verwendet werden soll
error message (in)	Fehlermeldungstext	Dies ist der Text, der auch im dbmlsync-Log angezeigt wird.
error id (in)	Nummerisch	Eine ID, die die Meldung eindeutig identifiziert.
error hook user state (in/out)	Ganzzahl	Dieser Wert kann vom Hook eingestellt werden, um Statusinformationen in zukünftigen Aufrufen an die Hooks sp_hook_dbmlsync_all_error, sp_hook_dbmlsync_communication_error, sp_hook_dbmlsync_misc_error, sp_hook_dbmlsync_sql_error oder sp_hook_dbmlsync_end zu übergeben. Wenn einer dieser Hooks zum ersten Mal aufgerufen wird, ist der Wert der Zeile 0. Wenn ein Hook den Wert der Zeile ändert, wird der neue Wert im nächsten Hook-Aufruf verwendet.
SQLCODE (in)	SQLCODE	Der SQLCODE, der von der Datenbank zurückgegeben wird, wenn der Vorgang fehlergeschlagen ist. Siehe „ SQL Anywhere-Fehlermeldungen - sortiert nach SQLCODE “ [Fehlermeldungen].
SQLSTATE (in)	SQLSTATE-Wert	Der SQLSTATE, der von der Datenbank zurückgegeben wird, wenn der Vorgang fehlergeschlagen ist.

Name	Wert	Beschreibung
subscription_ <i>n</i> (in)	Subskriptionsname(<i>n</i>)	Die synchronisierten Subskriptionen, wobei <i>n</i> eine Ganzzahl ist. Dies ist ein subscription_ <i>n</i> -Eintrag für jede synchronisierte Subskription. Die Nummerierung von <i>n</i> beginnt bei Null.

Bemerkungen

Wenn beim Start ein Fehler auftritt, bevor eine Synchronisation initiiert werden konnte, werden die #hook_dict-Einträge für den MobiLink-Benutzer und die Skriptversion auf eine leere Zeichenfolge zurückgesetzt, und in der #hook_dict-Tabelle werden keine publication_*n*- oder subscription_*n*-Zeilen eingestellt.

Sie können SQL-Fehler mit SQL Anywhere SQLCODE oder dem ANSI SQL-Standard SQLSTATE ermitteln. Eine Liste der SQLCODE- oder SQLSTATE-Werte finden Sie unter „[SQL Anywhere - Fehlermeldungen](#)“ [[Fehlermeldungen](#)].

Die Zeile "error hook user state" bietet ein nützliches Verfahren für die Übergabe der Informationen über die Art des Fehlers an den Hook sp_hook_dbmsync_end. Anhand dieser Informationen können Sie entscheiden, ob die Synchronisation erneut ausgeführt werden soll.

Diese Prozedur wird in einer eigenen Verbindung ausgeführt, damit sichergestellt ist, dass die dadurch bewirkten Vorgänge nicht verloren gehen, wenn ein Zurücksetzen der Synchronisationsverbindung erfolgt. Wenn dbmsync keine getrennte Verbindung herstellen kann, wird die Prozedur nicht aufgerufen.

Da dieser Hook in einer eigenen Verbindung ausgeführt wird, sollten Sie beim Zugriff auf Tabellen vorsichtig sein, die in Ihrer Hook-Prozedur synchronisiert werden, denn diese Tabellen werden von dbmsync möglicherweise gesperrt. Diese Sperren können zum Fehlschlagen von Vorgängen im Hook oder zu unbegrenzten Wartezeiten führen.

Die Aktionen dieser Prozedur werden sofort nach ihrer Ausführung festgeschrieben.

Privilegien

Hook-Prozeduren können von jedem Benutzer mit dem MANAGE REPLICATION-Systemprivileg erstellt werden. Um jedoch zu gewährleisten, dass ein Hook auf die #hook_dict-Tabelle zugreifen kann, die verwendet wird, um Daten an und aus Hooks zu übergeben, müssen Hooks eine der folgenden Anforderungen erfüllen:

- Eigentümer muss ein Benutzer sein, der die Systemprivilegien SELECT ANY TABLE und UPDATE ANY TABLE hat.
- Sie müssen durch die SQL SECURITY INVOKER-Klausel der CREATE PROCEDURE-Anweisung definiert sein.

Siehe auch

- „Behandlung von Fehlern und Warnungen in Ereignis-Hook-Prozeduren“ auf Seite 207
- „sp_hook_dbmlsync_all_error“ auf Seite 211
- „sp_hook_dbmlsync_communication_error“ auf Seite 215
- „sp_hook_dbmlsync_misc_error“ auf Seite 240
- „SQL Anywhere - Fehlermeldungen“ [*Fehlermeldungen*]

sp_hook_dbmlsync_upload_begin

Verwenden Sie diese gespeicherte Prozedur, um benutzerdefinierte Aktionen direkt vor der Übertragung des Uploads einzufügen.

Zeilen in der Tabelle #hook_dict

Name	Wert	Beschreibung
publication_ <i>n</i> (in)	Publikation	Nicht mehr empfohlen. Verwenden Sie stattdessen subscription_ <i>n</i> . Die synchronisierten Publikationen, wobei <i>n</i> eine Ganzzahl ist. Es gibt einen publication_ <i>n</i> -Eintrag für jede synchronisierte Publikation. Die Nummerierung von <i>n</i> beginnt bei Null.
MobiLink user (in)	MobiLink-Benutzername	Der MobiLink-Benutzer, für den Sie synchronisieren
script version (in)	Skriptversionsname	Die MobiLink-Skriptversion, die für die Synchronisation verwendet werden soll
subscription_ <i>n</i> (in)	Subskriptionsname(<i>n</i>)	Die synchronisierten Subskriptionen, wobei <i>n</i> eine Ganzzahl ist. Dies ist ein subscription_ <i>n</i> -Eintrag für jede synchronisierte Subskription. Die Nummerierung von <i>n</i> beginnt bei Null.

Bemerkungen

Wenn eine Prozedur dieses Namens vorhanden ist, wird sie aufgerufen, und zwar unmittelbar bevor dbmlsync den Upload sendet.

Die Aktionen dieser Prozedur werden sofort nach ihrer Ausführung festgeschrieben.

Privilegien

Hook-Prozeduren können von jedem Benutzer mit dem MANAGE REPLICATION-Systemprivileg erstellt werden. Um jedoch zu gewährleisten, dass ein Hook auf die #hook_dict-Tabelle zugreifen kann, die verwendet wird, um Daten an und aus Hooks zu übergeben, müssen Hooks eine der folgenden Anforderungen erfüllen:

- Eigentümer muss ein Benutzer sein, der die Systemprivilegien SELECT ANY TABLE und UPDATE ANY TABLE hat.
- Sie müssen durch die SQL SECURITY INVOKER-Klausel der CREATE PROCEDURE-Anweisung definiert sein.

Siehe auch

- [„Hook-Sequenz bei der Synchronisation“ auf Seite 203](#)

Beispiele

Nehmen wir an, Sie verwenden die folgende Tabelle, um Synchronisationsereignisse in der entfernten Datenbank zu protokollieren.

```
CREATE TABLE SyncLog
(
    "event_id"          integer NOT NULL DEFAULT autoincrement ,
    "event_time"        timestamp NULL,
    "event_name"        varchar(128) NOT NULL ,
    "subs"              varchar(1024) NULL ,
    PRIMARY KEY ("event_id")
)
```

Folgendes zeichnet den Upload-Beginn für jede Synchronisation auf.

```
CREATE PROCEDURE sp_hook_dbmlsync_upload_begin ()
BEGIN

    DECLARE subs_list VARCHAR(1024);

    -- build a list of subscriptions being synchronized
    SELECT LIST(value) INTO subs_list
    FROM #hook_dict
    WHERE name LIKE 'subscription_%';

    -- log the event
    INSERT INTO SyncLog(event_time, event_name, subs)
    VALUES( CURRENT TIMESTAMP, 'sp_hook_dbmlsync_upload_begin', subs_list );
END
```

sp_hook_dbmlsync_upload_end

Verwenden Sie diese gespeicherte Prozedur, um benutzerdefinierte Aktionen einzufügen, nachdem dbmlsync den Empfang des Uploads durch den MobiLink-Server geprüft hat.

Zeilen in der Tabelle #hook_dict

Name	Wert	Beschreibung
failure cause (in)	Weitere Hinweise finden Sie im Abschnitt "Bemerkungen".	Der Grund, warum ein Upload fehlgeschlagen ist. Weitere Hinweise finden Sie unter "Bemerkungen".

Name	Wert	Beschreibung
upload status (in)	retry committed failed unknown	<p>Gibt den Status an, der vom MobiLink-Server zurückgegeben wird, wenn dbmlsync versucht, den Empfang des Uploads zu prüfen.</p> <p>retry - Der MobiLink-Server und dbmlsync hatten unterschiedliche Werte für den Log-Offset, von dem der Upload starten sollte. Der Upload wurde vom MobiLink-Server nicht festgeschrieben. Das Dienstprogramm dbmlsync versucht, einen anderen Upload zu senden, der bei einem neuen Log-Offset beginnt.</p> <p>committed - Der Upload wurde vom MobiLink-Server empfangen und festgeschrieben.</p> <p>failed - Der MobiLink-Server hat den Upload nicht festgeschrieben.</p> <p>unknown - Der Upload wurde vom MobiLink-Server nicht bestätigt. Es gibt keine Möglichkeit zu erkennen, ob der Upload festgeschrieben wurde oder nicht.</p>
publication_ <i>n</i> (in)	Publikation	Nicht mehr empfohlen. Verwenden Sie stattdessen subscription_ <i>n</i> . Die synchronisierten Publikationen, wobei <i>n</i> eine Ganzzahl ist. Es gibt einen publication_ <i>n</i> -Eintrag für jede synchronisierte Publikation. Die Nummerierung von <i>n</i> beginnt bei Null.
MobiLink user (in)	MobiLink-Benutzername	Der MobiLink-Benutzer, für den Sie synchronisieren
script version (in)	Skriptversionsname	Die MobiLink-Skriptversion, die für die Synchronisation verwendet werden soll

Name	Wert	Beschreibung
authentication value (in)	Wert	Dieser Wert gibt die Ergebnisse des Versuchs von dbmlsync an, sich am MobiLink-Server zu authentifizieren. Er wird von den Skripten <code>authenticate_user</code> , <code>authenticate_user_hashed</code> oder <code>authenticate_parameters</code> auf dem Server generiert. Der Wert ist eine leere Zeichenfolge, wenn der Upload-Status unbekannt ist oder die Hook-Prozedur <code>upload_end</code> nach dem Neusenden eines Uploads aufgerufen wird, weil ein Konflikt zwischen den in der entfernten und der konsolidierten Datenbank gespeicherten Log-Offsets aufgetreten ist.
subscription_ <i>n</i> (in)	Subskriptionsname(<i>n</i>)	Die synchronisierten Subskriptionen, wobei <i>n</i> eine Ganzzahl ist. Dies ist ein <code>subscription_ <i>n</i></code> -Eintrag für jede synchronisierte Subskription. Die Nummerierung von <i>n</i> beginnt bei Null.

Bemerkungen

Wenn eine Prozedur dieses Namens vorhanden ist, wird sie aufgerufen, unmittelbar nachdem dbmlsync den Upload gesendet und die Bestätigung dafür vom MobiLink-Server erhalten hat.

Wenn Sie einen transaktionalen Upload oder einen inkrementellen Upload durchführen, wird dieser Hook nach dem Senden jedes Upload-Segments aufgerufen. In diesen Fällen ist der Upload-Status beim Aufrufen des Hook unbekannt, außer beim letzten Mal.

Die Aktionen dieser Prozedur werden sofort nach ihrer Ausführung festgeschrieben.

Folgende Parameterwerte sind für die Zeile "failure cause" in der Tabelle `#hook_dict` möglich:

- **UPLD_ERR_INVALID_USERID_OR_PASSWORD** Benutzer-ID oder Kennwort war falsch.
- **UPLD_ERR_USERID_OR_PASSWORD_EXPIRED** Benutzer-ID oder Kennwort ist abgelaufen.
- **UPLD_ERR_REMOTE_ID_ALREADY_IN_USE** Die entfernte ID wird bereits benutzt.
- **UPLD_ERR_SQLCODE_ *n*** Hier ist *n* eine Ganzzahl. In der konsolidierten Datenbank ist ein SQL-Fehler aufgetreten. Die angegebene Ganzzahl ist der SQLCODE für den eingetretenen Fehler.
- **UPLD_ERR_USER_ABORT_REQUEST** Der Upload wurde auf Anforderung des Benutzers abgebrochen.

Privilegien

Hook-Prozeduren können von jedem Benutzer mit dem MANAGE REPLICATION-Systemprivileg erstellt werden. Um jedoch zu gewährleisten, dass ein Hook auf die `#hook_dict`-Tabelle zugreifen kann, die verwendet wird, um Daten an und aus Hooks zu übergeben, müssen Hooks eine der folgenden Anforderungen erfüllen:

- Eigentümer muss ein Benutzer sein, der die Systemprivilegien SELECT ANY TABLE und UPDATE ANY TABLE hat.
- Sie müssen durch die SQL SECURITY INVOKER-Klausel der CREATE PROCEDURE-Anweisung definiert sein.

Siehe auch

- [„Hook-Sequenz bei der Synchronisation“ auf Seite 203](#)

Beispiele

Nehmen wir an, Sie verwenden die folgende Tabelle, um Synchronisationsereignisse in der entfernten Datenbank zu protokollieren.

```
CREATE TABLE SyncLog
(
    "event_id"          integer NOT NULL DEFAULT autoincrement ,
    "event_time"        timestamp NULL,
    "event_name"        varchar(128) NOT NULL ,
    "subs"              varchar(1024) NULL ,
    PRIMARY KEY ("event_id")
)
```

Folgendes zeichnet das Upload-Ende für jede Synchronisation auf.

```
CREATE PROCEDURE sp_hook_dbmlsync_upload_end ()
BEGIN

    DECLARE subs_list VARCHAR(1024);

    -- build a list of subscriptions being synchronized
    SELECT LIST(value) INTO subs_list
    FROM #hook_dict
    WHERE name LIKE 'subscription_%';

    -- log the event
    INSERT INTO SyncLog(event_time, event_name, subs)
    VALUES( CURRENT_TIMESTAMP, 'sp_hook_dbmlsync_upload_end', subs_list );
END
```

sp_hook_dbmlsync_validate_download_file

Verwenden Sie diesen Hook, um benutzerdefinierte Logik zu implementieren, die entscheiden kann, ob eine Download-Datei in einer entfernten Datenbank übernommen werden kann. Dieser Hook wird nur aufgerufen, wenn ein dateibasierter Download erfolgt.

Zeilen in der Tabelle #hook_dict

Name	Wert	Beschreibung
publication_ <i>n</i> (in)	Publikation	Nicht mehr empfohlen. Verwenden Sie stattdessen subscription_ <i>n</i> . Die synchronisierten Publikationen, wobei <i>n</i> eine Ganzzahl ist. Es gibt einen publication_ <i>n</i> -Eintrag für jede synchronisierte Publikation. Die Werte <i>n</i> in publication_ <i>n</i> und generation number_ <i>n</i> sind identisch. Die Nummerierung von <i>n</i> beginnt bei Null.
MobiLink user (in)	MobiLink-Benutzername	Der MobiLink-Benutzer, für den Sie synchronisieren
file last download time (in)		Die Zeit des letzten Downloads der Download-Datei. (Die Download-Datei enthält alle Zeilen, die zwischen der Zeit ihres letzten Downloads und dem nächsten Download-Zeitpunkt geändert wurden.)
file next last download time (in)		Die Zeit des nächsten Downloads der Download-Datei. (Die Download-Datei enthält alle Zeilen, die zwischen der Zeit ihres letzten Downloads und dem nächsten Download-Zeitpunkt geändert wurden.)
file creation time (in)		Die Zeit, zu der die Download-Datei erstellt wurde
file generation number_ <i>n</i> (in)	Zahl	Die Generierungsnummer aus der Download-Datei. Es gibt jeweils einen generation number_ <i>n</i> -Wert für jeden subscription_ <i>n</i> -Eintrag. Die Werte <i>n</i> in subscription_ <i>n</i> und generation number_ <i>n</i> sind identisch. Die Nummerierung von <i>n</i> beginnt bei Null.
user data (in)	Zeichenfolge	Die Zeichenfolge, die mit der dbmlsync-Option -be zu dem Zeitpunkt angegeben wurde, als die Download-Datei erstellt wurde oder die DnldFileExtra-Synchronisationsprofiloption.

Name	Wert	Beschreibung
apply file (in out)	True False	Wenn TRUE (Standardwert), wird die Download-Datei nur übernommen, wenn sie auch die anderen Validierungsprüfungen von dbmsync besteht. Wenn FALSE, wird die Download-Datei nicht in die entfernte Datenbank übernommen.
check generation number (in out)	True False	Wenn TRUE (der Standardwert), validiert dbmsync Generierungsnummern. Wenn die Generierungsnummern in der Download-Datei nicht mit denen in der entfernten Datenbank übereinstimmen, wendet dbmsync die Download-Datei nicht an. Wenn FALSE, überprüft dbmsync nicht die Generierungsnummern.
setting generation number (in)	true false	TRUE, wenn die Option -bg oder UpdateGenNum-Synchronisationsprofiloption beim Erstellen der Download-Datei benutzt wurde. Wenn TRUE, werden die Generationsnummern in der entfernten Datenbank aus der Download-Datei aktualisiert und normale Generationsnummerprüfungen nicht durchgeführt.
subscription_ <i>n</i> (in)	Subskriptionsname(<i>n</i>)	Die synchronisierten Subskriptionen, wobei <i>n</i> eine Ganzzahl ist. Dies ist ein subscription_ <i>n</i> -Eintrag für jede synchronisierte Subskription. Die Nummerierung von <i>n</i> beginnt bei Null.

Bemerkungen

Benutzen Sie diese gespeicherte Prozedur, um benutzerdefinierte Prüfungen einzurichten, mit denen entschieden werden kann, ob eine Download-Datei übernommen wird oder nicht.

Um die Generationsnummern oder Zeitstempel in der Datei mit denjenigen zu vergleichen, die in der entfernten Datenbank gespeichert sind, können Sie die Werte aus der SYSSYNC-Systemansicht abrufen.

Dieser Hook wird aufgerufen, bevor ein dateibasierter Download auf die entfernte Datenbank angewendet wird.

Die Aktionen dieses Hooks werden sofort nach seinem Abschluss festgeschrieben.

Privilegien

Hook-Prozeduren können von jedem Benutzer mit dem MANAGE REPLICATION-Systemprivileg erstellt werden. Um jedoch zu gewährleisten, dass ein Hook auf die #hook_dict-Tabelle zugreifen kann, die verwendet wird, um Daten an und aus Hooks zu übergeben, müssen Hooks eine der folgenden Anforderungen erfüllen:

- Eigentümer muss ein Benutzer sein, der die Systemprivilegien SELECT ANY TABLE und UPDATE ANY TABLE hat.
- Sie müssen durch die SQL SECURITY INVOKER-Klausel der CREATE PROCEDURE-Anweisung definiert sein.

Siehe auch

- „dbmlsync-Option -be“ auf Seite 113
- „dbmlsync-Option -bg“ auf Seite 114
- „MobiLink - dateibasierte Downloads“ [*MobiLink - Serveradministration*]

Beispiele

Das folgende Beispiel verhindert die Anwendung von Download-Dateien, die nicht die Benutzerzeichenfolge 'sales manager data' enthalten.

```
CREATE PROCEDURE sp_hook_dbmlsync_validate_download_file ()
BEGIN
  IF NOT exists(SELECT * FROM #hook_dict
    WHERE name = 'User data' AND value='sales manager data')
  THEN
    UPDATE #hook_dict
      SET value = 'false' WHERE name = 'Apply file';
  END IF;
END;
```

Dbmlsync C++-API-Referenz

Header-Datei

dbmlsynccli.hpp

Beispiel

Das untenstehende Beispiel zeigt eine typische Anwendung, in der die C++-Version der Dbmlsync-API verwendet wird, um eine Synchronisation zum Empfang von Ausgabeereignissen durchzuführen. Aus Gründen der Einfachheit wird die Fehlerbehandlung weggelassen. Es ist immer empfehlenswert, den Rückgabewert von jedem API-Aufruf zu überprüfen.

```
#include <stdio.h>
#include "dbmlsynccli.hpp"

int main( void ) {
  DbmlsyncClient *client;
  DBSC_SyncHdl   syncHdl;
  DBSC_Event     *ev1;

  client = DbmlsyncClient::InstantiateClient();
  if( client == NULL ) return( 1 );
  client->Init();

  // Setting the "server path" is usually required on Windows Mobile.
  // In other environments the server path is usually not required unless
  // your SQL Anywhere install is not in your path or you have multiple
  // versions of the product installed.
```

```

    client->SetProperty( "server path", "C:\\SQLAnywhere\\bin32" );

    client->StartServer( 3426,
        "-c server=remote;dbn=reml;uid=dba;pwd=sql -v+ -ot c:\\
        \\dbsync1.txt",
        5000, NULL );
    client->Connect( NULL, 3426, "dba", "sql");
    syncHdl = client->Sync( "my_sync_profile", "" );
    while( client->GetEvent( &ev1, 5000) == DBSC_GETEVENT_OK ) {
        if( ev1->hdl == syncHdl ) {
            //
            // Process events that interest you here
            //

            if( ev1->type == DBSC_EVENTTYPE_SYNC_DONE ) {
                client->FreeEventInfo( ev1 );
                break;
            }
            client->FreeEventInfo( ev1 );
        }
    }
    client->ShutdownServer( DBSC_SHUTDOWN_ON_EMPTY_QUEUE );
    client->WaitForServerShutdown( 10000 );
    client->Disconnect();
    client->Fini();
    delete client;

    return( 0 );
}

```

DbmlsyncClient-Klasse

Kommuniziert über TCP/IP mit einem eigenen Prozess, dem dbmlsync-Server, der eine Synchronisation durchführt, indem er eine Verbindung zum MobiLink-Server und zur entfernten Datenbank herstellt.

Syntax

```
public class DbmlsyncClient
```

Mitglieder

Alle Mitglieder der DbmlsyncClient-Klasse, einschließlich aller geerbten Mitglieder.

Name	Beschreibung
CancelSync-Methode	Bricht eine Synchronisationsanforderung ab.
Connect-Methode	Öffnet eine Verbindung mit einem dbmlsync-Server, der bereits auf diesem Computer ausgeführt wird.
Disconnect-Methode	Unterbricht die Verbindung mit dem dbmlsync-Server, die mit der Connect-Methode hergestellt wurde.
Fini-Methode	Gibt alle von dieser Instanz der Klasse verwendeten Ressourcen frei.

Name	Beschreibung
FreeEventInfo-Methode	Gibt Speicher frei, der einer DBSC_Event-Struktur zugeordnet ist, die von der GetEvent-Methode zurückgegeben wurde.
GetErrorInfo-Methode	Ruft zusätzliche Informationen über den Fehler ab, nachdem eine Methode der DbmlsyncClient-Klasse einen Fehlercode zurückgibt.
GetEvent-Methode	Ruft das nächste Rückmeldungseignis für vom Client angeforderte Synchronisationen ab.
GetProperty-Methode	Ruft den aktuellen Wert einer Eigenschaft ab.
Init-Methode	Initialisiert eine Instanz der DbmlsyncClient-Klasse.
InstantiateClient-Methode	Erstellt eine Instanz der dbmlsync-Clientklasse, mit der Synchronisationen gesteuert werden können.
Ping-Methode	Sendet eine Ping-Anforderung an den dbmlsync-Server, um zu prüfen, ob der Server aktiv ist und auf Anforderungen antwortet.
SetProperty-Methode	Legt verschiedene Eigenschaften fest, um das Verhalten der Klasseninstanz zu ändern.
ShutdownServer-Methode	Führt den dbmlsync-Server herunter, mit dem der Client verbunden ist.
StartServer-Methode	Startet einen neuen dbmlsync-Server, wenn noch keiner am angegebenen Port auf Nachrichten wartet.
Sync-Methode	Fordert an, dass der dbmlsync-Server eine Synchronisation ausführt.
WaitForServerShutdown-Methode	Kehrt zurück, wenn der Server heruntergefahren oder der Timeout abgelaufen ist, je nach dem, welches Ereignis früher eintritt.

Bemerkungen

Mehrere Clients können denselben dbmlsync-Server gemeinsam nutzen. Jeder dbmlsync-Server kann jedoch nur eine einzige entfernte Datenbank synchronisieren. Jede entfernte Datenbank kann jeweils nur von einem dbmlsync-Server synchronisiert werden.

Der dbmlsync-Server führt jeweils nur eine Synchronisation aus. Wenn der Server eine Synchronisationsanforderung empfängt, während er eine andere Synchronisation ausführt, stellt er diese Anforderung in eine Warteschlange und führt sie zu einem späteren Zeitpunkt aus.

Von den Synchronisationen generierte Statusinformationen werden über die GetEvent-Methode zurück an die Clientanwendung übertragen.

Siehe auch

- [DbmlsyncClient.GetEvent-Methode \[Dbmlsync C++\] auf Seite 275](#)

CancelSync-Methode

Bricht eine Synchronisationsanforderung ab.

Überladungsliste

Name	Beschreibung
CancelSync(DBSC_SyncHdl)-Methode (nicht mehr empfohlen)	Erlaubt es einem Client, eine Synchronisationsanforderung abubrechen, die zuvor mit der Sync-Methode gestellt wurde.
CancelSync(DBSC_SyncHdl, bool)-Methode	Erlaubt es einem Client, eine Synchronisationsanforderung abubrechen, die zuvor mit der Sync-Methode gestellt wurde.

CancelSync(DBSC_SyncHdl)-Methode (nicht mehr empfohlen)

Erlaubt es einem Client, eine Synchronisationsanforderung abubrechen, die zuvor mit der Sync-Methode gestellt wurde.

Syntax

```
public virtual bool CancelSync(DBSC_SyncHdl hdl)
```

Parameter

- **hdl** Das von der Sync-Methode zurückgegebene Synchronisations-Handle.

Rückgabe

TRUE, wenn die Synchronisationsanforderung erfolgreich abgebrochen wurde sonst FALSE. Wenn FALSE zurückgegeben wird, können Sie die Methode GetErrorInfo aufrufen, um weitere Informationen über den Fehler zu erhalten.

Bemerkungen

Es können nur Synchronisationsanforderungen abgebrochen werden, die darauf warten, verarbeitet zu werden. Um eine bereits begonnene Synchronisation zu stoppen, verwenden Sie die CancelSync(UInt32, Boolean)-Methode.

Vor der Verwendung dieser Methode muss eine Verbindung zum Server eingerichtet werden. Diese Methode kann nicht verwendet werden, wenn der Client nach dem Aufruf der Sync-Methode vom Server getrennt wurde.

Siehe auch

- [DbmlsyncClient.GetErrorInfo-Methode \[Dbmlsync C++\] auf Seite 274](#)
- [DbmlsyncClient.CancelSync-Methode \[Dbmlsync C++\] auf Seite 270](#)
- [DbmlsyncClient.ShutdownServer-Methode \[Dbmlsync C++\] auf Seite 279](#)

CancelSync(DBSC_SyncHdl, bool)-Methode

Erlaubt es einem Client, eine Synchronisationsanforderung abubrechen, die zuvor mit der Sync-Methode gestellt wurde.

Syntax

```
public virtual DBSC_CancelRet CancelSync(  
    DBSC_SyncHdl hdl,  
    bool cancel_active  
)
```

Parameter

- **hdl** Das von der Sync-Methode zurückgegebene Synchronisations-Handle.
- **cancel_active** Wenn dieser Parameter auf TRUE gesetzt ist, wird die Anforderung abgebrochen, auch wenn die Synchronisation bereits begonnen hat. Beim Wert FALSE wird die Anforderung nur abgebrochen, wenn die Synchronisation noch nicht begonnen hat.

Rückgabe

Ein Wert aus der DBSC_CancelRet-Enumeration. Wenn DBSC_CANCEL_FAILED zurückgegeben wird, können Sie die Methode GetErrorInfo aufrufen, um weitere Informationen über den Fehler zu erhalten.

Bemerkungen

Vor der Verwendung dieser Methode muss eine Verbindung zum Server eingerichtet werden. Diese Methode kann nicht verwendet werden, wenn der Client nach dem Aufruf der Sync-Methode vom Server getrennt wurde.

Siehe auch

- [DBSC_CancelRet-Enumeration \[Dbmlsync C++\] auf Seite 282](#)
- [DbmlsyncClient.GetErrorInfo-Methode \[Dbmlsync C++\] auf Seite 274](#)
- [DbmlsyncClient.ShutdownServer-Methode \[Dbmlsync C++\] auf Seite 279](#)

Connect-Methode

Öffnet eine Verbindung mit einem dbmlsync-Server, der bereits auf diesem Computer ausgeführt wird.

Syntax

```
public virtual bool Connect(  
    const char * host,  
    unsigned port,
```

```
    const char * uid,  
    const char * pwd  
)
```

Parameter

- **host** Dieser Wert ist reserviert. NULL verwenden.
- **port** Der TCP-Port, an dem der dbmlsync-Server auf Verbindungen wartet. Verwenden Sie denselben Port, den Sie für die StartServer-Methode angegeben haben.
- **uid** Eine gültige Datenbankbenutzer-ID mit DBA- oder REMOTE DBA-Berechtigung für die entfernte Datenbank, die synchronisiert werden soll.
- **pwd** Das Datenbankkennwort für den mit uid angegebenen Benutzer.

Rückgabe

TRUE, wenn eine Verbindung mit dem Server hergestellt wurde sonst FALSE. Wenn FALSE zurückgegeben wird, können Sie die Methode GetErrorInfo aufrufen, um weitere Informationen über den Fehler zu erhalten.

Bemerkungen

Dbmlsync-Server werden (entweder über die Befehlszeile oder über die StartServer-Methode) mithilfe einer Verbindungszeichenfolge gestartet, die eine Datenbankbenutzer-ID und das dazugehörige Kennwort (z.B. *server_userid*) enthält. Außerdem erfordert die Connect-Methode der Dbmlsync-API eine gültige Datenbankbenutzer-ID (z.B. *client_userid*).

Client_userid wird nur verwendet, um zu validieren, ob dieser Client ausreichende Berechtigungen für die Synchronisation der Datenbank hat. Wenn Synchronisationen ausgeführt werden, wird *server_userid* verwendet.

In SQL Anywhere 12 und früher waren sowohl für *client_userid* als auch für *server_userid* DBA- oder REMOTE DBA-Berechtigungen erforderlich.

In SQL Anywhere 16 und später muss *server_userid* ausreichende Privilegien für die Synchronisation haben. *Server_userid* muss mindestens die SYS_RUN_REPLICATION_ROLE-Systemrolle haben, aber für die Synchronisation können auch weitere Privilegien erforderlich sein. *Client_userid* muss eine der folgenden Bedingungen erfüllen:

- Mit *server_userid* übereinstimmen
- Die SYS_AUTH_DBA_ROLE-Systemrolle haben
- Eine auf *server_userid* basierende benutzererweiterte Rolle haben, z.B. CREATE ROLE FOR USER Serverbenutzer-ID, GRANT Serverbenutzer-ID TO Clientbenutzer-ID

Die letzte Option stellt sicher, dass *client_userid* mindestens ebenso viele Systemprivilegien hat wie *server_userid*. *server_userid* kann jedoch auch Privilegien auf Objektebene haben, die seiner Rolle nicht erteilt wurden. In diesem Fall hat *client_userid* diese Privilegien nicht. Wenn diese Privilegien während der Synchronisation verwendet werden, hat *client_userid* praktisch seine Privilegien erhöht, um die

Synchronisation durchzuführen. Falls dies nicht akzeptabel ist, stellen Sie sicher, dass alle Privilegien von *server_userid*s auf Objektebene den entsprechenden benutzererweiterten Rollen erteilt werden.

Siehe auch

- [DbmlsyncClient.StartServer-Methode \[Dbmlsync C++\] auf Seite 280](#)
- [DbmlsyncClient.GetErrorInfo-Methode \[Dbmlsync C++\] auf Seite 274](#)

Disconnect-Methode

Unterbricht die Verbindung mit dem dbmlsync-Server, die mit der Connect-Methode hergestellt wurde.

Syntax

```
public virtual bool Disconnect(void)
```

Rückgabe

TRUE, wenn die Verbindung mit dem Server unterbrochen wurde sonst FALSE. Wenn FALSE zurückgegeben wird, können Sie die Methode GetErrorInfo aufrufen, um weitere Informationen über den Fehler zu erhalten.

Bemerkungen

Sie sollten immer Disconnect aufrufen, wenn Sie eine Verbindung nicht mehr benötigen.

Siehe auch

- [DbmlsyncClient.Connect-Methode \[Dbmlsync C++\] auf Seite 271](#)
- [DbmlsyncClient.GetErrorInfo-Methode \[Dbmlsync C++\] auf Seite 274](#)

Fini-Methode

Gibt alle von dieser Instanz der Klasse verwendeten Ressourcen frei.

Syntax

```
public virtual bool Fini(void)
```

Rückgabe

TRUE, wenn die Klasseninstanz erfolgreich beendet wurde sonst FALSE. Wenn FALSE zurückgegeben wird, können Sie die Methode GetErrorInfo aufrufen, um weitere Informationen über den Fehler zu erhalten.

Bemerkungen

Diese Methode muss aufgerufen werden, bevor Sie die DbmlSyncClient-Klasseninstanz löschen können.

Hinweis

Verwenden Sie die Disconnect-Methode, um die Verbindung zu einem beliebigen verbundenen Server vor dem Beenden der Klasseninstanz zu trennen.

Siehe auch

- [DbmlsyncClient.Disconnect-Methode \[Dbmlsync C++\] auf Seite 273](#)
- [DbmlsyncClient.GetErrorInfo-Methode \[Dbmlsync C++\] auf Seite 274](#)

FreeEventInfo-Methode

Gibt Speicher frei, der einer DBSC_Event-Struktur zugeordnet ist, die von der GetEvent-Methode zurückgegeben wurde.

Syntax

```
public virtual bool FreeEventInfo(DBSC_Event * event)
```

Parameter

- **event** Ein Zeiger auf die DBSC_Event-Struktur, deren Speicher freigegeben werden soll.

Rückgabe

TRUE, wenn der Speicher erfolgreich freigegeben werden konnte sonst FALSE. Wenn FALSE zurückgegeben wird, können Sie die Methode GetErrorInfo aufrufen, um weitere Informationen über den Fehler zu erhalten.

Bemerkungen

FreeEventInfo muss für jede DBSC_Event-Struktur aufgerufen werden, die von der GetEvent-Methode zurückgegeben wird.

Siehe auch

- [DBSC_Event-Struktur \[Dbmlsync C++\] auf Seite 290](#)
- [DbmlsyncClient.GetEvent-Methode \[Dbmlsync C++\] auf Seite 275](#)
- [DbmlsyncClient.GetErrorInfo-Methode \[Dbmlsync C++\] auf Seite 274](#)

GetErrorInfo-Methode

Ruft zusätzliche Informationen über den Fehler ab, nachdem eine Methode der DbmlsyncClient-Klasse einen Fehlercode zurückgibt.

Syntax

```
public virtual const DBSC_ErrorInfo * GetErrorInfo(void)
```

Rückgabe

Zeiger auf eine DBSC_ErrorInfo-Struktur, die Informationen über den Fehler enthält. Der Inhalt dieser Struktur kann beim nächsten Aufruf einer beliebigen Klassenmethode überschrieben werden.

Siehe auch

- [DBSC_ErrorType-Enumeration \[Dbmlsync C++\] auf Seite 283](#)
- [DBSC_ErrorInfo-Struktur \[Dbmlsync C++\] auf Seite 289](#)
- [DbmlsyncClient.GetErrorInfo-Methode \[Dbmlsync C++\] auf Seite 274](#)

GetEvent-Methode

Ruft das nächste Rückmeldungseignis für vom Client angeforderte Synchronisationen ab.

Syntax

```
public virtual DBSC_GetEventRet GetEvent(  
    DBSC_Event ** event,  
    unsigned timeout  
)
```

Parameter

- **event** Wenn der Rückgabewert DBSC_GETEVENT_OK ist, wird der event-Parameter mit einem Zeiger auf eine DBSC_Event-Struktur gefüllt, die Informationen über das abgefragte Ereignis enthält. Wenn Sie die Ereignisstruktur nicht mehr benötigen, müssen Sie die Methode FreeEventInfo aufrufen, um den ihr zugeordneten Speicher freizugeben.
- **timeout** Gibt die maximale Anzahl von Millisekunden zurück, die gewartet werden soll, wenn kein Ereignis unmittelbar zur Rückgabe bereitsteht. Verwenden Sie DBSC_INFINITY, um unbegrenzt auf eine Antwort zu warten.

Rückgabe

Ein Wert aus der DBSC_GetEventRet-Enumeration. Wenn DBSC_GETEVENT_FAILED zurückgegeben wird, können Sie die Methode GetErrorInfo aufrufen, um weitere Informationen über den Fehler zu erhalten.

Bemerkungen

Rückmeldungseignisse enthalten Informationen, etwa Nachrichten, die durch die Synchronisation generiert wurden, Daten für die Aktualisierung des Fortschrittsbalkens und Synchronisationszyklus-Benachrichtigungen.

Wenn der dbmlsync-Server eine Synchronisation ausführt, generiert er eine Reihe von Ereignissen, die Informationen über den Verarbeitungsfortschritt der Synchronisation enthalten. Diese Ereignisse werden vom Server an die Klasse DbmlsyncClient gesendet, die sie in eine Warteschlange stellt. Wenn die GetEvent-Methode aufgerufen wird, wird die nächste Methode in der Warteschlange zurückgegeben, sofern dort eine wartet.

Wenn keine Ereignisse in der Warteschlange vorhanden sind, wartet diese Methode, bis ein Ereignis verfügbar ist oder bis der angegebene Timeout abgelaufen ist, bevor sie beendet wird.

Die für eine Synchronisation generierten Ereignistypen können mit Eigenschaften gesteuert werden.

Siehe auch

- [DBSC_GetEventRet-Enumeration \[Dbmlsync C++\] auf Seite 288](#)
- [DBSC_Event-Struktur \[Dbmlsync C++\] auf Seite 290](#)
- [DbmlsyncClient.FreeEventInfo-Methode \[Dbmlsync C++\] auf Seite 274](#)
- [DbmlsyncClient.SetProperty-Methode \[Dbmlsync C++\] auf Seite 278](#)
- [DbmlsyncClient.GetErrorInfo-Methode \[Dbmlsync C++\] auf Seite 274](#)

GetProperty-Methode

Ruft den aktuellen Wert einer Eigenschaft ab.

Syntax

```
public virtual bool GetProperty(const char * name, char * value)
```

Parameter

- **name** Der Name der abzurufenden Eigenschaft. Eine Liste der gültigen Eigenschaftsnamen finden Sie unter "SetProperty".
- **value** Ein Puffer von mindestens DBSC_MAX_PROPERTY_LEN Byte, in dem der Wert der Eigenschaft gespeichert wird.

Rückgabe

TRUE, wenn die Eigenschaft erfolgreich empfangen wurde sonst FALSE. Wenn FALSE zurückgegeben wird, können Sie die Methode GetErrorInfo aufrufen, um weitere Informationen über den Fehler zu erhalten.

Siehe auch

- [DbmlsyncClient.SetProperty-Methode \[Dbmlsync C++\] auf Seite 278](#)
- [DbmlsyncClient.GetErrorInfo-Methode \[Dbmlsync C++\] auf Seite 274](#)

Init-Methode

Initialisiert eine Instanz der DbmlsyncClient-Klasse.

Syntax

```
public virtual bool Init(void)
```

Rückgabe

TRUE, wenn die Klasseninstanz erfolgreich initialisiert wurde sonst FALSE. Wenn FALSE zurückgegeben wird, können Sie die Methode GetErrorInfo aufrufen, um weitere Informationen über den Fehler zu erhalten.

Bemerkungen

Diese Methode muss aufgerufen werden, nachdem die DbmlSyncClient-Klasseninstanz erstellt wurde. Andere DbmlSyncClient-Methoden können erst aufgerufen werden, wenn Sie die Instanz erfolgreich initialisiert haben.

Siehe auch

- [DbmlsyncClient.InstantiateClient-Methode \[Dbmlsync C++\] auf Seite 277](#)
- [DbmlsyncClient.GetErrorInfo-Methode \[Dbmlsync C++\] auf Seite 274](#)

InstantiateClient-Methode

Erstellt eine Instanz der dbmlsync-Clientklasse, mit der Synchronisationen gesteuert werden können.

Syntax

```
public static DbmlsyncClient * InstantiateClient(void)
```

Rückgabe

Zeiger auf die neu erstellte Instanz. Gibt NULL zurück, wenn ein Fehler auftritt.

Bemerkungen

Der von dieser Methode zurückgegebene Zeiger kann verwendet werden, um die übrigen Methoden in der Klasse aufzurufen. Sie können die Instanz entfernen, indem Sie den Standardoperator für Löschvorgänge an dem Zeiger aufrufen.

Ping-Methode

Sendet eine Ping-Anforderung an den dbmlsync-Server, um zu prüfen, ob der Server aktiv ist und auf Anforderungen antwortet.

Syntax

```
public virtual bool Ping(unsigned timeout)
```

Parameter

- **timeout** Die maximale Anzahl von Millisekunden, die gewartet werden soll, bis der Server auf die Ping-Anforderung antwortet. Verwenden Sie DBSC_INFINITY, um unbegrenzt auf eine Antwort zu warten.

Rückgabe

TRUE, wenn eine Antwort auf die Ping-Anforderung vom Server empfangen wurde sonst FALSE. Wenn FALSE zurückgegeben wird, können Sie die Methode GetErrorInfo aufrufen, um weitere Informationen über den Fehler zu erhalten.

Bemerkungen

Sie müssen mit dem Server verbunden sein, um diese Methode aufrufen zu können.

Siehe auch

- [DbmlsyncClient.GetErrorInfo-Methode \[Dbmlsync C++\] auf Seite 274](#)

SetProperty-Methode

Legt verschiedene Eigenschaften fest, um das Verhalten der Klasseninstanz zu ändern.

Syntax

```
public virtual bool SetProperty(const char * name, const char * value)
```

Parameter

- **name** Der Name der festzulegenden Eigenschaft. Eine Liste der gültigen Eigenschaftsnamen finden Sie in der Tabelle.
- **value** Der Wert, auf den die Eigenschaft gesetzt werden soll. Die Zeichenfolge muss weniger als DBCS_MAX_PROPERTY_LEN Byte lang sein.

Rückgabe

TRUE, wenn die Eigenschaft erfolgreich gesetzt werden konnte; sonst FALSE. Wenn FALSE zurückgegeben wird, können Sie die Methode GetErrorInfo aufrufen, um weitere Informationen über den Fehler zu erhalten.

Bemerkungen

Änderungen von Eigenschaftswerten haben nur Auswirkungen auf Synchronisationsanforderungen, die nach der Änderung des Eigenschaftswerts vorgenommen werden.

Mit der **server path**-Eigenschaft kann das Verzeichnis festgelegt werden, von dem aus der Client dbmlsync.exe starten soll, wenn die StartServer-Methode aufgerufen wird. Wenn diese Eigenschaft nicht festgelegt ist, wird dbmlsync.exe mithilfe der Umgebungsvariablen PATH gefunden. Wenn mehrere Versionen von SQL Anywhere auf Ihrem Computer installiert sind, ist es empfehlenswert, den Speicherort von dbmlsync.exe mit der **server path**-Eigenschaft anzugeben, da die Umgebungsvariable PATH möglicherweise die dbmlsync.exe einer anderen installierten Version von SQL Anywhere findet. Beispiel:

```
ret = cli->SetProperty("server path", "c:\\sa16\\bin32");
```

Die Eigenschaften steuern die Ereignistypen, die von der GetEvent-Methode zurückgegeben werden. Durch das Deaktivieren nicht benötigter Ereignisse ist es möglich, die Performance zu steigern. Ein Ereignistyp wird aktiviert, indem Sie die entsprechende Eigenschaft auf "1" setzen, und deaktiviert, indem Sie die Eigenschaft auf "0" setzen.

Die folgende Tabelle enthält die verfügbaren Eigenschaftsnamen und die Ereignistypen, die die einzelnen Namen steuern:

Eigenschaftsname	Gesteuerte Ereignistypen	Standardwert
enable errors	DBSC_EVENTTYPE_ERROR_MSG	1

Eigenschaftsname	Gesteuerte Ereignistypen	Standardwert
enable warnings	DBSC_EVENTTYPE_WARNING_MSG	1
enable info msgs	DBSC_EVENTTYPE_INFO_MSG	1
enable progress	DBSC_EVENTTYPE_PROGRESS_INDEX	0
enable progress text	DBSC_EVENTTYPE_PROGRESS_TEXT	0
enable title	DBSC_EVENTTYPE_TITLE	0
enable sync start and done	DBSC_EVENTTYPE_SYNC_START DBSC_EVENTTYPE_SYNC_DONE	1
enable status	DBSC_EVENTTYPE_ML_CONNECT DBSC_EVENTTYPE_UPLOAD_COMMITTED DBSC_EVENTTYPE_DOWNLOAD_COMMITTED	1

Siehe auch

- [DbmlsyncClient.StartServer-Methode \[Dbmlsync C++\] auf Seite 280](#)
- [DbmlsyncClient.GetEvent-Methode \[Dbmlsync C++\] auf Seite 275](#)
- [DbmlsyncClient.GetProperty-Methode \[Dbmlsync C++\] auf Seite 276](#)
- [DbmlsyncClient.GetErrorInfo-Methode \[Dbmlsync C++\] auf Seite 274](#)

ShutdownServer-Methode

Führt den dbmlsync-Server herunter, mit dem der Client verbunden ist.

Syntax

```
public virtual bool ShutdownServer(DBSC_ShutdownType how)
```

Parameter

- **how** Gibt die Dringlichkeit für das Herunterfahren des Servers an. Unterstützte Werte werden in der DBSC_ShutdownType-Enumeration aufgeführt.

Rückgabe

TRUE, wenn eine Anforderung zum Herunterfahren erfolgreich an den Server gesendet wurde sonst FALSE. Wenn FALSE zurückgegeben wird, können Sie die Methode GetErrorInfo aufrufen, um weitere Informationen über den Fehler zu erhalten.

Bemerkungen

Die Shutdown-Methode wird sofort beendet, doch es kann zu einer Verzögerung kommen, bevor der Server tatsächlich heruntergefahren wird.

Die Methode WaitForServerShutdown kann verwendet werden, um zu warten, bis der Server tatsächlich heruntergefahren wird.

Hinweis

Sie sollten jedoch die Disconnect-Methode nach dem Aufruf von ShutdownServer verwenden.

Siehe auch

- [DBSC_ShutdownType-Enumeration \[Dbmlsync C++\] auf Seite 288](#)
- [DbmlsyncClient.Disconnect-Methode \[Dbmlsync C++\] auf Seite 273](#)
- [DbmlsyncClient.WaitForServerShutdown-Methode \[Dbmlsync C++\] auf Seite 282](#)
- [DbmlsyncClient.GetErrorInfo-Methode \[Dbmlsync C++\] auf Seite 274](#)

StartServer-Methode

Startet einen neuen dbmlsync-Server, wenn noch keiner am angegebenen Port auf Nachrichten wartet.

Syntax

```
public virtual bool StartServer(  
    unsigned port,  
    const char * cmdline,  
    unsigned timeout,  
    DBSC_StartType * starttype  
)
```

Parameter

- **port** Der TCP-Port, an dem auf einen vorhandenen dbmlsync-Server geprüft werden soll. Wenn ein neuer Server gestartet wird, wird er so eingestellt, dass er an diesem Port auf Verbindungen wartet.
- **cmdline** Eine gültige Befehlszeile zum Starten eines dbmlsync-Servers. Die Befehlszeile kann nur die folgenden Optionen enthalten, die die gleiche Bedeutung wie für das Dienstprogramm dbmlsync haben: -a, -c, -dl, -do, -ek, -ep, -k, -l, -o, -os, -ot, -p, -pc+, -pc-, -pd, -pp, -q, -qi, -qc, -sc, -sp, -uc, -ud, -ui, -um, -un, -ux, -v[cnoprst], -wc, -wh. Die Option -c muss angegeben werden.
- **timeout** Die maximale Dauer in Millisekunden, die nach dem Start eines dbmlsync-Servers gewartet werden soll, bis der Server bereit zur Entgegennahme von Anforderungen ist. Verwenden Sie DBSC_INFINITY, um unbegrenzt auf eine Antwort zu warten.
- **starttype** Ein OUT-Parameter, der anzeigt, ob der Server gefunden oder gestartet wurde. Wenn starttype beim Eintritt Nicht-NULL ist und StartServer TRUE zurückgibt, wird beim Beenden die Variable, auf die starttype zeigt, auf einen der Werte der DBSC_StartType-Enumeration gesetzt.

Rückgabe

TRUE, wenn der Server bereits lief oder erfolgreich gestartet werden konnte; sonst FALSE. Wenn FALSE zurückgegeben wird, können Sie die Methode `GetErrorInfo` aufrufen, um weitere Informationen über den Fehler zu erhalten.

Bemerkungen

Wenn ein Server vorhanden ist, setzt diese Methode den `starttype`-Parameter auf `DBSC_SS_ALREADY_RUNNING` und wird ohne weitere Aktion beendet. Wenn kein Server gefunden wird, startet die Methode mit den im Argument `cmdline` angegebenen Optionen einen neuen Server und wartet, dass der Server Anforderungen akzeptiert, bevor die Methode beendet wird.

Auf Windows Mobile-Geräten ist es üblicherweise notwendig, die **server path**-Eigenschaft zu setzen, bevor `StartServer` erfolgreich aufgerufen werden kann. Die **server path**-Eigenschaft muss in den folgenden Fällen nicht gesetzt werden:

- Ihre Anwendung befindet sich in demselben Verzeichnis wie *dbmlsync.exe*.
- *dbmlsync.exe* befindet sich im Windows-Verzeichnis.

Siehe auch

- [DbmlsyncClient.GetErrorInfo-Methode \[Dbmlsync C++\] auf Seite 274](#)

Sync-Methode

Fordert an, dass der `dbmlsync`-Server eine Synchronisation ausführt.

Syntax

```
public virtual DBSC_SyncHdl Sync(
    const char * profile_name,
    const char * extra_opts
)
```

Parameter

- **profile_name** Der Name eines Synchronisationsprofil, das in der entfernten Datenbank definiert ist, welche die Optionen für die Synchronisation enthält. Wenn `profile_name` NULL ist, wird kein Profil verwendet und der Parameter `extra_opts` muss alle Optionen für die Synchronisation enthalten.
- **extra_opts** Eine Zeichenfolge, die nach denselben Regeln gebildet wird, die auch beim Definieren einer Optionszeichenfolge für ein Synchronisationsprofil verwendet werden. Dabei handelt es sich um eine Zeichenfolge, die durch Semikola getrennte Elemente in der Form `<Optionsname>=<Optionswert>` enthält. Wenn `profile_name` Nicht-NULL ist, werden die von `extra_opts` festgelegten Optionen zu den Optionen hinzugefügt, die bereits im von `profile_name` angegebenen Synchronisationsprofil enthalten sind. Wenn eine in der Zeichenfolge enthaltene Option im Profil bereits vorhanden ist, ersetzt der Wert aus der Zeichenfolge den bereits im Profil gespeicherten Wert. Wenn `profile_name` NULL ist, muss `extra_opts` alle Optionen für die Synchronisation festlegen. Siehe „[CREATE SYNCHRONIZATION PROFILE-Anweisung \[MobiLink\]](#)“ [[SQL Anywhere Server - SQL-Referenzhandbuch](#)].

Rückgabe

Ein DBSC_SyncHdl-Wert, der diese Synchronisationsanforderung eindeutig identifiziert und nur gültig ist, bis der Client vom Server getrennt wird. Gibt NULL_SYNCHDL zurück, wenn ein Fehler verhindert, dass die Synchronisationsanforderung erstellt wird. Wenn NULL_SYNCHDL zurückgegeben wird, können Sie die Methode GetErrorInfo aufrufen, um weitere Informationen über den Fehler zu erhalten.

Bemerkungen

Sie müssen mit dem Server verbunden sein, um diese Methode aufrufen zu können. Mindestens einer der beiden Parameter profile_name oder extra_opts muss Nicht-NULL sein.

Der Rückgabewert gibt die Synchronisationsanforderung an und kann verwendet werden, um die Anforderung abzubrechen oder die von der Synchronisation zurückgegeben Ereignisse zu verarbeiten.

Siehe auch

- [DbmlsyncClient.GetErrorInfo-Methode \[Dbmlsync C++\] auf Seite 274](#)

WaitForServerShutdown-Methode

Kehrt zurück, wenn der Server heruntergefahren oder der Timeout abgelaufen ist, je nach dem, welches Ereignis früher eintritt.

Syntax

```
public virtual bool WaitForServerShutdown(unsigned timeout)
```

Parameter

- **timeout** Zeigt die maximale Dauer in Millisekunden an, die gewartet werden soll, bis der Server heruntergefahren wird. Verwenden Sie DBSC_INFINITY, um unbegrenzt auf eine Antwort zu warten.

Rückgabe

TRUE, wenn die Methode beendet wurde, weil der Server heruntergefahren wurde sonst FALSE. Wenn FALSE zurückgegeben wird, können Sie die Methode GetErrorInfo aufrufen, um weitere Informationen über den Fehler zu erhalten.

Bemerkungen

WaitForServerShutdown kann nur aufgerufen werden, nachdem die Methode ShutdownServer aufgerufen wurde.

Siehe auch

- [DbmlsyncClient.GetErrorInfo-Methode \[Dbmlsync C++\] auf Seite 274](#)

DBSC_CancelRet-Enumeration

Gibt das Ergebnis des Versuchs an, eine Synchronisation abzubrechen.

Syntax

```
public enum DBSC_CancelRet
```

Mitglieder

Mitgliedsname	Beschreibung	Wert
DBSC_CAN- CEL_OK_QUEUED	Synchronisation abgebrochen, die sich in der Warteschlange befand.	1
DBSC_CANCEL_OK_ACTIVE	Aktive Synchronisation abgebrochen.	2
DBSC_CANCEL_FAILED	Synchronisation konnte nicht abgebrochen werden.	3

Siehe auch

- [DbmlsyncClient.CancelSync-Methode \[Dbmlsync C++\] auf Seite 270](#)

DBSC_ErrorType-Enumeration

Gibt den Grund für einen fehlgeschlagenen Methodenaufruf an.

Syntax

```
public enum DBSC_ErrorType
```

Mitglieder

Mitgliedsname	Beschreibung	Wert
DBSC_ERR_OK	Es ist kein Fehler aufgetreten.	1
DBSC_ERR_NOT_INITIALIZED	Die Klasse wurde nicht durch den Aufruf der init-Methode initialisiert.	2
DBSC_ERR_ALREADY_INITIALIZED	Die init-Methode wurde für eine Klasse aufgerufen, die bereits initialisiert war.	3
DBSC_ERR_NOT_CONNECTED	Es besteht keine Verbindung mit einem dbmlsync-Server.	4
DBSC_ERR_CANT_RESOLVE_HOST	Hostinformationen konnten nicht aufgelöst werden.	5
DBSC_ERR_CONNECT_FAILED	Verbindung mit dem dbmlsync-Server ist fehlgeschlagen.	6

Mitgliedsname	Beschreibung	Wert
DBSC_ERR_INITIALIZING_TCP_LAYER	Bei der Initialisierung der TCP-Schicht ist ein Fehler aufgetreten.	7
DBSC_ERR_ALREADY_CONNECTED	Die Connect-Methode ist fehlgeschlagen, da bereits eine Verbindung bestand.	8
DBSC_ERR_PROTOCOL_ERROR	Dies ist ein interner Fehler.	9
DBSC_ERR_CONNECTION_REJECTED	Die Verbindung wurde vom dbmlsync-Server zurückgewiesen. str1 zeigt auf eine vom Server zurückgegebene Zeichenfolge, die weitere Informationen darüber bereitstellen kann, warum der Verbindungsversuch abgelehnt wurde.	10
DBSC_ERR_TIMED_OUT	Beim Warten auf eine Antwort des Servers ist der Timeout abgelaufen.	11
DBSC_ERR_STILL_CONNECTED	Die Fini-Methode konnte nicht für die Klasse aufgerufen werden, da sie noch mit dem Server verbunden ist.	12
DBSC_ERR_SYNC_NOT_CANCELED	Der Server konnte die Synchronisationsanforderung nicht abbrechen. Dies liegt wahrscheinlich daran, dass die Synchronisation bereits ausgeführt wurde.	14
DBSC_ERR_INVALID_VALUE	Ein ungültiger Eigenschaftswert wurde an die SetProperty-Methode übergeben.	15
DBSC_ERR_INVALID_PROP_NAME	Der angegebene Eigenschaftsname ist ungültig.	16
DBSC_ERR_VALUE_TOO_LONG	Der Eigenschaftswert ist zu lang; Eigenschaften müssen weniger als DBCS_MAX_PROPERTY_LEN Byte lang sein.	17
DBSC_ERR_SERVER_SIDE_ERROR	Ein serverseitiger Fehler ist beim Abbrechen oder Hinzufügen von einer Synchronisation aufgetreten. str1 zeigt auf eine vom Server zurückgegebene Zeichenfolge, die weitere Informationen über den Fehler bereitstellen kann.	18

Mitgliedsname	Beschreibung	Wert
DBSC_ERR_CREATE_PROCESS_FAILED	Es konnte kein neuer dbmlsync-Server gestartet werden.	20
DBSC_ERR_READ_FAILED	Ein TCP-Fehler ist beim Lesen von Daten vom dbmlsync-Server aufgetreten.	21
DBSC_ERR_WRITE_FAILED	Ein TCP-Fehler ist beim Senden von Daten an den dbmlsync-Server aufgetreten.	22
DBSC_ERR_NO_SERVER_RESPONSE	Eine Antwort vom Server, die für die Ausführung der angeforderten Aktion erforderlich ist, konnte nicht empfangen werden.	23
DBSC_ERR_UID_OR_PWD_TOO_LONG	Der angegebene Wert für UID oder PWD ist zu lang.	24
DBSC_ERR_UID_OR_PWD_NOT_VALID	Der angegebene Wert für UID oder PWD ist ungültig oder hat keine ausreichenden Privilegien, um eine Verbindung herzustellen.	25
DBSC_ERR_INVALID_PARAMETER	Einer der an die Funktion übergebenen Parameter war ungültig.	26
DBSC_ERR_WAIT_FAILED	Während darauf gewartet wurde, dass der Server herunterfährt, ist ein Fehler aufgetreten.	27
DBSC_ERR_SHUTDOWN_NOT_CALLED	Die WaitForServerShutdown-Methode wurde aufgerufen, ohne dass zuerst die ShutdownServer-Methode aufgerufen wurde.	28
DBSC_ERR_NO_SYNC_ACK	<p>Es wurde eine Synchronisationsanforderung an den Server gesendet, jedoch wurde keine Bestätigung empfangen. Es gibt keine Möglichkeit zu erkennen, ob der Server die Anforderung empfangen hat.</p> <p>hdl1 ist das Handle für die gesendete Synchronisationsanforderung. Wenn der Server die Anforderung empfangen hat, kann dieses Handle verwendet werden, um die Ereignisse für die mit der GetEvent-Methode abgerufene Synchronisation zu identifizieren.</p>	29

Mitgliedsname	Beschreibung	Wert
DBSC_ERR_ACTIVE_SYNC_NOT_CAN-CELED	Der Server konnte die Synchronisationsanforderung nicht abbrechen, da die Synchronisation bereits aktiv war.	30
DBSC_ERR_DEAD_SERVER	Beim Starten des dbmlsync-Servers ist ein Fehler aufgetreten. Der Server wird nun heruntergefahren. Verwenden Sie die dbmlsync-Option -o zum Aufzeichnen der Fehlermeldung in einer Datei.	31

DBSC_EventType-Enumeration

Zeigt den Typ des Ereignisses an, das von einer Synchronisation generiert wurde.

Syntax

```
public enum DBSC_EventType
```

Mitglieder

Mitgliedsname	Beschreibung	Wert
DBSC_EVENT-TYPE_ERROR_MSG	Die Synchronisation hat einen Fehler generiert; str1 zeigt auf den Fehlertext.	1
DBSC_EVENT-TYPE_WARNING_MSG	Die Synchronisation hat eine Warnung generiert; str1 zeigt auf den Text der Warnung.	2
DBSC_EVENT-TYPE_INFO_MSG	Die Synchronisation hat eine Meldung generiert; str1 zeigt auf den Meldungstext.	3
DBSC_EVENT-TYPE_PROGRESS_INDEX	Stellt Informationen für die Aktualisierung des Fortschrittsbalkens bereit; val1 enthält den neuen Wert des Verarbeitungsfortschritts. Dieser Wert kann im Bereich von 0 bis 1000 liegen, wobei 0 für 0 % erledigt und 1000 für 100 % erledigt steht.	4
DBSC_EVENT-TYPE_PROGRESS_TEXT	Der dem Fortschrittsbalken zugeordnete Text wurde aktualisiert; str1 zeigt auf den neuen Wert.	6
DBSC_EVENT-TYPE_TITLE	Der Titel des Synchronisationsfensters/Steuerelements wurde geändert; str1 zeigt auf den neuen Titel.	7

Mitgliedsname	Beschreibung	Wert
DBSC_EVENT-TYPE_SYNC_START	Die Synchronisation wurde gestartet. Es gibt keine zusätzlichen Informationen zu diesem Ereignis.	8
DBSC_EVENT-TYPE_SYNC_DONE	Die Synchronisation ist abgeschlossen; val1 enthält den Beendigungscode aus der Synchronisation. Der Wert 0 zeigt Erfolg an. Ein Wert ungleich Null zeigt an, dass die Synchronisation fehlgeschlagen ist.	9
DBSC_EVENT-TYPE_ML_CONNECT	Eine Verbindung zum MobiLink-Server wurde hergestellt; str1 gibt das verwendete Kommunikationsprotokoll an und str2 enthält die verwendeten Netzwerkprotokolloptionen.	10
DBSC_EVENT-TYPE_UPLOAD_COMMITTED	Der MobiLink-Server bestätigt, dass der Upload erfolgreich in der konsolidierten Datenbank festgeschrieben wurde.	11
DBSC_EVENT-TYPE_DOWNLOAD_COMMITTED	Der Download wurde erfolgreich in der entfernten Datenbank festgeschrieben. val1 enthält die Anzahl der festgeschriebenen Einfüge-/Aktualisierungsvorgänge. val2 enthält die Anzahl der festgeschriebenen Löschvorgänge.	12
DBSC_EVENT-TYPE_UPLOAD_START	Der entfernte Client hat mit dem Hochladen auf den Server begonnen.	13
DBSC_EVENT-TYPE_UPLOAD_SENT	Der entfernte Client hat das Senden eines Uploadsegments an den Server abgeschlossen. Für inkrementelle und transaktionale Uploads wird beim Senden jedes Upload-Segments ein Ereignis generiert. val1 enthält die Anzahl der gesendeten Einfügevorgänge. val2 enthält die Anzahl der gesendeten Aktualisierungsvorgänge. val3 enthält die Anzahl der gesendeten Löschvorgänge.	14
DBSC_EVENT-TYPE_DOWNLOAD_START	Der entfernte Client hat begonnen, den vom Server erhaltenen Download zu verarbeiten.	15

Siehe auch

- [DBSC_Event-Struktur \[Dbmlsync C++\] auf Seite 290](#)
- [DbmlsyncClient.GetEvent-Methode \[Dbmlsync C++\] auf Seite 275](#)

DBSC_GetEventRet-Enumeration

Gibt das Ergebnis eines Versuchs zur Abfrage eines Ereignisses an.

Syntax

```
public enum DBSC_GetEventRet
```

Mitglieder

Mitgliedsname	Beschreibung	Wert
DBSC_GETEVENT_OK	Zeigt an, dass ein Ereignis erfolgreich abgerufen wurde.	1
DBSC_GETE- VENT_TIMED_OUT	Zeigt an, dass der Timeout abgelaufen ist, ohne dass ein Ereignis für die Rückgabe verfügbar wurde.	2
DBSC_GETEVENT_FAI- LED	Zeigt an, dass aufgrund einer Fehlerbedingung kein Ereignis zurückgegeben wurde.	3

Siehe auch

- [DbmlsyncClient.GetEvent-Methode \[Dbmlsync C++\] auf Seite 275](#)

DBSC_ShutdownType-Enumeration

Gibt an, wie dringend der Server heruntergefahren werden sollte.

Syntax

```
public enum DBSC_ShutdownType
```

Mitglieder

Mitgliedsname	Beschreibung	Wert
DBSC_SHUT- DOWN_ON_EMP- TY_QUEUE	Zeigt an, dass der Server alle noch anstehenden Synchronisationsanforderungen ausführen und dann heruntergefahren werden soll. Wenn der Server die Anforderung zum Herunterfahren empfängt, nimmt er keine weiteren Synchronisationsanforderungen mehr an.	1
DBSC_SHUT- DOWN_CLEANLY	Zeigt an, dass der Server so schnell wie möglich sauber heruntergefahren werden soll. Noch anstehende Synchronisationsanforderungen werden nicht ausgeführt, und wenn gerade eine Synchronisation ausgeführt wird, kann sie unterbrochen werden.	2

Siehe auch

- [DbmlsyncClient.ShutdownServer-Methode \[Dbmlsync C++\] auf Seite 279](#)

DBSC_StartType-Enumeration

Zeigt die Aktion an, die bei einem Versuch, den dbmlsync-Server zu starten, ausgeführt wurde.

Syntax

```
public enum DBSC_StartType
```

Mitglieder

Mitgliedsname	Beschreibung	Wert
DBSC_SS_STARTED	Zeigt an, dass ein neuer dbmlsync-Server gestartet wurde.	1
DBSC_SS_ALREADY_RUNNING	Zeigt an, dass ein vorhandener dbmlsync-Server gefunden und daher kein neuer Server gestartet wurde.	2

Siehe auch

- [DbmlsyncClient.StartServer-Methode \[Dbmlsync C++\] auf Seite 280](#)

DBSC_ErrorInfo-Struktur

Enthält Informationen über einen Fehler eines vorherigen Methodenaufrufs.

Syntax

```
public typedef struct DBSC_ErrorInfo
```

Mitglieder

Mitgliedsname	Typ	Beschreibung
hdl	DBSC_SyncHdl	Enthält zusätzliche Informationen über den Fehler. Die Bedeutung dieser Informationen hängt vom Wert der type-Variable ab.
str	const char *	Enthält zusätzliche Informationen über den Fehler. Die Bedeutung dieser Informationen hängt vom Wert der type-Variable ab.

Mitgliedsname	Typ	Beschreibung
str2	const char *	Enthält zusätzliche Informationen über den Fehler. Die Bedeutung dieser Informationen hängt vom Wert der type-Variable ab.
type	DBSC_ErrorType	Enthält einen Wert, der den Grund für einen Fehlschlag angibt. Unterstützte Werte werden in der DBSC_ErrorType-Enumeration aufgeführt.
val1	long int	Enthält zusätzliche Informationen über den Fehler. Die Bedeutung dieser Informationen hängt vom Wert der type-Variable ab.
val2	long int	Enthält zusätzliche Informationen über den Fehler. Die Bedeutung dieser Informationen hängt vom Wert der type-Variable ab.

Bemerkungen

str1, str2, val1, val2 und hdl1 enthalten zusätzliche Informationen über den Fehler. Ihre Bedeutung hängt vom Fehlertyp ab. Die folgenden Fehlertypen verwenden Felder in dieser Struktur zum Speichern zusätzlicher Informationen:

- DBSC_ERR_CONNECTION_REJECTED
- DBSC_ERR_SERVER_SIDE_ERROR
- DBSC_ERR_NO_SYNC_ACK

Siehe auch

- [DBSC_ErrorType-Enumeration \[Dbmlsync C++\] auf Seite 283](#)

DBSC_Event-Struktur

Enthält Informationen über ein Ereignis, das durch eine Synchronisation generiert wurde.

Syntax

```
public typedef struct DBSC_Event
```

Mitglieder

Mitgliedsname	Typ	Beschreibung
data	void *	Enthält zusätzliche Informationen zum Ereignis. Die Bedeutung dieser Informationen hängt vom Wert der type-Variable ab.
hdl	DBSC_SyncHdl	Gibt die Synchronisation an, die das Ereignis generiert hat. Dieser Wert entspricht dem Wert, der von der Sync-Methode zurückgegeben wurde.
str1	const char *	Enthält zusätzliche Informationen zum Ereignis. Die Bedeutung dieser Informationen hängt vom Wert der type-Variable ab.
str2	const char *	Enthält zusätzliche Informationen zum Ereignis. Die Bedeutung dieser Informationen hängt vom Wert der type-Variable ab.
type	DBSC_EventType	Gibt den Typ des gemeldeten Ereignisses an.
val1	long int	Enthält zusätzliche Informationen zum Ereignis. Die Bedeutung dieser Informationen hängt vom Wert der type-Variable ab.
val2	long int	Enthält zusätzliche Informationen zum Ereignis. Die Bedeutung dieser Informationen hängt vom Wert der type-Variable ab.
val3	long int	Enthält zusätzliche Informationen zum Ereignis. Die Bedeutung dieser Informationen hängt vom Wert der type-Variable ab.

Siehe auch

- [DBSC_EventType-Enumeration \[Dbmlsync C++\] auf Seite 286](#)

Dbmlsync .NET-API-Referenz

Namespace

```
iAnywhere.MobiLink.Client
```

Beispiel

Das untenstehende Beispiel zeigt eine typische Anwendung, in der die .NET-Version der Dbmlsync-API zur Synchronisation verwendet wird, um Ausgabeereignisse zu empfangen. Aus Gründen der Einfachheit wird die Fehlerbehandlung weggelassen. Es ist immer empfehlenswert, den Rückgabewert von jedem API-Aufruf zu überprüfen.

```
using System;
using System.Collections.Generic;
using System.Text;
using iAnywhere.MobiLink.Client;

namespace ConsoleApplication6
{
    class Program
    {
        static void Main(string[] args)
        {
            DbmlsyncClient cli1;
            DBSC_StartType st1;
            DBSC_Event ev1;
            UInt32 syncHdl;

            cli1 = DbmlsyncClient.InstantiateClient();
            cli1.Init();

            // Setting the "server path" is usually required on Windows
            // Mobile/CE. In other environments the server path is usually
            // not required unless you SA install is not in your path or
            // you have multiple versions of the product installed
            cli1.SetProperty("server path", "d:\\sybase\\asall00r\\bin32");

            cli1.StartServer(3426,
                "-c server=cons;dbn=rem1;uid=dba;pwd=sql -ve+ -ot c:\\\
            \\dbsync1.txt",
                5000, out st1);
            cli1.Connect(null, 3426, "dba", "sql");
            syncHdl = cli1.Sync("spl", "");
            while (cli1.GetEvent(out ev1, 5000)
                == DBSC_GetEventRet.DBSC_GETEVENT_OK)
            {
                if (ev1.hdl == syncHdl)
                {
                    Console.WriteLine("Event Type : {0}", ev1.type);
                    if (ev1.type == DBSC_EventType.DBSC_EVENTTYPE_INFO_MSG)
                    {
                        Console.WriteLine("Info : {0}", ev1.str1);
                    }
                    if (ev1.type == DBSC_EventType.DBSC_EVENTTYPE_SYNC_DONE)
                    {
                        break;
                    }
                }
            }

            cli1.ShutdownServer(DBSC_ShutdownType.DBSC_SHUTDOWN_ON_EMPTY_QUEUE);
            cli1.WaitForServerShutdown(10000);
            cli1.Disconnect();
            cli1.Fini();
            Console.ReadLine();
        }
    }
}
```

DbmlsyncClient-Klasse

Diese Klasse kommuniziert über TCP/IP mit einem eigenen Prozess, dem dbmlsync-Server, der die Synchronisation durchführt, indem er eine Verbindung zum MobiLink-Server und zur entfernten Datenbank herstellt.

Visual Basic-Syntax

```
Public Class DbmlsyncClient
```

C#-Syntax

```
public class DbmlsyncClient
```

Mitglieder

Alle Mitglieder der DbmlsyncClient-Klasse, einschließlich aller geerbten Mitglieder.

Name	Beschreibung
CancelSync-Methode	Bricht eine Synchronisationsanforderung ab.
Connect-Methode	Öffnet eine Verbindung mit einem dbmlsync-Server, der bereits auf diesem Computer ausgeführt wird.
Disconnect-Methode	Unterbricht die Verbindung mit dem dbmlsync-Server, die mit der Connect-Methode hergestellt wurde.
Fini-Methode	Gibt alle von dieser Klasseninstanz verwendeten Ressourcen frei.
GetErrorInfo-Methode	Ruft zusätzliche Informationen über den Fehler ab, nachdem eine DbmlsyncClient-Klassenmethode einen Fehler zurückgegeben hat.
GetEvent-Methode	Ruft das nächste Rückmeldungsereignis für Synchronisationen ab, die vom Client angefordert wurden.
GetProperty-Methode	Ruft den aktuellen Wert einer Eigenschaft ab.
Init-Methode	Initialisiert eine DbmlsyncClient-Klasseninstanz.
InstantiateClient-Methode	Erstellt eine Instanz der dbmlsync-Clientklasse, die verwendet werden kann, um Synchronisationen zu steuern.
Ping-Methode	Sendet eine Ping-Anforderung an den dbmlsync-Server, um zu prüfen, ob der Server aktiv ist und auf Anforderungen antwortet.
SetProperty-Methode	Legt verschiedene Eigenschaften fest, um das Verhalten der Klasseninstanz zu ändern.
ShutdownServer-Methode	Führt den dbmlsync-Server herunter, mit dem der Client verbunden ist.

Name	Beschreibung
StartServer-Methode	Startet einen neuen dbmlsync-Server, wenn noch keiner am angegebenen Port auf Nachrichten wartet.
Sync-Methode	Fordert an, dass der dbmlsync-Server eine Synchronisation ausführt.
WaitForServerShutdown-Methode	Kehrt zurück, wenn der Server heruntergefahren oder der Timeout abgelaufen ist, je nach dem, welches Ereignis früher eintritt.

Bemerkungen

Mehrere Clients können denselben dbmlsync-Server gemeinsam nutzen. Jeder dbmlsync-Server kann nur eine einzige entfernte Datenbank synchronisieren. Jede entfernte Datenbank kann jeweils nur von einem dbmlsync-Server synchronisiert werden.

Der dbmlsync-Server führt jeweils nur eine Synchronisation aus. Wenn der Server eine Synchronisationsanforderung empfängt, während er eine andere Synchronisation ausführt, stellt er diese Anforderung in eine Warteschlange und führt sie zu einem späteren Zeitpunkt aus.

Von den Synchronisationen generierte Statusinformationen werden über die GetEvent-Methode zurück an die Clientanwendung übertragen.

Siehe auch

- [DbmlsyncClient.GetEvent-Methode \[Dbmlsync .NET\] auf Seite 299](#)

CancelSync-Methode

Bricht eine Synchronisationsanforderung ab.

Überladungsliste

Name	Beschreibung
CancelSync(UInt32)-Methode (nicht mehr empfohlen)	Erlaubt es einem Client, eine Synchronisationsanforderung abbrechen, die zuvor mit der Sync-Methode gestellt wurde.
CancelSync(UInt32, Boolean)-Methode	Erlaubt es einem Client, eine Synchronisationsanforderung abbrechen, die zuvor mit der Sync-Methode gestellt wurde.

CancelSync(UInt32)-Methode (nicht mehr empfohlen)

Erlaubt es einem Client, eine Synchronisationsanforderung abbrechen, die zuvor mit der Sync-Methode gestellt wurde.

Visual Basic-Syntax

```
Public Function CancelSync(ByVal hdl As UInt32) As Boolean
```

C#-Syntax

```
public Boolean CancelSync(UInt32 hdl)
```

Parameter

- **hdl** Das von der Sync-Methode zurückgegebene Synchronisations-Handle.

Rückgabe

TRUE, wenn die Synchronisationsanforderung erfolgreich abgebrochen wurde sonst FALSE. Wenn FALSE zurückgegeben wird, können Sie die Methode GetErrorInfo aufrufen, um weitere Informationen über den Fehler zu erhalten.

Bemerkungen

Es können nur Synchronisationsanforderungen abgebrochen werden, die darauf warten, verarbeitet zu werden. Um eine bereits begonnene Synchronisation zu stoppen, verwenden Sie die CancelSync(UInt32, Boolean)-Methode.

Vor der Verwendung dieser Methode muss eine Verbindung zum Server eingerichtet werden. Diese Methode kann nicht verwendet werden, wenn der Client nach dem Aufruf der Sync-Methode vom Server getrennt wurde.

Siehe auch

- [DbmlsyncClient.GetErrorInfo-Methode \[Dbmlsync .NET\] auf Seite 298](#)
- [DbmlsyncClient.CancelSync-Methode \[Dbmlsync .NET\] auf Seite 294](#)
- [DbmlsyncClient.ShutdownServer-Methode \[Dbmlsync .NET\] auf Seite 304](#)

CancelSync(UInt32, Boolean)-Methode

Erlaubt es einem Client, eine Synchronisationsanforderung abzubrechen, die zuvor mit der Sync-Methode gestellt wurde.

Visual Basic-Syntax

```
Public Function CancelSync(  
    ByVal hdl As UInt32,  
    ByVal cancel_active As Boolean  
) As DBSC_CancelRet
```

C#-Syntax

```
public DBSC_CancelRet CancelSync(UInt32 hdl, Boolean cancel_active)
```

Parameter

- **hdl** Das von der Sync-Methode zurückgegebene Synchronisations-Handle.
- **cancel_active** Wenn dieser Parameter auf TRUE gesetzt ist, wird die Anforderung abgebrochen, auch wenn die Synchronisation bereits begonnen hat. Beim Wert FALSE wird die Anforderung nur abgebrochen, wenn die Synchronisation noch nicht begonnen hat.

Rückgabe

Ein Wert aus der `DBSC_CancelRet`-Enumeration. Wenn `DBSC_CANCEL_FAILED` zurückgegeben wird, können Sie die Methode `GetErrorInfo` aufrufen, um weitere Informationen über den Fehler zu erhalten.

Bemerkungen

Vor der Verwendung dieser Methode muss eine Verbindung zum Server eingerichtet werden. Diese Methode kann nicht verwendet werden, wenn der Client nach dem Aufruf der `Sync`-Methode vom Server getrennt wurde.

Siehe auch

- [DBSC_CancelRet-Enumeration \[Dbmlsync .NET\] auf Seite 308](#)
- [DbmlsyncClient.GetErrorInfo-Methode \[Dbmlsync .NET\] auf Seite 298](#)
- [DbmlsyncClient.ShutdownServer-Methode \[Dbmlsync .NET\] auf Seite 304](#)

Connect-Methode

Öffnet eine Verbindung mit einem `dbmlsync`-Server, der bereits auf diesem Computer ausgeführt wird.

Visual Basic-Syntax

```
Public Function Connect(  
    ByVal host As String,  
    ByVal port As Int32,  
    ByVal uid As String,  
    ByVal pwd As String  
) As Boolean
```

C#-Syntax

```
public Boolean Connect(String host, Int32 port, String uid, String pwd)
```

Parameter

- **host** Dieser Wert ist reserviert. Geben Sie null bei Verwendung von C# an. Geben Sie bei Verwendung von Visual Basic nichts an.
- **port** Der TCP-Port, an dem der `dbmlsync`-Server auf Verbindungen wartet. Verwenden Sie denselben Port, den Sie für die `StartServer`-Methode angegeben haben.
- **uid** Eine gültige Datenbankbenutzer-ID mit DBA- oder REMOTE DBA-Berechtigung für die entfernte Datenbank, die synchronisiert werden soll.
- **pwd** Das Datenbankkennwort für den mit `uid` angegebenen Benutzer.

Rückgabe

TRUE, wenn eine Verbindung mit dem Server hergestellt wurde sonst FALSE. Wenn FALSE zurückgegeben wird, können Sie die Methode `GetErrorInfo` aufrufen, um weitere Informationen über den Fehler zu erhalten.

Bemerkungen

Dbmlsync-Server werden (entweder über die Befehlszeile oder über die StartServer-Methode) mithilfe einer Verbindungszeichenfolge gestartet, die eine Datenbankbenutzer-ID und das dazugehörige Kennwort (z.B. *server_userid*) enthält. Außerdem erfordert die Connect-Methode der Dbmlsync-API eine gültige Datenbankbenutzer-ID (z.B. *client_userid*).

Client_userid wird nur verwendet, um zu validieren, ob dieser Client ausreichende Berechtigungen für die Synchronisation der Datenbank hat. Wenn Synchronisationen ausgeführt werden, wird *server_userid* verwendet.

In SQL Anywhere 12 und früher waren sowohl für *client_userid* als auch für *server_userid* DBA- oder REMOTE DBA-Berechtigungen erforderlich.

In SQL Anywhere 16 und später muss *server_userid* ausreichende Privilegien für die Synchronisation haben. *Server_userid* muss mindestens die SYS_RUN_REPLICATION_ROLE-Systemrolle haben, aber für die Synchronisation können auch weitere Privilegien erforderlich sein. *Client_userid* muss eine der folgenden Bedingungen erfüllen:

- Mit *server_userid* übereinstimmen
- Die SYS_AUTH_DBA_ROLE-Systemrolle haben
- Eine auf *server_userid* basierende benutzererweiterte Rolle haben, z.B. CREATE ROLE FOR USER Serverbenutzer-ID, GRANT Serverbenutzer-ID TO Clientbenutzer-ID

Die letzte Option stellt sicher, dass *client_userid* mindestens ebenso viele Systemprivilegien hat wie *server_userid*. *server_userid* kann jedoch auch Privilegien auf Objektebene haben, die seiner Rolle nicht erteilt wurden. In diesem Fall hat *client_userid* diese Privilegien nicht. Wenn diese Privilegien während der Synchronisation verwendet werden, hat *client_userid* praktisch seine Privilegien erhöht, um die Synchronisation durchzuführen. Falls dies nicht akzeptabel ist, stellen Sie sicher, dass alle Privilegien von *server_userid*s auf Objektebene den entsprechenden benutzererweiterten Rollen erteilt werden.

Siehe auch

- [DbmlsyncClient.StartServer-Methode \[Dbmlsync .NET\] auf Seite 305](#)
- [DbmlsyncClient.GetErrorInfo-Methode \[Dbmlsync .NET\] auf Seite 298](#)

Disconnect-Methode

Unterbricht die Verbindung mit dem dbmlsync-Server, die mit der Connect-Methode hergestellt wurde.

Visual Basic-Syntax

```
Public Function Disconnect() As Boolean
```

C#-Syntax

```
public Boolean Disconnect()
```

Rückgabe

TRUE, wenn die Verbindung mit dem Server unterbrochen wurde sonst FALSE. Wenn FALSE zurückgegeben wird, können Sie die Methode GetErrorInfo aufrufen, um weitere Informationen über den Fehler zu erhalten.

Bemerkungen

Sie sollten immer Disconnect aufrufen, wenn Sie eine Verbindung nicht mehr benötigen.

Siehe auch

- [DbmlsyncClient.Connect-Methode \[Dbmlsync .NET\] auf Seite 296](#)
- [DbmlsyncClient.GetErrorInfo-Methode \[Dbmlsync .NET\] auf Seite 298](#)

Fini-Methode

Gibt alle von dieser Klasseninstanz verwendeten Ressourcen frei.

Visual Basic-Syntax

```
Public Function Fini() As Boolean
```

C#-Syntax

```
public Boolean Fini()
```

Rückgabe

TRUE, wenn die Klasseninstanz erfolgreich beendet wurde sonst FALSE. Wenn FALSE zurückgegeben wird, können Sie die Methode GetErrorInfo aufrufen, um weitere Informationen über den Fehler zu erhalten.

Bemerkungen

Diese Methode muss aufgerufen werden, bevor Sie die DbmlSyncClient-Klasseninstanz löschen können.

Hinweis

Verwenden Sie die Disconnect-Methode, um die Verbindung zu einem beliebigen verbundenen Server vor dem Beenden der Klasseninstanz zu trennen.

Siehe auch

- [DbmlsyncClient.Disconnect-Methode \[Dbmlsync .NET\] auf Seite 297](#)
- [DbmlsyncClient.GetErrorInfo-Methode \[Dbmlsync .NET\] auf Seite 298](#)

GetErrorInfo-Methode

Ruft zusätzliche Informationen über den Fehler ab, nachdem eine DbmlsyncClient-Klassenmethode einen Fehler zurückgegeben hat.

Visual Basic-Syntax

```
Public Function GetErrorInfo() As DBSC_ErrorInfo
```

C#-Syntax

```
public DBSC_ErrorInfo GetErrorInfo()
```

Rückgabe

Zeiger auf eine DBSC_ErrorInfo-Struktur, die Informationen über den Fehler enthält. Der Inhalt dieser Struktur kann beim nächsten Aufruf einer beliebigen Klassenmethode überschrieben werden.

Siehe auch

- [DBSC_ErrorType-Enumeration \[Dbmlsync .NET\] auf Seite 308](#)
- [DBSC_ErrorInfo-Struktur \[Dbmlsync .NET\] auf Seite 315](#)
- [DbmlsyncClient.GetErrorInfo-Methode \[Dbmlsync .NET\] auf Seite 298](#)

GetEvent-Methode

Ruft das nächste Rückmeldungereignis für Synchronisationen ab, die vom Client angefordert wurden.

Visual Basic-Syntax

```
Public Function GetEvent(  
    ByVal ev As DBSC_Event,  
    ByVal timeout As UInt32  
) As DBSC_GetEventRet
```

C#-Syntax

```
public DBSC_GetEventRet GetEvent(out DBSC_Event ev, UInt32 timeout)
```

Parameter

- **ev** Wenn der Rückgabewert DBSC_GETEVENT_OK ist, wird der Ereignisparameter mit Informationen über das Ereignis gefüllt, das abgerufen wurde.
- **timeout** Gibt die maximale Anzahl von Millisekunden zurück, die gewartet werden soll, wenn kein Ereignis unmittelbar zur Rückgabe bereitsteht. Verwenden Sie DbmlsyncClient.DBSC_INFINITY, um unbegrenzt auf eine Antwort zu warten.

Rückgabe

Ein Wert aus der DBSC_GetEventRet-Enumeration. Wenn DBSC_GETEVENT_FAILED zurückgegeben wird, können Sie die Methode GetErrorInfo aufrufen, um weitere Informationen über den Fehler zu erhalten.

Bemerkungen

Rückmeldungereignisse enthalten Informationen, etwa Nachrichten, die durch die Synchronisation generiert wurden, Daten für die Aktualisierung des Fortschrittsbalkens und Synchronisationszyklus-Benachrichtigungen.

Wenn der dbmlsync-Server eine Synchronisation ausführt, generiert er eine Reihe von Ereignissen, die Informationen über den Verarbeitungsfortschritt der Synchronisation enthalten. Diese Ereignisse werden vom Server an die Klasse DbmlsyncClient gesendet, die sie in eine Warteschlange stellt. Wenn die GetEvent-Methode aufgerufen wird, wird die nächste Methode in der Warteschlange zurückgegeben, sofern dort eine wartet.

Wenn keine Ereignisse in der Warteschlange vorhanden sind, wartet diese Methode, bis ein Ereignis verfügbar ist oder bis der angegebene Timeout abgelaufen ist, bevor sie beendet wird.

Die für eine Synchronisation generierten Ereignistypen können mit Eigenschaften gesteuert werden.

Siehe auch

- [DBSC_GetEventRet-Enumeration \[Dbmlsync .NET\] auf Seite 313](#)
- [DBSC_Event-Struktur \[Dbmlsync .NET\] auf Seite 316](#)
- [DbmlsyncClient.SetProperty-Methode \[Dbmlsync .NET\] auf Seite 302](#)
- [DbmlsyncClient.GetErrorInfo-Methode \[Dbmlsync .NET\] auf Seite 298](#)

GetProperty-Methode

Ruft den aktuellen Wert einer Eigenschaft ab.

Visual Basic-Syntax

```
Public Function GetProperty(  
    ByVal name As String,  
    ByVal value As String  
) As Boolean
```

C#-Syntax

```
public Boolean GetProperty(String name, out String value)
```

Parameter

- **name** Der Name der abzurufenden Eigenschaft. Eine Liste der gültigen Eigenschaftsnamen finden Sie unter "SetProperty".
- **value** Bei der Beendigung der Methode wird der Wert der Eigenschaft in dieser Variablen gespeichert.

Rückgabe

TRUE, wenn die Eigenschaft erfolgreich empfangen wurde sonst FALSE. Wenn FALSE zurückgegeben wird, können Sie die Methode GetErrorInfo aufrufen, um weitere Informationen über den Fehler zu erhalten.

Siehe auch

- [DbmlsyncClient.SetProperty-Methode \[Dbmlsync .NET\] auf Seite 302](#)
- [DbmlsyncClient.GetErrorInfo-Methode \[Dbmlsync .NET\] auf Seite 298](#)

Init-Methode

Initialisiert eine DbmlsyncClient-Klasseninstanz.

Visual Basic-Syntax

```
Public Function Init() As Boolean
```

C#-Syntax

```
public Boolean Init()
```

Rückgabe

TRUE, wenn die Klasseninstanz erfolgreich initialisiert wurde sonst FALSE. Wenn FALSE zurückgegeben wird, können Sie die Methode GetErrorInfo aufrufen, um weitere Informationen über den Fehler zu erhalten.

Bemerkungen

Diese Methode muss aufgerufen werden, nachdem die DbmlSyncClient-Klasseninstanz erstellt wurde. Andere DbmlSyncClient-Methoden können erst aufgerufen werden, wenn Sie die Instanz erfolgreich initialisiert haben.

Siehe auch

- [DbmlsyncClient.InstantiateClient-Methode \[Dbmlsync .NET\] auf Seite 301](#)
- [DbmlsyncClient.GetErrorInfo-Methode \[Dbmlsync .NET\] auf Seite 298](#)

InstantiateClient-Methode

Erstellt eine Instanz der dbmlsync-Clientklasse, die verwendet werden kann, um Synchronisationen zu steuern.

Visual Basic-Syntax

```
Public Shared Function InstantiateClient() As DbmlsyncClient
```

C#-Syntax

```
public static DbmlsyncClient InstantiateClient()
```

Rückgabe

Die erstellte DbmlsyncClient-Instanz. Gibt NULL zurück, wenn ein Fehler auftritt.

Bemerkungen

Das von dieser Methode zurückgegebene Objekt kann verwendet werden, um die übrigen Methoden in der Klasse aufzurufen.

Ping-Methode

Sendet eine Ping-Anforderung an den dbmlsync-Server, um zu prüfen, ob der Server aktiv ist und auf Anforderungen antwortet.

Visual Basic-Syntax

```
Public Function Ping(ByVal timeout As UInt32) As Boolean
```

C#-Syntax

```
public Boolean Ping(UInt32 timeout)
```

Parameter

- **timeout** Die maximale Anzahl von Millisekunden, die gewartet werden soll, bis der Server auf die Ping-Anforderung antwortet. Verwenden Sie DbmlsyncClient.DBSC_INFINITY, um unbegrenzt auf eine Antwort zu warten.

Rückgabe

TRUE, wenn eine Antwort auf die Ping-Anforderung vom Server empfangen wurde sonst FALSE. Wenn FALSE zurückgegeben wird, können Sie die Methode GetErrorInfo aufrufen, um weitere Informationen über den Fehler zu erhalten.

Bemerkungen

Sie müssen mit dem Server verbunden sein, um diese Methode aufrufen zu können.

Siehe auch

- [DbmlsyncClient.GetErrorInfo-Methode \[Dbmlsync .NET\] auf Seite 298](#)

SetProperty-Methode

Legt verschiedene Eigenschaften fest, um das Verhalten der Klasseninstanz zu ändern.

Visual Basic-Syntax

```
Public Function SetProperty(  
    ByVal name As String,  
    ByVal value As String  
) As Boolean
```

C#-Syntax

```
public Boolean SetProperty(String name, String value)
```

Parameter

- **name** Der Name der festzulegenden Eigenschaft. Eine Liste der gültigen Eigenschaftsnamen finden Sie in der Tabelle.
- **value** Der Wert, auf den die Eigenschaft gesetzt werden soll.

Rückgabe

TRUE, wenn die Eigenschaft erfolgreich gesetzt werden konnte sonst FALSE. Wenn FALSE zurückgegeben wird, können Sie die Methode GetErrorInfo aufrufen, um weitere Informationen über den Fehler zu erhalten.

Bemerkungen

Änderungen von Eigenschaftswerten haben nur Auswirkungen auf Synchronisationsanforderungen, die nach der Änderung des Eigenschaftswerts vorgenommen werden.

Mit der **server path**-Eigenschaft kann das Verzeichnis festgelegt werden, von dem aus der Client dbmlsync.exe starten soll, wenn die StartServer-Methode aufgerufen wird. Wenn diese Eigenschaft nicht festgelegt ist, wird dbmlsync.exe mithilfe der Umgebungsvariablen PATH gefunden. Wenn mehrere Versionen von SQL Anywhere auf Ihrem Computer installiert sind, ist es empfehlenswert, den Speicherort von dbmlsync.exe mit der **server path**-Eigenschaft anzugeben, da die Umgebungsvariable PATH möglicherweise die dbmlsync.exe einer anderen installierten Version von SQL Anywhere findet. Beispiel:

```
ret = cli->SetProperty("server path", "c:\\sa12\\bin32");
```

Die Eigenschaften steuern die Ereignistypen, die von der GetEvent-Methode zurückgegeben werden. Durch das Deaktivieren nicht benötigter Ereignisse ist es möglich, die Performance zu steigern. Ein Ereignistyp wird aktiviert, indem Sie die entsprechende Eigenschaft auf "1" setzen, und deaktiviert, indem Sie die Eigenschaft auf "0" setzen.

Die folgende Tabelle enthält die verfügbaren Eigenschaftsnamen und die Ereignistypen, die die einzelnen Namen steuern:

Eigenschaftsname	Gesteuerte Ereignistypen	Standardwert
enable errors	DBSC_EVENTTYPE_ERROR_MSG	1
enable warnings	DBSC_EVENTTYPE_WARNING_MSG	1
enable info msgs	DBSC_EVENTTYPE_INFO_MSG	1
enable progress	DBSC_EVENTTYPE_PROGRESS_INDEX	0
enable progress text	DBSC_EVENTTYPE_PROGRESS_TEXT	0
enable title	DBSC_EVENTTYPE_TITLE	0
enable sync start	DBSC_EVENTTYPE_SYNC_START	1
enable sync done	DBSC_EVENTTYPE_SYNC_DONE	1
enable sync start and done	DBSC_EVENTTYPE_SYNC_START DBSC_EVENTTYPE_SYNC_DONE	1

Eigenschaftsname	Gesteuerte Ereignistypen	Standardwert
enable status	DBSC_EVENTTYPE_ML_CONNECT DBSC_EVENTTYPE_UPLOAD_START DBSC_EVENTTYPE_UPLOAD_SENT DBSC_EVENTTYPE_UPLOAD_COMMITTED DBSC_EVENTTYPE_DOWNLOAD_START DBSC_EVENTTYPE_DOWNLOAD_COMMITTED	1

Siehe auch

- [DbmlsyncClient.StartServer-Methode \[Dbmlsync .NET\] auf Seite 305](#)
- [DbmlsyncClient.GetEvent-Methode \[Dbmlsync .NET\] auf Seite 299](#)
- [DbmlsyncClient.GetProperty-Methode \[Dbmlsync .NET\] auf Seite 300](#)
- [DbmlsyncClient.GetErrorInfo-Methode \[Dbmlsync .NET\] auf Seite 298](#)

ShutdownServer-Methode

Führt den dbmlsync-Server herunter, mit dem der Client verbunden ist.

Visual Basic-Syntax

```
Public Function ShutdownServer(  
    ByVal how As DBSC_ShutdownType  
) As Boolean
```

C#-Syntax

```
public Boolean ShutdownServer(DBSC_ShutdownType how)
```

Parameter

- **how** Gibt die Dringlichkeit für das Herunterfahren des Servers an. Unterstützte Werte werden in der DBSC_ShutdownType-Enumeration aufgeführt.

Rückgabe

TRUE, wenn eine Anforderung zum Herunterfahren erfolgreich an den Server gesendet wurde sonst FALSE. Wenn FALSE zurückgegeben wird, können Sie die Methode GetErrorInfo aufrufen, um weitere Informationen über den Fehler zu erhalten.

Bemerkungen

Die Shutdown-Methode wird sofort beendet, doch es kann zu einer Verzögerung kommen, bevor der Server tatsächlich heruntergefahren wird.

Die Methode `WaitForServerShutdown` kann verwendet werden, um zu warten, bis der Server tatsächlich heruntergefahren wird.

Hinweis

Sie sollten jedoch die `Disconnect`-Methode nach dem Aufruf von `ShutdownServer` verwenden.

Siehe auch

- [DBSC_ShutdownType-Enumeration \[Dbmlsync .NET\] auf Seite 314](#)
- [DbmlsyncClient.Disconnect-Methode \[Dbmlsync .NET\] auf Seite 297](#)
- [DbmlsyncClient.WaitForServerShutdown-Methode \[Dbmlsync .NET\] auf Seite 307](#)
- [DbmlsyncClient.GetErrorInfo-Methode \[Dbmlsync .NET\] auf Seite 298](#)

StartServer-Methode

Startet einen neuen dbmlsync-Server, wenn noch keiner am angegebenen Port auf Nachrichten wartet.

Visual Basic-Syntax

```
Public Function StartServer(
    ByVal port As Int32,
    ByVal cmdline As String,
    ByVal timeout As UInt32,
    ByVal starttype As DBSC_StartType
) As Boolean
```

C#-Syntax

```
public Boolean StartServer(
    Int32 port,
    String cmdline,
    UInt32 timeout,
    out DBSC_StartType starttype
)
```

Parameter

- **port** Der TCP-Port, an dem auf einen vorhandenen dbmlsync-Server geprüft werden soll. Wenn ein neuer Server gestartet wird, wird er so eingestellt, dass er an diesem Port auf Verbindungen wartet.
- **cmdline** Eine gültige Befehlszeile zum Starten eines dbmlsync-Servers. Die Befehlszeile kann nur die folgenden Optionen enthalten, die die gleiche Bedeutung wie für das Dienstprogramm dbmlsync haben: -a, -c, -dl, -do, -ek, -ep, -k, -l, -o, -os, -ot, -p, -pc+, -pc-, -pd, -pp, -q, -qi, -qc, -sc, -sp, -uc, -ud, -ui, -um, -un, -ux, -v[cnoprsut], -wc, -wh. Die Option -c muss angegeben werden.
- **timeout** Die maximale Dauer in Millisekunden, die nach dem Start eines dbmlsync-Servers gewartet werden soll, bis der Server bereit zur Entgegennahme von Anforderungen ist. Verwenden Sie `DbmlsyncClient.DBSC_INFINITY`, um unbegrenzt auf eine Antwort zu warten.
- **starttype** Ein OUT-Parameter, der anzeigt, ob der Server gefunden oder gestartet wurde. Wenn `starttype` beim Eintritt Nicht-NULL ist und `StartServer` TRUE zurückgibt, wird beim Beenden die Variable, auf die `starttype` zeigt, auf einen der Werte der `DBSC_StartType`-Enumeration gesetzt.

Rückgabe

TRUE, wenn der Server bereits lief oder erfolgreich gestartet werden konnte sonst FALSE. Wenn FALSE zurückgegeben wird, können Sie die Methode `GetErrorInfo` aufrufen, um weitere Informationen über den Fehler zu erhalten.

Bemerkungen

Wenn ein Server vorhanden ist, setzt diese Methode den `starttype`-Parameter auf `DBSC_SS_ALREADY_RUNNING` und wird ohne weitere Aktion beendet. Wenn kein Server gefunden wird, startet die Methode mit den im Argument `cmdline` angegebenen Optionen einen neuen Server und wartet, dass der Server Anforderungen akzeptiert, bevor die Methode beendet wird.

Auf Windows Mobile-Geräten ist es üblicherweise notwendig, die **server path**-Eigenschaft zu setzen, bevor `StartServer` erfolgreich aufgerufen werden kann. Die **server path**-Eigenschaft muss in den folgenden Fällen nicht gesetzt werden:

- Ihre Anwendung befindet sich in demselben Verzeichnis wie `dbmlsync.exe`.
- `dbmlsync.exe` befindet sich im Windows-Verzeichnis.

Siehe auch

- [DbmlsyncClient.GetErrorInfo-Methode \[Dbmlsync .NET\] auf Seite 298](#)

Sync-Methode

Fordert an, dass der `dbmlsync`-Server eine Synchronisation ausführt.

Visual Basic-Syntax

```
Public Function Sync(  
    ByVal syncName As String,  
    ByVal opts As String  
) As UInt32
```

C#-Syntax

```
public UInt32 Sync(String syncName, String opts)
```

Parameter

- **syncName** Der Name eines Synchronisationsprofil, das in der entfernten Datenbank definiert ist, welche die Optionen für die Synchronisation enthält. Wenn `syncName` NULL ist, wird kein Profil verwendet und der Parameter `opts` muss alle Optionen für die Synchronisation enthalten.
- **opts** Eine Zeichenfolge, die nach denselben Regeln gebildet wird, die auch beim Definieren einer Optionszeichenfolge für ein Synchronisationsprofil verwendet werden. Dabei handelt es sich um eine Zeichenfolge, die durch Semikola getrennte Elemente in der Form `<Optionsname>=<Optionswert>` enthält. Wenn `syncName` Nicht-NULL ist, werden die von `opts` festgelegten Optionen zu den Optionen hinzugefügt, die bereits im von `syncName` angegebenen Synchronisationsprofil enthalten sind. Wenn eine in der Zeichenfolge enthaltene Option im Profil bereits vorhanden ist, ersetzt der Wert aus der Zeichenfolge den bereits im Profil gespeicherten Wert.

Wenn syncName NULL ist, muss opts alle Optionen für die Synchronisation festlegen. Siehe „CREATE SYNCHRONIZATION PROFILE-Anweisung [MobiLink]“ [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Rückgabe

Ein ganzzahliger Wert, der diese Synchronisationsanforderung eindeutig identifiziert und nur gültig ist, bis der Client vom Server getrennt wird. Gibt NULL_SYNCHDL zurück, wenn ein Fehler verhindert, dass die Synchronisationsanforderung erstellt wird. Wenn NULL_SYNCHDL zurückgegeben wird, können Sie die Methode GetErrorInfo aufrufen, um weitere Informationen über den Fehler zu erhalten.

Bemerkungen

Sie müssen mit dem Server verbunden sein, um diese Methode aufrufen zu können. Mindestens eine der Optionen syncName oder opts muss Nicht-NULL sein.

Der Rückgabewert gibt die Synchronisationsanforderung an und kann verwendet werden, um die Anforderung abzubrechen oder die von der Synchronisation zurückgegeben Ereignisse zu verarbeiten.

Das folgende C#-Beispiel zeigt, wie Fehlercodes nach dem Aufrufen der Sync-Methode angezeigt werden.

```
// Insert code to initialize the synchronization client.

UInt32 request = syncClient.Sync("syncName", null);
if (request == DbmlsyncClient.NULL_SYNCHDL) {
    string error_code = syncClient.GetErrorInfo().type.ToString();
    MessageBox.Show(error_code, "Sync Error");
}
```

Siehe auch

- [DbmlsyncClient.GetErrorInfo-Methode \[Dbmlsync .NET\] auf Seite 298](#)

WaitForServerShutdown-Methode

Kehrt zurück, wenn der Server heruntergefahren oder der Timeout abgelaufen ist, je nach dem, welches Ereignis früher eintritt.

Visual Basic-Syntax

```
Public Function WaitForServerShutdown(  
    ByVal timeout As UInt32  
) As Boolean
```

C#-Syntax

```
public Boolean WaitForServerShutdown(UInt32 timeout)
```

Parameter

- **timeout** Zeigt die maximale Dauer in Millisekunden an, die gewartet werden soll, bis der Server heruntergefahren wird. Verwenden Sie DbmlsyncClient.DBSC_INFINITY, um unbegrenzt auf eine Antwort zu warten.

Rückgabe

TRUE, wenn die Methode beendet wurde, weil der Server heruntergefahren wurde, sonst FALSE. Wenn FALSE zurückgegeben wird, können Sie die Methode `GetErrorInfo` aufrufen, um weitere Informationen über den Fehler zu erhalten.

Bemerkungen

`WaitForServerShutdown` kann nur aufgerufen werden, nachdem die Methode `ShutdownServer` aufgerufen wurde.

Siehe auch

- [DbmlsyncClient.GetErrorInfo-Methode \[Dbmlsync .NET\] auf Seite 298](#)

DBSC_CancelRet-Enumeration

Gibt das Ergebnis des Versuchs an, eine Synchronisation abubrechen.

Visual Basic-Syntax

```
Public Enum DBSC_CancelRet
```

C#-Syntax

```
public enum DBSC_CancelRet
```

Mitglieder

Mitgliedsname	Beschreibung	Wert
DBSC_CAN- CEL_OK_QUEUED	Synchronisation abgebrochen, die sich in der Warteschlange befand.	1
DBSC_CANCEL_OK_ACTI- VE	Aktive Synchronisation abgebrochen.	2
DBSC_CANCEL_FAILED	Synchronisation konnte nicht abgebrochen werden.	3

Siehe auch

- [DbmlsyncClient.CancelSync-Methode \[Dbmlsync .NET\] auf Seite 294](#)

DBSC_ErrorType-Enumeration

Gibt den Grund für einen fehlgeschlagenen Methodenaufruf an.

Visual Basic-Syntax

```
Public Enum DBSC_ErrorType
```

C#-Syntax

```
public enum DBSC_ErrorType
```

Mitglieder

Mitgliedsname	Beschreibung	Wert
DBSC_ERR_OK	Es ist kein Fehler aufgetreten.	1
DBSC_ERR_NOT_INITIALIZED	Die Klasse wurde nicht durch den Aufruf der init-Methode initialisiert.	2
DBSC_ERR_ALREADY_INITIALIZED	Die init-Methode wurde für eine Klasse aufgerufen, die bereits initialisiert war.	3
DBSC_ERR_NOT_CONNECTED	Es besteht keine Verbindung mit einem dbmlsync-Server.	4
DBSC_ERR_CANT_RESOLVE_HOST	Hostinformationen konnten nicht aufgelöst werden.	5
DBSC_ERR_CONNECT_FAILED	Verbindung mit dem dbmlsync-Server ist fehlgeschlagen.	6
DBSC_ERR_INITIALIZING_TCP_LAYER	Bei der Initialisierung der TCP-Schicht ist ein Fehler aufgetreten.	7
DBSC_ERR_ALREADY_CONNECTED	Die Connect-Methode ist fehlgeschlagen, da bereits eine Verbindung bestand.	8
DBSC_ERR_PROTOCOL_ERROR	Dies ist ein interner Fehler.	9
DBSC_ERR_CONNECTION_REJECTED	Die Verbindung wurde vom dbmlsync-Server zurückgewiesen. str1 zeigt auf eine vom Server zurückgegebene Zeichenfolge, die weitere Informationen darüber bereitstellen kann, warum der Verbindungsversuch abgelehnt wurde.	10
DBSC_ERR_TIMED_OUT	Beim Warten auf eine Antwort des Servers ist der Timeout abgelaufen.	11
DBSC_ERR_STILL_CONNECTED	Die Fini-Methode konnte nicht für die Klasse aufgerufen werden, da sie noch mit dem Server verbunden ist.	12

Mitgliedsname	Beschreibung	Wert
DBSC_ERR_SYNC_NOT_CANCELED	Der Server konnte die Synchronisationsanforderung nicht abbrechen. Dies liegt wahrscheinlich daran, dass die Synchronisation bereits ausgeführt wurde.	14
DBSC_ERR_INVALID_VALUE	Ein ungültiger Eigenschaftswert wurde an die SetProperty-Methode übergeben.	15
DBSC_ERR_INVALID_PROP_NAME	Der angegebene Eigenschaftsname ist ungültig.	16
DBSC_ERR_VALUE_TOO_LONG	Der Eigenschaftswert ist zu lang; Eigenschaften müssen weniger als DBCS_MAX_PROPERTY_LEN Byte lang sein.	17
DBSC_ERR_SERVER_SIDE_ERROR	Ein serverseitiger Fehler ist beim Abbrechen oder Hinzufügen von einer Synchronisation aufgetreten. str1 zeigt auf eine vom Server zurückgegebene Zeichenfolge, die weitere Informationen über den Fehler bereitstellen kann.	18
DBSC_ERR_CREATE_PROCESS_FAILED	Es konnte kein neuer dbmlsync-Server gestartet werden.	20
DBSC_ERR_READ_FAILED	Ein TCP-Fehler ist beim Lesen von Daten vom dbmlsync-Server aufgetreten.	21
DBSC_ERR_WRITE_FAILED	Ein TCP-Fehler ist beim Senden von Daten an den dbmlsync-Server aufgetreten.	22
DBSC_ERR_NO_SERVER_RESPONSE	Eine Antwort vom Server, die für die Ausführung der angeforderten Aktion erforderlich ist, konnte nicht empfangen werden.	23
DBSC_ERR_UID_OR_PWD_TOO_LONG	Der angegebene Wert für UID oder PWD ist zu lang.	24
DBSC_ERR_UID_OR_PWD_NOT_VALID	Der angegebene Wert für UID oder PWD ist ungültig oder hat keine ausreichenden Privilegien, um eine Verbindung herzustellen.	25
DBSC_ERR_INVALID_PARAMETER	Einer der an die Funktion übergebenen Parameter war ungültig.	26

Mitgliedsname	Beschreibung	Wert
DBSC_ERR_WAIT_FAILED	Während darauf gewartet wurde, dass der Server herunterfährt, ist ein Fehler aufgetreten.	27
DBSC_ERR_SHUTDOWN_NOT_CALLED	Die WaitForServerShutdown-Methode wurde aufgerufen, ohne dass zuerst die ShutdownServer-Methode aufgerufen wurde.	28
DBSC_ERR_NO_SYNC_ACK	<p>Es wurde eine Synchronisationsanforderung an den Server gesendet, jedoch wurde keine Bestätigung empfangen. Es gibt keine Möglichkeit zu erkennen, ob der Server die Anforderung empfangen hat.</p> <p>hdl1 ist das Handle für die gesendete Synchronisationsanforderung. Wenn der Server die Anforderung empfangen hat, kann dieses Handle verwendet werden, um die Ereignisse für die mit der GetEvent-Methode abgerufene Synchronisation zu identifizieren.</p>	29
DBSC_ERR_ACTIVE_SYNC_NOT_CANCELED	Der Server konnte die Synchronisationsanforderung nicht abbrechen, da die Synchronisation bereits aktiv war.	30
DBSC_ERR_DEAD_SERVER	<p>Beim Starten des dbmlsync-Servers ist ein Fehler aufgetreten.</p> <p>Der Server wird nun heruntergefahren. Verwenden Sie die dbmlsync-Option -o zum Aufzeichnen der Fehlermeldung in einer Datei.</p>	31

DBSC_EventType-Enumeration

Zeigt den Typ des Ereignisses an, das von einer Synchronisation generiert wurde.

Visual Basic-Syntax

```
Public Enum DBSC_EventType
```

C#-Syntax

```
public enum DBSC_EventType
```

Mitglieder

Mitgliedsname	Beschreibung	Wert
DBSC_EVENT- TYPE_ERROR_MSG	Die Synchronisation hat einen Fehler generiert; str1 enthält den Fehlertext.	1
DBSC_EVENT- TYPE_WARNING_MSG	Die Synchronisation hat eine Warnung generiert; str1 enthält den Text der Warnung.	2
DBSC_EVENT- TYPE_INFO_MSG	Die Synchronisation hat eine Meldung generiert; str1 enthält den Meldungstext.	3
DBSC_EVENT- TYPE_PROGRESS_INDEX	Stellt Informationen für die Aktualisierung des Fortschrittsbalkens bereit; val1 enthält den neuen Wert des Verarbeitungsfortschritts. Dieser Wert kann im Bereich von 0 bis 1000 liegen, wobei 0 für 0 % erledigt und 1000 für 100 % erledigt steht.	4
DBSC_EVENT- TYPE_PROGRESS_TEXT	Der dem Fortschrittsbalken zugeordnete Text wurde aktualisiert und str1 enthält den neuen Wert.	5
DBSC_EVENT- TYPE_TITLE	Der Titel für das Synchronisationsfenster/Steuerelement wurde geändert, und str1 enthält den neuen Titel.	6
DBSC_EVENT- TYPE_SYNC_START	Die Synchronisation wurde gestartet. Es gibt keine zusätzlichen Informationen zu diesem Ereignis.	7
DBSC_EVENT- TYPE_SYNC_DONE	Die Synchronisation ist abgeschlossen; val1 enthält den Beendigungscode aus der Synchronisation. Der Wert 0 zeigt Erfolg an. Ein Wert ungleich Null zeigt an, dass die Synchronisation fehlgeschlagen ist.	8
DBSC_EVENT- TYPE_ML_CONNECT	Eine Verbindung zum MobiLink-Server wurde hergestellt; str1 gibt das verwendete Kommunikationsprotokoll an und str2 enthält die verwendeten Netzwerkprotokolloptionen.	10
DBSC_EVENT- TYPE_UPLOAD_COMMITTED	Der MobiLink-Server bestätigt, dass der Upload erfolgreich in der konsolidierten Datenbank festgeschrieben wurde.	11

Mitgliedsname	Beschreibung	Wert
DBSC_EVENT- TYPE_DOWNLOAD- COMMITTED	Der Download wurde erfolgreich in der entfernten Datenbank festgeschrieben. val1 enthält die Anzahl der festgeschriebenen Einfüge-/Aktualisierungsvorgänge. val2 enthält die Anzahl der festgeschriebenen Löschvorgänge.	12
DBSC_EVENT- TYPE_UPLOAD_START	Der entfernte Client hat mit dem Hochladen auf den Server begonnen.	13
DBSC_EVENT- TYPE_UPLOAD_SENT	Der entfernte Client hat das Senden eines Uploadsegments an den Server abgeschlossen. Für inkrementelle und transaktionale Uploads wird beim Senden jedes Upload-Segments ein Ereignis generiert. val1 enthält die Anzahl der gesendeten Einfügevorgänge. val2 enthält die Anzahl der gesendeten Aktualisierungsvorgänge. val3 enthält die Anzahl der gesendeten Löschvorgänge.	14
DBSC_EVENT- TYPE_DOWNLOAD- START	Der entfernte Client hat begonnen, den vom Server erhaltenen Download zu verarbeiten.	15

Siehe auch

- [DBSC_Event-Struktur \[Dbmlsync .NET\] auf Seite 316](#)
- [DbmlsyncClient.GetEvent-Methode \[Dbmlsync .NET\] auf Seite 299](#)

DBSC_GetEventRet-Enumeration

Gibt das Ergebnis eines Versuchs zur Abfrage eines Ereignisses an.

Visual Basic-Syntax

```
Public Enum DBSC_GetEventRet
```

C#-Syntax

```
public enum DBSC_GetEventRet
```

Mitglieder

Mitgliedsname	Beschreibung	Wert
DBSC_GETEVENT_OK	Zeigt an, dass ein Ereignis erfolgreich abgerufen wurde.	1

Mitgliedsname	Beschreibung	Wert
DBSC_GETE- VENT_TIMED_OUT	Zeigt an, dass der Timeout abgelaufen ist, ohne dass ein Ereignis für die Rückgabe verfügbar wurde.	2
DBSC_GETEVENT_FAI- LED	Zeigt an, dass aufgrund einer Fehlerbedingung kein Ereignis zurückgegeben wurde.	3

Siehe auch

- [DbmlsyncClient.GetEvent-Methode \[Dbmlsync .NET\] auf Seite 299](#)

DBSC_ShutdownType-Enumeration

Gibt an, wie dringend der Server heruntergefahren werden sollte.

Visual Basic-Syntax

```
Public Enum DBSC_ShutdownType
```

C#-Syntax

```
public enum DBSC_ShutdownType
```

Mitglieder

Mitgliedsname	Beschreibung	Wert
DBSC_SHUT- DOWN_ON_EMP- TY_QUEUE	Zeigt an, dass der Server alle noch anstehenden Synchronisationsanforderungen ausführen und dann heruntergefahren werden soll. Wenn der Server die Anforderung zum Herunterfahren empfängt, nimmt er keine weiteren Synchronisationsanforderungen mehr an.	1
DBSC_SHUT- DOWN_CLEANLY	Zeigt an, dass der Server so schnell wie möglich sauber heruntergefahren werden soll. Noch anstehende Synchronisationsanforderungen werden nicht ausgeführt, und wenn gerade eine Synchronisation ausgeführt wird, kann sie unterbrochen werden.	2

Siehe auch

- [DbmlsyncClient.ShutdownServer-Methode \[Dbmlsync .NET\] auf Seite 304](#)

DBSC_StartType-Enumeration

Zeigt die Aktion an, die bei einem Versuch, den dbmlsync-Server zu starten, ausgeführt wurde.

Visual Basic-Syntax

```
Public Enum DBSC_StartType
```

C#-Syntax

```
public enum DBSC_StartType
```

Mitglieder

Mitgliedsname	Beschreibung	Wert
DBSC_SS_STARTED	Zeigt an, dass ein neuer dbmlsync-Server gestartet wurde.	1
DBSC_SS_ALREADY_RUNNING	Zeigt an, dass ein vorhandener dbmlsync-Server gefunden und daher kein neuer Server gestartet wurde.	2

Siehe auch

- [DbmlsyncClient.StartServer-Methode \[Dbmlsync .NET\] auf Seite 305](#)

DBSC_ErrorInfo-Struktur

Enthält Informationen über einen Fehler eines vorherigen Methodenaufrufs.

Visual Basic-Syntax

```
Structure DBSC_ErrorInfo
```

C#-Syntax

```
public typedef struct DBSC_ErrorInfo
```

Mitglieder

Mitgliedsname	Typ	Beschreibung
hdl1	UInt32	Enthält zusätzliche Informationen über den Fehler. Die Bedeutung dieser Informationen hängt vom Wert der type-Variable ab.
str1	Zeichenfolge	Enthält zusätzliche Informationen über den Fehler. Die Bedeutung dieser Informationen hängt vom Wert der type-Variable ab.
str2	Zeichenfolge	Enthält zusätzliche Informationen über den Fehler. Die Bedeutung dieser Informationen hängt vom Wert der type-Variable ab.

Mitgliedsname	Typ	Beschreibung
type	DBSC_Error-Type	Enthält einen Wert, der den Grund für einen Fehlschlag angibt. Unterstützte Werte werden in der DBSC_ErrorType-Enumeration aufgeführt.
val1	Int32	Enthält zusätzliche Informationen über den Fehler. Die Bedeutung dieser Informationen hängt vom Wert der type-Variable ab.
val2	Int32	Enthält zusätzliche Informationen über den Fehler. Die Bedeutung dieser Informationen hängt vom Wert der type-Variable ab.

Bemerkungen

str1, str2, val1, val2 und hdl1 enthalten zusätzliche Informationen über den Fehler. Ihre Bedeutung hängt vom Fehlertyp ab. Die folgenden Fehlertypen verwenden Felder in dieser Struktur zum Speichern zusätzlicher Informationen:

- DBSC_ERR_CONNECTION_REJECTED
- DBSC_ERR_SERVER_SIDE_ERROR
- DBSC_ERR_NO_SYNC_ACK

Siehe auch

- [DBSC_ErrorType-Enumeration \[Dbmlsync .NET\] auf Seite 308](#)

DBSC_Event-Struktur

Enthält Informationen über ein Ereignis, das durch eine Synchronisation generiert wurde.

Visual Basic-Syntax

```
Structure DBSC_Event
```

C#-Syntax

```
public typedef struct DBSC_Event
```

Mitglieder

Mitgliedsname	Typ	Beschreibung
hdl	UInt32	Gibt die Synchronisation an, die das Ereignis generiert hat. Dieser Wert entspricht dem Wert, der von der Sync-Methode zurückgegeben wurde.
str1	Zeichenfolge	Enthält zusätzliche Informationen zum Ereignis. Die Bedeutung dieser Informationen hängt vom Wert der type-Variable ab.
str2	Zeichenfolge	Enthält zusätzliche Informationen zum Ereignis. Die Bedeutung dieser Informationen hängt vom Wert der type-Variable ab.
type	DBSC_Event-Type	Gibt den Typ des gemeldeten Ereignisses an.
val1	Int32	Enthält zusätzliche Informationen zum Ereignis. Die Bedeutung dieser Informationen hängt vom Wert der type-Variable ab.
val2	Int32	Enthält zusätzliche Informationen zum Ereignis. Die Bedeutung dieser Informationen hängt vom Wert der type-Variable ab.
val3	Int32	Enthält zusätzliche Informationen zum Ereignis. Die Bedeutung dieser Informationen hängt vom Wert der type-Variable ab.

Siehe auch

- [DBSC_EventType-Enumeration \[Dbmlsync .NET\] auf Seite 311](#)

DBTools-Schnittstelle für dbmlsync

"Database Tools (DBTools)" ist eine Bibliothek, mit der Sie Datenbankverwaltungsaufgaben, einschließlich der Synchronisation, in Ihre Anwendungen integrieren können. Alle Dienstprogramme für die Datenbankverwaltung sind auf DBTools aufgebaut.

Siehe „Datenbanktools-Schnittstelle (DBTools)“ [[SQL Anywhere Server - Programmierung](#)].

Hinweis

Die Dbmlsync-API ist die bevorzugte Schnittstelle für die Integration der Synchronisation in Ihre Anwendungen. Sie verfügt über Funktionen, die der DBTools-Schnittstelle sehr ähnlich sind, und ist einfacher zu verwenden.

Siehe „Dbmlsync-API“ auf Seite 103.

Mit der DBTools-Schnittstelle für dbmlsync können Sie Synchronisationsfunktionen in Ihre Clientanwendungen für die MobiLink-Synchronisation integrieren. Sie können zum Beispiel die Schnittstelle verwenden, um Synchronisationen zu starten und dbmlsync-Ausgabemeldungen auf einer speziell definierten Benutzeroberfläche anzuzeigen.

Die DBTools-Schnittstelle für dbmlsync besteht aus den folgenden Elementen, mit denen Sie den MobiLink-Synchronisationsclient konfigurieren und ausführen können:

- **a_sync_db-Struktur** Diese Struktur enthält Einstellungen, die den dbmlsync-Befehlszeilenoptionen entsprechen, mit denen Sie die Synchronisation anpassen können. Die Struktur enthält auch Zeiger auf Callback-Funktionen, die Synchronisations- und Fortschrittsinformationen empfangen.

Siehe [a_sync_db-Struktur \[Datenbanktools\]](#) [*SQL Anywhere Server - Programmierung*].

- **a_syncpub-Struktur** Diese Struktur enthält Publikations- oder Subskriptionsinformationen. Sie können eine verknüpfte Liste von Publikationen oder Subskriptionen für die Synchronisation angeben.

Siehe [a_syncpub-Struktur \[Datenbanktools\]](#) [*SQL Anywhere Server - Programmierung*].

- **DBSynchronizeLog-Funktion** Diese Funktion startet den Synchronisationsprozess. Ihr einziger Parameter ist ein Zeiger auf eine a_sync_db-Instanz.

Siehe [DBSynchronizeLog-Methode \[Datenbanktools\]](#) [*SQL Anywhere Server - Programmierung*].

DBTools-Schnittstelle für dbmlsync einrichten

Dieser Abschnitt enthält die wichtigsten Schritte zum Konfigurieren und Starten von dbmlsync unter Verwendung der DBTools-Schnittstelle in C oder C++.

Voraussetzungen

Es gibt keine Voraussetzungen für diese Aufgabe.

Kontext und Bemerkungen

Weitere Hinweise zur DBTools-Bibliothek finden Sie unter „[Datenbanktools-Schnittstelle \(DBTools\)](#)“ [*SQL Anywhere Server - Programmierung*].

Weitere Hinweise zur Verwendung von Importbibliotheken für Ihre Entwicklungsumgebung finden Sie unter „[DBTools-Importbibliotheken](#)“ [*SQL Anywhere Server - Programmierung*].

Aufgabe

1. Beziehen Sie die DBTools-Headerdatei ein.

Die DBTools-Headerdatei *dbtools.h* listet die Eintrittspunkte zur DBTools-Bibliothek auf und definiert die erforderlichen Datentypen.

```
#include "dbtools.h"
```

2. Starten Sie die DBTools-Schnittstelle.

- Deklarieren und initialisieren Sie die `a_dbtools_info`-Struktur.

```
a_dbtools_info  info;
short ret;
...
// clear a_dbtools_info fields
memset( &info, 0, sizeof( info ) );
info.errorrtn = dbsyncErrorCallBack;
```

Die `dbsyncErrorCallBack`-Funktion handhabt Fehlermeldungen und wird in Schritt 4 dieses Verfahrens beschrieben.

- Verwenden Sie die `DBToolsInit`-Funktion, um DBTools zu initialisieren.

```
ret = DBToolsInit( &info );
if( ret != 0 ) {
    printf("dbtools initialization failure \n");
}
```

Weitere Hinweise zur DBTools-Initialisierung finden Sie unter

- „Initialisierung und Finalisierung der DBTools-Bibliothek“ [[SQL Anywhere Server - Programmierung](#)]
- `a_dbtools_info`-Struktur [[Datenbanktools](#)] [[SQL Anywhere Server - Programmierung](#)]
- `DBToolsInit`-Methode [[Datenbanktools](#)] [[SQL Anywhere Server - Programmierung](#)]

3. Initialisieren Sie die `a_sync_db`-Struktur.

- Deklarieren Sie eine `a_sync_db`-Instanz. Beispiel: Deklarieren Sie eine Instanz namens `dbsync_info`:

```
a_sync_db dbsync_info;
```

- Löschen Sie die `a_sync_db`-Strukturfelder.

```
memset( &dbsync_info, 0, sizeof( dbsync_info ) );
```

- Legen Sie die erforderlichen `a_sync_db`-Felder fest.

```
dbsync_info.version = DB_TOOLS_VERSION_NUMBER;
dbsync_info.output_to_mobile_link = 1;
dbsync_info.default_window_title
    = "dbmlsync dbtools sample";
```

- Legen Sie die Datenbank-Verbindungszeichenfolge fest.

```
dbsync_info.connectparms = "UID=DBA;PWD=sql";
```

- Legen Sie die anderen a_sync_db-Felder fest, um die Synchronisation anzupassen.

Die meisten Felder entsprechen dbmlsync-Befehlszeilenoptionen.

Im unten stehenden Beispiel ist "Vorgang ausführlich darstellen" aktiviert.

```
dbsync_info.verbose_upload = 1;
dbsync_info.verbose_option_info = 1;
dbsync_info.verbose_row_data = 1;
dbsync_info.verbose_row_cnts = 1;
```

4. Erstellen Sie Callback-Funktionen, um während der Synchronisation Rückmeldungen zu erhalten, und ordnen Sie diese Funktionen den entsprechenden a_sync_db-Feldern zu.

Die folgenden Funktionen verwenden den Standard-Ausgebdatenstrom, um dbmlsync-Fehler-, Log- und Fortschrittsinformationen anzuzeigen.

- Beispiel: Erstellen Sie eine Funktion namens dbsyncErrorCallBack, um generierte Fehlermeldungen zu verarbeiten:

```
extern short _callback dbsyncErrorCallBack( char *str )
{
    if( str != NULL ) {
        printf( "Error Msg      %s\n", str );
    }
    return 0;
}
```

- Beispiel: Erstellen Sie eine Funktion namens dbsyncWarningCallBack, um generierte Warmmeldungen zu verarbeiten:

```
extern short _callback dbsyncWarningCallBack( char *str )
{
    if( str != NULL ) {
        printf( "Warning Msg   %s\n", str );
    }
    return 0;
}
```

- Beispiel: Erstellen Sie eine Funktion namens dbsyncLogCallBack, um ausführliche Informationsmeldungen zu erhalten, die Sie in einer Datei protokollieren, anstatt sie in einem Fenster anzuzeigen:

```
extern short _callback dbsyncLogCallBack( char *str )
{
    if( str != NULL ) {
        printf( "Log Msg        %s\n", str );
    }
    return 0;
}
```

- Beispiel: Erstellen Sie eine Funktion namens dbsyncMsgCallBack, um Informationsmeldungen zu erhalten, die während der Synchronisation generiert werden.

```
extern short _callback dbsyncMsgCallBack( char *str )
{
    if( str != NULL ) {
        printf( "Display Msg   %s\n", str );
    }
    return 0;
}
```

- Beispiel: Erstellen Sie eine Funktion namens `dbsyncProgressMessageCallBack`, um den Fortschritts-Text zu erhalten. Im `dbmlsync`-Dienstprogramm wird dieser Text direkt über dem Fortschrittsbalken angezeigt.

```
extern short _callback dbsyncProgressMessageCallBack(
    char *str )
{
    if( str != NULL ) {
        printf( "ProgressText %s\n", str );
    }
    return 0;
}
```

- Beispiel: Erstellen Sie eine Funktion namens `dbsyncProgressIndexCallBack`, um Informationen zum Aktualisieren einer Fortschrittsanzeige oder eines Fortschrittsbalkens zu erhalten. Diese Funktion erhält zwei Parameter:
 - **index** Eine Ganzzahl, die den aktuellen Stand des Fortschritts einer Synchronisation angibt
 - **max** Der maximale Fortschrittswert. Wenn dieser Wert Null ist, hat sich der maximale Wert nicht verändert, seitdem das Ereignis zuletzt ausgelöst wurde.

```
extern short _callback dbsyncProgressIndexCallBack(
    a_sql_uint32 index, a_sql_uint32 max )
{
    printf( "ProgressIndex      Index %d Max: %d\n",
            index, max );
    return 0;
}
```

Im Folgenden finden Sie eine typische Sequenz von Aufrufen an diese Callback-Funktion:

```
// example calling sequence
dbsyncProgressIndexCallBack( 0, 100 );
dbsyncProgressIndexCallBack( 25, 0 );
dbsyncProgressIndexCallBack( 50, 0 );
dbsyncProgressIndexCallBack( 75, 0 );
dbsyncProgressIndexCallBack( 100, 0 );
```

Diese Sequenz sollte dazu führen, dass der Fortschrittsbalken mit 0%, 25%, 50%, 75% und 100% dargestellt wird.

- Beispiel: Erstellen Sie eine Funktion namens `dbsyncWindowTitleCallBack`, um Statusinformationen zu erhalten. Im Dienstprogramm `dbmlsync` werden diese Informationen in der Titelleiste angezeigt.

```
extern short _callback dbsyncWindowTitleCallBack(
    char *title )
{
    printf( "Window Title      %s\n", title );
    return 0;
}
```

- Die `dbsyncMsgQueueCallBack`-Funktion wird aufgerufen, wenn eine Verzögerung oder ein Ruhezustand erforderlich ist. Sie muss einen der folgenden Werte zurückgeben, die in `dllapi.h` definiert sind.
 - **MSGQ_SLEEP_THROUGH** Zeigt an, dass die Routine über die angeforderte Anzahl von Millisekunden im Ruhezustand war.

- **MSGQ_SHUTDOWN_REQUESTED** Zeigt an, dass Sie die Synchronisation so schnell wie möglich beenden möchten
- **MSGQ_SYNC_REQUESTED** Zeigt an, dass die Routine für die angeforderte Anzahl von Millisekunden im Ruhezustand war und die nächste Synchronisation unmittelbar beginnen sollte, wenn derzeit keine Synchronisation durchgeführt wird.

```
extern short _callback dbsyncMsgQueueCallBack(  
    a_sql_uint32 sleep_period_in_milliseconds )  
{  
  
    printf( "Sleep %d ms\n", sleep_period_in_milliseconds );  
    Sleep( sleep_period_in_milliseconds );  
    return MSGQ_SLEEP_THROUGH;  
}
```

- Ordnen Sie Callback-Funktionszeiger den entsprechenden a_sync_db-Synchronisationsstrukturfeldern zu.

```
// set call back functions  
dbsync_info.errorrtn      = dbsyncErrorCallBack;  
dbsync_info.warningrtn   = dbsyncWarningCallBack;  
dbsync_info.logrtn       = dbsyncLogCallBack;  
dbsync_info.msgrtn       = dbsyncMsgCallBack;  
dbsync_info.msgqueue rtn = dbsyncMsgQueueCallBack;  
dbsync_info.progress_index_rtn  
    = dbsyncProgressIndexCallBack;  
dbsync_info.progress_msg_rtn  
    = dbsyncProgressMessageCallBack;  
dbsync_info.set_window_title_rtn  
    = dbsyncWindowTitleCallBack;
```

5. Erstellen Sie eine verknüpfte Liste von a_syncpub-Strukturen, um festzulegen, welche Subskriptionen synchronisiert werden sollen.

Jeder Knoten in der verknüpften Liste entspricht einer Instanz der Option -s in der dbmlsync-Befehlszeile.

- Deklarieren Sie eine a_syncpub-Instanz. Beispiel: Nennen Sie sie publication_info:

```
a_syncpub publication_info;
```

- Initialisieren Sie a_syncpub-Felder, indem Sie angeben, welche Subskriptionen Sie synchronisieren wollen.

Beispiel, um die Subskriptionen template_p1- und template_p2 zusammen in einer einzigen Synchronisationssitzung zu synchronisieren:

```
publication_info.next = NULL; // linked list terminates  
publication_info.subscription = "template_p1,template_p2";  
publication_info.ext_opt = "dir=c:\\logs";  
publication_info.allocated_by_dbsync = 0;  
publication_info.pub_name = NULL;
```

Das entspricht der Eingabe von -s template_p1,template_p2 in der dbmlsync-Befehlszeile.

Die Angabe von erweiterten Optionen mit dem ext_opt-Feld bietet dieselbe Funktionalität wie die Option -eu von dbmlsync.

- Ordnen Sie die Publikationsstruktur dem upload_defs-Feld Ihrer a_sync_db-Instanz zu.

```
dbsync_info.upload_defs = &publication_info;
```

Sie können eine verknüpfte Liste von a_syncpub-Strukturen erstellen. Jede a_syncpub-Instanz in der verknüpften Liste entspricht einer Spezifikation der Option -n oder -s in der dbmsync-Befehlszeile.

6. Führen Sie dbmsync aus, indem Sie die DBSynchronizeLog-Funktion verwenden.

In der folgenden Liste mit Codes enthält sync_ret_val den Rückgabewert "0" für "Erfolgreich" und "nicht-0" für "Fehlschlag".

```
short sync_ret_val;
printf("Running dbmsync using dbtools interface...\n");
sync_ret_val = DBSynchronizeLog(&dbsync_info);
printf("\n Done... synchronization return value is: %I \n", sync_ret_val);
```

Sie können Schritt 6 mit denselben oder mit anderen Parameterwerten mehrfach wiederholen.

7. Fahren Sie die DBTools-Schnittstelle herunter.

```
DBToolsFini( &info );
```

Die DBToolsFini-Funktion setzt DBTools-Ressourcen frei.

Ergebnisse

Dbmsync wird konfiguriert und gestartet.

Siehe auch

- „dbmsync-Option -c“ auf Seite 116
- „dbmsync-Option -eu“ auf Seite 123
- „dbmsync-Option -s“ auf Seite 133
- „dbmsync-Option -n (nicht mehr empfohlen)“ auf Seite 125
- a_syncpub-Struktur [Datenbanktools] [*SQL Anywhere Server - Programmierung*]
- DBToolsFini-Methode [Datenbanktools] [*SQL Anywhere Server - Programmierung*]
- „Initialisierung und Finalisierung der DBTools-Bibliothek“ [*SQL Anywhere Server - Programmierung*]
- a_dblic_info-Struktur [Datenbanktools] [*SQL Anywhere Server - Programmierung*]
- DBToolsInit-Methode [Datenbanktools] [*SQL Anywhere Server - Programmierung*]
- a_sync_db-Struktur [Datenbanktools] [*SQL Anywhere Server - Programmierung*]
- „Callback-Funktionen“ [*SQL Anywhere Server - Programmierung*]

Skriptgesteuerter Upload

Ein skriptgesteuerter Upload bezieht sich ausschließlich auf MobiLink-Anwendungen, die entfernte SQL Anywhere-Datenbanken verwenden.

Vorsicht

Wenn Sie einen skriptgesteuerten Upload implementieren, verwendet dbmlsync kein Transaktionslog, um zu ermitteln, welche Daten heraufgeladen werden sollen. Wenn Ihre Skripten also nicht alle Änderungen aufzeichnen, können in der entfernten Datenbank Daten verloren gehen. Daher wird für die meisten Anwendungen die logbasierte Synchronisation empfohlen.

Bei den meisten MobiLink-Anwendungen wird der Upload durch das Datenbank-Transaktionslog bestimmt. Auf diese Weise wird sichergestellt, dass Änderungen, die seit dem letzten Upload in der entfernten Datenbank durchgeführt wurden, synchronisiert werden. Dies ist für die meisten Anwendungen zu empfehlen und garantiert, dass in der entfernten Datenbank keine Daten verloren gehen.

In einigen Fällen kann es jedoch sinnvoll sein, das Transaktionslog zu ignorieren und den Upload selbst festzulegen. Mit dem skriptgesteuerten Upload können Sie genau definieren, welche Daten Sie heraufladen möchten. Bei skriptgesteuerten Uploads ist kein Transaktionslog für die entfernte Datenbank notwendig. Transaktionslogs erfordern Speicherplatz, der gerade bei kleineren Geräten begrenzt ist. Transaktionslogs sind jedoch für das Sichern und Wiederherstellen von Datenbanken und zur Leistungssteigerung der Datenbank äußerst wichtig.

Für einen skriptgesteuerten Upload erzeugen Sie eine spezielle Publikation, die die Namen von erstellten gespeicherten Prozeduren festlegt. Die gespeicherten Prozeduren definieren einen Upload, indem sie Ergebnismengen mit den Zeilen zurückgeben, die in der konsolidierten Datenbank eingefügt, aktualisiert oder gelöscht werden sollen.

Hinweis: Verwechseln Sie skriptgesteuerte Uploads nicht mit Upload-Skripten. Upload-Skripten sind MobiLink-Ereignisskripten in der konsolidierten Datenbank, die Sie erstellen, um dem MobiLink-Server Anweisungen zur Verarbeitung des Uploads zu geben. Bei skriptgesteuerten Uploads müssen Sie immer noch Upload-Skripten schreiben, um den Upload in die konsolidierte Datenbank zu übernehmen, und Download-Skripten, um festzulegen, was heruntergeladen werden soll.

Szenarios

In den folgenden Szenarios erhalten Sie Beispiele dafür, wann ein skriptgesteuerter Upload sinnvoll sein kann:

- Ihre entfernte Datenbank läuft auf einem Gerät mit begrenzter Speicherkapazität und Sie haben nicht genügend Platz für ein Transaktionslog.
- Sie möchten einen Upload aller Daten von allen entfernten Datenbanken durchführen, um eine neue konsolidierte Datenbank zu erstellen.
- Sie möchten eine benutzerdefinierte Logik erstellen, um die Änderungen festzulegen, die in die konsolidierte Datenbank geladen werden sollen.

Warnungen

Lesen Sie diesen Abschnitt ganz, bevor Sie skriptgesteuerte Uploads implementieren. Beachten Sie insbesondere die folgenden Punkte:

- Wenn Sie den skriptgesteuerten Upload nicht korrekt definieren, können Daten verloren gehen.

- Bei der Arbeit mit skriptgesteuerten Uploads müssen Sie Dinge berücksichtigen oder referenzieren, die normalerweise von dbmsync übernommen werden. Dazu gehören Pre- und Post-Images von Daten und der Synchronisationsfortschritt.
- Wahrscheinlich müssen Sie in der entfernten Datenbank für eine Synchronisation über skriptgesteuerte Uploads Tabellen sperren. Bei einer logbasierten Synchronisation ist eine Sperrung nicht erforderlich.
- Eine Implementierung von transaktionalen Uploads über skriptgesteuerte Uploads ist äußerst schwierig.

Skriptgesteuerte Uploads einrichten

Die folgenden Schritte geben einen Überblick darüber, wie Sie einen skriptgesteuerten Upload einrichten. Dabei wird vorausgesetzt, dass Sie bereits eine MobiLink-Synchronisation eingerichtet haben.

Vorsicht

Wenn Sie einen skriptgesteuerten Upload verwenden, wird nachdrücklich empfohlen, die Standardeinstellung für die erweiterte dbmsync-Option LockTables zu verwenden.

Sie können viele der Probleme mit skriptgesteuerten Uploads vermeiden, indem Sie die Standardeinstellung für LockTables verwenden. Dann erhält dbmsync Sperren für alle Synchronisationstabellen, bevor der Upload durchgeführt wird. Dadurch wird verhindert, dass andere Verbindungen die Synchronisationstabellen ändern, während die Skripte den Upload aufbauen. Außerdem wird so sichergestellt, dass keine nicht festgeschriebenen Transaktionen in offenen Synchronisationstabellen vorhanden sind, wenn der Upload aufgebaut wird.

1. **Erstellen Sie gespeicherte Prozeduren, um die Zeilen zu definieren, die heraufgeladen werden sollen** Sie können pro Tabelle drei gespeicherte Prozeduren definieren: Jeweils eine für Upload, Einfügen und Löschen.

Siehe „[Gespeicherte Prozeduren für skriptgesteuerten Upload](#)“ auf Seite 332.

2. **Erstellen Sie eine Publikation mit den Schlüsselwörtern WITH SCRIPTED UPLOAD und in denen die Namen der gespeicherten Prozeduren festgelegt sind** Siehe „[Publikationen für skriptgesteuerte Uploads](#)“ auf Seite 337.

Weitere Ressourcen für die ersten Schritte

- „[Praktische Einführung: Verwenden des skriptgesteuerten Uploads](#)“

Hinweise zur Planung von skriptgesteuerten Uploads

Ein Vorgang pro Zeile

Der Upload darf nicht mehr als einen Vorgang (Einfügen, Aktualisieren oder Löschen) pro Einzelzeile enthalten. Sie können jedoch mehrere Vorgänge in einem einzigen Upload kombinieren. Wenn beispielsweise eine Zeile eingefügt und dann aktualisiert wird, können Sie diese beiden Vorgänge in einem Vorgang kombinieren, indem Sie einmalig die endgültigen Werte einfügen.

Reihenfolge der Vorgänge

Wenn der Upload in die konsolidierte Datenbank aufgenommen wird, werden Einfügungen und Aktualisierungen vor Löschungen vorgenommen. Sie können diese Reihenfolge der Vorgänge in einer gegebenen Tabelle nicht ändern.

Konfliktbehandlung

Ein Konflikt tritt auf, wenn eine Zeile zwischen Synchronisationen in mehr als einer Datenbank aktualisiert wird. Der MobiLink-Server kann Konflikte erkennen, da jede Aktualisierung in einem Upload das Pre-Image der Zeile enthält, die aktualisiert werden soll. Das Pre-Image ist der Wert jeder Spalte in der Zeile zum Zeitpunkt des letzten erfolgreichen Uploads oder Downloads. Der MobiLink-Server meldet einen Konflikt, wenn das Pre-Image nicht mit den Werten in der konsolidierten Datenbank übereinstimmt.

Wenn Sie in Ihrer Anwendung die Konflikterkennung implementieren müssen und Sie skriptgesteuerte Uploads verwenden, müssen Sie in der entfernten Datenbank den Wert jeder Zeile zum Zeitpunkt des letzten erfolgreichen Uploads oder Downloads überprüfen. Auf diese Weise können Sie die korrekten Pre-Images hochladen.

Eine Möglichkeit, die Daten von Pre-Images zu pflegen, besteht darin, eine Pre-Image-Tabelle zu erstellen, die mit der Synchronisationstabelle identisch ist. Sie können dann in Ihrer Synchronisationstabelle einen Trigger erstellen, der die Pre-Image-Tabelle bei jeder Aktualisierung neu mit Daten füllt. Nach einem erfolgreichen Upload können Sie die Zeilen in der Pre-Image-Tabelle löschen.

Ein Beispiel zur Lösung von Konflikten finden Sie unter [„Praktische Einführung: Verwenden des skriptgesteuerten Uploads“](#) auf Seite 337.

Konflikte nicht behandeln

Wenn Sie Konflikte nicht behandeln müssen, können Sie die Anwendung deutlich vereinfachen, indem Sie Pre-Images nicht protokollieren. Stattdessen können Sie Aktualisierungen einfach in einem Einfügevorgang hochladen. Schreiben Sie dazu ein `upload_insert`-Skript in der konsolidierten Datenbank, das eine Zeile einfügt, wenn diese noch nicht vorhanden ist, oder eine bereits existierende Zeile aktualisiert. Wenn Sie eine konsolidierte SQL Anywhere-Datenbank verwenden, erreichen Sie dies mit der `ON EXISTING`-Klausel in der `INSERT`-Anweisung Ihres `upload_insert`-Skripts.

Siehe [„INSERT-Anweisung“](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Wenn Sie Konflikte nicht beheben und zwei oder mehr entfernte Datenbanken dieselbe Zeile ändern, überschreibt die zuletzt synchronisierte Zeile die früheren Änderungen.

Sperren

Sie können viele der Probleme mit skriptgesteuerten Uploads vermeiden, indem Sie die Standardeinstellung für die erweiterte `dbmlsync`-Option `LockTables` verwenden. Dann erhält `dbmlsync` Exklusivsperrungen für alle Synchronisationstabellen, bevor der Upload aufgebaut wird. Dadurch wird verhindert, dass andere Verbindungen die Synchronisationstabellen ändern, während die Skripten den Upload aufbauen. Außerdem wird so sichergestellt, dass keine nicht festgeschriebenen Transaktionen in offenen Synchronisationstabellen vorhanden sind, wenn der Upload aufgebaut wird.

Weitere Hinweise zur Aufhebung der Tabellensperrung finden Sie unter [„Skriptgesteuerter Upload ohne Tabellensperre“](#) auf Seite 329.

Redundante Uploads

Üblicherweise muss jeder Vorgang nur genau einmal in die entfernte Datenbank geladen werden. MobiLink unterstützt Sie dabei, indem für jede Subskription ein Fortschrittswert verwaltet wird. Standardmäßig entspricht der Fortschrittswert dem Zeitpunkt, an dem dbmlsync mit dem letzten erfolgreichen Upload begonnen hat. Dieser Fortschrittswert kann mithilfe des Hooks `sp_hook_dbmlsync_set_upload_end_progress` mit einem anderen Wert überschrieben werden.

Siehe „[sp_hook_dbmlsync_set_upload_end_progress](#)“ auf Seite 255.

Jedes Mal, wenn eine der Upload-Prozeduren aufgerufen wird, werden Werte über die Tabelle `#hook_dict` an sie übergeben. Dazu gehören die Werte für den 'Start-Verarbeitungsfortschritt' und 'Abschluss-Verarbeitungsfortschritt'. Diese Werte definieren den Zeitraum, in dem der Upload, der gerade aufgebaut wird, Änderungen der entfernten Datenbank enthalten sollte. Vorgänge, die vor dem 'Start-Verarbeitungsfortschritt' aufgetreten sind, wurden bereits heraufgeladen. Änderungen, die nach dem 'Abschluss-Verarbeitungsfortschritt' aufgetreten sind, sollten während der nächsten Synchronisation heraufgeladen werden.

Unbekannter Upload-Status

Ein häufiger Fehler bei der Implementierung von skriptgesteuerten Uploads ist die Erstellung von gespeicherten Prozeduren, die nur über die Hooks `sp_hook_dbmlsync_upload_end` oder `sp_hook_dbmlsync_end` ermitteln können, ob ein Upload erfolgreich in die konsolidierte Datenbank übernommen wurde. Diese Methode ist unzuverlässig.

Im folgenden Beispiel werden Einfügungen über ein Bit auf jeder Zeile durchgeführt, um zu überprüfen, ob eine Zeile aktualisiert werden muss. Das Bit wird beim Einfügen einer Zeile eingestellt und nach einem erfolgreichen Upload durch den Hook `sp_hook_dbmlsync_upload_end` gelöscht.

```
//
// DO NOT DO THIS!
//
CREATE TABLE t1 (
    pk            integer primary key,
    val           varchar( 256 ),
    to_upload     bit DEFAULT 1
);

CREATE PROCEDURE t1_ins()
RESULT( pk integer, val varchar(256) )
BEGIN
    SELECT pk, val
    FROM t1
    WHERE to_upload = 1;
END;

CREATE PROCEDURE sp_hook_dbmlsync_upload_end()
BEGIN
    DECLARE    upload_status    varchar(256);

    SELECT value
    INTO upload_status
    FROM #hook_dict
    WHERE name = 'upload status';

    if upload_status = 'committed' THEN
        UPDATE t1 SET to_upload = 0;
    end if;
END;
```

```
END IF
END;

CREATE PUBLICATION p1 WITH SCRIPTED UPLOAD (
  TABLE t1 USING ( PROCEDURE t1_ins FOR UPLOAD INSERT )
);
```

Dieser Ansatz funktioniert meistens. Der Vorgang schlägt fehl, wenn ein Hardware- oder Softwarefehler auftritt, durch den dbmlsync gestoppt wird, nachdem der Upload gesendet wurde, aber bevor er vom Server bestätigt wurde. In diesem Fall wird der Upload zwar möglicherweise in die konsolidierte Datenbank übernommen, aber der Hook `sp_hook_dbmlsync_upload_end` wird nicht aufgerufen und die `to_upload`-Bits werden nicht gelöscht. Das bedeutet, dass bei der nächsten Synchronisation Einfügungen für Zeilen heraufgeladen werden, für die bereits ein Upload erfolgt ist. In diesem Fall schlägt die Synchronisation normalerweise fehl, da in der konsolidierten Datenbank ein Fehler wegen mehrfach vorhandener Primärschlüssel generiert wird.

Probleme können außerdem auftreten, wenn die Verbindung mit dem MobiLink-Server abbricht, nachdem der Upload gesendet wurde, aber bevor er bestätigt wurde. In diesem Fall kann dbmlsync nicht feststellen, ob der Upload erfolgreich übernommen wurde. Das Dienstprogramm dbmlsync ruft den Hook `sp_hook_dbmlsync_upload_end` auf und setzt den Upload-Status auf unbekannt. Dieser Hook verhindert das Löschen der `to_upload`-Bits. Das ist korrekt, wenn der Upload nicht vom Server übernommen wurde. Wurde der Upload jedoch übernommen, tritt das Problem auf, das bereits im vorherigen Absatz beschrieben wurde. In beiden Fällen ist die betroffene entfernte Datenbank nicht in der Lage, eine weitere Synchronisation durchzuführen, solange das Problem nicht manuell behoben wurde.

Datenverlust während des Downloads verhindern

Wenn Sie skriptgesteuerte Uploads verwenden, kann es vorkommen, dass Daten in der entfernten Datenbank, für die ein Upload erfolgen muss, durch Daten überschrieben werden, die aus der konsolidierten Datenbank heruntergeladen werden. Dies bewirkt, dass Änderungen verloren gehen, die in der entfernten Datenbank durchgeführt wurden. Um diesen Datenverlust zu verhindern, müssen Ihre Upload-Prozeduren alle Änderungen enthalten, die in der entfernten Datenbank festgeschrieben wurden, bevor der Hook `sp_hook_dbmlsync_set_upload_end_progress` in den einzelnen Uploads aufgerufen wird.

Im nachfolgenden Beispiel wird gezeigt, wie Daten verloren gehen können, wenn Sie diese Regel verletzen:

Zeit	
1:05:00	Eine Zeile R, die sowohl in der konsolidierten Datenbank als auch in entfernten Datenbanken existiert, wird mit neuen Werten R1 in der entfernten Datenbank aktualisiert und die Änderung wird festgeschrieben.
1:06:00	Die Zeile R wird in der konsolidierten Datenbank mit neuen Werten R2 aktualisiert und die Änderung wird festgeschrieben.

Zeit	
1:07:00	Eine Synchronisation wird vorgenommen. Die Upload-Skripten sind so geschrieben, dass der Upload nur Vorgänge enthält, die vor 1:00:00 festgeschrieben wurden. Dies stellt einen Regelverstoß dar, weil dadurch die Vorgänge, die vor dem Aufbau des Uploads ausgeführt wurden, vom Upload nicht erfasst werden. Die Änderung von Zeile R wird nicht einbezogen, da sie nach 1:00:00 durchgeführt wurde. Der vom Server erhaltene Download enthält die Zeile R2. Wenn der Download übernommen wird, ersetzt die Zeile R2 die Zeile R1 in der entfernten Datenbank. Die Aktualisierung in der entfernten Datenbank geht verloren.

Skriptgesteuerter Upload ohne Tabellensperre

Standardmäßig sperrt dbmsync die zu synchronisierenden Tabellen, bevor Upload-Skripten aufgerufen werden, und hält diese Sperren aufrecht, bis der Download festgeschrieben wurde. Sie können die Tabellensperre verhindern, indem Sie die erweiterte Option LockTables auf "off" setzen.

Wir empfehlen, grundsätzlich das Standardverhalten für die Tabellensperre zu aktivieren. Bei skriptgesteuerten Uploads ohne Tabellensperren erhöht sich das Risiko von Problemen und der Schwierigkeitsgrad bei der Erarbeitung einer sauberen und durchführbaren Lösung. Nur erfahrene Anwender mit umfassenden Kenntnissen im Bereich der Datenbankparallelität und der Synchronisationskonzepte sollten vom Standardverhalten abweichende Einstellungen vornehmen.

Isolationsstufen ohne Tabellensperren verwenden

Wenn die Tabellensperren deaktiviert sind, ist die Isolationsstufe, unter der der Upload gespeicherter Prozeduren erfolgt, besonders wichtig, weil damit festgelegt wird, wie nicht festgeschriebene Transaktionen verwaltet werden. Dies ist kein Problem, wenn Tabellensperren aktiviert sind, weil Tabellensperren dafür sorgen, dass keine nicht festgeschriebenen Änderungen in den synchronisierten Tabellen vorhanden sind, wenn der Upload aufgebaut wird.

Ihre gespeicherten Upload-Prozeduren werden auf der Standardisolationsstufe für den Datenbankbenutzer ausgeführt, der in der dbmsync-Befehlszeile angegeben ist, sofern Sie die Isolationsstufe nicht ausdrücklich in der gespeicherten Upload-Prozedur geändert haben.

Isolationsstufe 0 ist zwar die Standardisolationsstufe für die Datenbank, es wird aber empfohlen, dass Sie Ihre Upload-Prozeduren nicht auf Isolationsstufe 0 ausführen, wenn Sie skriptgesteuerte Uploads ohne Tabellensperren ausführen. Wenn Sie skriptgesteuerte Uploads ohne Tabellensperren implementieren und Isolationsstufe 0 verwenden, können Sie einen Upload von Änderungen vornehmen, die nicht festgeschrieben wurden, woraus sich folgende Probleme ergeben könnten:

- Die nicht festgeschriebenen Änderungen können zurückgesetzt werden, was zur Folge hat, dass falsche Daten in die konsolidierte Datenbank eingelesen werden.
- Die nicht festgeschriebene Transaktion ist möglicherweise nicht vollständig, sodass der Upload nur einen Teil einer Transaktion enthält und die konsolidierte Datenbank inkonsistent wird.

Als Alternative können Sie Isolationsstufen 1, 2, 3 oder snapshot verwenden. Bei allen Isolationsstufen wird sichergestellt, dass kein Upload einer nicht festgeschriebenen Transaktion erfolgt.

Wenn Sie Isolationsstufen 1, 2 oder 3 verwenden, kann dies dazu führen, dass Ihre gespeicherten Upload-Prozeduren blockieren, wenn nicht festgeschriebene Änderungen in der Tabelle vorhanden sind. Da Ihre gespeicherten Upload-Prozeduren aufgerufen werden, während dbmlsync mit dem MobiLink-Server verbunden ist, kann dies Serververbindungen binden. Wenn Sie Isolationsstufe 1 verwenden, können Sie das Blockieren verhindern, indem Sie die Tabellen-Hint-Klausel READPAST in Ihren Anweisungen verwenden.

Snapshot-Isolation ist eine gute Alternative, da sie sowohl das Blockieren als auch das Lesen von nicht festgeschriebenen Änderungen verhindert.

Verlust von nicht festgeschriebenen Änderungen

Wenn Sie sich dafür entscheiden, Tabellensperren zu deaktivieren, müssen Sie einen Mechanismus verwenden, mit dem Sie Vorgänge verarbeiten können, die beim Eintritt einer Synchronisation nicht festgeschrieben sind. Im nachstehenden Beispiel können Sie erkennen, warum dies ein Problem ist.

Angenommen, eine Tabelle wird mit einem skriptgesteuerten Upload synchronisiert. Der Einfachheit halber wird angenommen, dass nur Einfügungen heraufgeladen werden. Die Tabelle enthält eine insert_time-Spalte, also einen Zeitstempel, der den Zeitpunkt angibt, zu dem eine Zeile eingefügt wurde.

Jeder Upload wird aufgebaut, indem alle festgeschriebenen Zeilen in der Tabelle ausgewählt werden, bei denen der Wert der insert_time-Spalte nach dem Zeitpunkt des letzten erfolgreichen Uploads und vor dem Zeitpunkt liegt, an dem Sie mit dem Aufbauen des aktuellen Uploads begonnen haben (also der Zeitpunkt, zu dem der Hook sp_hook_dbmlsync_set_upload_end_progress aufgerufen wurde). Angenommen, Folgendes tritt ein.

Zeit	
1:00:00	Eine erfolgreiche Synchronisation wird vorgenommen.
1:04:00	Zeile R wird in die Tabelle eingefügt, aber nicht festgeschrieben. Die insert_time-Spalte für R ist auf 1:04:00 gesetzt.
1:05:00	Eine Synchronisation wird vorgenommen. Der Upload der Zeilen mit insert_time-Werten zwischen 1:00:00 und 1:05:00 wird vorgenommen. Der Upload der Zeile R wird nicht vorgenommen, weil die Zeile nicht festgeschrieben ist. Der Synchronisationsfortschritt wird auf 1:05:00 gesetzt.
1:07:00	Die um 1:04:00 eingefügte Zeile wird festgeschrieben. Die insert_time-Spalte für R bleibt auf 1:04:00 gesetzt.
1:10:00	Eine Synchronisation wird vorgenommen. Der Upload der Zeilen mit insert_time-Werten zwischen 1:05:00 und 1:10:00 wird vorgenommen. Der Upload der Zeile R wird nicht vorgenommen, weil ihr insert_time-Wert außerhalb des Bereichs liegt. Das heißt, die Zeile R wird nie übertragen.

Grundsätzlich kann davon ausgegangen werden, dass ein Vorgang, der vor der Synchronisation eintritt und danach festgeschrieben wird, bei dieser Vorgehensweise verloren geht.

Verarbeitung nicht festgeschriebener Transaktionen

Die einfachste Art der Verarbeitung nicht festgeschriebener Transaktionen besteht darin, den Hook `sp_hook_dbmsync_set_upload_end_progress` zu verwenden, um das Verarbeitungsende für jede Synchronisation auf den Startzeitpunkt der ältesten, nicht festgeschriebenen Transaktion auf den Zeitpunkt zu setzen, zu dem der Hook aufgerufen wird. Sie können diesen Zeitpunkt mit der Systemprozedur `sa_transactions` wie folgt festlegen:

```
SELECT min( start_time )
FROM sa_transactions()
```

In diesem Fall müssen Ihre gespeicherten Upload-Prozeduren den Abschluss-Verarbeitungsfortschritt ignorieren, das im Hook `sp_hook_dbmsync_set_upload_end_progress` mit `sa_transactions` kalkuliert und über die `#hook_dict`-Tabelle übergeben wurde. Die gespeicherten Prozeduren müssen einfach einen Upload aller festgeschriebenen Vorgänge vornehmen, die nach dem Start-Verarbeitungsfortschritt eingetreten sind. Damit wird sichergestellt, dass der Download keine Zeilen mit Änderungen überschreibt, für die noch ein Upload durchgeführt werden muss. Außerdem wird damit gewährleistet, dass ein Upload von Vorgängen rechtzeitig erfolgt, auch wenn nicht festgeschriebene Transaktionen vorhanden sind.

Diese Lösung stellt sicher, dass keine Vorgänge verloren gehen. Allerdings kann es vorkommen, dass ein Upload einiger Vorgänge mehr als einmal erfolgt. Ihre serverseitigen Skripten müssen so geschrieben sein, dass sie Vorgänge verarbeiten, deren Upload mehr als einmal vorgenommen wird. Nachstehend sehen Sie ein Beispiel, in dem gezeigt wird, wie bei dieser Vorgehensweise der Upload einer Zeile mehr als einmal erfolgen kann.

Zeit	
1:00:00	Eine erfolgreiche Synchronisation wird vorgenommen.
2:00:00	Zeile R1 wird eingefügt, aber nicht festgeschrieben.
2:10:00	Zeile R2 wird eingefügt und festgeschrieben.
3:00:00	Eine Synchronisation wird vorgenommen. Ein Upload der Vorgänge, die zwischen 1:00 und 3:00 eingetreten sind, wird vorgenommen. Ein Upload der Zeile R2 wird vorgenommen und der Verarbeitungsfortschritt wird auf 2:00 gesetzt, weil dies der Startzeitpunkt der ältesten, nicht festgeschriebenen Transaktion ist.
4:00:00	Zeile R1 wird festgeschrieben.
5:00:00	Eine Synchronisation wird vorgenommen. Vorgänge, die zwischen 2:00 und 5:00 aufgetreten sind, werden hochgeladen und der Fortschritt wird auf 5:00 gesetzt. Der Upload enthält die Zeilen R1 und R2, da die Zeitstempel dieser Zeilen innerhalb des Upload-Zeitraums liegen. Daher wurde der Upload von R2 zweimal durchgeführt.

Wenn Ihre konsolidierte Datenbank eine SQL Anywhere-Datenbank ist, können Sie redundant hochgeladene Einfügevorgänge verarbeiten, indem Sie in Ihrem `upload_insert`-Skript in der konsolidierten Datenbank die `INSERT...ON EXISTING UPDATE`-Anweisung verwenden.

Bei anderen konsolidierten Datenbanken können Sie eine ähnliche Programmlogik in einer gespeicherten Prozedur implementieren, die von Ihrem `upload_insert`-Skript aufgerufen wird. Schreiben Sie eine

Prüfroutine, um zu verifizieren, ob eine Zeile mit dem Primärschlüssel der eingefügten Zeile bereits in der konsolidierten Datenbank existiert. Wenn die Zeile existiert, aktualisieren Sie sie, sonst fügen Sie die neue Zeile ein.

Redundante Uploads von Lösch- und Aktualisierungsvorgängen sind ein Problem, wenn Sie Konflikterkennung oder Konfliktbereinigungslogik auf der Serverseite eingerichtet haben. Wenn Sie Konflikterkennungs- und Konfliktlösungsskripten auf der Serverseite schreiben, müssen diese auch für die Verarbeitung redundanter Uploads ausgelegt sein.

Redundante Uploads von Löschvorgängen können wichtig werden, wenn Primärschlüsselwerte von der konsolidierten Datenbank wiederverwendet werden können. Es gibt die folgende Abfolge von Ereignissen:

1. Zeile R mit dem Primärschlüssel 100 wird in einer entfernten Datenbank eingefügt und ein Upload in die konsolidierte Datenbank erfolgt.
2. Zeile R wird in der entfernten Datenbank gelöscht und ein Upload des Löschvorgangs erfolgt.
3. Eine neue Zeile R' mit dem Primärschlüssel 100 wird in die konsolidierte Datenbank eingefügt.
4. Der Upload des Löschvorgangs in Zeile R aus Schritt 2 aus der entfernten Datenbank wird nochmals durchgeführt. Dies kann dazu führen, dass R' aus der konsolidierten Datenbank fälschlicherweise gelöscht wird.

Siehe auch

- „sa_transactions-Systemprozedur“ [[SQL Anywhere Server - SQL-Referenzhandbuch](#)]
- „So legen Sie die Isolationsstufe fest“ [[SQL Anywhere Server - SQL-Benutzerhandbuch](#)]

Gespeicherte Prozeduren für skriptgesteuerten Upload

Um einen skriptgesteuerten Upload zu implementieren, müssen Sie gespeicherte Prozeduren erstellen, die einen Upload definieren, indem sie Ergebnismengen mit den Zeilen zurückgeben, die in der konsolidierten Datenbank eingefügt, aktualisiert oder gelöscht werden sollen.

Wenn die gespeicherten Prozeduren aufgerufen werden, wird eine temporäre Tabelle mit dem Namen #hook_dict erstellt, die zwei Spalten enthält: Name und Value (Name und Wert). Die Tabelle wird verwendet, um Name-Wert-Paare an die gespeicherten Prozeduren zu übergeben. Die gespeicherten Prozeduren können aus dieser Tabelle wichtige Informationen abrufen.

Um sicherzustellen, dass gespeicherte Prozeduren auf die #hook_dict-Tabelle zugreifen können, müssen die Prozeduren eine der folgenden Anforderungen erfüllen:

- Eigentümer muss ein Benutzer sein, der die Systemprivilegien SELECT ANY TABLE und UPDATE ANY TABLE hat.
- Sie müssen mit der SQL SECURITY INVOKER-Klausel der CREATE PROCEDURE-Anweisung erstellt worden sein.

Die folgenden Name-Wert-Paare sind definiert:

Name	Wert	Beschreibung
start progress	Zeitstempel als Zeichenfolge	Die Zeit, in der alle Änderungen in der entfernten Datenbank heraufgeladen sind. Ihr Upload sollte nur Vorgänge nach dieser Zeit enthalten.
raw start progress	64-Bit-Ganzzahl ohne Vorzeichen	Der Start-Verarbeitungsfortschritt wird als Ganzzahl ohne Vorzeichen dargestellt.
end progress	Zeitstempel als Zeichenfolge	Das Ende des Upload-Zeitraums. Ihr Upload sollte nur Vorgänge vor dieser Zeit enthalten.
raw end progress	64-Bit-Ganzzahl ohne Vorzeichen	Der Abschluss-Verarbeitungsfortschritt wird als Ganzzahl ohne Vorzeichen dargestellt.
generating download exclusion list	TRUE FALSE	TRUE bei reinen Download- oder dateibasierten Synchronisationen. In diesen Fällen wird kein Upload gesendet und der Download wird nicht übernommen, wenn er sich auf eine Zeile bezieht, die von der gespeicherten Prozedur eines skriptgesteuerten Uploads ausgewählt wurde. (Damit wird sichergestellt, dass Änderungen in der entfernten Datenbank, die hochgeladen werden müssen, nicht durch den Download überschrieben werden.)
subscription_ <i>n</i>	Subskriptionsname(<i>n</i>)	Die synchronisierten Subskriptionen, wobei <i>n</i> eine Ganzzahl ist. Dies ist ein subscription_ <i>n</i> -Eintrag für jede synchronisierte Subskription. Die Nummerierung von <i>n</i> beginnt bei Null.
publication_ <i>n</i>	Publikationsname	Nicht mehr empfohlen. Verwenden Sie stattdessen subscription_ <i>n</i> . Die synchronisierten Publikationen, wobei <i>n</i> eine Ganzzahl ist. Die Nummerierung von <i>n</i> beginnt bei Null.
script version	Versionsname	Die MobiLink-Skriptversion, die für die Synchronisation verwendet werden soll
MobiLink user	MobiLink-Benutzername	Der MobiLink-Benutzer, für den Sie synchronisieren

Siehe auch

- „#hook_dict-Tabelle“ auf Seite 205

Benutzerdefinierte Fortschrittswerte in skriptgesteuerten Uploads

Standardmäßig sind die Start- und Abschluss-Verarbeitungsfortschrittswerte, die an die Prozeduren des skriptgesteuerten Uploads übergeben werden, Zeitstempel. Der Abschluss-Verarbeitungsfortschrittswert entspricht standardmäßig dem Zeitpunkt, an dem dbmlsync mit dem Upload beginnt. Der Start-Verarbeitungsfortschritt für eine Synchronisation entspricht immer dem Abschluss-Verarbeitungsfortschritt des letzten erfolgreichen Uploads dieser Subskription. Dies ist für die meisten Implementierungen zutreffend.

Der Hook `sp_hook_dbmlsync_set_upload_end_progress` ist in den seltenen Fällen notwendig, in denen ein anderes Verhalten erforderlich ist. Mit diesem Hook können Sie den Abschluss-Verarbeitungsfortschritt einstellen, der für einen Upload verwendet wird. Der Abschluss-Verarbeitungsfortschritt muss größer als der Start-Verarbeitungsfortschritt sein. Der Start-Verarbeitungsfortschritt kann nicht geändert werden.

Mit dem `sp_hook_dbmlsync_set_upload_end_progress`-Hook können Sie den Abschluss-Verarbeitungsfortschritt entweder als `TIMESTAMP`- oder als `UNSIGNED INTEGER`-Wert definieren. Die gespeicherten Prozeduren des Uploads erkennen beide Formen. Zur Vereinfachung können Sie die Funktionen `sa_convert_ml_progress_to_timestamp` und `sa_convert_timestamp_to_ml_progress` verwenden, um die Fortschrittswerte von einer Form in die andere zu konvertieren.

Siehe auch

- „`sp_hook_dbmlsync_set_upload_end_progress`“ auf Seite 255
- „`sa_convert_ml_progress_to_timestamp`-Systemprozedur“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- „`sa_convert_timestamp_to_ml_progress`-Systemprozedur“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]

Gespeicherte Prozeduren für Einfügungen

Die gespeicherten Prozeduren für Einfügungen müssen Ergebnismengen mit allen Spalten, die hochgeladen werden sollen, so zurückgeben, wie in der Anweisung `CREATE PUBLICATION` definiert, wobei die Spalten in der Reihenfolge aufgeführt werden müssen, die in der Anweisung `CREATE TABLE` festgelegt wurde.

Spaltenreihenfolge

Mit der folgenden Abfrage können Sie die Reihenfolge der Erstellung von Spalten in einer Tabelle mit dem Namen "t1" ermitteln:

```
SELECT column_name
FROM SYSTAB JOIN SYSTABCOL
WHERE table_name = 't1'
ORDER BY column_id
```

Beispiel

Eine ausführliche Erläuterung, wie Sie gespeicherte Prozeduren für Einfügungen definieren, finden Sie unter „[Praktische Einführung: Verwenden des skriptgesteuerten Uploads](#)“ auf Seite 337.

Im folgenden Beispiel wird eine Tabelle mit dem Namen "t1" und eine Publikation mit dem Namen "p1" erstellt. Die Publikation definiert WITH SCRIPTED UPLOAD und registriert die gespeicherte Prozedur t1_insert als Einfügeprozedur. In der Definition der gespeicherten Prozedur t1_insert enthält die Ergebnismenge alle Spalten, die in der Anweisung CREATE PUBLICATION aufgelistet wurden, jedoch in der Reihenfolge, in der die Spalten in der Anweisung CREATE TABLE festgelegt wurden.

```
CREATE TABLE t1(
  //The column ordering is taken from here
  pk integer primary key,
  c1 char( 30),
  c2 float,
  c3 double );

CREATE PROCEDURE t1_insert ( )
RESULT( pk integer, c1 char(30), c3 double )
begin
  ...
end

CREATE PUBLICATION WITH SCRIPTED UPLOAD p1(
  // Order of columns here is ignored
  TABLE t1( c3, pk, c1 ) USING (
    PROCEDURE t1_insert FOR UPLOAD INSERT
  )
)
```

Gespeicherte Prozeduren für Löschungen

Die gespeicherten Prozeduren für Löschungen müssen Ergebnismengen mit allen Spalten, die hochgeladen werden sollen, so zurückgeben, wie in der Anweisung CREATE PUBLICATION definiert, wobei die Spalten in der Reihenfolge aufgeführt werden müssen, die in der Anweisung CREATE TABLE festgelegt wurde.

Spaltenreihenfolge

Mit der folgenden Abfrage können Sie die Reihenfolge der Erstellung von Spalten in einer Tabelle mit dem Namen "t1" ermitteln:

```
SELECT column_name
FROM SYSTAB JOIN SYSTABCOL
WHERE table_name = 't1'
ORDER BY column_id
```

Beispiel

Eine ausführliche Erläuterung, wie Sie gespeicherte Prozeduren für Löschungen definieren, finden Sie unter [„Praktische Einführung: Verwenden des skriptgesteuerten Uploads“ auf Seite 337](#).

Im folgenden Beispiel wird eine Tabelle mit dem Namen "t1" und eine Publikation mit dem Namen "p1" erstellt. Die Publikation definiert WITH SCRIPTED UPLOAD und registriert die gespeicherte Prozedur t1_delete als Löschungsprozedur. In der Definition der gespeicherten Prozedur t1_delete enthält die Ergebnismenge alle Spalten, die in der Anweisung CREATE PUBLICATION aufgelistet wurden, jedoch in der Reihenfolge, in der die Spalten in der Anweisung CREATE TABLE festgelegt wurden.

```
CREATE TABLE t1(
  //The column ordering is taken from here
```

```
pk integer primary key,  
c1 char( 30),  
c2, float,  
c3 double );  
  
CREATE PROCEDURE t1_delete ()  
RESULT( pk integer, c1 char(30), c3 double )  
begin  
    ...  
end  
  
CREATE PUBLICATION p1 WITH SCRIPTED UPLOAD(  
    // Order of columns here is ignored  
    TABLE t1( c3, pk, c1 ) USING (  
        PROCEDURE t1_delete FOR UPLOAD DELETE  
    )  
)
```

Gespeicherte Prozeduren für Aktualisierungen

Die gespeicherte Prozedur für Aktualisierungen muss eine Ergebnismenge zurückgeben, die zwei Wertemengen enthält:

- Die erste Wertemenge legt das Pre-Image der Aktualisierung fest (die Werte in der Zeile zu dem Zeitpunkt, als sie das letzte Mal vom MobiLink-Server empfangen oder erfolgreich heraufgeladen wurden).
- Die zweite Wertemenge legt das Post-Image der Aktualisierung fest (die Werte, die die aktualisierte Zeile in der konsolidierten Datenbank enthalten soll).

Die gespeicherte Prozedur für Aktualisierungen muss eine Ergebnismenge mit doppelt so vielen Spalten zurückgeben wie die gespeicherte Einfüge- oder Löschprozedur.

Beispiel

Eine ausführliche Erläuterung, wie Sie gespeicherte Prozeduren für Aktualisierungen definieren, finden Sie unter [„Praktische Einführung: Verwenden des skriptgesteuerten Uploads“ auf Seite 337](#).

Im folgenden Beispiel wird eine Tabelle mit dem Namen "t1" und eine Publikation mit dem Namen "p1" erstellt. Die Publikation definiert WITH SCRIPTED UPLOAD und registriert die gespeicherte Prozedur t1_update als Aktualisierungsprozedur. Die Publikation legt die drei zu synchronisierenden Spalten fest: pk, c1 und c3. Die Aktualisierungsprozedur gibt eine Ergebnismenge mit sechs Spalten zurück. Die ersten drei Spalten enthalten das Pre-Image der Spalten pk, c1 und c3. Die letzten drei Spalten enthalten das Post-Image derselben Spalten. In beiden Fällen sind die Spalten so sortiert wie bei der Erstellung der Tabelle und nicht wie in der CREATE PUBLICATION-Anweisung festgelegt.

```
CREATE TABLE t1(  
    //Column ordering is taken from here  
    pk integer primary key,  
    c1 char( 30),  
    c2 float,  
    c3 double );  
  
CREATE PROCEDURE t1_update ()  
RESULT( preimage_pk integer, preimage_c1 char(30), preimage_c3 double,  
    postimage_pk integer, postimage_c1 char(30), postimage_c3 double )
```

```

BEGIN
...
END

CREATE PUBLICATION WITH SCRIPTED UPLOAD p1 (
    // Order of columns here is ignored
    TABLE t1( c3, pk, c1 ) USING (
        PROCEDURE t1_update FOR UPLOAD UPDATE
    )
)

```

Publikationen für skriptgesteuerte Uploads

Verwenden Sie bei der Erstellung eines skriptgesteuerten Uploads die Schlüsselwörter WITH SCRIPTED UPLOAD und definieren Sie die gespeicherte Prozedur in der USING-Klausel.

Wenn Sie keine gespeicherte Prozedur für eine Tabelle in der Publikation des skriptgesteuerten Uploads definieren, werden keine Vorgänge in die Tabelle geladen. Sie können normale Publikationen nicht mit ALTER PUBLICATION in eine skriptgesteuerte Upload-Publikation umwandeln.

Beispiel

Die folgende Publikation verwendet gespeicherte Prozeduren, um Daten für die zwei Tabellen "t1" und "t2" heraufzuladen. Für Tabelle t1 werden Einfügungen, Löschungen und Aktualisierungen heraufgeladen. Für Tabelle t2 werden nur Einfügungen heraufgeladen.

```

CREATE PUBLICATION pub WITH SCRIPTED UPLOAD (
    TABLE t1 (col1, col2, col3) USING (
        PROCEDURE my.t1_ui FOR UPLOAD INSERT,
        PROCEDURE my.t1_ud FOR UPLOAD DELETE,
        PROCEDURE my.t1_uu FOR UPLOAD UPDATE
    ),
    TABLE t2 USING (
        PROCEDURE my.t2_ui FOR UPLOAD INSERT
    )
)

```

Siehe auch

- „CREATE PUBLICATION-Anweisung [MobiLink] [SQL Remote]“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- „ALTER PUBLICATION-Anweisung [MobiLink] [SQL Remote]“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]

Praktische Einführung: Verwenden des skriptgesteuerten Uploads

Privilegien

Sie müssen die folgenden Rollen und Privilegien haben:

- SYS_REPLICATION_ADMIN_ROLE-Systemrolle

- SYS_RUN_REPLICATION_ROLE-Systemrolle
- CREATE ANY TRIGGER-Systemprivileg

Lektion 1: Erstellen der konsolidierten Datenbank

In dieser praktischen Einführung erfahren Sie, wie Sie einen skriptgesteuerten Upload mit Konflikterkennung einrichten. In der praktischen Einführung werden die konsolidierte und die entfernte Datenbank, gespeicherte Prozeduren, Publikationen und Subskriptionen erstellt, die für den skriptgesteuerten Upload erforderlich sind.

Voraussetzungen

In dieser Lektion wird davon ausgegangen, dass Sie die Rollen und Privilegien haben, die im Abschnitt "Privilegien" am Anfang dieser praktischen Einführung aufgeführt sind: „[Praktische Einführung: Verwenden des skriptgesteuerten Uploads](#)“.

Kontext und Bemerkungen

In dieser praktischen Einführung werden Dateinamen angegeben und es wird davon ausgegangen, dass sich die Dateien im aktuellen Verzeichnis befinden, nämlich *scriptedupload*. In einer Echtanwendung müssen Sie den vollen Pfad angeben.

Sie können entweder die praktische Einführung einfach durchlesen oder den Text ausschneiden und einfügen, um das Beispiel auszuführen.

Aufgabe

1. Führen Sie den folgenden Befehl aus, um ein Verzeichnis für die Dateien der praktischen Einführung zu erstellen, und wechseln Sie zu diesem Verzeichnis.

```
md c:\scriptedupload
cd c:\scriptedupload
```

2. Geben Sie folgenden Befehl ein, um eine konsolidierte Datenbank zu erstellen:

```
dbinit -dba DBA,sql consol.db
```

3. Führen Sie dann den folgenden Befehl aus, um eine ODBC-Datenquelle für die konsolidierte Datenbank zu definieren:

```
dbdsn -w dsn_consol -y -c "UID=DBA;PWD=sql;DBF=consol.db;SERVER=consol"
```

4. Um eine Datenbank als konsolidierte MobiLink-Datenbank zu verwenden, müssen Sie ein Setup-Skript ausführen, das Systemtabellen, Ansichten und gespeicherte Prozeduren hinzufügt, die von MobiLink verwendet werden. Der folgende Befehl richtet *consol.db* als konsolidierte Datenbank ein:

```
dbisql -c "DSN=dsn_consol" "%SQLANY16%\MobiLink\setup\syncsa.sql"
```

5. Führen Sie den folgenden Befehl aus, um Interactive SQL zu öffnen und eine Verbindung mit *consol.db* unter Verwendung von "dsn_consol" herzustellen:

```
dbisql -c "DSN=dsn_consol"
```

6. Führen Sie die folgenden SQL-Anweisungen aus. Sie erstellen die Tabelle employee in der konsolidierten Datenbank, fügen Werte in die Tabelle ein und erstellen die erforderlichen Synchronisationsskripten.

```
CREATE TABLE employee (
    id      unsigned integer primary key,
    name    varchar( 256),
    salary  numeric( 9, 2 )
);

INSERT INTO employee VALUES( 100, 'smith', 225000 );
COMMIT;

CALL ml_add_table_script( 'default', 'employee', 'upload_insert',
    'INSERT INTO employee ( id, name, salary ) VALUES ( {ml r.id}, {ml
r.name}, {ml r.salary} )' );

CALL ml_add_table_script( 'default', 'employee', 'upload_update',
    'UPDATE employee SET name = {ml r.name}, salary = {ml r.salary}
WHERE id = {ml r.id}' );

CALL ml_add_table_script( 'default', 'employee', 'upload_delete',
    'DELETE FROM employee WHERE id = {ml r.id}' );

CALL ml_add_table_script( 'default', 'employee', 'download_cursor',
    'SELECT * from employee' );
```

Lassen Sie nach dem Ausführen der SQL-Anweisungen Interactive SQL aktiviert und mit der konsolidierten Datenbank verbunden, da Sie weitere SQL-Anweisungen in der Datenbank ausführen werden, während Sie die praktische Einführung durcharbeiten.

Ergebnisse

Die konsolidierte Datenbank wird erstellt und für die Verwendung mit MobiLink eingerichtet.

Nächste Schritte

Gehen Sie weiter zu [„Lektion 2: Erstellen der entfernten Datenbank“ auf Seite 339](#).

Lektion 2: Erstellen der entfernten Datenbank

In dieser Lektion erstellen Sie eine entfernte Datenbank und füllen sie mit Objekten.

Voraussetzungen

In dieser Lektion wird davon ausgegangen, dass Sie die Rollen und Privilegien haben, die im Abschnitt "Privilegien" am Anfang dieser praktischen Einführung aufgeführt sind: [„Praktische Einführung: Verwenden des skriptgesteuerten Uploads“](#).

In dieser Lektion wird davon ausgegangen, dass Sie bereits alle vorherigen Lektionen abgeschlossen haben. Siehe [„Lektion 1: Erstellen der konsolidierten Datenbank“ auf Seite 338](#).

Aufgabe

1. Führen Sie an der Eingabeaufforderung in Ihrem Beispielverzeichnis den folgenden Befehl aus, um die entfernte Datenbank zu erstellen.

```
dbinit -dba DBA,sql remote.db
```

2. Führen Sie dann den folgenden Befehl aus, um eine ODBC-Datenquelle zu definieren:

```
dbdsn -w dsn_remote -y -c "UID=DBA;PWD=sql;DBF=remote.db;SERVER=remote"
```

3. Führen Sie den folgenden Befehl aus, um Interactive SQL zu öffnen und eine Verbindung mit *remote.db* unter Verwendung von "dsn_remote" herzustellen:

```
dbisql -c "DSN=dsn_remote"
```

4. Führen Sie folgende Anweisungen aus, um in der entfernten Datenbank Objekte zu erstellen:

Erstellen Sie zunächst die Tabelle, die synchronisiert werden soll. Die Spalten `insert_time` und `delete_time` werden nicht synchronisiert, enthalten aber Informationen, die von der gespeicherten Upload-Prozedur verwendet werden, um festzulegen, welche Zeilen hochgeladen werden sollen.

```
CREATE TABLE employee (  
    id            unsigned integer primary key,  
    name          varchar( 256),  
    salary        numeric( 9, 2 ),  
    insert_time   timestamp default '1900-01-01'  
);
```

Lassen Sie nach dem Ausführen der SQL-Anweisungen Interactive SQL aktiviert und mit der entfernten Datenbank verbunden, da Sie weitere SQL-Anweisungen in der Datenbank ausführen werden, während Sie die praktische Einführung durcharbeiten.

Ergebnisse

Die entfernte Datenbank wird erstellt.

Nächste Schritte

Gehen Sie weiter zu „[Lektion 3: Umgang mit Einfügungen](#)“ auf Seite 340.

Lektion 3: Umgang mit Einfügungen

Sie müssen gespeicherte Prozeduren und andere Komponenten definieren, um den Upload verarbeiten zu können. Dies geschieht für Einfügungen, Aktualisierungen und Löschungen getrennt.

Voraussetzungen

In dieser Lektion wird davon ausgegangen, dass Sie die Rollen und Privilegien haben, die im Abschnitt "Privilegien" am Anfang dieser praktischen Einführung aufgeführt sind: „[Praktische Einführung: Verwenden des skriptgesteuerten Uploads](#)“.

In dieser Lektion wird davon ausgegangen, dass Sie bereits alle vorherigen Lektionen abgeschlossen haben. Siehe „[Lektion 1: Erstellen der konsolidierten Datenbank](#)“ auf Seite 338.

Aufgabe

1. Wählen Sie die mit der entfernten Datenbank verbundene Instanz von Interactive SQL und erstellen Sie mithilfe der folgenden SQL-Anweisungen einen Trigger, um den insert_time-Wert für jede Zeile festzulegen, wenn sie eingefügt wird.

```
CREATE TRIGGER emp_ins AFTER INSERT ON employee
REFERENCING NEW AS newrow
FOR EACH ROW
BEGIN
    UPDATE employee SET insert_time = CURRENT_TIMESTAMP
    WHERE id = newrow.id
END;
```

Dieser Zeitstempel wird verwendet, um festzustellen, ob seit der letzten Synchronisation eine Zeile eingefügt wurde. Dieser Trigger wird nicht ausgelöst, wenn dbmsync übertragene Einfügungen aus der konsolidierten Datenbank anwendet, da Sie später in diesem Beispiel die erweiterte Option FireTriggers deaktivieren werden. Zeilen, die durch den Download eingefügt werden, erhalten den insert_time-Wert "1900-01-01", also den Standardwert, der bei der Erstellung der Tabelle employee definiert wurde. Dieser Wert sollte immer vor dem Beginn des Verarbeitungsfortschritts liegen, damit solche Zeilen nicht als neue Einfügungen behandelt und bei der nächsten Synchronisation nicht hochgeladen werden.

2. Erstellen Sie als Nächstes, ebenfalls in der entfernten Datenbank, eine Prozedur, die als Ergebnismenge alle eingefügten Zeilen zurückgibt, die Teil des Uploads sein sollen.

```
CREATE PROCEDURE employee_insert()
RESULT( id unsigned integer,
        name varchar( 256 ),
        salary numeric( 9,2 )
)
BEGIN
    DECLARE start_time timestamp;

    SELECT value
    INTO start_time
    FROM #hook_dict
    WHERE name = 'start progress as timestamp';

    // Upload as inserts all rows inserted after the start_time
    // that were not subsequently deleted
    SELECT id, name, salary
    FROM employee e
    WHERE insert_time > start_time AND
        NOT EXISTS( SELECT id FROM employee_delete ed WHERE ed.id =
e.id );

END;
```

Ergebnisse

Diese Prozedur gibt alle Zeilen zurück, die basierend auf insert_time seit dem letzten erfolgreichen Upload eingefügt und anschließend nicht gelöscht wurden. Der Zeitpunkt des letzten erfolgreichen Uploads wird durch den Wert des Start-Verarbeitungsfortschritts in der Tabelle #hook_dict definiert.

Dieses Beispiel benutzt die Standardeinstellung für die erweiterte dbmsync-Option LockTables, sodass dbmsync die zu synchronisierenden Tabellen sperrt. Daher brauchen Sie keine Zeilen auszuschließen, die nach dem Bearbeitungsfortschritts-Abschluss eingefügt wurden. Die Tabellensperren verhindern jegliche Vorgänge nach dem Bearbeitungsfortschritts-Abschluss, während der Upload durchgeführt wird.

Nächste Schritte

Gehen Sie weiter zu [„Lektion 4: Umgang mit Aktualisierungen“ auf Seite 342](#).

Lektion 4: Umgang mit Aktualisierungen

In dieser Lektion erstellen Sie eine Tabelle, einen Trigger und eine gespeicherte Prozedur zum Verarbeiten von Aktualisierungen.

Voraussetzungen

In dieser Lektion wird davon ausgegangen, dass Sie die Rollen und Privilegien haben, die im Abschnitt "Privilegien" am Anfang dieser praktischen Einführung aufgeführt sind: [„Praktische Einführung: Verwenden des skriptgesteuerten Uploads“](#).

In dieser Lektion wird davon ausgegangen, dass Sie bereits alle vorherigen Lektionen abgeschlossen haben. Siehe [„Lektion 1: Erstellen der konsolidierten Datenbank“ auf Seite 338](#).

Kontext und Bemerkungen

Bei Aktualisierungen müssen Sie zunächst sicherstellen, dass während des Uploads das korrekte Pre-Image basierend auf dem Start-Verarbeitungsfortschritt verwendet wird.

Aufgabe

1. Wählen Sie die mit der entfernten Datenbank verbundene Instanz von Interactive SQL und erstellen Sie eine Tabelle, in der die Pre-Images der aktualisierten Zeilen verwaltet werden. Die Pre-Images werden für das Generieren der skriptgesteuerten Uploads benötigt.

```
CREATE TABLE employee_preimages (  
    id            unsigned integer NOT NULL,  
    name          varchar( 256),  
    salary        numeric( 9, 2 ),  
    img_time      timestamp default CURRENT_TIMESTAMP,  
    primary key( id, img_time )  
);
```

2. Erstellen Sie dann einen Trigger, um ein Pre-Image für jede Zeile zu speichern, wenn sie aktualisiert wird. Wie bereits der Insert-Trigger wird dieser Trigger beim Download nicht ausgeführt.

```
CREATE TRIGGER emp_upd AFTER UPDATE OF name,salary ON employee  
    REFERENCING OLD AS oldrow  
    FOR EACH ROW  
BEGIN  
    INSERT INTO employee_preimages ON EXISTING SKIP VALUES(  
        oldrow.id, oldrow.name, oldrow.salary, CURRENT_TIMESTAMP );  
END;
```

Dieser Trigger speichert ein Pre-Image der Zeile, sobald die Zeile aktualisiert wird (es sei denn, zwei Aktualisierungen liegen so nah beieinander, dass sie denselben Zeitstempel erhalten). Auf den ersten Blick erscheint dies überflüssig. Man könnte versucht sein, ein Pre-Image nur dann zu speichern, wenn dieses für die entsprechende Zeile in der Tabelle nicht existiert, und dann mithilfe des Hooks `sp_hook_dbmsync_upload_end` die Pre-Images zu löschen, sobald sie heraufgeladen wurden.

Der Hook `sp_hook_dbmsync_upload_end` ist hierfür jedoch nicht zuverlässig genug. Der Hook wird beispielsweise nicht aufgerufen, wenn `dbmsync` aufgrund eines Hard- oder Softwarefehlers gestoppt wird, nachdem der Upload gesendet, aber noch nicht bestätigt wurde. Dies führt dazu, dass Zeilen nicht aus Pre-Image-Tabellen gelöscht werden, obwohl sie erfolgreich heraufgeladen wurden. Auch ein Verbindungsfehler kann dazu führen, dass `dbmsync` keine Bestätigung für einen Upload vom Server erhält. In diesem Fall wird dem Hook der Upload-Status 'unknown' übergeben. Dann kann der Hook nicht erkennen, ob die Pre-Image-Tabelle bereinigt werden muss oder unverändert bleiben soll. Beim Speichern mehrerer Pre-Images kann das korrekte Pre-Image basierend auf dem Start-Verarbeitungsfortschritt beim Upload ausgewählt werden.

3. Erstellen Sie als Nächstes eine Upload-Prozedur für das Bearbeiten von Aktualisierungen.

```
CREATE PROCEDURE employee_update()
RESULT(
    preimage_id unsigned integer,
    preimage_name varchar( 256),
    preimage_salary numeric( 9,2 ),
    postimage_id unsigned integer,
    postimage_name varchar( 256),
    postimage_salary numeric( 9,2 )
)
BEGIN
    DECLARE start_time timestamp;

    SELECT value
    INTO start_time
    FROM #hook_dict
    WHERE name = 'start progress as timestamp';

    // Upload as an update all rows that have been updated since
    // start_time that were not newly inserted or deleted.
    SELECT ep.id, ep.name, ep.salary, e.id, e.name, e.salary
    FROM employee e JOIN employee_preimages ep
    ON ( e.id = ep.id )
    // Do not select rows inserted since the start time. These should be
    // uploaded as inserts.
    WHERE insert_time <= start_time
    // Do not upload deleted rows.
    AND NOT EXISTS( SELECT id FROM employee_delete ed WHERE ed.id =
e.id )
    // Select the earliest pre-image after the start time.
    AND ep.img_time = ( SELECT MIN( img_time )
        FROM employee_preimages
        WHERE id = ep.id
        AND img_time > start_time );

END;
```

Diese gespeicherte Prozedur gibt eine Ergebnismenge mit doppelt so vielen Spalten zurück wie andere Skripten: Sie enthält das Pre-Image (die Werte in der Zeile zu dem Zeitpunkt, als sie das letzte Mal vom MobiLink-Server empfangen oder erfolgreich heraufgeladen wurden) und das Post-Image (die Werte, die in die konsolidierte Datenbank eingegeben werden sollen).

Das Pre-Image ist die erste Wertemenge in employee_preimages, die nach start_progress aufgezeichnet wurde. In diesem Beispiel werden vorhandene Zeilen, die gelöscht und erneut eingefügt werden, nicht korrekt verarbeitet. In einer komplexeren Lösung würden diese als Aktualisierung hochgeladen.

Ergebnisse

Erstellt werden eine Tabelle zum Speichern von Pre-Images, ein Trigger zum Speichern von Pre-Images jeder aktualisierten Zeile und eine gespeicherte Prozedur zum Verarbeiten der Aktualisierungen.

Nächste Schritte

Gehen Sie weiter zu [„Lektion 5: Umgang mit Löschungen“ auf Seite 344](#).

Lektion 5: Umgang mit Löschungen

In dieser Lektion erstellen Sie eine Tabelle, einen Trigger und eine gespeicherte Prozedur zum Verarbeiten von Löschungen.

Voraussetzungen

In dieser Lektion wird davon ausgegangen, dass Sie die Rollen und Privilegien haben, die im Abschnitt "Privilegien" am Anfang dieser praktischen Einführung aufgeführt sind: [„Praktische Einführung: Verwenden des skriptgesteuerten Uploads“](#).

In dieser Lektion wird davon ausgegangen, dass Sie bereits alle vorherigen Lektionen abgeschlossen haben. Siehe [„Lektion 1: Erstellen der konsolidierten Datenbank“ auf Seite 338](#).

Aufgabe

1. Wählen Sie die mit der entfernten Datenbank verbundene Instanz von Interactive SQL und erstellen Sie eine Tabelle, in der eine Liste der gelöschten Zeilen verwaltet wird:

```
CREATE TABLE employee_delete (  
    id            unsigned integer  primary key NOT NULL,  
    name          varchar( 256 ),  
    salary        numeric( 9, 2 ),  
    delete_time  timestamp  
);
```

2. Erstellen Sie dann einen Trigger, um die Tabelle employee_delete beim Löschen von Zeilen aus der Tabelle employee zu füllen.

```
CREATE TRIGGER emp_del AFTER DELETE ON employee  
REFERENCING OLD AS delrow  
FOR EACH ROW  
BEGIN  
    INSERT INTO employee_delete  
VALUES( delrow.id, delrow.name, delrow.salary, CURRENT_TIMESTAMP );  
END;
```

Dieser Trigger wird während eines Downloads nicht aufgerufen, da Sie später die erweiterte Option FireTriggers in dbmlsync auf FALSE setzen. Dieser Trigger setzt voraus, dass eine gelöschte Zeile nicht wieder eingefügt wird. Daher verarbeitet er das Löschen derselben Zeile nicht mehr als einmal.

3. Die nächste SQL-Anweisung erstellt eine Upload-Prozedur für das Bearbeiten von Löschungen.

```
CREATE PROCEDURE employee_delete()
RESULT( id unsigned integer,
        name varchar( 256),
        salary numeric( 9,2 )
      )
BEGIN
  DECLARE start_time timestamp;

  SELECT value
  INTO start_time
  FROM #hook_dict
  WHERE name = 'start progress as timestamp';

  // Upload as a delete all rows that were deleted after the
  // start_time that were not inserted after the start_time.
  // If a row was updated before it was deleted, then the row
  // to be deleted is the pre-image of the update.
  SELECT IF ep.id IS NULL THEN ed.id ELSE ep.id ENDIF,
         IF ep.id IS NULL THEN ed.name ELSE ep.name ENDIF,
         IF ep.id IS NULL THEN ed.salary ELSE ep.salary ENDIF
  FROM employee_delete ed LEFT OUTER JOIN employee_preimages ep
  ON( ed.id = ep.id AND ep.img_time > start_time )
  WHERE
    // Only upload deletes that occurred since the last sync.
    ed.delete_time > start_time
    // Don't upload a delete for rows that were inserted since
    // the last upload and then deleted.
    AND NOT EXISTS (
      SELECT id
      FROM employee e
      WHERE e.id = ep.id AND e.insert_time > start_time )
  // Select the earliest preimage after the start time.
  AND ( ep.id IS NULL OR ep.img_time = (SELECT MIN( img_time )
                                         FROM employee_preimages
                                         WHERE id = ep.id
                                         AND img_time > start_time ) );

END;
```

Diese gespeicherte Prozedur gibt eine Ergebnismenge zurück, die die Zeilen enthält, die in der konsolidierten Datenbank gelöscht werden sollen. Die gespeicherte Prozedur verwendet die Tabelle employee_preimages. Wenn also eine Zeile aktualisiert und dann gelöscht wird, handelt es sich bei dem Image, das für die Löschung heraufgeladen wird, um das Image, das als Letztes erfolgreich herunter- oder heraufgeladen wurde.

Ergebnisse

Erstellt werden eine Tabelle zum Speichern einer Liste von Löschungen, ein Trigger, der die employee_delete-Tabelle beim Löschen von Zeilen aus der Tabelle "employee" ausfüllt, und eine Upload-Prozedur zum Verarbeiten der Löschungen.

Nächste Schritte

Gehen Sie weiter zu [„Lektion 6: Pre-Image leeren und Tabellen löschen“ auf Seite 346](#).

Lektion 6: Pre-Image leeren und Tabellen löschen

In dieser Lektion erstellen Sie einen upload_end-Hook, um das Pre-Image zu leeren und Tabellen zu löschen.

Voraussetzungen

In dieser Lektion wird davon ausgegangen, dass Sie die Rollen und Privilegien haben, die im Abschnitt "Privilegien" am Anfang dieser praktischen Einführung aufgeführt sind: [„Praktische Einführung: Verwenden des skriptgesteuerten Uploads“](#).

In dieser Lektion wird davon ausgegangen, dass Sie bereits alle vorherigen Lektionen abgeschlossen haben. Siehe [„Lektion 1: Erstellen der konsolidierten Datenbank“](#) auf Seite 338.

Kontext und Bemerkungen

In dieser praktischen Einführung wird die Standardeinstellung für die erweiterte dbmlsync-Option LockTables verwendet, sodass die Tabellen während der Synchronisation gesperrt sind. Sie brauchen sich daher nicht darum zu kümmern, dass Zeilen in den Tabellen für Vorgänge zurückbleiben, die nach end_progress eingetreten sind. Durch die Sperre werden solche Vorgänge verhindert.

Aufgabe

- Wählen Sie die mit der entfernten Datenbank verbundene Instanz von Interactive SQL und erstellen Sie einen upload_end-Hook, um die Tabellen employee_preimage und employee_delete nach einem erfolgreichen Upload zu bereinigen.

```
CREATE PROCEDURE sp_hook_dbmlsync_upload_end()  
BEGIN  
    DECLARE val    varchar(256);  
  
    SELECT value  
    INTO val  
    FROM #hook_dict  
    WHERE name = 'upload status';  
  
    IF val = 'committed' THEN  
        DELETE FROM employee_delete;  
        DELETE FROM employee_preimages;  
    END IF;  
END;
```

Ergebnisse

Das Pre-Image und die Löschergebnisse werden aus den Tabellen entfernt, wenn ein Upload erfolgreich verläuft.

Nächste Schritte

Gehen Sie weiter zu [„Lektion 7: Erstellen einer Publikation, eines MobiLink-Benutzers und einer Subskription“](#) auf Seite 347.

Lektion 7: Erstellen einer Publikation, eines MobiLink-Benutzers und einer Subskription

In dieser Lektion erstellen Sie eine Publikation, einen MobiLink-Benutzer und eine Subskription.

Voraussetzungen

In dieser Lektion wird davon ausgegangen, dass Sie die Rollen und Privilegien haben, die im Abschnitt "Privilegien" am Anfang dieser praktischen Einführung aufgeführt sind: [„Praktische Einführung: Verwenden des skriptgesteuerten Uploads“](#).

In dieser Lektion wird davon ausgegangen, dass Sie bereits alle vorherigen Lektionen abgeschlossen haben. Siehe [„Lektion 1: Erstellen der konsolidierten Datenbank“](#) auf Seite 338.

Aufgabe

- Wählen Sie die mit der entfernten Datenbank verbundene Instanz von Interactive SQL und führen Sie die folgenden SQL-Anweisungen aus: Die Publikation mit dem Namen "pub1" verwendet die skriptgesteuerte Upload-Syntax (WITH SCRIPTED UPLOAD). Sie erstellt einen Artikel für die Tabelle employee und registriert die drei gespeicherten Prozeduren, die Sie gerade zur Verwendung im skriptgesteuerten Upload erstellt haben. Sie erstellt außerdem einen MobiLink-Benutzer mit dem Namen "u1" und eine Subskription zwischen "v1" und "pub1". Die erweiterte Option FireTriggers wird deaktiviert, um zu verhindern, dass Trigger in der entfernten Datenbank während des Downloads ausgelöst werden. Dadurch wird vermieden, dass heruntergeladene Änderungen bei der nächsten Synchronisation heraufgeladen werden.

```
CREATE PUBLICATION pub1 WITH SCRIPTED UPLOAD (
  TABLE employee( id, name, salary ) USING (
    PROCEDURE employee_insert FOR UPLOAD INSERT,
    PROCEDURE employee_update FOR UPLOAD UPDATE,
    PROCEDURE employee_delete FOR UPLOAD DELETE
  )
);

CREATE SYNCHRONIZATION USER u1;

CREATE SYNCHRONIZATION SUBSCRIPTION TO pub1 FOR u1
TYPE 'tcpip'
ADDRESS 'host=localhost'
OPTION FireTriggers='off'
SCRIPT VERSION 'default';
```

Ergebnisse

Eine Publikation, ein MobiLink-Benutzer und eine Subskription werden erstellt.

Nächste Schritte

Gehen Sie weiter zu [„Lektion 8: Demonstration des skriptgesteuerten Uploads“](#) auf Seite 348.

Lektion 8: Demonstration des skriptgesteuerten Uploads

In dieser Lektion führen Sie eine SQL-Anweisung und Befehle aus, um einen skriptgesteuerten Upload zu demonstrieren.

Voraussetzungen

In dieser Lektion wird davon ausgegangen, dass Sie die Rollen und Privilegien haben, die im Abschnitt "Privilegien" am Anfang dieser praktischen Einführung aufgeführt sind: [„Praktische Einführung: Verwenden des skriptgesteuerten Uploads“](#).

In dieser Lektion wird davon ausgegangen, dass Sie bereits alle vorherigen Lektionen abgeschlossen haben. Siehe [„Lektion 1: Erstellen der konsolidierten Datenbank“](#) auf Seite 338.

Aufgabe

1. Wählen Sie die mit der entfernten Datenbank verbundene Instanz von Interactive SQL und fügen Sie Daten ein, die mithilfe eines skriptgesteuerten Uploads synchronisiert werden sollen. Führen Sie die folgenden SQL-Anweisungen für die entfernte Datenbank aus:

```
INSERT INTO employee(id, name, salary) VALUES( 7, 'black', 700 );
INSERT INTO employee(id, name, salary) VALUES( 8, 'anderson', 800 );
INSERT INTO employee(id, name, salary) VALUES( 9, 'dilon', 900 );
INSERT INTO employee(id, name, salary) VALUES( 10, 'dwit', 1000 );
INSERT INTO employee(id, name, salary) VALUES( 11, 'dwit', 1100 );
COMMIT;
```

2. Starten Sie den MobiLink-Server an einer Eingabeaufforderung:

```
mlsrv16 -c "DSN=dsn_consol" -o mlserver.mls -v+ -dl -zu+
```

3. Starten Sie eine Synchronisation mit dbmlsync:

```
dbmlsync -c "DSN=dsn_remote" -k -uo -o remote.mlc -v+
```

4. Wählen Sie die mit der entfernten Datenbank verbundene Instanz von Interactive SQL und führen Sie die folgende SQL-Anweisung aus, um zu überprüfen, ob die Einfügungen hochgeladen wurden.

```
select * from employee
```

Ergebnisse

Danach sollten die Werte angezeigt werden, die Sie am Anfang dieser Lektion eingefügt haben.

Nächste Schritte

Gehen Sie weiter zu [„Aufräumen“](#) auf Seite 348.

Aufräumen

Wenn Sie Ihren Computer nach Abschluss der praktischen Einführung bereinigen möchten, führen Sie die folgenden Befehle aus:

```
mlstop -h -w
dbstop -y -c server=consol
dbstop -y -c server=remote

dberase -y consol.db
dberase -y remote.db

del remote.mlc mlserver.mls
del remote.mlc mlserver.mls remote.rid
```

Index

Symbole

#hook_dict, Tabelle

dbmlsync, 205

Info über MobiLink, 205

Skriptgesteuerter Upload, 332

-a, Option

MobiLink SQL Anywhere-Client-Dienstprogramm
(dbmlsync), 112

-ap, Option

MobiLink SQL Anywhere-Client-Dienstprogramm
(dbmlsync), 112

MobiLink-Dienstprogramm für die
Dateiübertragung (mlfiletransfer), 23

-ba, Option

MobiLink SQL Anywhere-Client-Dienstprogramm
(dbmlsync), 112

-bc, Option

MobiLink SQL Anywhere-Client-Dienstprogramm
(dbmlsync), 113

-be, Option

MobiLink SQL Anywhere-Client-Dienstprogramm
(dbmlsync), 113

-bg, Option

MobiLink SQL Anywhere-Client-Dienstprogramm
(dbmlsync), 114

-bk, Option

MobiLink SQL Anywhere-Client-Dienstprogramm
(dbmlsync), 115

-bkr, Option

MobiLink SQL Anywhere-Client-Dienstprogramm
(dbmlsync), 115

-c, Option

MobiLink SQL Anywhere-Client-Dienstprogramm
(dbmlsync), 116

-ci, Option

MobiLink SQL Anywhere-Client-Dienstprogramm
(dbmlsync), 116

-cl, Option

MobiLink SQL Anywhere-Client-Dienstprogramm
(dbmlsync), 117

-cm, Option

MobiLink SQL Anywhere-Client-Dienstprogramm
(dbmlsync), 117

-d, Option

MobiLink SQL Anywhere-Client-Dienstprogramm
(dbmlsync), 118

-dc, Option

MobiLink SQL Anywhere-Client-Dienstprogramm
(dbmlsync), 118

-dl, Option

MobiLink SQL Anywhere-Client-Dienstprogramm
(dbmlsync), 119

-do, Option

MobiLink SQL Anywhere-Client-Dienstprogramm
(dbmlsync), 119

-drs, Option

MobiLink SQL Anywhere-Client-Dienstprogramm
(dbmlsync), 120

-ds, Option

MobiLink SQL Anywhere-Client-Dienstprogramm
(dbmlsync), 120

-dst, Option

MobiLink-Dienstprogramm für Microsoft
ActiveSync-Provider (mlasinst), 20

-e adr

dbmlsync, erweiterte Option, 149
Optionen für, 25

-e bd

dbmlsync, erweiterte Option, 148

-e BufferDownload

dbmlsync, erweiterte Option, 148

-e cd

dbmlsync, erweiterte Option, 151

-e CommunicationAddress

dbmlsync, erweiterte Option, 149
Optionen für, 25

-e CommunicationType

dbmlsync, erweiterte Option, 150

-e ConflictRetries

dbmlsync, erweiterte Option, 150

-e ContinueDownload

dbmlsync, erweiterte Option, 151

-e cr

dbmlsync, erweiterte Option, 150

-e ctp

dbmlsync, erweiterte Option, 150

-e dir

dbmlsync, erweiterte Option, 164

-e DisablePolling

dbmlsync, erweiterte Option, 152

-e DownloadOnly

dbmlsync, erweiterte Option, 153

- e DownloadReadSize
 - dbmlsync, erweiterte Option, 154
- e drs
 - dbmlsync, erweiterte Option, 154
- e ds
 - dbmlsync, erweiterte Option, 153
- e eh
 - dbmlsync, erweiterte Option, 158
- e el
 - dbmlsync, erweiterte Option, 155
- e ErrorLogSendLimit
 - dbmlsync, erweiterte Option, 155
- e FireTriggers
 - dbmlsync, erweiterte Option, 156
- e ft
 - dbmlsync, erweiterte Option, 156
- e HoverRescanThreshold
 - dbmlsync, erweiterte Option, 157
- e hrt
 - dbmlsync, erweiterte Option, 157
- e IgnoreHookErrors
 - dbmlsync, erweiterte Option, 158
- e IgnoreScheduling
 - dbmlsync, erweiterte Option, 158
- e inc
 - dbmlsync, erweiterte Option, 159
- e Increment
 - dbmlsync, erweiterte Option, 159
- e isc
 - dbmlsync, erweiterte Option, 158
- e LockTables
 - dbmlsync, erweiterte Option, 160
- e lt
 - dbmlsync, erweiterte Option, 160
- e MirrorLogDirectory
 - dbmlsync, erweiterte Option, 161
- e mld
 - dbmlsync, erweiterte Option, 161
- e mn
 - dbmlsync, erweiterte Option, 163
- e MobiLinkPwd
 - dbmlsync, erweiterte Option, 162
- e mp
 - dbmlsync, erweiterte Option, 162
- e NewMobiLinkPwd
 - dbmlsync, erweiterte Option, 163
- e NoSyncOnStartup
 - dbmlsync, erweiterte Option, 163
- e nss
 - dbmlsync, erweiterte Option, 163
- e OfflineDirectory
 - dbmlsync, erweiterte Option, 164
- e p
 - dbmlsync, erweiterte Option, 152
- e PollingPeriod
 - dbmlsync, erweiterte Option, 165
- e pp
 - dbmlsync, erweiterte Option, 165
- e sa
 - dbmlsync, erweiterte Option, 169
- e sch
 - dbmlsync, erweiterte Option, 166
- e Schedule
 - dbmlsync, erweiterte Option, 166
- e ScriptVersion
 - dbmlsync, erweiterte Option, 168
- e SendDownloadACK
 - erweiterte dbmlsync-Option, 169
- e SendTriggers
 - dbmlsync, erweiterte Option, 169
- e st
 - dbmlsync, erweiterte Option, 169
- e sv
 - dbmlsync, erweiterte Option, 168
- e TableOrder
 - dbmlsync, erweiterte Option, 171
- e TableOrderChecking
 - dbmlsync, erweiterte Option, 172
- e toc
 - dbmlsync, erweiterte Option, 172
- e tor
 - dbmlsync, erweiterte Option, 171
- e uo
 - dbmlsync, erweiterte Option, 173
- e UploadOnly
 - dbmlsync, erweiterte Option, 173
- e v
 - dbmlsync, erweiterte Option, 174
- e Verbose
 - dbmlsync, erweiterte Option, 174
- e VerboseHooks
 - dbmlsync, erweiterte Option, 175
- e VerboseMin
 - dbmlsync, erweiterte Option, 176
- e VerboseOptions
 - dbmlsync, erweiterte Option, 176

-
- e VerboseRowCounts
dbmlsync, erweiterte Option,177
 - e VerboseRowValues
dbmlsync, erweiterte Option,178
 - e VerboseUpload
dbmlsync, erweiterte Option,179
 - e vm
dbmlsync, erweiterte Option,176
 - e vn
dbmlsync, erweiterte Option,177
 - e vo
dbmlsync, erweiterte Option,176
 - e vr
dbmlsync, erweiterte Option,178
 - e vs
dbmlsync, erweiterte Option,175
 - e vu
dbmlsync, erweiterte Option,179
 - e, Option
MobiLink SQL Anywhere-Client-Dienstprogramm
(dbmlsync),121
 - eh, Option
MobiLink SQL Anywhere-Client-Dienstprogramm
(dbmlsync),122
 - ek, Option
MobiLink SQL Anywhere-Client-Dienstprogramm
(dbmlsync),122
 - ep, Option
MobiLink SQL Anywhere-Client-Dienstprogramm
(dbmlsync),123
 - eu, Option
MobiLink SQL Anywhere-Client-Dienstprogramm
(dbmlsync),123
 - f, Option
MobiLink-Dienstprogramm für die
Dateiübertragung (mlfiletransfer),23
 - g, Option
MobiLink-Dienstprogramm für die
Dateiübertragung (mlfiletransfer),23
 - is, Option
MobiLink SQL Anywhere-Client-Dienstprogramm
(dbmlsync),123
 - k, Option
MobiLink SQL Anywhere-Client-Dienstprogramm
(dbmlsync) (nicht mehr empfohlen),124
MobiLink-Dienstprogramm für Microsoft
ActiveSync-Provider (mlasinst),20
 - l, Option
MobiLink SQL Anywhere-Client-Dienstprogramm
(dbmlsync),124
 - mn, Option
MobiLink SQL Anywhere-Client-Dienstprogramm
(dbmlsync),124
 - mp, Option
MobiLink SQL Anywhere-Client-Dienstprogramm
(dbmlsync),125
 - n, Option
MobiLink SQL Anywhere-Client-Dienstprogramm
(dbmlsync),125
MobiLink-Dienstprogramm für Microsoft
ActiveSync-Provider (mlasinst),20
 - o, Option
MobiLink SQL Anywhere-Client-Dienstprogramm
(dbmlsync),126
 - os, Option
MobiLink SQL Anywhere-Client-Dienstprogramm
(dbmlsync),127
 - ot, Option
MobiLink SQL Anywhere-Client-Dienstprogramm
(dbmlsync),127
 - p, Option
MobiLink SQL Anywhere-Client-Dienstprogramm
(dbmlsync),127
MobiLink-Dienstprogramm für die
Dateiübertragung (mlfiletransfer),23
 - pc+, Option
MobiLink SQL Anywhere-Client-Dienstprogramm
(dbmlsync),128
 - pd, Option
MobiLink SQL Anywhere-Client-Dienstprogramm
(dbmlsync),129
 - pi, Option
MobiLink SQL Anywhere-Client-Dienstprogramm
(dbmlsync),129
 - po, Option
MobiLink SQL Anywhere-Client-Dienstprogramm
(dbmlsync),130
 - pp, Option
MobiLink SQL Anywhere-Client-Dienstprogramm
(dbmlsync),130
 - q, Option
MobiLink SQL Anywhere-Client-Dienstprogramm
(dbmlsync),131
 - qc, Option
MobiLink SQL Anywhere-Client-Dienstprogramm
(dbmlsync),131

- qi, Option
 - MobiLink SQL Anywhere-Client-Dienstprogramm (dbmlsync),131
- r, Option
 - MobiLink SQL Anywhere-Client-Dienstprogramm (dbmlsync),132
 - MobiLink-Dienstprogramm für die Dateiübertragung (mlfiletransfer),23
- ra, Option
 - MobiLink SQL Anywhere-Client-Dienstprogramm (dbmlsync),132
- rb, Option
 - MobiLink SQL Anywhere-Client-Dienstprogramm (dbmlsync),132
- s, Option
 - MobiLink SQL Anywhere-Client-Dienstprogramm (dbmlsync),133
- sc, Option
 - MobiLink SQL Anywhere-Client-Dienstprogramm (dbmlsync),134
- sm, Option
 - MobiLink SQL Anywhere-Client-Dienstprogramm (dbmlsync),134
- sp, Option
 - MobiLink SQL Anywhere-Client-Dienstprogramm (dbmlsync),135
- src, Option
 - MobiLink-Dienstprogramm für Microsoft ActiveSync-Provider (mlasinst),20
- ts, Option
 - MobiLink SQL Anywhere-Clientdienstprogramm (dbmlsync),135
- tu, Option
 - MobiLink SQL Anywhere-Client-Dienstprogramm (dbmlsync),136
- u, Option
 - MobiLink SQL Anywhere-Client-Dienstprogramm (dbmlsync),138
 - MobiLink-Dienstprogramm für die Dateiübertragung (mlfiletransfer),23
 - MobiLink-Dienstprogramm für Microsoft ActiveSync-Provider (mlasinst),20
- ud, Option
 - MobiLink SQL Anywhere-Client-Dienstprogramm (dbmlsync),139
- ui, Option
 - MobiLink SQL Anywhere-Client-Dienstprogramm (dbmlsync),139
- uo, Option
 - MobiLink SQL Anywhere-Client-Dienstprogramm (dbmlsync),139
- urc, Option
 - MobiLink SQL Anywhere-Client-Dienstprogramm (dbmlsync),140
- ux, Option
 - MobiLink SQL Anywhere-Client-Dienstprogramm (dbmlsync),140
- v+, Option
 - MobiLink SQL Anywhere-Client-Dienstprogramm (dbmlsync),141
- v, Option
 - MobiLink SQL Anywhere-Client-Dienstprogramm (dbmlsync),141
 - MobiLink-Dienstprogramm für die Dateiübertragung (mlfiletransfer),23
 - MobiLink-Dienstprogramm für Microsoft ActiveSync-Provider (mlasinst),20
- vc, Option
 - MobiLink SQL Anywhere-Client-Dienstprogramm (dbmlsync),141
- vn, Option
 - MobiLink SQL Anywhere-Client-Dienstprogramm (dbmlsync),141
- vo, Option
 - MobiLink SQL Anywhere-Client-Dienstprogramm (dbmlsync),141
- vp, Option
 - MobiLink SQL Anywhere-Client-Dienstprogramm (dbmlsync),141
- vr, Option
 - MobiLink SQL Anywhere-Client-Dienstprogramm (dbmlsync),141
- vs, Option
 - MobiLink SQL Anywhere-Client-Dienstprogramm (dbmlsync),141
- vu, Option
 - MobiLink SQL Anywhere-Client-Dienstprogramm (dbmlsync),141
- wc, Option
 - MobiLink SQL Anywhere-Client-Dienstprogramm (dbmlsync),142
- x, Option
 - MobiLink SQL Anywhere-Client-Dienstprogramm (dbmlsync),143
 - MobiLink-Dienstprogramm für die Dateiübertragung (mlfiletransfer),23

.NET

MobiLink-Benutzerauthentifizierung,16

@data-Option

MobiLink SQL Anywhere-Client-Dienstprogramm
(dbmlsync),111

A

a_dbtools_info, Struktur

Initialisierung,318

a_sync_db, Struktur

Einführung,318

Initialisieren,318

a_syncpub, Struktur

Einführung,318

Abfolgeplanung

erweiterte Option im MobiLink SQL Anywhere -
Client-Dienstprogramm (dbmlsync),166

in MobiLink mit sp_hook_dbmlsync_delay,218

in MobiLink mit sp_hook_dbmlsync_end,232

MobiLink SQL Anywhere-Clients,100

Abruf

dbmlsync, Logscan-Abruf,152

adr, erweiterte dbmlsync-Option

Info,149

Optionen für,25

Ändern

Artikel für MobiLink SQL Anywhere-Clients,79

MobiLink-Publikationen für SQL Anywhere-
Clients,79

Subskriptionen für SQL Anywhere-Clients,87

Ändern von MobiLink-Subskriptionen

SQL Anywhere-Clients,87

Ändern, Kennwörter

MobiLink,9

Ändern, vorhandene Publikationen

MobiLink SQL Anywhere-Clients,79

Angepasste Benutzerauthentifizierung

MobiLink,15

MobiLink-Clients,15

Anpassen

Synchronisationsprozess von SQL Anywhere-
Clients,202

Anpassen der dbmlsync-Synchronisation

MobiLink SQL Anywhere-Clients,102

Anpassen von Client-Synchronisationsprozessen

SQL Anywhere-Clients,202

Anweisungen

MobiLink,181

args, Option

MobiLink-Dienstprogramm für Microsoft
ActiveSync-Provider (mlasinst),20

Artikel

aus MobiLink SQL Anywhere-Clients entfernen,79

für MobiLink SQL Anywhere-Clients ändern,79

für MobiLink SQL Anywhere-Clients erstellen,74

für MobiLink SQL Anywhere-Clients

hinzufügen,79

MobiLink-Synchronisationssubskriptionen,86

Artikel erstellen, Assistent

in MobiLink verwenden,80

Assistent zum Erstellen von Benutzern

MobiLink-Plug-In,6

Assistent zum Hinzufügen von Benutzern

MobiLink-Plug-In,6

Ausführlichkeitsstufe

Einstellung des MobiLink SQL Anywhere-Client-
Dienstprogramms (dbmlsync),141

Ausführlichkeitsstufe, Option

MobiLink SQL Anywhere-Client-Dienstprogramm
(dbmlsync),141

Auswählen

UltraLite, Netzwerkprotokolle,30

authenticate_user

Authentifizierungsprozess,13

Info,15

vordefinierte Skripten verwenden,17

Authentifizieren, MobiLink-Benutzer

Info,4

Authentifizierung

Authentifizierungsprozess in MobiLink,13

MobiLink, bei externen Servern,17

MobiLink-Benutzer,4

Authentifizierungsprozess

MobiLink,13

Automatisches Einwählen

Verbindungsoption des MobiLink-Clients,49

B

Beendigungscodes

dbmlsync [sp_hook_dbmlsync_abort],208

dbmlsync

[sp_hook_dbmlsync_process_exit_code],247

Befehlszeile

dbmlsync starten,106

- Befehlszeilen-Dienstprogramme
 - MobiLink-Clients,19
 - Syntax für MobiLink SQL Anywhere-Client-Dienstprogramm (dbmlsync) ,106
 - Syntax für MobiLink-Dienstprogramm für die Dateiübertragung (mlfiletransfer),22
 - Syntax für MobiLink-Dienstprogramm für Microsoft ActiveSync-Provider (mlasinst),20
- Behandlung von Fehlern
 - MobiLink dbmlsync-Clients,207
- Benannte Parameter
 - remote_id,11
 - username,11
- Benutzer
 - in MobiLink erstellen,5
 - Info über MobiLink,4
 - MobiLink-Erstellung in SQL Anywhere-Clients,83
- Benutzerauthentifizierung
 - .NET-Synchronisationslogik,16
 - in MobiLink ein Verfahren auswählen,12
 - Java-Synchronisationslogik,16
 - Kennwörter,8
 - MobiLink, angepasstes Verfahren,15
 - MobiLink, neue Benutzer,7
 - MobiLink-Architektur,13
 - MobiLink-Kennwörter,6
 - MobiLink-Kennwörter ändern,9
 - MobiLink-Sicherheit,4
- Benutzerauthentifizierungsarchitektur
 - MobiLink,13
- Benutzerdefinierte Header
 - Verbindungsoption des MobiLink-Clients,37
- Benutzername
 - in MobiLink erstellen,5
 - Info über MobiLink,4
- Benutzernamen
 - in MobiLink erstellen,5
 - Info über MobiLink,4
 - MobiLink, in Skripten verwenden,11
- buffer_size-Protokolloption
 - Verbindungsoption des MobiLink-Clients,30
- BufferDownload, erweiterte Option
 - dbmlsync,144
 - Info,148
- C**
- cac, Option
 - MobiLink-Clients,47
- cac-Authentifizierung
 - Verbindungsoption des MobiLink-Clients,47
- CancelSync-Methode
 - DbmlsyncClient-Klasse [Dbmlsync .NET-API],294
 - DbmlsyncClient-Klasse [Dbmlsync C++-API],270
- cd, erweiterte dbmlsync-Option
 - Info,151
- CERTIFICATE, Option
 - Verbindungsoption des MobiLink-Clients,47
- certificate_company-Protokolloption
 - Verbindungsoption des MobiLink-Clients,31
- certificate_name-Protokolloption
 - Verbindungsoption des MobiLink-Clients,33
- certificate_unit-Protokolloption
 - Verbindungsoption des MobiLink-Clients,34
- class, Option
 - MobiLink-Dienstprogramm für Microsoft ActiveSync-Provider (mlasinst),20
- Client, Datenbanken
 - MobiLink, dbmlsync-Optionen,106
- Client, Netzwerkprotokolloptionen
 - MobiLink,25
- Client-Hook, Prozeduren
 - MobiLink SQL Anywhere-Clients,202
- client_port-Protokolloption
 - Verbindungsoption des MobiLink-Clients,36
- Clients
 - MobiLink SQL Anywhere Client (dbmlsync),106
 - SQL Anywhere als MobiLink,1
 - SQL Anywhere MobiLink-Clients,69
 - UltraLite-Anwendungen als MobiLink,2
- COMMIT-Anweisung
 - Hook-Prozeduren,204
- CommunicationAddress, erweiterte dbmlsync-Option
 - Info,149
 - Optionen für,25
- CommunicationAddress, erweiterte Option
 - dbmlsync,144
- CommunicationType, erweiterte dbmlsync-Option
 - Info,150
- CommunicationType, erweiterte Option
 - dbmlsync,144
- compression
 - Verbindungsoption des MobiLink-Clients,36
- compression-Protokolloption
 - UltraLite-Deploymentanforderungen,30

Verbindungsoption des MobiLink-Clients,36
 ConflictRetries, erweiterte dbmsync-Option
 Info,150
 Parallelität während der Synchronisation,94
 ConflictRetries, erweiterte Option
 dbmsync,144
 Connect-Methode
 DbmsyncClient-Klasse [Dbmsync .NET-API],296
 DbmsyncClient-Klasse [Dbmsync C++-API],271
 ContinueDownload, erweiterte dbmsync-Option
 Info,151
 ContinueDownload, erweiterte Option
 dbmsync,144
 cr, erweiterte dbmsync-Option
 Info,150
 CREATE PUBLICATION-Anweisung
 SQL Anywhere-Datenbanksyntax,74
 CREATE SYNCHRONIZATION SUBSCRIPTION-Anweisung
 Microsoft ActiveSync für MobiLink SQL
 Anywhere-Clients,96
 CREATE SYNCHRONIZATION USER-Anweisung
 Microsoft ActiveSync für MobiLink SQL
 Anywhere-Clients,96
 ctp, erweiterte dbmsync-Option
 Info,150
 custom_header-Protokolloption
 Verbindungsoption des MobiLink-Clients,37

D

Dateibasierte Downloads
 dbmsync, Option -bc,113
 dbmsync, Option -be,113
 dbmsync, Option -bg,114
 Dateiübertragungen
 MobiLink-Dienstprogramm für die
 Dateiübertragung (mlfiletransfer),22
 Datenbanken
 entfernte MobiLink-Datenbanken,1
 Datenbanktools, Schnittstelle
 dbmsync,317
 für dbmsync einrichten,318
 Datenbanktools, Schnittstelle für dbmsync
 Info,317
 Datenstromparameter
 MobiLink-Clients,25

Dauerhafte Verbindungen
 MobiLink dbmsync, Option -pc,128
 db dbmsync, erweiterte Option
 Info,148
 dbmsync
 Optionen,106
 Dbmsync .NET, API
 DBSC_Event-Struktur,316
 Dbmsync .NET-API
 DbmsyncClient-Klasse,293
 DBSC_CancelRet-Enumeration,308
 DBSC_ErrorInfo-Struktur,315
 DBSC_ErrorType-Enumeration,308
 DBSC_EventType-Enumeration,311
 DBSC_GetEventRet-Enumeration,313
 DBSC_ShutdownType-Enumeration,314
 DBSC_StartType-Enumeration,314
 Dbmsync .NET-API-Referenz
 iAnywhere.MobiLink.Client-Namespace,291
 Dbmsync C++, API
 DBSC_Event-Struktur,290
 Dbmsync C++-API
 DbmsyncClient-Klasse,268
 DBSC_CancelRet-Enumeration,282
 DBSC_ErrorInfo-Struktur,289
 DBSC_ErrorType-Enumeration,283
 DBSC_EventType-Enumeration,286
 DBSC_GetEventRet-Enumeration,288
 DBSC_ShutdownType-Enumeration,288
 DBSC_StartType-Enumeration,289
 Dbmsync C++-API-Referenz
 dbmsynccli.h-Headerdatei,267
 Dbmsync, API
 Einführung,103
 dbmsync, Dienstprogramm
 #hook_dict-Tabelle,205
 alphabetische Liste der Optionen,106
 DBTools-Schnittstelle,317
 Ereignis-Hook,202
 erweiterte Optionen,143
 Fehlerbehandlung von Ereignis-Hooks,207
 Hooks,202
 Info,106
 Kennwörter,8
 Kennwörter ändern,9
 Mac OS X,104
 mit der entfernten Datenbank verbinden,116
 MobiLink-Synchronisation anpassen,202

- Offsets für den Verarbeitungsfortschritt,73
- Optionen,106
- Parallelität,94
- Privilegien,89
- sp_hook_dbmlsync_abort, Hook,208
- sp_hook_dbmlsync_all_error,211
- sp_hook_dbmlsync_begin,214
- sp_hook_dbmlsync_communication_error,215
- sp_hook_dbmlsync_delay,218
- sp_hook_dbmlsync_download_begin,221
- sp_hook_dbmlsync_download_end,222
- sp_hook_dbmlsync_download_log_ri_violation,224
- sp_hook_dbmlsync_download_ri_violation,226
- sp_hook_dbmlsync_download_table_begin,229
- sp_hook_dbmlsync_download_table_end,230
- sp_hook_dbmlsync_end,232
- sp_hook_dbmlsync_log_rescan,235
- sp_hook_dbmlsync_logscan_begin,237
- sp_hook_dbmlsync_logscan_end,239
- sp_hook_dbmlsync_misc_error,240
- sp_hook_dbmlsync_ml_connect_failed,244
- sp_hook_dbmlsync_process_exit_code,247
- sp_hook_dbmlsync_schema_upgrade,249
- sp_hook_dbmlsync_set_extended_options,252
- sp_hook_dbmlsync_set_ml_connect_info,253
- sp_hook_dbmlsync_set_upload_end_progress,255
- sp_hook_dbmlsync_sql_error,257
- sp_hook_dbmlsync_upload_begin,260
- sp_hook_dbmlsync_upload_end,261
- sp_hook_dbmlsync_validate_download_file,264
- Syntax,106
- verwenden,88
- dbmlsync, erweiterte Optionen
 - Info,143
 - Liste,144
 - verwenden,92
- dbmlsync, Netzwerkprotokolloptionen
 - Info,93
- Dbmlsync-API
 - Architektur,103
 - Schnittstellen,103
- dbmlsync-Client, Ereignis-Hooks
 - Einführung,102
- dbmlsync-Dienstprogramm
 - Microsoft ActiveSync für MobiLink SQL Anywhere-Clients,95
 - Programmierschnittstelle,103
 - Synchronisation von einer Anwendung aus initialisieren,95
 - Transaktionslogs,93
- dbmlsync-Fehler
 - Behandlung,207
- dbmlsync-Integrationskomponente (*Siehe* Dbmlsync-API)
- dbmlsync-Meldungslog
 - Info,103
- dbmlsynccli.h-Headerdatei
 - Dbmlsync C++-API-Referenz,267
- DbmlsyncClient-Klasse [Dbmlsync .NET-API]
 - Beschreibung,293
 - CancelSync-Methode,294
 - Connect-Methode,296
 - Disconnect-Methode,297
 - Fini-Methode,298
 - GetErrorInfo-Methode,298
 - GetEvent-Methode,299
 - GetProperty-Methode,300
 - Init-Methode,301
 - InstantiateClient-Methode,301
 - Ping-Methode,302
 - SetProperty-Methode,302
 - ShutdownServer-Methode,304
 - StartServer-Methode,305
 - Sync-Methode,306
 - WaitForServerShutdown-Methode,307
- DbmlsyncClient-Klasse [Dbmlsync C++-API]
 - Beschreibung,268
 - CancelSync-Methode,270
 - Connect-Methode,271
 - Disconnect-Methode,273
 - Fini-Methode,273
 - FreeEventInfo-Methode,274
 - GetErrorInfo-Methode,274
 - GetEvent-Methode,275
 - GetProperty-Methode,276
 - Init-Methode,276
 - InstantiateClient-Methode,277
 - Ping-Methode,277
 - SetProperty-Methode,278
 - ShutdownServer-Methode,279
 - StartServer-Methode,280
 - Sync-Methode,281
 - WaitForServerShutdown-Methode,282
- DBSC_CancelRet-Enumeration [Dbmlsync .NET-API]

Beschreibung,308
 DBSC_CancelRet-Enumeration [Dbmlsync C++-API]
 Beschreibung,282
 DBSC_ErrorInfo-Struktur [Dbmlsync .NET-API]
 Beschreibung,315
 DBSC_ErrorInfo-Struktur [Dbmlsync C++-API]
 Beschreibung,289
 DBSC_ErrorType-Enumeration [Dbmlsync .NET-API]
 Beschreibung,308
 DBSC_ErrorType-Enumeration [Dbmlsync C++-API]
 Beschreibung,283
 DBSC_Event-Struktur [Dbmlsync .NET-API]
 Beschreibung,316
 DBSC_Event-Struktur [Dbmlsync C++-API]
 Beschreibung,290
 DBSC_EventType-Enumeration [Dbmlsync .NET-API]
 Beschreibung,311
 DBSC_EventType-Enumeration [Dbmlsync C++-API]
 Beschreibung,286
 DBSC_GetEventRet-Enumeration [Dbmlsync .NET-API]
 Beschreibung,313
 DBSC_GetEventRet-Enumeration [Dbmlsync C++-API]
 Beschreibung,288
 DBSC_ShutdownType-Enumeration [Dbmlsync .NET-API]
 Beschreibung,314
 DBSC_ShutdownType-Enumeration [Dbmlsync C++-API]
 Beschreibung,288
 DBSC_StartType-Enumeration [Dbmlsync .NET-API]
 Beschreibung,314
 DBSC_StartType-Enumeration [Dbmlsync C++-API]
 Beschreibung,289
 DBSynchronizeLog-Funktion
 Einführung,318
 DBTools, Schnittstelle
 dbmlsync,317
 für dbmlsync einrichten,318
 DBTools-Schnittstelle
 SQL Anywhere-Clients synchronisieren,95
 DBTools-Schnittstelle für dbmlsync
 Info,317
 dbtools.h
 SQL Anywhere-Clients synchronisieren,95
 DBToolsFinis-Funktion
 anwenden,321
 DBToolsInit-Funktion
 dbtools starten,318
 DDL
 entfernte MobiLink-Datenbanken,62
 default_internet
 Einstellung der network_name-Protokolloption,49
 default_work
 Einstellung der network_name-Protokolloption,49
 Deployment durchführen
 Fehlerbehandlung bei MobiLink-Deployment von SQL Anywhere-Clients,72
 MobiLink SQL Anywhere-Clients,70
 Deployment entfernter Datenbanken durchführen
 MobiLink SQL Anywhere-Clients,70
 Dialogfreier Modus
 MobiLink SQL Anywhere-Client-Dienstprogramm (dbmlsync),131
 Dienstprogramm dbmlsync
 mit dem MobiLink-Server verbinden,149
 Dienstprogramm zur Installation des Microsoft ActiveSync-Providers (mlasinst)
 Syntax,20
 Dienstprogramme
 MobiLink, Liste der Dienstprogramme,19
 Syntax für MobiLink für Microsoft ActiveSync-Provider (mlasinst),20
 Syntax für MobiLink SQL Anywhere-Client (dbmlsync),106
 Syntax für MobiLink-Dienstprogramm für die Dateiübertragung (mlfiletransfer),22
 dir, erweiterte dbmlsync-Option
 Info,164
 DisablePolling, erweiterte dbmlsync-Option
 Info,152
 DisablePolling, erweiterte Option
 dbmlsync,144
 Disconnect-Methode
 DbmlsyncClient-Klasse [Dbmlsync .NET-API],297
 DbmlsyncClient-Klasse [Dbmlsync C++-API],273
 dllapi.h
 DBTools-Schnittstelle für dbmlsync,321
 Download von Zeilen
 MobiLink RI-Verletzungen klären,224

- Download, fortsetzen
 - dbmsync, Option -dc,118
- Downloadbestätigung senden
 - dbmsync, erweiterte Option,169
- DownloadOnly, erweiterte dbmsync-Option
 - Info,153
- DownloadOnly, erweiterte Option
 - dbmsync,144
- DownloadReadSize, erweiterte dbmsync-Option
 - Info,154
- DownloadReadSize, erweiterte Option
 - dbmsync,144
- DROP PUBLICATION-Anweisung
 - MobiLink verwenden,82
- DROP SYNCHRONIZATION SUBSCRIPTION-Anweisung
 - verwenden,88
- drs, erweiterte dbmsync-Option
 - Info,154
- ds, erweiterte dbmsync-Option
 - Info,153

E

- e2ee_public_key-Protokolloption
 - Verbindungsoption des MobiLink-Clients,38
- eh, erweiterte dbmsync-Option
 - Info,158
- Einrichten
 - MobiLink, skriptgesteuerte Uploads,325
 - MobiLink. DBTools-Schnittstelle für dbmsync,318
- Einwahl
 - dbmsync-Verbindung,149
 - MobiLink-Client, Protokolloptionen,25
- el, erweiterte dbmsync-Option
 - Info,155
- Ende-zu-Ende-Verschlüsselung, öffentlicher Schlüssel
 - Verbindungsoption des MobiLink-Clients,38
- Entfernen
 - Artikel aus MobiLink SQL Anywhere-Clients,79
- Entfernte Datenbanken
 - aus Archiven wiederherstellen,132
 - Dateien übertragen,22
 - Deployment von SQL Anywhere-Clients durchführen,70
 - MobiLink SQL Anywhere-Clients,69
 - SQL Anywhere-Clients erstellen,69

- Entfernte Datenbanken erstellen
 - SQL Anywhere-Clients,69
- Entfernte DBA-Privilegien
 - MobiLink-Synchronisation von SQL Anywhere-Clients,89
- Entfernte IDs
 - in SQL Anywhere-Datenbanken einstellen,72
 - Info,9
- Entfernte MobiLink-Datenbanken
 - Schemaänderungen,62
- Entfernte SQL Anywhere-Datenbanken
 - Info zu MobiLink,69
- Ereignis-Hook
 - Info,202
 - sp_hook_dbmsync_abort,208
 - sp_hook_dbmsync_begin,214,221
 - sp_hook_dbmsync_delay,218
 - sp_hook_dbmsync_download_begin,221
 - sp_hook_dbmsync_download_end,222
 - sp_hook_dbmsync_download_log_ri_violation,224
 - sp_hook_dbmsync_download_ri_violation,226
 - sp_hook_dbmsync_download_table_begin,229
 - sp_hook_dbmsync_download_table_end,230
 - sp_hook_dbmsync_end,232
 - sp_hook_dbmsync_log_rescan,235
 - sp_hook_dbmsync_logscan_begin,237
 - sp_hook_dbmsync_logscan_end,239
 - sp_hook_dbmsync_set_extended_options,252
 - sp_hook_dbmsync_upload_end,261
 - sp_hook_dbmsync_validate_download_file,264
- Ereignis-Hook-Sequenz
 - SQL Anywhere-Clients,203
- Ereignis-Hooks
 - #hook_dict-Tabelle,205
 - Ereignisargumente,205
 - Fehler ignorieren,208
 - Fehlerbehandlung,207
 - Festschreiben nicht zulässig,204
 - Hook-Sequenz,203
 - Prozedureigentümer,204
 - schwere Fehler,207
 - sp_hook_dbmsync_all_error,211
 - sp_hook_dbmsync_communication_error,215
 - sp_hook_dbmsync_misc_error,240
 - sp_hook_dbmsync_ml_connect_failed,244
 - sp_hook_dbmsync_process_exit_code,247
 - sp_hook_dbmsync_schema_upgrade,249

- sp_hook_dbmlsync_set_ml_connect_info,253
- sp_hook_dbmlsync_set_upload_end_progress,255
- sp_hook_dbmlsync_sql_error,257
- sp_hook_dbmlsync_upload_begin,260
- Synchronisationsprozess von SQL Anywhere-Clients anpassen,202
- Verbindungen,207
- verwenden,204
- Zurücksetzen nicht zulässig,204
- Ereignis-Hooks für SQL Anywhere-Clients
 - Info,202
- Ereignisargumente
 - SQL Anywhere-Clients,205
- ErrorLogSendLimit, erweiterte dbmlsync-Option
 - Info,155
- ErrorLogSendLimit, erweiterte Option
 - dbmlsync,144
- Erste Orientierung
 - SyncConsole,104
- Erste Synchronisation erfolgreich
 - dbmlsync,73
- Erstellen
 - Artikel für MobiLink SQL Anywhere-Clients,74
 - entfernte SQL Anywhere-Datenbanken,69
 - MobiLink-Benutzer,5
 - MobiLink-Benutzer in SQL Anywhere-Clients,83
 - Publikationen für MobiLink SQL Anywhere-Clients,74
 - Publikationen mit ganzen Tabellen für MobiLink SQL-Clients,75
 - Publikationen mit spaltenweiser Verteilung für MobiLink SQL Anywhere-Clients,76
 - Publikationen mit zeilenweiser Verteilung für MobiLink SQL Anywhere-Clients,77
- Erstellen entfernter IDs
 - SQL Anywhere-Datenbanken,72
- Erstellen und Registrieren von MobiLink-Benutzern
 - Info,5
- Erstellen von MobiLink-Benutzern
 - Info zu SQL Anywhere-Clients,83
- Erstellen von MobiLink-Benutzern in der entfernten Datenbank
 - Info,83
- Erstellen von MobiLink-Benutzern, Assistent
 - verwenden,83
- Erstellen von Publikationen für skriptgesteuerte Uploads
 - Info,337
- Erstellen, MobiLink-Benutzer
 - Info,5
- Erweiterte Optionen
 - dbmlsync,143
 - für SQL Anywhere-Clients konfigurieren,84,85
 - Prioritätenfolge für SQL Anywhere-Clients,143
- Erweiterte Optionen, Liste
 - dbmlsync,144
- Externe Authentifizierer, Eigenschaften
 - MobiLink,19
- Externe Server
 - Authentifizierung bei MobiLink-Anwendungen,17
- F**
- Failover
 - MobiLink SQL Anywhere-Clients mit
 - sp_hook_dbmlsync_ml_connect_failed,244
- Fehler
 - MobiLink dbmlsync-Clients,207
- Fehlerbehandlung
 - entfernte Datenbank aus einer Sicherung
 - wiederherstellen,132
 - MobiLink dbmlsync-Log,103
 - MobiLink-Deployment von SQL Anywhere-Clients,72
- Fehlerbehandlung und Warnungen in Hook-Prozeduren
 - MobiLink, dbmlsync-Clients,207
- Fehlersuche
 - MobiLink dbmlsync-Log,103
- Festlegen, Netzwerkprotokoll für Clients
 - MobiLink,2
- Fini-Methode
 - DbmlsyncClient-Klasse [Dbmlsync .NET-API],298
 - DbmlsyncClient-Klasse [Dbmlsync C++-API],273
- FIPS
 - Verbindungsoption des MobiLink-Clients,39
- FIPS-Protokolloption
 - Verbindungsoption des MobiLink-Clients,39
- FireTriggers, erweiterte dbmlsync-Option
 - Info,156
- FireTriggers, erweiterte Option
 - dbmlsync,144
- Folge von Tabellen
 - dbmlsync, erweiterte Optionen,171
- Fortschritt

Skriptgesteuerter Upload,332
 FreeEventInfo-Methode
 DbmlsyncClient-Klasse [Dbmlsync C++-API],274
 ft, erweiterte dbmlsync-Option
 Info,156

G

Gespeicherte Prozeduren
 MobiLink dbmlsync-Ereignis-Hooks,202
 MobiLink, Clientprozeduren,202
 sp_hook_dbmlsync_abort, Syntax,208
 sp_hook_dbmlsync_all_error, Syntax,211
 sp_hook_dbmlsync_begin, Syntax,214
 sp_hook_dbmlsync_communication_error,
 Syntax,215
 sp_hook_dbmlsync_delay, Syntax,218
 sp_hook_dbmlsync_download_begin, Syntax,221
 sp_hook_dbmlsync_download_end, Syntax,222
 sp_hook_dbmlsync_download_log_ri_violation,
 224
 sp_hook_dbmlsync_download_ri_violation,226
 sp_hook_dbmlsync_download_table_begin,
 Syntax,229
 sp_hook_dbmlsync_download_table_end,
 Syntax,230
 sp_hook_dbmlsync_end, Syntax,232
 sp_hook_dbmlsync_log_rescan, Syntax,235
 sp_hook_dbmlsync_logscan_begin, Syntax,237
 sp_hook_dbmlsync_logscan_end, Syntax,239
 sp_hook_dbmlsync_misc_error, Syntax,240
 sp_hook_dbmlsync_ml_connect_failed,
 Syntax,244
 sp_hook_dbmlsync_process_exit_code, Syntax,247
 sp_hook_dbmlsync_schema_upgrade, Syntax,249
 sp_hook_dbmlsync_set_extended_options,
 Syntax,252
 sp_hook_dbmlsync_set_ml_connect_info,
 Syntax,253
 sp_hook_dbmlsync_set_upload_end_progress,
 Syntax,255
 sp_hook_dbmlsync_sql_error, Syntax,257
 sp_hook_dbmlsync_upload_begin, Syntax,260
 sp_hook_dbmlsync_upload_end, Syntax,261
 sp_hook_dbmlsync_validate_download_file,
 Syntax,264
 GetErrorInfo-Methode

DbmlsyncClient-Klasse [Dbmlsync .NET-
 API],298
 DbmlsyncClient-Klasse [Dbmlsync C++-API],274
 GetEvent-Methode
 DbmlsyncClient-Klasse [Dbmlsync .NET-
 API],299
 DbmlsyncClient-Klasse [Dbmlsync C++-API],275
 GetProperty-Methode
 DbmlsyncClient-Klasse [Dbmlsync .NET-
 API],300
 DbmlsyncClient-Klasse [Dbmlsync C++-API],276

H

Herunterfahren
 dbmlsync, automatisch,131
 Hinzufügen
 Artikel für MobiLink SQL Anywhere-Clients,79
 MobiLink-Benutzer zu einem SQL Anywhere-
 Client,83
 MobiLink-Benutzer zur konsolidierten
 Datenbank,5
 Spalten zu entfernten MobiLink-Datenbanken,65
 Tabellen zu entfernten MobiLink SQL Anywhere-
 Datenbanken,64
 Hook
 Fehler ignorieren,158
 Info zu dbmlsync-Hook-Prozeduren,202
 sp_hook_dbmlsync_abort,208
 sp_hook_dbmlsync_begin,214
 sp_hook_dbmlsync_delay,218
 sp_hook_dbmlsync_download_begin,221
 sp_hook_dbmlsync_download_end,222
 sp_hook_dbmlsync_download_log_ri_violation,
 224
 sp_hook_dbmlsync_download_ri_violation,226
 sp_hook_dbmlsync_download_table_begin,229
 sp_hook_dbmlsync_download_table_end,230
 sp_hook_dbmlsync_end,232
 sp_hook_dbmlsync_log_rescan,235
 sp_hook_dbmlsync_logscan_begin,237
 sp_hook_dbmlsync_logscan_end,239
 sp_hook_dbmlsync_schema_upgrade,249
 sp_hook_dbmlsync_set_extended_options,252
 sp_hook_dbmlsync_upload_begin,260
 sp_hook_dbmlsync_upload_end,261
 sp_hook_dbmlsync_validate_download_file,264
 Synchronisation, Ereignis-Hook,202

Hook-Prozedur, Eigentümer
SQL Anywhere-Clients,204

Hooks

Fehlerbehandlung,207
sp_hook_dbmlsync_all_error,211
sp_hook_dbmlsync_communication_error,215
sp_hook_dbmlsync_misc_error,240
sp_hook_dbmlsync_ml_connect_failed,244
sp_hook_dbmlsync_process_exit_code,247
sp_hook_dbmlsync_set_ml_connect_info,253
sp_hook_dbmlsync_set_upload_end_progress,255
sp_hook_dbmlsync_sql_error,257
Synchronisation, Hook-Sequenz,203

host-Protokolloption

Verbindungsoption des MobiLink-Clients,41

Hovering

dbmlsync,100

HoverRescanThreshold, erweiterte dbmlsync-Option
Info,157

HoverRescanThreshold, erweiterte Option
dbmlsync,144

hrt, erweiterte dbmlsync-Option
Info,157

HTTP

MobiLink-Clientoptionen,27

HTTP-Synchronisation

MobiLink-Clientoptionen,27

http_buffer_responses-Protokolloption

Verbindungsoption des MobiLink-Clients,41

http_password-Protokolloption

Verbindungsoption des MobiLink-Clients,42

http_proxy_password-Protokolloption

Verbindungsoption des MobiLink-Clients,43

http_proxy_userid-Protokolloption

Verbindungsoption des MobiLink-Clients,44

http_userid-Protokolloption

Verbindungsoption des MobiLink-Clients,45

HTTPS

MobiLink-Clientoptionen,28

HTTPS-Synchronisation

MobiLink-Clientoptionen,28

I

iAnywhere.MobiLink.Client-Namespace
Dbmlsync .NET-API-Referenz,291

Identity, Option

Verbindungsoption des MobiLink-Clients,46

identity_password-Option

Verbindungsoption des MobiLink-Clients,47

IDs

entfernte IDs in MobiLink,9

IgnoreHookErrors, erweiterte dbmlsync-Option
Info,158

IgnoreHookErrors, erweiterte Option
dbmlsync,144

IgnoreScheduling, erweiterte dbmlsync-Option
Info,158

IgnoreScheduling, erweiterte Option
dbmlsync,144

Ignorieren von Fehlern in Hook-Prozeduren
SQL Anywhere-Clients,208

IMAP-Authentifizierung

MobiLink-Skripten,17

inc, erweiterte dbmlsync-Option
Info,159

Increment, erweiterte dbmlsync-Option
Info,159

Increment, erweiterte Option
dbmlsync,144

Init-Methode

DbmlsyncClient-Klasse [Dbmlsync .NET-API],301

DbmlsyncClient-Klasse [Dbmlsync C++-API],276

Initiieren

Synchronisation für SQL Anywhere-Clients,88

Initiieren der Synchronisation von einer Anwendung
aus

SQL Anywhere-Clients,95

Initiieren einer Synchronisation
SQL Anywhere-Clients,88

Inkrementelle Uploads

MobiLink-Synchronisation,159

Installieren

MobiLink-Provider für Microsoft ActiveSync für
SQL Anywhere-Clients,97

Installieren, MobiLink-Provider für Microsoft
ActiveSync

SQL Anywhere-Clients,97

InstantiateClient-Methode

DbmlsyncClient-Klasse [Dbmlsync .NET-API],301

DbmlsyncClient-Klasse [Dbmlsync C++-API],277

isc, erweiterte dbmlsync-Option
Info,158

J

Java

- MobiLink-Benutzerauthentifizierung,16
- Java- und .NET-Benutzerauthentifizierung
MobiLink,16
- java.naming.provider.url
externe Authenticator-Eigenschaften in
MobiLink,19

K

Kennwörter

- für MobiLink ändern,9
- MobiLink, Authentifizierung durch Endbenutzer,8
- MobiLink-Benutzerauthentifizierung,
Einrichtung,6

Klassennamen

- Microsoft ActiveSync,142

Kommunikation

- für MobiLink festlegen,2
- MobiLink dbmsync, Option -adr,149
- MobiLink dbmsync, Option -c,116
- MobiLink dbmsync, Option -ctp,150
- MobiLink-Clients,25

Konfigurieren

- entfernte SQL Anywhere-Datenbanken für
Microsoft ActiveSync,96
- MobiLink-Benutzereigenschaften für SQL
Anywhere-Clients,84,85

L

LDAP

- MobiLink-Skripten,17

LockTables, erweiterte dbmsync-Option

- Info,160
- Parallelität während der Synchronisation,94

LockTables, erweiterte Option

- dbmsync,144

Log-Offsets

- MobiLink SQL Anywhere-Clients,73

Logdateien

- MobiLink, SQL Anywhere-Clients,103
- Transaktionslogs für MobiLink SQL Anywhere
Client-Dienstprogramm (dbmsync),93

Logscan-Abruf

- Info,152

Löschen

- Artikel aus MobiLink SQL Anywhere-Clients,79

- MobiLink-Subskriptionen aus SQL Anywhere-
Clients,88

- Publikationen aus MobiLink SQL Anywhere-
Clients,82

Löschen von MobiLink-Benutzern

- SQL Anywhere-Clients,88

Löschen von Publikationen

- MobiLink SQL Anywhere-Clients,82

lt, erweiterte dbmsync-Option

- Info,160

M

Mac OS X

- MobiLink,104

mail.imap.host

- externe Authenticator-Eigenschaften in
MobiLink,19

mail.imap.port

- externe Authenticator-Eigenschaften in
MobiLink,19

mail.pop3.host

- externe Authenticator-Eigenschaften in
MobiLink,19

mail.pop3.port

- externe Authenticator-Eigenschaften in
MobiLink,19

Meldungslog

- Info zu MobiLink SQL Anywhere Client
Dienstprogramm (dbmsync),103

Meldungslogdatei

- MobiLink SQL Anywhere- Client-
Dienstprogramm (dbmsync), Option -os,127
- MobiLink SQL Anywhere- Client-
Dienstprogramm (dbmsync), Option -ot,127
- MobiLink SQL Anywhere-Client-Dienstprogramm
(dbmsync),126

Microsoft ActiveSync

- Anwendungen für SQL Anywhere-Clients
registrieren,98
- CREATE SYNCHRONIZATION USER-
Anweisung für MobiLink SQL Anywhere-
Clients,96
- Klassennamen für dbmsync,142
- MobiLink-Dienstprogramm zur Installation des
Microsoft ActiveSync-Providers (mlasinst),20
- MobiLink-Provider für SQL Anywhere-Clients
installieren,97

- MobiLink-Provider installieren,20
- Mircrosoft ActiveSync
 - MobiLink SQL Anywhere-Clients,95
- MirrorLogDirectory, erweiterte dbmlsync-Option
 - Info,161
- MirrorLogDirectory, erweiterte Option
 - dbmlsync,144
- ml_remote_id, Option
 - SQL Anywhere-Clients,72
- ml_user
 - SQL Anywhere-Client über alte Version installieren,72
- ml_username
 - erstellen,5
 - Info,4
- mlasdesk.dll
 - Installation,20
- mlasdev.dll
 - Installation,20
- mlasinst, Dienstprogramm
 - Syntax,20
- mlasinst-Dienstprogramm
 - MobiLink-Provider für Microsoft ActiveSync für SQL Anywhere-Clients installieren,97
 - Verwendung von dbmlsync,95
- mld, erweiterte dbmlsync-Option
 - Info,161
- mlfiletransfer-Dienstprogramm
 - Optionen,23
 - Syntax,22
- mluser, Dienstprogramm
 - verwenden,6
- mn, erweiterte dbmlsync-Option
 - Info,163
- MobiLink
 - Benutzer,4
 - dbmlsync-Ereignis-Hooks,202
 - dbmlsync-Optionen,106
 - Dienstprogramme für Clients,19
 - Ereignis-Hook,202
 - RI-Verletzungen protokollieren,224
 - Skriptgesteuerter Upload,323
 - SQL Anywhere-Clients,69
 - SQL Anywhere-Clients planen,100
 - Verbindungsparameter für Clients,25
- MobiLink, skriptgesteuerter Upload
 - einrichten,325
- MobiLink, SQL-Anweisungen
 - Liste,181
- MobiLink, Synchronisationsprofile
 - Einführung,182
- MobiLink-Benutzer
 - Eigenschaften für SQL Anywhere-Clients konfigurieren,84,85
 - erstellen,5
 - erstellen in SQL Anywhere-Clients,83
 - Info,4
 - Info zu SQL Anywhere-Clients,82
- MobiLink-Benutzer, anfängliches Kennwort
 - mit mluser einrichten,6
 - mit Sybase Central festlegen,6
- MobiLink-Benutzername
 - Info,4
- MobiLink-Benutzernamen
 - erstellen,5
 - in Skripten verwenden,11
- MobiLink-Client
 - erweiterte Optionen,143
- MobiLink-Client, erweiterte Optionen
 - Info,143
- MobiLink-Client, Netzwerkprotokolloptionen
 - Info,25
- MobiLink-Client-Dienstprogramm (dbmlsync)
 - Optionen,106
- MobiLink-Clients, Dienstprogramme
 - Info,19
- MobiLink-Dienstprogramm für die Dateiübertragung (mlfiletransfer)
 - Syntax,22
- MobiLink-Dienstprogramm zur Installation des Microsoft ActiveSync-Providers (mlasinst)
 - Syntax,20
- MobiLink-Dienstprogramme
 - Client,19
 - Syntax für MobiLink für Microsoft ActiveSync-Provider (mlasinst),20
 - Syntax für MobiLink SQL Anywhere-Client (dbmlsync),106
 - Syntax für MobiLink-Dienstprogramm für die Dateiübertragung (mlfiletransfer),22
- MobiLink-Performance
 - Anzahl der Upload-Zeilen schätzen,140
- MobiLink-Server
 - Mac OS X,104
- MobiLink-Sicherheit
 - angepasste Benutzerauthentifizierung,15

- Benutzerauthentifizierung,4
- Benutzerauthentifizierung, Architektur,13
- Benutzerauthentifizierung, Kennwörter,8
- Benutzerauthentifizierungsverfahren wählen,12
- Kennwörter,6
- Kennwörter ändern,9
- neue Benutzer,7
- MobiLink-Synchronisation
 - Skriptgesteuerter Upload,323
 - SQL Anywhere-Clients,69
 - SQL Anywhere-Clients planen,100
- MobiLink-Synchronisationsclient
 - Optionen,106
- MobiLink-Synchronisationsprofile
 - Optionen,182
- MobiLink-Synchronisationssubskriptionen
 - SQL Anywhere-Clients,86
- MobiLinkPwd, erweiterte dbmlsync-Option
 - Info,162
- MobiLinkPwd, erweiterte Option
 - dbmlsync,144
- mp, erweiterte dbmlsync-Option
 - Info,162
- MSGQ_SHUTDOWN_REQUESTED
 - DBTools-Schnittstelle für dbmlsync,321
- MSGQ_SLEEP_THROUGH
 - DBTools-Schnittstelle für dbmlsync,321
- MSGQ_SYNC_REQUESTED
 - DBTools-Schnittstelle für dbmlsync,321

N

- name, Option
 - MobiLink-Dienstprogramm für Microsoft ActiveSync-Provider (mlasinst),20
- network_adapter_name-Protokolloption
 - Verbindungsoption des MobiLink-Clients,48
- network_leave_open-Protokolloption
 - Verbindungsoption des MobiLink-Clients,49
- network_name-Protokolloption
 - Verbindungsoption des MobiLink-Clients,49
- Netzwerkparameter
 - MobiLink-Clients,25
- Netzwerkprotokolle
 - angeben für dbmlsync,150
 - angeben für MobiLink,2
 - MobiLink-Clientoptionen für HTTP,27
 - MobiLink-Clientoptionen für HTTPS,28

- MobiLink-Clientoptionen für TCP/IP,26
- MobiLink-Clientoptionen für TLS,26
- UltraLite-Unterstützung,30
- Netzwerkprotokolloptionen
 - dbmlsync,149
 - MobiLink-Clients,25
- Neu startbare Downloads
 - dbmlsync, Option -dc,118
 - sp_hook_dbmlsync_end,232
- Neue Benutzer
 - MobiLink-Benutzerauthentifizierung,7
- NewMobiLinkPwd, erweiterte dbmlsync-Option
 - Info,163
- NewMobiLinkPwd, erweiterte Option
 - dbmlsync,144
- NoSyncOnStartup, erweiterte dbmlsync-Option
 - Info,163
- NoSyncOnStartup, erweiterte Option
 - dbmlsync,144
- nss dbmlsync, erweiterte Option
 - Info,163

O

- OfflineDirectory, erweiterte dbmlsync-Option
 - Info,164
- OfflineDirectory, erweiterte Option
 - dbmlsync,144
- Offsets
 - MobiLink SQL Anywhere-Clients,73
- Offsets für den Verarbeitungsfortschritt
 - MobiLink SQL Anywhere-Clients,73
- Optionen
 - MobiLink dbmlsync, erweiterte Optionen,143
 - MobiLink SQL Anywhere Client-Dienstprogramm (dbmlsync),106
 - MobiLink-Dienstprogramm für die Dateiübertragung (mlfiletransfer),23
 - MobiLink-Dienstprogramm für Microsoft ActiveSync-Provider (mlasinst),20
 - UltraLite, Netzwerkprotokolle,30
- Optionen für die Performance, Optimierung
 - MobiLink SQL Anywhere-Clients,92

P

- p, erweiterte dbmlsync-Option
 - Info,152
- Palm OS

-
- Version 12, nicht mehr empfohlene Funktion, MobiLink,1
 - Parallelität
 - MobiLink SQL Anywhere-Clients,94
 - Parameter
 - MobiLink SQL Anywhere-Client-Dienstprogramm (dbmlsync),106
 - MobiLink-Clientverbindung,25
 - MobiLink-Dienstprogramm für die Dateiübertragung (mlfiletransfer),23
 - MobiLink-Dienstprogramm für Microsoft ActiveSync-Provider (mlasinst),20
 - Performance
 - MobiLink SQL Anywhere-Clients,92
 - persistent-Protokolloption
 - Verbindungsoption des MobiLink-Clients,51
 - Ping
 - MobiLink-Server,129
 - Synchronisationsparameter in dbmlsync,129
 - Ping-Methode
 - DbmlsyncClient-Klasse [Dbmlsync .NET-API],302
 - DbmlsyncClient-Klasse [Dbmlsync C++-API],277
 - Planen
 - MobiLink SQL Anywhere-Clients,100
 - PollingPeriod, erweiterte dbmlsync-Option
 - Info,165
 - PollingPeriod, erweiterte Option
 - dbmlsync,144
 - POP3-Authentifizierung
 - MobiLink-Skripten,17
 - port-Protokolloption
 - Verbindungsoption des MobiLink-Clients,51
 - pp, erweiterte dbmlsync-Option
 - Info,165
 - Praktische Einführungen
 - skriptgesteuerter Upload,337
 - Prioritätenfolge für erweiterte Optionen und Verbindungsparameter
 - SQL Anywhere-Clients,143
 - Programmierschnittstellen
 - dbmlsync,102
 - Protokolle
 - angeben für dbmlsync,150
 - MobiLink-Clientoptionen für HTTP,27
 - MobiLink-Clientoptionen für HTTPS,28
 - MobiLink-Clientoptionen für TCP/IP,26
 - MobiLink-Clientoptionen für TLS,26
 - UltraLite-Liste,30
 - Protokollieren
 - MobiLink RI-Verletzungen,224
 - Protokollierung
 - MobiLink SQL Anywhere Client-Dienstprogrammaktionen (dbmlsync-Aktionen),103
 - MobiLink SQL Anywhere-Client-Dienstprogramm (dbmlsync), Option -v,141
 - Transaktionslogs für MobiLink SQL Anywhere Client-Dienstprogramm (dbmlsync),93
 - Protokollierung von dbmlsync-Aktivitäten
 - Info,103
 - Protokolloptionen
 - dbmlsync,149
 - MobiLink-Clients,25
 - proxy_host-Protokolloption
 - Verbindungsoption des MobiLink-Clients,52,53
 - proxy_hostname, Option
 - Verbindungsoption des MobiLink-Clients,52
 - proxy_portnumber, Option
 - Verbindungsoption des MobiLink-Clients,53
 - Prozeduren
 - MobiLink dbmlsync-Ereignis-Hooks,202
 - Publikation erstellen, Assistent
 - spaltenweise Verteilung in MobiLink,76
 - zeilenweise Verteilung in MobiLink SQL Anywhere-Clients,77
 - Publikationen
 - einfache Publikationen für MobiLink SQL Anywhere-Clients,75
 - für MobiLink SQL Anywhere-Clients ändern,79
 - für MobiLink SQL Anywhere-Clients erstellen,74
 - Info zu MobiLink SQL Anywhere-Clients,74
 - löschen aus MobiLink SQL Anywhere-Clients,82
 - Offsets für MobiLink SQL Anywhere-Clients,73
 - reiner Download,78
 - spaltenweise Verteilung für MobiLink SQL Anywhere-Clients,76
 - WHERE-Klausel in MobiLink verwenden,77
 - zeilenweise Verteilung für MobiLink SQL Anywhere-Clients,77
 - Publizieren
 - ausgewählte Spalten für MobiLink SQL Anywhere-Clients,76
 - ausgewählte Zeilen in MobiLink,77
 - MobiLink, ausgewählte Spalten (SQL Anywhere-Clients),76

- MobiLink, ausgewählte Zeilen (SQL Anywhere-Clients),77
- MobiLink, ganze Tabellen (SQL Anywhere-Clients),75
- SQL Anywhere-Client, ganze Tabellen,75
- SQL Anywhere-Clienttabellen,74
- Publizieren von Daten
 - MobiLink SQL Anywhere-Clients,74
- Puffergröße
 - Verbindungsoption des MobiLink-Clients,30

R

- Referenzielle Integrität
 - MobiLink RI-Verletzungen klären,224
- Registrieren
 - MobiLink SQL Anywhere-Anwendungen mit Microsoft ActiveSync,98
 - MobiLink-Benutzer,5
- Registrieren von MobiLink-Benutzern
 - Info,5
- Reine Download-Synchronisation
 - dbmlsync, erweiterte Option DownloadOnly,153
 - dbmlsync-Option -ds,120
 - Info,78
- Reine Upload-Synchronisation
 - dbmlsync, Option -uo,139
 - entfernte SQL Anywhere-Datenbanken,173
- Reiner Download
 - dbmlsync, erweiterte Option DownloadOnly,153
 - dbmlsync-Option -ds,120
 - Publikationen,78
 - unterschiedliche Ansätze,78
- RI-Verletzungen
 - MobiLink dbmlsync-Clients,207
- ROLLBACK-Anweisung
 - Hook-Prozeduren,204
- Rollen
 - MobiLink-Client,89
 - Sicherheitshinweise zur Synchronisation,89
- Rückgabecodes
 - dbmlsync [sp_hook_dbmlsync_abort],208
 - dbmlsync
 - [sp_hook_dbmlsync_process_exit_code],247

S

- s.remote_id
 - Verwendung,11

- s.username
 - MobiLink, in Skripten verwenden,11
- sa, erweiterte dbmlsync-Option
 - Info,169
- sch, erweiterte dbmlsync-Option
 - Info,166
- Schedule, erweiterte dbmlsync-Option
 - Info,166
- Schedule, erweiterte Option
 - dbmlsync,144
- Schema
 - Änderungen in entfernten MobiLink-Datenbanken,62
- Schema-Upgrades
 - entfernte UltraLite-Datenbanken,66
 - Ereignis-Hook
 - sp_hook_dbmlsync_schema_upgrade,249
- Schemaänderungen
 - entfernte MobiLink-Datenbanken,62
 - entfernte SQL Anywhere-Datenbanken,64
- Schnittstellen
 - DBTools für dbmlsync,317
- ScriptVersion, erweiterte dbmlsync-Option
 - Info,168
- ScriptVersion, erweiterte Option
 - dbmlsync,144
- SendDownloadACK, erweiterte dbmlsync-Option
 - Info,169
- SendDownloadAck, erweiterte Option
 - dbmlsync,144
- SendTriggers, erweiterte dbmlsync-Option
 - Info,169
- SendTriggers, erweiterte Option
 - dbmlsync,144
- Sequenzen
 - Synchronisation, Hook-Prozeduren,203
- Serverspeicherte Prozeduren
 - MobiLink dbmlsync-Ereignis-Hooks,202
- set_cookie-Protokolloption
 - Verbindungsoption des MobiLink-Clients,54
- SetProperty-Methode
 - DbmlsyncClient-Klasse [Dbmlsync .NET-API],302
 - DbmlsyncClient-Klasse [Dbmlsync C++-API],278
- ShutdownServer-Methode
 - DbmlsyncClient-Klasse [Dbmlsync .NET-API],304
 - DbmlsyncClient-Klasse [Dbmlsync C++-API],279

Sicherheit	sp_hook_dbmlsync_download_log_ri_violation
angepasste MobiLink-Benutzerauthentifizierung,15	Syntax,224
Benutzerauthentifizierung, Kennwörter,8	sp_hook_dbmlsync_download_ri_violation
MobiLink, neue Benutzer,7	Syntax,226
MobiLink-Benutzerauthentifizierung,4	sp_hook_dbmlsync_download_table_begin
MobiLink-Kennwörter ändern,9	Syntax,229
MobiLink-Synchronisation von SQL Anywhere-	sp_hook_dbmlsync_download_table_end
Clients,89	Syntax,230
Sicherungen	sp_hook_dbmlsync_end
Wiederherstellung entfernter Datenbanken,132	Syntax,232
Skriptbasierter Upload	sp_hook_dbmlsync_log_rescan
Info zu MobiLink,323	Syntax,235
Skripten	sp_hook_dbmlsync_logscan_begin
MobiLink, Parameter remote_id,11	Syntax,237
Skriptgesteuerter Upload	sp_hook_dbmlsync_logscan_end
Info zu MobiLink,323	Syntax,239
MobiLink, Beispiel,338	sp_hook_dbmlsync_misc_error
MobiLink, benutzerdefinierte Fortschrittswerte,334	Syntax,240
MobiLink, gespeicherte Prozeduren für	sp_hook_dbmlsync_ml_connect_failed
Aktualisierungen definieren,336	Syntax,244
MobiLink, gespeicherte Prozeduren für	sp_hook_dbmlsync_process_exit_code
Einfügungen definieren,334	Syntax,247
MobiLink, gespeicherte Prozeduren für	sp_hook_dbmlsync_schema_upgrade
Löschungen definieren,335	Syntax,249
MobiLink, gespeicherte Prozeduren für	sp_hook_dbmlsync_set_extended_options
skriptgesteuerten Upload definieren,332	Syntax,252
MobiLink-Design,325	sp_hook_dbmlsync_set_ml_connect_info
Rollen und Privilegien,337	Syntax,253
Skriptparameter	sp_hook_dbmlsync_set_upload_end_progress
remote_id,11	Syntax,255
username,11	sp_hook_dbmlsync_sql_error
Skriptversionen	Syntax,257
Beispiel für Zuordnung zu Subskription,63	sp_hook_dbmlsync_upload_begin
Subskriptionen zuordnen,63	Syntax,260
sp_hook_dbmlsync_abort	sp_hook_dbmlsync_upload_end
Syntax,208	Syntax,261
sp_hook_dbmlsync_all_error	sp_hook_dbmlsync_validate_download_file
Syntax,211	Syntax,264
sp_hook_dbmlsync_begin	Spalten
Syntax,214	zu entfernten MobiLink-Datenbanken
sp_hook_dbmlsync_communication_error	hinzufügen,65
Syntax,215	Spaltenweise Verteilung
sp_hook_dbmlsync_delay	MobiLink SQL Anywhere-Clients,76
Syntax,218	Sperre
sp_hook_dbmlsync_download_begin	MobiLink SQL Anywhere-Clients,94
Syntax,221	Spiegellogs
sp_hook_dbmlsync_download_end	für dbmlsync löschen,161
Syntax,222	SQL Anywhere

- als MobiLink-Clients, 1
- SQL Anywhere Client, Dienstprogramm (dbmlsync)
 - Syntax, 106
- SQL Anywhere-Clientprotokollierung
 - Info, 103
- SQL Anywhere-Clients
 - Einführung, 1
 - für Microsoft ActiveSync registrieren, 98
 - Info zu MobiLink, 69
 - MobiLink SQL Anywhere-Client-Dienstprogramm (dbmlsync), 106
- SQL-Anweisungen
 - MobiLink, 181
- st, erweiterte dbmlsync-Option
 - Info, 169
- StartServer-Methode
 - DbmlsyncClient-Klasse [Dbmlsync .NET-API], 305
 - DbmlsyncClient-Klasse [Dbmlsync C++-API], 280
- Subskriptionen
 - MobiLink SQL Anywhere-Clients, 86
- sv, erweiterte dbmlsync-Option
 - Info, 168
- Sync-Methode
 - DbmlsyncClient-Klasse [Dbmlsync .NET-API], 306
 - DbmlsyncClient-Klasse [Dbmlsync C++-API], 281
- SyncConsole
 - Erste Orientierung, 104
- Synchronisation
 - Abfolgeplanung in dbmlsync, 166
 - Abfolgeplanung von MobiLink SQL Anywhere-Clients, 100
 - angepasste Benutzerauthentifizierung, 15
 - anpassen, 202
 - dbmlsync, erste Synchronisation, 73
 - Dienstprogramme für MobiLink-Clients, 19
 - initiiieren für SQL Anywhere-Clients, 88
 - Kennwörter ändern, 9
 - Microsoft ActiveSync für MobiLink SQL Anywhere-Clients, 95
 - MobiLink dbmlsync-Ereignis-Hooks, 202
 - SQL Anywhere-Clients, 69
 - Transaktionen, 204
 - Verbindungsparameter für Clients, 25
- Synchronisation einer Ereignis-Hook-Sequenz
 - SQL Anywhere-Clients, 203
- Synchronisationsbenutzer
 - Eigenschaften für SQL Anywhere-Clients konfigurieren, 84, 85
 - erstellen, 5
 - erstellen in SQL Anywhere-Clients, 83
 - Info, 4
- Synchronisationsplanung
 - SQL Anywhere-Clients, 100
- Synchronisationsprofil, Optionen
 - Info, SQL Anywhere-Clients, 182
- Synchronisationsprofile
 - Option -sp, 135
 - Option ApplyDnldFile, 182
 - Option AuthParms, 182
 - Option Background, 182
 - Option BackgroundRetry, 182
 - Option CacheInit, 182
 - Option CacheMax, 182
 - Option CacheMin, 182
 - Option ContinueDownload, 182
 - Option CreateDnldFile, 182
 - Option DnldFileExtra, 182
 - Option DownloadOnly, 182
 - Option DownloadReadSize, 182
 - Option ExtOpt, 182
 - Option IgnoreHookErrors, 182
 - Option IgnoreScheduling, 182
 - Option KillConnections, 182
 - Option LogRenameSize, 182
 - Option MLUser, 182
 - Option MobiLinkPwd, 182
 - Option NewMobiLinkPwd, 182
 - Option ping, 182
 - Option Publication, 182
 - Option RemoteProgressGreater, 182
 - Option RemoteProgressLess, 182
 - Option Subscription, 182
 - Option TransactionalUpload, 182
 - Option UpdateGenNum, 182
 - Option UploadOnly, 182
 - Option UploadRowCnt, 182
 - Option Verbosity, 182
 - SQL Anywhere-Clients, 182
- Synchronisationssubskriptionen
 - ändern für SQL Anywhere-Clients, 87
 - löschen aus SQL Anywhere-Clients, 88
 - Prioritätenfolge für erweiterte Optionen und Verbindungsparameter, 143
 - SQL Anywhere-Clients, 86

Synchronisationssubskriptionen erstellen

SQL Anywhere-Clients,86

Synchronisieren

MobiLink, neue Benutzer,7

Syntax

MobiLink dbmlsync-Ereignis-Hooks,202

MobiLink SQL Anywhere-Client-Dienstprogramm
(dbmlsync),106

MobiLink-Dienstprogramm für die
Dateiübertragung (mlfiletransfer),22

MobiLink-Dienstprogramm zur Installation des
Microsoft ActiveSync-Providers (mlasinst),20

MobiLink-Synchronisation, Dienstprogramme für
Clients,19

SYS_RUN_REPLICATION_ROLE-Systemrolle

MobiLink-Client,89

Systemprozeduren

MobiLink dbmlsync-Ereignis-Hooks,202

Systemtabellen

MobiLink-Clients,3

T

Tabellen

publizieren für MobiLink SQL Anywhere-
Clients,74

spaltenweise Verteilung für MobiLink SQL
Anywhere-Clients,76

zeilenweise Verteilung für MobiLink SQL
Anywhere-Clients,77

zu entfernten MobiLink SQL Anywhere-
Datenbanken hinzufügen,64

Tabellenfolge

dbmlsync, erweiterte Optionen,171

TableOrder, erweiterte dbmlsync-Option

Info,171

TableOrder, erweiterte Option

dbmlsync,144

TableOrderChecking, erweiterte dbmlsync-Option

Info,172

TableOrderChecking, erweiterte Option

dbmlsync,144

TCP/IP

MobiLink-Clientoptionen,26

MobiLink-Clientoptionen für TLS,26

TCP/IP-Synchronisation

MobiLink-Clientoptionen,26

MobiLink-Clientoptionen für TLS,26

timeout-Protokolloption

Verbindungsoption des MobiLink-Clients,54

TLS

MobiLink-Clientoptionen,26

TLS-Synchronisation

MobiLink-Clientoptionen,26

toc, erweiterte dbmlsync-Option

Info,172

tor, erweiterte dbmlsync-Option

Info,171

Transaktionslog

MobiLink SQL Anywhere Client-Dienstprogramm
(dbmlsync),93

Spiegellog löschen für dbmlsync,161

Transaktionslog-Spiegel

für dbmlsync löschen,161

Transaktionslogdateien

MobiLink SQL Anywhere-Client Dienstprogramm
(dbmlsync),93

Transaktionsstufe, Uploads

dbmlsync, Option -tu,136

trusted_certificate_name-Protokolloption

Verbindungsoption des MobiLink-Clients,57

trusted_certificates-Protokolloption

Verbindungsoption des MobiLink-Clients,56

U

Überprüfen der Tabellenfolge

dbmlsync, erweiterte Optionen,172

Übertragen von Dateien

MobiLink-Dienstprogramm für die
Dateiübertragung (mlfiletransfer),22

Überwachen

MobiLink RI-Verletzungen protokollieren,224

UltraLite

MobiLink-Clients,2

UltraLite Netzwerkprotokolle

Deploymentanforderungen für Komprimierung,30

UltraLite, Netzwerkprotokolle

Synchronisationsoptionen für HTTP,30

Synchronisationsoptionen für HTTPS,30

Synchronisationsoptionen für TCP/IP,30

Synchronisationsoptionen für TLS,30

UltraLite, Protokolle

Synchronisationsoptionen für HTTP,30

Synchronisationsoptionen für HTTPS,30

Synchronisationsoptionen für TCP/IP,30

- Synchronisationsoptionen für TLS,30
- UltraLite, Synchronisation
 - HTTP-Clientoptionen,30
 - HTTPS-Clientoptionen,30
 - komprimierte Synchronisation,
 - Deploymentanforderungen,30
 - TCP/IP-Clientoptionen,30
 - TLS-Clientoptionen,30
- UltraLite-Anwendungen
 - als MobiLink-Clients,2
- UltraLite-Clients
 - Einführung,2
- Unterstützte Netzwerkprotokolle
 - UltraLite-Liste,30
- uo, erweiterte dbmlsync-Option
 - Info,173
- Upgrade von entfernten Datenbanken
 - MobiLink SQL Anywhere-Clients,72
- UPLD_ERR_INVALID_USERID_OR_PASSWORD
 - dbmlsync-Fehlermeldung,263
- UPLD_ERR_REMOTE_ID_ALREADY_IN_USE
 - dbmlsync-Fehlermeldung,263
- UPLD_ERR_SQLCODE_n
 - dbmlsync-Fehlermeldung,263
- UPLD_ERR_USERID_OR_PASSWORD_EXPIRED
 - dbmlsync-Fehlermeldung,263
- Upload von Daten
 - Skriptgesteuerter Upload in MobiLink,323
- UploadOnly, erweiterte dbmlsync-Option
 - Info,173
- UploadOnly, erweiterte Option
 - dbmlsync,144
- Uploads
 - MobiLink SQL Anywhere-Client-Dienstprogramm (dbmlsync), Option -uo für reine Upload-Synchronisation,139
 - skriptgesteuerter Upload in MobiLink,323
- url_suffix-Protokolloption
 - Verbindungsoption des MobiLink-Clients,59
- username
 - MobiLink, in Skripten verwenden,11
- V**
- v, erweiterte dbmlsync-Option
 - Info,174
- Verbinden
 - MobiLink dbmlsync, Option -adr,149
 - MobiLink dbmlsync, Option -c,116
 - MobiLink dbmlsync, Option -ctp,150
 - MobiLink-Clients,25
- Verbindungen
 - MobiLink dbmlsync, Option -adr,149
 - MobiLink dbmlsync, Option -c,116
 - MobiLink dbmlsync, Option -ctp,150
 - MobiLink-Clients,25
- Verbindungen für Hook-Prozeduren
 - SQL Anywhere-Clients,207
- Verbindungsfehler
 - MobiLink dbmlsync-Clients,207
- Verbindungsoptionen
 - dbmlsync,149
- Verbindungsparameter
 - MobiLink SQL Anywhere-Clients,93
 - MobiLink-Clients,25
 - Prioritätenfolge für MobiLink SQL Anywhere-Clients,143
- Verbindungszeichenfolgen
 - MobiLink dbmlsync,116
- Verbose, erweiterte dbmlsync-Option
 - Info,174
- Verbose, erweiterte Option
 - dbmlsync,144
- VerboseHooks, erweiterte dbmlsync-Option
 - Info,175
- VerboseHooks, erweiterte Option
 - dbmlsync,144
- VerboseMin, erweiterte dbmlsync-Option
 - Info,176
- VerboseMin, erweiterte Option
 - dbmlsync,144
- VerboseOptions, erweiterte dbmlsync-Option
 - Info,176
- VerboseOptions, erweiterte Option
 - dbmlsync,144
- VerboseRowCounts, erweiterte dbmlsync-Option
 - Info,177
- VerboseRowCounts, erweiterte Option
 - dbmlsync,144
- VerboseRowValues, erweiterte dbmlsync-Option
 - Info,178
- VerboseRowValues, erweiterte Option
 - dbmlsync,144
- VerboseUpload, erweiterte dbmlsync-Option
 - Info,179
- VerboseUpload, erweiterte Option

- dbmlsync,144
- version-Protokolloption
 - Verbindungsoption des MobiLink-Clients,59
- Verteilung
 - spaltenweise für MobiLink SQL Anywhere-Clients,76
 - zeilenweise für MobiLink SQL Anywhere-Clients,77
- viewcert-Dienstprogramm
 - Optionen,20
- vm, erweiterte dbmlsync-Option
 - Info,176
- vn, erweiterte dbmlsync-Option
 - Info,177
- vo, erweiterte dbmlsync-Option
 - Info,176
- Vorbereiten
 - entfernte Datenbanken für MobiLink,70
- vr, erweiterte dbmlsync-Option
 - Info,178
- vs, erweiterte dbmlsync-Option
 - Info,175
- vu, erweiterte dbmlsync-Option
 - Info,179

W

- WaitForServerShutdown-Methode
 - DbmlsyncClient-Klasse [Dbmlsync .NET-API],307
 - DbmlsyncClient-Klasse [Dbmlsync C++-API],282
- WHERE-Klausel
 - MobiLink-Publikationen,77
- Wiederherstellung
 - entfernte Datenbanken aus Archiven wiederherstellen,132
- Windows Mobile
 - dbmlsync, DLLs vorab einlesen,129

Z

- Zeilenweise Verteilung
 - MobiLink SQL Anywhere-Clients,77
- Zeitplanung
 - ignorieren für dbmlsync,158
- Zertifikatsfelder
 - MobiLink TLS certificate_company, Option,31
 - MobiLink TLS certificate_name, Option,33
 - MobiLink TLS certificate_unit, Option,34

- Zertifikatsfelder prüfen
 - MobiLink TLS certificate_company, Option,31
 - MobiLink TLS certificate_name, Option,33
 - MobiLink TLS certificate_unit, Option,34
- zlib-Komprimierung
 - MobiLink-Synchronisation,36
- zlib_download_window_size-Protokolloption
 - Verbindungsoption des MobiLink-Clients,60
- zlib_upload_window_size-Protokolloption
 - Verbindungsoption des MobiLink-Clients,61
- Zustand
 - MobiLink SQL Anywhere-Clients,73

