MobiLink™
Serverinitiierte Synchronisation

Version 16.0

Februar 2013

Version 16.0
Februar 2013

Inhalt

Über diese Dokumentation

Diese Dokumentation beschreibt die serverinitiierte Synchronisation mit MobiLink, eine Funktion, die es dem MobiLink-Server gestattet, eine Synchronisation zu initiieren oder Aktionen auf entfernten Geräten auszuführen.

Serverinitiierte Synchronisation

Hinweis
Sie können mit Sybase Central entfernte Datenbanken verwalten und anschließend serverinitiierte entfernte Aufgaben (SIRT) als Alternative zur serverinitiierten Synchronisation verwenden. Weitere Hinweise finden Sie unter „Zentrale Administration von entfernten Datenbanken“ [*MobiLink - Serveradministration*] und „Serverinitiierte entfernte Aufgaben (SIRT)“ [*MobiLink - Serveradministration*].

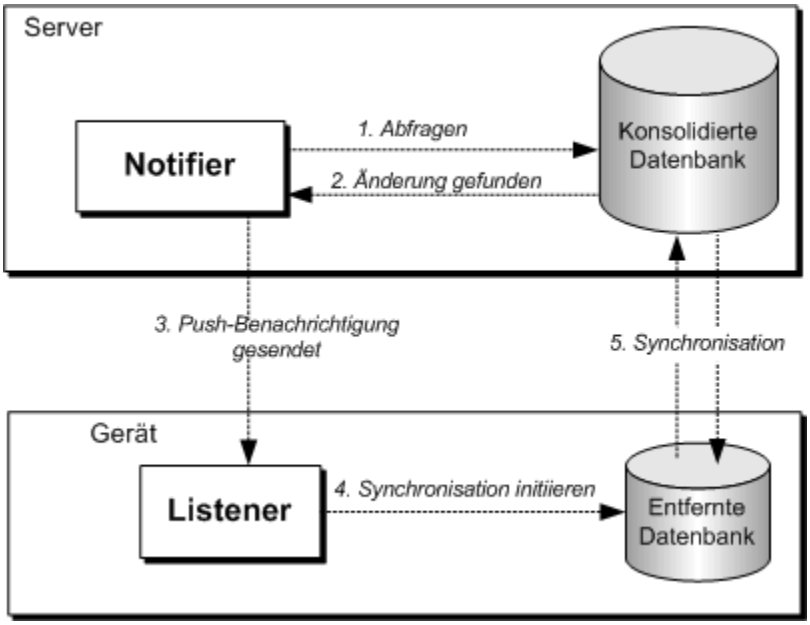
Die serverinitiierte MobiLink-Synchronisation gestattet es Ihnen, die Synchronisation von der konsolidierten Datenbank aus zu initiieren. Sie können Push-Benachrichtigungen an entfernte Datenbanken senden, sodass die entfernten Datenbanken die konsolidierte Datenbank aktualisieren. Diese MobiLink-Komponente stellt programmierbare Optionen zur Erkennung von Änderungen in der konsolidierten Datenbank für die Initiierung der Synchronisation bereit, sowie zur Auswahl von Geräten für den Empfang von Push-Benachrichtigungen und zur Festlegung, wie Geräte auf diese Push-Benachrichtigungen reagieren.

Beispiel

Ein Speditionsunternehmen stattet seine Fahrer mit mobilen Geräten aus. Auf jedem Gerät läuft eine Datenbank, die Routen und Zustellungsadressen enthält. Wenn ein Fahrer einen Hinweis auf eine Verkehrsstörung eingibt, wird der Bericht an eine konsolidierte Datenbank gesendet. Eine serverseitige MobiLink-Komponente, die als Notifier bezeichnet wird, erkennt den Bericht und sendet eine Push-Benachrichtigung an andere Fahrer, deren Routen von der Verkehrsstörung betroffen sind. Diese Push-Benachrichtigung bewirkt, dass die entfernten Datenbanken synchronisiert werden, sodass die Fahrer eine andere Route verwenden können.

Prozess der serverinitiierten Synchronisation

In der folgenden Abbildung prüft der Notifier eine konsolidierte Datenbank auf Änderungen. Der Notifier sendet eine Push-Benachrichtigung an ein Gerät, was dazu führt, dass die entfernte Datenbank mit der konsolidierten Datenbank synchronisiert wird.



Der Prozess der serverinitiierten Synchronisation umfasst folgende Schritte:

1. Mithilfe einer auf Geschäftslogik basierenden Abfrage prüft der Notifier eine konsolidierte Datenbank auf Änderungen, die mit der entfernten Datenbank synchronisiert werden müssen.

2. Wenn eine Änderung festgestellt wird, bereitet der Notifier eine Push-Benachrichtigung vor, die an das Gerät gesendet wird.

3. Der Notifier sendet die Push-Benachrichtigung. Push-Benachrichtigungen können über ein Device Tracker-, UDP-, SMTP- oder SYNC-Gateway gesendet werden.

4. Der Listener vergleicht Betreff, Inhalt und Absender mit einem Nachrichtenfilter.

5. Wenn die Filterbedingungen erfüllt sind, wird eine Aktion initiiert. In einer typischen Implementierung könnte eine Aktion z.B. den MobiLink-Client ausführen oder eine UltraLite-Anwendung starten.

Komponenten der serverinitiierten Synchronisation

Die serverinitiierte MobiLink-Synchronisation erfordert folgende Komponenten:

- **Push-Anforderungen** Eine Push-Anforderung ist eine Zeile von Werten in einer Ergebnismenge, die einen Notifier darüber informiert, dass Sie eine Push-Benachrichtigung an ein Gerät senden möchten. Push-Anforderungen bewirken die Ausführung von serverinitiierten Synchronisationen. Jede Datenbankanwendung kann Push-Anforderungen erstellen, auch der Notifier. Eine Push-Anforderung kann z.B. mithilfe eines Datenbank-Triggers erstellt werden, der aktiviert wird, wenn sich ein Preis ändert. Siehe „Push-Anforderungen“ auf Seite 7.

- **MobiLink-Notifier** Ein Notifier ist ein in den MobiLink-Server integriertes Programm. Es führt regelmäßige Prüfungen der konsolidierten Datenbank auf Push-Anforderungen aus. Sie können steuern, wie häufig der Notifier auf Push-Anforderungen prüft, indem Sie seine Eigenschaften festlegen. Sie müssen die Prüfung auf Push-Anforderungen in der Geschäftslogik festlegen und bestimmen, welche Geräte benachrichtigt werden müssen. Wenn der Notifier eine Push-Anforderung erkennt, wird eine Push-Benachrichtigung an ein Gerät gesendet. Siehe „Notifier“ auf Seite 12.

- **MobiLink-Listener** Ein Listener ist ein Programm, das auf einem Gerät ausgeführt wird. Es empfängt Push-Benachrichtigungen vom Notifier und filtert dann die Nachrichten und initiiert eine Aktion mithilfe eines Message-Handlers. In einer typischen Anwendung handelt es sich bei Aktionen um Synchronisationsaufrufe, doch Anwendungen sind in der Lage, auch andere Aktionen auszuführen. Sie können den MobiLink Listener so konfigurieren, dass er auf Push-Benachrichtigungen von ausgewählten Serverquellen und auf Push-Benachrichtigungen mit einem bestimmten Inhalt unterschiedlich reagiert.

  Auf Windows-Geräten ist der MobiLink Listener ein ausführbares Programm, das Sie mithilfe von Befehlszeilenoptionen konfigurieren. Um Push-Benachrichtigungen zu empfangen, muss das Gerät eingeschaltet sein und der MobiLink Listener muss ausgeführt werden. Siehe „MobiLink Listener-Dienstprogramm für Windows-Geräte (dblsn)“ auf Seite 55.

- **Lightweight-Polling-Modul** Ein Lightweight-Polling-Modul ist eine Geräteanwendung, die Push-Benachrichtigungen in einer angegebenen Zeitspanne abruft. Die Verwendung eines Lightweight-Polling-Moduls ist eine Alternative zum Einrichten eines Gateways und wird empfohlen, da hierbei keine beständige Verbindung zum Server erforderlich ist und dies dazu beiträgt, die Lebensdauer des Akkus zu verlängern.

  Der MobiLink Listener ist ein Lightweight-Polling-Modul, das Sie mithilfe von MobiLink Listener-Befehlszeilenoptionen konfigurieren können. Alternativ dazu können Sie die Lightweight-Polling-API verwenden, um ein eigenes Lightweight-Polling-Modul zu erstellen.

  Siehe:

  - „Lightweight-Polling-Option einrichten“ auf Seite 19
  - „Lightweight-Polling-API“ auf Seite 83

- **Gateway (Alternative zu einem Lightweight-Polling-Modul)** Ein Gateway stellt eine Notifier-Schnittstelle zum Senden von Push-Benachrichtigungen an ein Gerät bereit. Gateways sind eine Alternative zu Lightweight-Polling-Modulen. Sie können Nachrichten über ein Device Tracking-, SYNC-, UDP- oder SMTP-Gateway senden. Siehe „Gateways und Netzbetreiber“ auf Seite 22.

Hinweis
Sie können mit Sybase Central entfernte Datenbanken verwalten und anschließend serverinitiierte entfernte Aufgaben (SIRT) als Alternative zu serverinitiierter Synchronisation verwenden. Weitere Hinweise finden Sie unter „Zentrale Administration von entfernten Datenbanken“ [*MobiLink - Serveradministration*] und „Serverinitiierte entfernte Aufgaben (SIRT)“ [*MobiLink - Serveradministration*].

Hinweise zum Deployment der serverinitiierten Synchronisation

Berücksichtigen Sie die folgenden Punkte, bevor Sie serverinitiierte Synchronisationsanwendungen bereitstellen.

Geräteinschränkungen bei der Verwendung von UDP-Gateways

- Die IP-Adresse auf dem Gerät muss direkt vom MobiLink-Server aus adressierbar sein.
- Die IP-Protokollierung für die UDP-Benachrichtigung funktioniert nicht, wenn die IP-Adresse auf einem Windows-Gerät nicht direkt vom MobiLink-Server aus adressiert werden kann.

Einschränkungen für das Device Tracking

MobiLink Listener von SQL Anywhere 9.0.0 oder früher unterstützen Device Tracking (Geräteprotokollierung) nicht. Um Device Tracking zusammen mit diesen MobiLink Listenern zu verwenden, müssen Sie das Device Tracking manuell einrichten.

Unterstützte Plattformen für Geräte

Der MobiLink Listener wird unter Windows und Windows Mobile unterstützt.

Siehe auch

- „Device Tracking für MobiLink Listener 9.0.0 einrichten“ auf Seite 24

Schnellstart für serverinitiierte Synchronisation

Vor dem Ausführen dieser Prozedur müssen Sie MobiLink für die normale Synchronisation einrichten. Siehe MobiLink - Erste Orientierung.

1. Bereiten Sie auf dem MobiLink-Server die konsolidierte Datenbank zum Speichern von Push-Anforderungen vor. Siehe „Voraussetzungen für Push-Anforderungen“ auf Seite 7.

2. Konfigurieren Sie auf dem MobiLink-Server die Notifier-Ereignisse zum Erstellen und Verwalten von Push-Anforderungen. Siehe „Konfiguration von Notifier-Ereignissen und -Eigenschaften “ auf Seite 13.

3. Richten Sie auf dem Gerät ein Lightweight-Polling-Modul ein. Siehe „Lightweight-Polling-Module“ auf Seite 21.

   Wenn Sie kein Lightweight-Polling-Modul verwenden möchten, richten Sie ein unterstütztes Gateway auf dem MobiLink-Server ein. Wenn Sie ein SMTP-Gateway verwenden, müssen Sie auch einen Netzbetreiber konfigurieren. Siehe „Gateways und Netzbetreiber“ auf Seite 22.

4. Auf dem Gerät richten Sie einen MobiLink Listener zum Filtern von Nachrichten und zum Ausführen von Aktionen ein. Siehe „Message-Handler“ auf Seite 15.

Weitere Ressourcen

- Beispielanwendungen werden im Verzeichnis *%SQLANYSAMP16%\MobiLink* installiert. Alle Anwendungen im Zusammenhang mit der serverinitiierten Synchronisation befinden sich in Verzeichnissen mit dem Präfix **SIS_**.

Einrichten der serverinitiierten Synchronisation

In den folgenden Abschnitten wird beschrieben, wie Sie eine serverinitiierte Synchronisation einrichten:

- „Push-Anforderungen“
- „Notifier“
- „Listener“
- „Lightweight-Polling-Module“
- „Gateways und Netzbetreiber“

Push-Anforderungen

Eine Push-Anforderung ist eine Zeile von Werten in einer Ergebnismenge, die von einem Notifier darauf geprüft wird, ob Push-Benachrichtigungen an ein Gerät gesendet werden müssen. Ein Notifier stellt eine Push-Anforderung in eine Push-Benachrichtigung und sendet dann die Push-Benachrichtigung. Eine Push-Anforderung in einem typischen Setup einer serverinitiierten Synchronisation enthält Inhalte von Nachrichten und Informationen über das Zielgerät. Zum Senden einer Push-Benachrichtigung müssen Sie ein Notifier-Ereignis so konfigurieren, dass der Notifier die Push-Anforderung erkennen kann.

Voraussetzungen für Push-Anforderungen

Die Voraussetzungen für Push-Anforderungen sind von der Methode abhängig, die vom MobiLink-Server verwendet wird, um mit Geräten zu kommunizieren. Für alle Push-Anforderungen sind Spalten für Betreff und Inhalt erforderlich.

Wenn Sie Lightweight-Polling-Module für Push-Benachrichtigungen verwenden, müssen Sie eine Polling-Schlüsselspalte für ihre Identifizierung erstellen.

Wenn Sie Gateways zum Senden von Push-Benachrichtigungen verwenden, müssen Sie Spalten für Gateways und Adressen erstellen.

Es ist nicht erforderlich, Spalten für Push-Anforderungen zu erstellen, wenn sie auf dem System bereits vorhanden sind. Wenn die Voraussetzungen für die Push-Anforderungen erfüllt sind, können sie verwendet werden. Siehe „Mit Push-Anforderungen arbeiten“ auf Seite 9.

Voraussetzungen für Push-Anforderungen bei der Verwendung von Lightweight-Polling-Modulen (empfohlen)

Wenn Sie Lightweight-Polling-Module zum Abrufen von Push-Benachrichtigungen verwenden, erstellen Sie folgende Spalten:

Spalte	Typ	Beschreibung
Polling-Schlüssel	VARCHAR	Der Schlüssel, mit dem ein Lightweight-Polling-Modul identifiziert wird. Jedes Lightweight-Polling-Modul sendet einen Unique-Schlüssel, um sich beim MobiLink-Server zu identifizieren.
Betreff	VARCHAR	Die Betreffzeile der Nachricht
Inhalt	VARCHAR	Der Inhalt der Nachricht.

Voraussetzungen für Push-Anforderungen bei der Verwendung von Gateways

Sofern nicht anders angegeben, erstellen Sie folgende Spalten, wenn Sie Gateways zum Senden von Push-Benachrichtigungen verwenden:

Spalte	Typ	Beschreibung
Anforderungs-ID	INTEGER	Optional. Die eindeutige ID einer Push-Anforderung. Dieser Spaltenname ist für einige Notifier erforderlich. Siehe „Notifier-Ereignisse“ auf Seite 35.
Gateway	VARCHAR	Der Name des Gateways, an das die Nachricht gesendet wird.
Betreff	VARCHAR	Die Betreffzeile der Nachricht
Inhalt	VARCHAR	Der Inhalt der Nachricht.
Adresse	VARCHAR	Die Zieladresse eines Geräts.
Neusendeintervall	VARCHAR	Optional. Die Zeitspanne zwischen wiederholten Sendeversuchen für Nachrichten. Das Neusendeintervall ist nützlich, wenn ein UDP-Gateway in einem störungsanfälligen Netzwerk verwendet wird. Der Notifier geht davon aus, dass sich die Attribute, die der Push-Anforderung zugeordnet sind, nicht ändern. Nachfolgende Aktualisierungen nach dem ersten Abruf der Anforderung werden ignoriert. Der Notifier passt das nächste Polling-Intervall automatisch an, wenn eine Push-Benachrichtigung vor dem nächsten Polling-Zeitpunkt gesendet werden muss. Sie können das Senden einer Push-Anforderung mithilfe der Synchronisationslogik im request_cursor-Ereignis stoppen. Die Zustellbestätigung vom vorgesehenen MobiLink Listener kann eine nachfolgende Neusendung unterbinden. Siehe „request_cursor-Ereignis“ auf Seite 40.
Restzeit	VARCHAR	Optional. Angabe der Zeitdauer, bis die Neusendung abgelaufen ist.

Siehe auch

- „Konfiguration von Notifier-Ereignissen und -Eigenschaften “ auf Seite 13
- „request_cursor-Ereignis“ auf Seite 40

Beispiel

Im folgenden Beispiel werden die Voraussetzungen für Push-Anforderungen zur Verwendung von Lightweight-Polling erfüllt, indem die erforderlichen Tabellenspalten in einer konsolidierten SQL Anywhere-Datenbank erstellt werden:

```
CREATE TABLE PushRequest (
    req_id INTEGER DEFAULT AUTOINCREMENT PRIMARY KEY,
    poll_key VARCHAR(128),
    subject VARCHAR(128),
    content VARCHAR(128)
)
```

Sie müssen eine solche Tabelle nur erstellen, wenn die Push-Anforderungsspalten an keiner anderen Stelle verfügbar sind. Diese Spalten können sich in verschiedenen Tabellen, in bereits vorhandenen Tabellen oder in einer Ansicht befinden.

Mit Push-Anforderungen arbeiten

Push-Anforderungen generieren

Damit Push-Anforderungen generiert werden können, muss die konsolidierte Datenbank die für die serverinitiierte Synchronisation erforderlichen Push-Anforderungsspalten enthalten. Außerdem müssen Sie in der Lage sein, die Werte mit einer einzigen Datenbankabfrage zu erhalten. Push-Anforderungen werden automatisch generiert, wenn Sie im request_cursor-Ereignis eine Datenbankabfrage bereitstellen, die Push-Anforderungsspalten auswählt. Weitere Hinweise zum Einrichten der Voraussetzungen für Push-Anforderungen finden Sie unter „Voraussetzungen für Push-Anforderungen“ auf Seite 7.

Beispiel der Erzeugung von Push-Anforderungen für ein Lightweight-Polling-Modul

Angenommen, ein entferntes Gerät wird auf dem MobiLink-Server als unique_device_ID identifiziert und die konsolidierte Datenbank enthält die Tabelle PushRequest, die mit folgender SQL-Anweisung erstellt wurde:

```
CREATE TABLE PushRequest (
    req_id INTEGER DEFAULT AUTOINCREMENT PRIMARY KEY,
    poll_key VARCHAR(128),
    subject VARCHAR(128),
    content VARCHAR(128)
)
```

In diesem Beispiel können Sie die folgende SQL-Anweisung in der konsolidierten Datenbank für die Vorbereitung einer Push-Anforderung ausführen:

```
INSERT INTO PushRequest (poll_key, subject, content) VALUES
('unique_device_ID', 'synchronize', 'ASAP');
```

Bei der Verwendung des obenstehenden Skripts zum Einfügen von Werten in die Tabelle PushRequest wird keine Push-Anforderung generiert. Sie müssen eine Datenbankabfrage im request_cursor-Ereignis

auf dem MobiLink-Server erstellen, sodass die eingefügten Werte ausgewählt werden können und eine Push-Anforderung generiert werden kann.

In diesem Beispiel können Sie die folgende SQL-Anweisung für das request_cursor-Ereignisskript auf dem MobiLink-Server erstellen:

```
SELECT poll_key, subject, content FROM PushRequest;
```

Eine Push-Anforderung wird generiert, wenn das unique_device_ID-Gerät Push-Benachrichtigungen vom Server abruft und das request_cursor-Ereignis Daten in der PushRequest-Tabelle erkennt. Wenn eine Push-Benachrichtigung an das Gerät gesendet wird, ist der Betreff als **synchronize** und der Inhalt als **ASAP** definiert.

Einschränkungen für Push-Anforderungen

In der folgenden Tabelle sind die Einschränkungen für Push-Anforderungen für die einzelnen Spalten aufgelistet:

Spalte	Typ	Beschränkung
Anforderungs-ID	INTEGER	Dieser Wert muss ein eindeutiger Primärschlüssel sein.
Polling-Schlüssel	VARCHAR	Nur erforderlich, wenn Lightweight-Polling-Module verwendet werden. Für den Polling-Schlüssel gibt es keine Einschränkungen.
Gateway	VARCHAR	Nur erforderlich, wenn Gateways verwendet werden. Dieser Wert muss auf den Namen eines aktivierten Gateways gesetzt werden. Geben Sie Ihren eigenen benutzerdefinierten Gateway-Namen an oder wählen Sie einen der folgenden vorkonfigurierten Gateway-Namen aus: • **Default-DeviceTracker** • **Default-SMTP** • **Default-SYNC** • **Default-UDP** Siehe „Gateways als Alternative zu Lightweight-Polling-Modulen“ auf Seite 22.
Betreff	VARCHAR	Vermeiden Sie die Verwendung von nicht-alphanumerischen Zeichen bei der Einstellung dieses Werts. Geschweifte, runde und eckige Klammern, Anführungszeichen und Apostrophe sind für die interne Verwendung reserviert und dürfen in der Betreffspalte nicht verwendet werden.
Inhalt	VARCHAR	Für den Nachrichteninhalt gibt es keine Einschränkungen.

Spalte	Typ	Beschränkung
Adresse	VARCHAR	Nur erforderlich, wenn Gateways verwendet werden. Für UDP-Gateways muss dieser Wert eine IP-Adresse oder ein Hostname sein. Suffixe für Portnummern werden in folgenden Formaten unterstützt: ● *IP-address*:*port-number* ● *hostname*:*port-number* Für SMTP-Gateways muss dieser Wert eine E-Mail-Adresse sein. Für SYNC- und Device Tracking-Gateways muss dieser Wert der Empfängername sein, der mit der MobiLink Listener-Option -t+ angegeben wurde. Siehe „dblsn-Option -t“ auf Seite 68.
Neusendeintervall	VARCHAR	Standardmäßig wird dieser Wert in Minuten gemessen. Sie können **S**, **M** und **H** für die Einheiten Sekunden, Minuten bzw. Stunden angeben. Sie können auch Einheiten kombinieren. Beispiel: `1H 30M 10S` weist den Notifier an, die Nachrichten jeweils nach 1 Stunde, 30 Minuten und 10 Sekunden neu zu senden. Wenn dieser Wert NULL oder nicht angegeben ist, wird standardmäßig genau einmal gesendet, ohne Wiederholung.
Restzeit	VARCHAR	Standardmäßig wird dieser Wert in Minuten gemessen. Sie können **S**, **M** und **H** für die Einheiten Sekunden, Minuten bzw. Stunden angeben. Sie können auch Einheiten kombinieren. Beispiel: `3H 30M 10S` weist den Notifier an, die Wiederholung der Nachrichtensendung 3 Stunden, 30 Minuten und 10 Sekunden nach dem ersten Sendeversuch zu beenden. Wenn dieser Wert NULL oder nicht angegeben ist, wird standardmäßig genau einmal gesendet, ohne Wiederholung.

Push-Anforderungen erkennen und Push-Benachrichtigungen senden

Ein Notifier erkennt Push-Anforderungen, indem er das request_cursor-Ereignis regelmäßig auslöst. Standardmäßig wird kein Skript für dieses Ereignis angegeben. Sie müssen ein request_cursor-Ereignisskript bereitstellen, sodass der Notifier Push-Anforderungen erkennen kann. In einer typischen Anwendung ist ein request_cursor-Ereignisskript eine SELECT-Anweisung. Siehe „request_cursor-Ereignis“ auf Seite 40.

Im folgenden Beispiel wird mithilfe der ml_add_property-Systemprozedur ein request_cursor-Ereignisskript für einen benutzerdefinierten Notifier namens Simple erstellt. Die SELECT-Anweisung informiert den Notifier darüber, dass Push-Anforderungen in einer Tabelle namens PushRequest zu finden sind.


```
CALL ml_add_property('SIS', 'Notifier(Simple)', 'request_cursor',
    'SELECT poll_key, subject, content FROM PushRequest'
);
```

Hinweis
Sie müssen Spalten in derselben Reihenfolge auswählen, in der sie in der Push-Anforderung angegeben sind. Siehe „Voraussetzungen für Push-Anforderungen“ auf Seite 7.

Weitere Hinweise zum Einrichten von Notifier-Ereignissen finden Sie unter „MobiLink-Server-Einstellungen für serverinitiierte Synchronisation“ auf Seite 29.

Push-Anforderungen löschen

Der Notifier sendet Push-Benachrichtigungen erneut, wenn die Informationen zum benachrichtigten Gerät nie aktualisiert werden, nachdem sie gemäß den Geschäftsregeln gesendet und erfüllt wurden. Wenn die Push-Anforderungen erfüllt wurden, müssen Sie verhindern, dass der Notifier alte Push-Anforderungen findet. Wenn die Push-Benachrichtigungen zu Synchronisationszwecken gesendet wurden, können Sie die Push-Anforderungen mithilfe eines Synchronisationsskripts löschen.

Sie können Push-Anforderungen mithilfe des request_delete-Ereignisses anhand ihrer Anforderungs-ID löschen. Die Push-Anforderung muss jedoch eine Anforderungs-ID-Spalte enthalten und Sie müssen die Zustellungsbestätigung aktivieren.

Siehe:

- „Voraussetzungen für Push-Anforderungen“ auf Seite 7
- „request_delete-Ereignis“ auf Seite 41
- „MobiLink-Server-Einstellungen für serverinitiierte Synchronisation“ auf Seite 29

Siehe auch

- „Praktische Einführung: Konfigurieren der serverinitiierten Synchronisation unter Verwendung von Lightweight-Polling“ auf Seite 101
- „Praktische Einführung: Konfigurieren der serverinitiierten Synchronisation unter Verwendung von Gateways“ auf Seite 115

Notifier

Ein Notifier ist ein in den MobiLink-Server integriertes Programm, das die konsolidierte Datenbank regelmäßig auf Push-Anforderungen prüft. Wenn Push-Anforderungen erkannt werden, sendet er Push-Benachrichtigungen an Geräte. Ein Notifier führt auch eine Reihe von Ereignissen aus, die es Ihnen gestatten, Skripten zur Überwachung von Daten zu erstellen, Push-Anforderungen zu verwalten, Zustellungsbestätigungen zu verarbeiten und Fehler zu behandeln.

Notifier werden beim ersten Laden des MobiLink-Servers gestartet. Sie können mehrere Notifier in einer einzigen Instanz des MobiLink-Servers ausführen. Ein Beispiel zur Verwendung mehrerer Notifier finden Sie in der Beispielanwendung unter *%SQLANYSAMP16%\MobiLink\SIS_MultipleNotifier*.

Wenn ein Notifier die Datenbankverbindung verliert, versucht er, die Verbindung wiederherzustellen, bis er erneut Zugriff erhält. Nach der Wiederherstellung fährt der Notifier mit denselben Konfigurationseinstellungen fort.

Notifier in einer MobiLink-Serverfarm

In MobiLink 11.0 und früher konnte die serverinitiierte Synchronisation in einer MobiLink-Serverfarm redundante Push-Benachrichtigungen verursachen, die zu einer gesteigerten Auslastung und zusätzlichen Synchronisationen der konsolidierten Datenbank in einer MobiLink-Serverfarm führten. Ein Notifier kann nun auf jedem MobiLink-Server in der Farm ausgeführt werden. Die Notifier stellen gemeinsam sicher, dass keine redundanten Push-Benachrichtigungen für einen MobiLink Listener vorhanden sind. Mit der mlsrv16-Serveroption -lsc werden Informationen an andere Server übermittelt, wenn sie eine Verbindung mit dem lokalen MobiLink-Server herstellen wollen. Siehe „mlsrv16-Option -lsc“ [*MobiLink - Serveradministration*].

Diese Funktion macht einen bestimmten Notifier zum primären Notifier. Alle anderen Notifier werden zu sekundären Notifiern. Der primäre Notifier steuert Push-Benachrichtigungen direkt oder indirekt über die sekundären Notifier. Die sekundären Notifier leiten auch MobiLink Listener-Informationen an den primären Notifier, sodass dieser weiß, wo sich die MobiLink Listener befinden und wie sie erreicht werden können.

Wenn der MobiLink-Server ausfällt, auf dem der primäre Notifier läuft, wird von der Serverfarm ein neuer primärer Notifier gewählt und die Benachrichtigungen werden fortgesetzt.

MobiLink Listener können zu jedem MobiLink-Server in der Farm eine Verbindung herstellen, ohne wissen zu müssen, welcher Server der Primärserver ist.

Um diese Funktion verwenden zu können, sind die folgenden mlsrv16-Befehlszeilenoptionen auf jedem MobiLink-Server in der Farm erforderlich:

- „mlsrv16-Option -lsc“ [*MobiLink - Serveradministration*]
- „mlsrv16-Option -notifier “ [*MobiLink - Serveradministration*]
- „mlsrv16-Option -zs “ [*MobiLink - Serveradministration*]

Beispiel

Auf host001:

```
mlsrv16 -notifier -zs ml001 -lsc tcpip(host=host001;port=2439) ...
```

Auf host007:

```
mlsrv16 -notifier -zs ml007 -lsc tcpip(host=host007;port=2439) ...
```

Konfiguration von Notifier-Ereignissen und -Eigenschaften

Notifier-Ereignisse gestatten es Ihnen, Skripten zur Verwaltung des gesamten serverinitiierten Synchronisationsvorgangs einzubetten. Weitere Hinweise zur Notifier-Ereignissequenz und deren Auslösung finden Sie unter „Notifier-Ereignisse“ auf Seite 35.

Sie können Notifier-Ereignisse z.B. so konfigurieren, dass folgende Aufgaben ausgeführt werden:

- Ermittlung, welche Informationen wie in einer Push-Anforderung mit dem request_cursor-Ereignis an wen gesendet werden sollen.
- Erstellung von Push-Anforderungen, die auf Änderungen in der konsolidierten Datenbank mithilfe des begin_poll-Ereignisses reagieren (erweiterter Verwendungszweck).
- Löschen von Push-Anforderungen mit dem request_delete-Ereignis (wenn nötig)
- Protokollierung von Notifier-Abrufen und Bereinigung von Tabellendaten mit dem end_poll-Ereignis (erweiterter Verwendungszweck).

Notifier-Eigenschaften sind ähnlich Ereignissen. Ebenso wie der Nachrichtenprozess von Ereignissen gesteuert wird, steuern Eigenschaften das Notifier-Verhalten. Beispiel: Eigenschaften des Notifiers bestimmen, wie häufig der Notifier die konsolidierte Datenbank abfragt und ob der Notifier beim Start aktiviert wird. Notifier-Eigenschaften und -Ereignisse werden als serverseitige Einstellungen konfiguriert.

Siehe auch

- „MobiLink-Server-Einstellungen für serverinitiierte Synchronisation“ auf Seite 29

Notifier-Start

Wenn der MobiLink-Server geladen wird, startet er alle aktivierten Notifier. Um einen Notifier zu deaktivieren, müssen Sie den Wert der Notifier-Eigenschaft auf FALSE festlegen. Siehe „Notifier-Eigenschaften“ auf Seite 47.

Zum Starten der Notifier verwenden Sie eine der folgenden Methoden:

- Konfigurieren Sie die Notifier und führen Sie mlsrv16 mit der Option -notifier aus.
- Wenn die Einstellungen in einer Notifier-Konfigurationsdatei gespeichert sind, laden Sie die Datenbank von der Befehlszeile aus, indem Sie mlsrv16 ausführen, und geben Sie die Datei mit der Option -notifier an. Um beispielsweise eine Datei namens *myfirst.Notifier* zu verwenden, konfiguriert der folgende Befehl den MobiLink-Server so, dass er die in der Datei angegebenen Eigenschaften und Ereignisse verwendet:

```
mlsrv16 ... -notifier c:\myfirst.Notifier
```

Siehe auch

- „mlsrv16-Option -notifier “ [*MobiLink - Serveradministration*]
- „MobiLink-Server-Einstellungen für serverinitiierte Synchronisation“ auf Seite 29
- „Konfiguration von Notifier-Ereignissen und -Eigenschaften “ auf Seite 13

Listener

Ein Listener ist ein Programm, das auf einem Gerät ausgeführt wird. Es empfängt Push-Benachrichtigungen von einem Notifier und initiiert Aktionen. Wenn ein Gateway für die serverinitiierte

Synchronisation verwendet wird, können Device Tracking-Informationen von Listenern an die konsolidierte Datenbank übertragen werden.

Siehe auch

- „Device Tracking-Gateways“ auf Seite 23
- „Message-Handler“ auf Seite 15
- „MobiLink Listener-Dienstprogramm für Windows-Geräte (dblsn)“ auf Seite 55

Beispiel

Der folgende Befehl startet das MobiLink Listener-Dienstprogramm für Windows-Geräte:

```
dblsn -v2 -m -ot dblsn.log
    -l "poll_connect='host=localhost';
    poll_notifier=notifier_name1;
    poll_key=sis_user1;
    poll_every=10;
    subject=sync;
    action='start dbmlsync.exe
        -c SERVER=rem1;UID=DBA;PWD=sql
        -ot dbmlsyncOut.txt -qc';"
```

Dieser Befehl lädt einen MobiLink Listener, legt die Ausführlichkeitsstufe 2 fest, aktiviert die Nachrichtenprotokollierung und gibt an, dass sich der Server unter **localhost** befindet. Die Datei *dblsn.log* wird gekürzt, bevor die Ausgabe hineingeschrieben wird. Der MobiLink Listener ruft alle 10 Sekunden Push-Benachrichtigungen ab. Wenn der MobiLink Listener eine Push-Benachrichtigung mit dem Betreff **sync** empfängt, wird die MobiLink-Clientanwendung gestartet.

Weitere Hinweise zu den Befehlszeilenoptionen des MobiLink Listeners für Windows finden Sie unter „MobiLink Listener-Optionen für Windows-Geräte“ auf Seite 57.

Message-Handler

Ein Message-Handler ist eine MobiLink Listener-Komponente, die den Nachrichteninhalt einer Push-Benachrichtigung durchsucht, um eine Aktion zu initiieren. Er kann auch verwendet werden, um Lightweight-Polling-Optionen anzugeben, wie z.B. den Serverstandort und die Polling-Häufigkeit.

Message-Handler haben folgende Komponenten:

- **Filterschlüsselwörter** Nachdem eine Push-Benachrichtigung vorverarbeitet wurde, können Sie Filterschlüsselwörter verwenden, um den Nachrichteninhalt zu durchsuchen. Wenn eine Filterbedingung erfüllt ist, wird eine Aktion initiiert. Sie können das Schlüsselwort **subject** angeben, um eine Nachricht mit einem bestimmten Betreff auszufiltern. Bei der Angabe des Schlüsselworts **sender** werden Nachrichten ausgefiltert, die von einem bestimmten MobiLink-Server empfangen wurden.

- **Aktion** Wenn die Filterbedingungen für eine Nachricht erfüllt sind, wird eine Aktion initiiert. In einer typischen Anwendung geben Sie eine Aktion zum Initiieren der Synchronisation an. Sie können aber auch andere Vorgänge ausführen. Zur Unterstützung bei der Fehlerverarbeitung können Sie eine alternative Aktion festlegen, die ausgeführt werden soll, wenn die ursprüngliche Aktion fehlschlägt.

- **Polling-Einstellungen** Mithilfe von Polling-Einstellungen können Sie konfigurieren, wie der MobiLink Listener den MobiLink-Server nach Push-Benachrichtigungen abfragt.

- **Optionen** Mithilfe von Optionen können Sie entfernte Einstellungen steuern, wie z.B. die Zustellungs- und die Aktionsbestätigung.

Sie können einen Message-Handler mit der dblsn-Option -l erstellen. Es können mehrere Message-Handler angegeben werden.

Siehe auch

- „dblsn-Option -l “ auf Seite 63
- „MobiLink Listener-Schlüsselwörter für Windows-Geräte“ auf Seite 72
- „MobiLink Listener-Aktionsbefehle für Windows-Geräte“ auf Seite 74

Nachrichtenfilter

Beim Empfang einer Push-Benachrichtigung durch einen MobiLink Listener wird die Nachricht vom MobiLink Listener extrahiert und in mehrere Schlüsselwörter unterteilt. Das Schlüsselwort **message** enthält die gesamte unformatierte Nachricht. Die Nachricht wird dann in die Schlüsselwörter **subject**, **content** und **sender** unterteilt. Diese Schlüsselwörter werden über den Nachrichtenfilter ausgeführt, um zu ermitteln, welche Aktionen initiiert werden sollen. Weitere Hinweise zur Verwendung der Schlüsselwörter zum Filtern von Nachrichten finden Sie unter „MobiLink Listener-Schlüsselwörter für Windows-Geräte“ auf Seite 72.

Mithilfe eines Filterschlüsselworts wird ein Teil einer Push-Benachrichtigung mit einer benutzerdefinierten Phrase verglichen. Wenn der Text der beiden Phrasen gleich ist, wird eine Aktion initiiert. Weitere Informationen zur Vorverarbeitung von Push-Benachrichtigungen für die Nachrichtenfilterung finden Sie unter „Nachrichtensyntax“ auf Seite 97.

Filterschlüsselwörter können durch Ausführen des MobiLink Listeners mit folgender Syntax angegeben werden:

```
dblsn ... -l "filter-keyword-name='content to filter';action='...'"
```

Sie können die Option -l mehrfach verwenden, um mehrere Filter zu erstellen, doch Sie müssen für jede Instanz von -l auch eine Aktion festlegen. Aktionen werden nur initiiert, wenn alle Filterbedingungen erfüllt sind.

Die folgenden Schlüsselwörter können nur einmal in einem Message-Handler verwendet werden:

- **content** Dieses Schlüsselwort und das Schlüsselwort subject werden zum Filtern von Nachrichten empfohlen. Verwenden Sie dieses Schlüsselwort, um Nachrichten nach ihrem Inhalt zu filtern. Beispiel:

  ```
  dblsn -l "content='your content filter here';action='...'"
  ```

- **subject** Dieses Schlüsselwort und das Schlüsselwort content werden zum Filtern von Nachrichten empfohlen. Verwenden Sie dieses Schlüsselwort, um Nachrichten nach ihrem Betreff zu filtern. Beispiel:

```
dblsn -l "subject='your subject filter here';action='...'"
```

- **message** Verwenden Sie dieses Schlüsselwort, um Nachrichten nach ihren unformatierten Daten zu filtern. Ihr Filterwert muss genau mit der Länge der Nachricht übereinstimmen. Dieses Schlüsselwort wird nicht empfohlen, da es eine variable Struktur hat. Weitere Informationen zur Vorverarbeitung von Push-Benachrichtigungen für die Nachrichtenfilterung finden Sie unter „Nachrichtensyntax“ auf Seite 97.

- **message_start** Verwenden Sie dieses Schlüsselwort, um Nachrichten nach einem Teil ihrer unformatierten Daten zu filtern, und zwar beginnend am Anfang. Weitere Informationen zur Vorverarbeitung von Push-Benachrichtigungen für die Nachrichtenfilterung finden Sie unter „Nachrichtensyntax“ auf Seite 97.

  Wenn Sie dieses Schlüsselwort angeben, erstellt der MobiLink Listener die Aktionsvariablen $message_start und $message_end.

- **sender** Verwenden Sie dieses Schlüsselwort, um Nachrichten nach ihrem Absender zu filtern. Dieses Schlüsselwort ist nützlich zum Protokollieren von Push-Benachrichtigungen, die von einem bestimmten Notifier gesendet wurden. Der Wert hängt von dem verwendeten Gateway ab. Bei UDP-Gateways ist er die IP-Adresse des Gateway-Hosts. Bei SYNC-Gateways ist er **MobiLink**. Bei SMTP-Gateways hängt der Wert vom Mobilfunkanbieter ab. Siehe „Gateways und Netzbetreiber“ auf Seite 22.

Siehe auch

- „MobiLink Listener-Schlüsselwörter für Windows-Geräte“ auf Seite 72
- „Aktionsvariablen“ auf Seite 18

Initiierung von Aktionen

Wenn eine Nachricht die Bedingungen eines Filters erfüllt, wird eine Aktion initiiert. Weitere Informationen zum Filtern von Push-Benachrichtigungen finden Sie unter „Nachrichtenfilter“ auf Seite 16.

Aktionen können durch Ausführen des MobiLink Listeners mit folgender Syntax festgelegt werden:

```
dblsn ... -l "...;action='action-command command statement'"
```

Folgende Aktionsbefehle gestatten es, verschiedene Aufgaben auszuführen, wenn eine Nachricht gefiltert wird:

- **START** Eine Anwendung starten und im Hintergrund ausführen.

- **RUN** Eine Anwendung ausführen und warten, bis sie abgeschlossen wurde, bevor weitere Push-Benachrichtigungen empfangen werden.

- **POST** Eine Fensternachricht an einen Prozess senden, der bereits ausgeführt wird. Dieser Befehl kann nur auf Windows-Geräten verwendet werden.

- **SOCKET** Eine Nachricht über eine TCP/IP-Verbindung an eine Anwendung senden.

- **DBLSN FULL SHUTDOWN** Den MobiLink Listener herunterfahren.

Siehe auch

- „MobiLink Listener-Aktionsbefehle für Windows-Geräte“ auf Seite 74
- „Aktionsvariablen“ auf Seite 18

Aktionsvariablen

Aktionsvariablen gestatten es Ihnen, Teile einer Push-Benachrichtigung von einem Nachrichtenfilter oder einer Aktion aus zu referenzieren. Siehe „Initiierung von Aktionen“ auf Seite 17 und „Nachrichtenfilter“ auf Seite 16.

Aktionsvariablen festlegen

Die meisten Aktionsvariablen werden bei jedem Empfang einer Push-Benachrichtigung automatisch festgelegt. Die Variablennamen sind ähnlich den in der Nachrichtensyntax angegebenen Namen. Beispiel: *message* legt die Aktionsvariable $message fest, *subject* die Aktionsvariable $subject, *sender* die Aktionsvariable $sender und *content* die Aktionsvariable $content. Siehe „Nachrichtensyntax“ auf Seite 97.

Aktionsvariablen verwenden

Aktionsvariablen werden beim Aufruf des MobiLink Listeners in der Befehlszeile verwendet. Wie sie verwendet werden, hängt vom Message-Handler und der Aktion ab, die initiiert werden soll. Das folgende Beispiel zeigt die Verwendung des Aktionsbefehls RUN, mit dem die MobiLink-Clientanwendung initiiert wird.

```
dblsn ... -l "subject=publish;action='RUN dbmlsync.exe @dbmlsync.txt -n
$content'"
```

Dieser Message-Handler filtert Nachrichten aus, bei denen der Betreff dem Text "publish" entspricht. Nach dem Filtern wird dbmlsync mit der Option -n ausgeführt und die Aktionsvariable $content wird als Parameter übergeben. Unter der Annahme, dass *content* den Namen einer Synchronisationspublikation referenziert, verwendet dbmlsync die Publikation, um die Gerätedatenbank mit der konsolidierten Datenbank zu synchronisieren.

Das folgende Beispiel zeigt die Verwendung einer Aktionsvariablen zum Filtern einer Nachricht:

```
dblsn ... -l "subject=$content;action='RUN script.bat"
```

Dieser Message-Handler filtert Nachrichten, wenn **subject** dem Text von **content** entspricht. Nach dem Filtern führt das Gerät ein benutzerdefiniertes Batchskript aus.

Siehe auch

- „MobiLink Listener-Aktionsbefehle für Windows-Geräte“ auf Seite 74
- „MobiLink Listener-Aktionsvariablen für Windows-Geräte“ auf Seite 78
- „Entfernte ID als Filter“ auf Seite 19

Lightweight-Polling-Option einrichten

Mit Message-Handlern können Sie Polling-Vorgänge verarbeiten. Mit den Lightweight-Polling-Optionen können Sie den Standort des Servers, den Notifier-Namen, die Polling-Häufigkeit und einen Polling-Schlüssel angeben. Alternativ dazu können Sie die Lightweight-Polling-API verwenden, um diese Eigenschaften festzulegen.

Lightweight-Polling-Optionen können durch das Ausführen des MobiLink Listeners mit folgender Syntax festgelegt werden:

```
dblsn ... -l
    "poll_connect=protocol-options;
    poll_notifier=Notifier-name;
    poll_key=identifier-string;
    poll_every=number-of-seconds;..."
```

Ein einzelner Message-Handler kann nur eine der folgenden Optionen enthalten:

- **poll_connect** Verwenden Sie diese Option, um die Protokolloptionen festzulegen, die erforderlich sind, um eine Verbindung zum Server herzustellen. Alternativ dazu können Sie mit der dblsn-Option -x die Standardprotokolloptionen festlegen. Die Option poll_connect setzt die Standardprotokolloptionen für den Message-Handler außer Kraft.
- **poll_notifier** Verwenden Sie diese Option, um den Notifier anzugeben, der vom MobiLink-Server zur Verarbeitung von Push-Anforderungen verwendet wird. Diese Option ist erforderlich, wenn der MobiLink-Server mehrere Notifier hosten kann.
- **poll_key** Verwenden Sie diese Option, um den MobiLink Listener beim Notifier zu identifizieren. Der MobiLink-Server verwendet diesen Wert, um Push-Benachrichtigungen für das Gerät zu senden. In einer typischen Anwendung sollte dieser Wert die entfernte ID des Geräts sein.
- **poll_every** Mit dieser Option geben Sie an, wie häufig der MobiLink Listener den Notifier abrufen soll. Standardmäßig ruft der MobiLink Listener diesen Wert automatisch vom MobiLink-Server ab. Dieser Wert wird in Sekunden gemessen.

Siehe auch

- „Voraussetzungen für Push-Anforderungen“ auf Seite 7
- „Konfiguration von Notifier-Ereignissen und -Eigenschaften “ auf Seite 13
- „Lightweight-Polling-Module“ auf Seite 21
- „MobiLink Listener-Schlüsselwörter für Windows-Geräte“ auf Seite 72
- „Lightweight-Polling-API“ auf Seite 83

Erweiterte Message-Handler-Funktionen

Entfernte ID als Filter

Sie können Nachrichten mit der dblsn-Option -r und der Aktionsvariablen $remote_id nach der entfernten ID filtern.

Wenn eine entfernte SQL Anywhere-Datenbank erstmals synchronisiert wird, wird eine Datei mit der ID der Datenbank erstellt. Der Name der Datei entspricht dem der Datenbank. Die Datei hat die

Erweiterung *.rid* und wird in demselben Verzeichnis gespeichert wie die Datenbank. Für UltraLite-Datenbanken gibt es keine solche Datei. Die entfernte ID wird aus der Datenbank direkt extrahiert.

Starten Sie den MobiLink Listener mit der dblsn-Option -r, um den Namen und den Speicherort der Datei mit der entfernten ID oder die UltraLite-Datenbank anzugeben, und verwenden Sie dann die dblsn-Option -l, um den Message-Handler zu erstellen.

Sie können die entfernte ID direkt in den Nachrichtenfilter eingeben. Entfernte IDs sind standardmäßig GUIDs. Um sich die entfernte ID einfach merken zu können, sollte ein aussagekräftiger Name angegeben werden.

Hinweis
In der dblsn-Befehlszeile können Sie mehrere Instanzen der Optionen -r und -l angeben. Die in der Option -l verwendete Aktionsvariable $remote_id wird immer in der Option -r angegeben, die ihr vorangeht. Daher ist es wichtig, die Option -r vor der Option -l anzugeben.

Das folgende Beispiel zeigt die Verwendung mehrerer entfernter IDs. Es wird vorausgesetzt, dass auf dem Gerät eine SQL Anywhere-Datenbank mit dem Namen *business.db* und eine UltraLite-Datenbank mit dem Namen *personal.udb* vorhanden sind. In diesem Beispiel ist **ulpersonal** der Fensterklassenname der UltraLite-Anwendung.

```
dblsn ... -r "c:\app\db\business.rid"
    -l "subject=$remote_id;action='dbmlsync.exe -k -c dsn=business';"
    -r "c:\ulapp\personal.udb"
    -l "subject=$remote_id;action=post dbas_synchronize to ulpersonal;"
```

Siehe auch

- „Aktionsvariablen“ auf Seite 18
- „Entfernte IDs“ [*MobiLink - Clientadministration*]
- „dblsn-Option -r“ auf Seite 67
- „MobiLink Listener-Aktionsvariablen für Windows-Geräte“ auf Seite 78

Verbindungsinitiierte Synchronisation

Auf Windows-Geräten können Sie die Synchronisation initiieren, wenn sich die Netzwerkverbindung ändert.

Wenn die IP-Netzwerkverbindung hergestellt oder abgebrochen wird, sendet das Gerät eine Push-Benachrichtigung mit der Nachricht **_IP_CHANGED_** an den MobiLink Listener. Wenn das Gerät einen neuen optimalen Pfad zum MobiLink-Server findet, sendet es eine Push-Benachrichtigung mit der Nachricht **_BEST_IP_CHANGED_** an den MobiLink Listener. Mit einem Message-Handler können Sie diese Änderungen der Netzwerkverbindung erkennen und eine Aktion initiieren.

Änderungen der Netzwerkverbindung erkennen

Die Nachricht **_IP_CHANGED_** zeigt an, dass eine Änderung der IP-Netzwerkverbindung aufgetreten ist. Eine Änderung tritt normalerweise ein, wenn ein Gerät in die Reichweite eines WiFi-Netzwerks kommt, wenn der Benutzer eine RAS-Verbindung herstellt oder wenn er das Gerät in eine Dockingstation stellt. Sie können die Nachricht **_IP_CHANGED_** referenzieren, indem Sie den MobiLink Listener mit folgender Syntax ausführen:

```
dblsn ... -l "message=_IP_CHANGED_;action='...'"
```

Das folgende Beispiel zeigt die Verwendung der Nachricht **_IP_CHANGED_**. Der Message-Handler filtert die Nachricht und sendet sie an den Server. Bei einer Trennung der Verbindung wird ein Fehler generiert.

```
dblsn -l "message=_IP_CHANGED_;
    action='
        SOCKET port=12345;
        sendText=IP changed: $adapters|$network_names;
        recvText=beeperAck;
        timeout=5';
    continue=yes;"
```

Änderung des optimalen Pfads zu einem MobiLink-Server erkennen

Die Nachricht **_BEST_IP_CHANGED_** zeigt an, dass sich der optimale Pfad zum MobiLink-Server geändert hat. Sie können diese Nachricht referenzieren, wenn Sie den MobiLink Listener mit folgender Syntax ausführen:

```
dblsn ... -x MobiLink-protocol-options -l
"message=_BEST_IP_CHANGED_;action='...'"
```

Beim Filtern der Nachricht **_BEST_IP_CHANGED_** können Sie mithilfe der Aktionsvariablen $best_ip nützliche Aktionen initiieren. $best_ip ersetzt die IP-Adresse, die die vorherige beste IP-Verbindung darstellt. Wenn keine IP-Verbindung vorhanden ist, gibt $best_ip den Wert 0.0.0.0 zurück.

Im folgenden Beispiel wird die Nachricht **_BEST_IP_CHANGED_** verwendet, um eine Synchronisation zu initiieren, wenn die beste IP-Verbindung sich ändert. Bei einer Trennung der Verbindung wird ein Fehler generiert.

```
dblsn -x http(host=mlserver.company.com)
   -v2 -m -i 3 -ot dblsn.log
   -l "message=_BEST_IP_CHANGED_;
       action='
           START dbmlsync.exe -ra -c SERVER=remote;UID=DBA;PWD=sql -n
test_pub'"
```

Hinweis
Führen Sie den MobiLink Listener auf einem anderen Computer als dem MobiLink-Server aus, wenn Sie die verbindungsinitiierte Synchronisation mit Ihren Anwendungen testen.

Siehe auch

- „MobiLink Listener-Dienstprogramm für Windows-Geräte (dblsn)“ auf Seite 55
- „MobiLink Listener-Aktionsbefehle für Windows-Geräte“ auf Seite 74
- „MobiLink Listener-Aktionsvariablen für Windows-Geräte“ auf Seite 78

Lightweight-Polling-Module

Ein Lightweight-Polling-Modul ist eine Geräteanwendung, die Push-Benachrichtigungen in einer angegebenen Zeitspanne abruft. Die Verwendung eines Lightweight-Polling-Moduls ist eine Alternative zum Einrichten eines Gateways. Sie wird empfohlen, da hierbei anders als beim SYNC-Gateway keine

beständige Verbindung zum Server erforderlich ist und anders als beim UDP-Gateway auch keine kontinuierliche Verbindung.

Wenn ein Gerät den Server abruft, sendet es einen Polling-Schlüssel und einen Notifier-Namen. Der MobiLink-Server prüft den Notifier-Namen, um zu ermitteln, welcher Notifier den Cache auf Push-Anforderungen prüfen soll. Der Polling-Schlüssel identifiziert das Gerät beim Notifier, der ihn verwendet, um an das Gerät gerichtete Push-Anforderungen zu erkennen. Nachdem die Push-Anforderungen erkannt wurden, werden Push-Benachrichtigungen gesendet.

Mit MobiLink Listener-Befehlszeilenoptionen können Sie ein Lightweight-Polling-Modul konfigurieren. Alternativ dazu können Sie die Lightweight-Polling-API verwenden, um ein Lightweight-Polling-Modul in die Geräteanwendung zu integrieren.

Hinweis
Sie können mit Sybase Central entfernte Datenbanken verwalten und anschließend eine serverinitiierte entfernte Aufgabe (SIRT) verwenden, um eine Push-Benachrichtigung zu implementieren. Weitere Hinweise finden Sie unter „Zentrale Administration von entfernten Datenbanken“ [*MobiLink - Serveradministration*] und „Serverinitiierte entfernte Aufgaben (SIRT)“ [*MobiLink - Serveradministration*].

Siehe auch

- „Lightweight-Polling-API“ auf Seite 83
- „Lightweight-Polling-Option einrichten“ auf Seite 19
- „MobiLink Listener-Schlüsselwörter für Windows-Geräte“ auf Seite 72

Gateways und Netzbetreiber

Gateways als Alternative zu Lightweight-Polling-Modulen

Ein Gateway ist ein MobiLink-Objekt, das in MobiLink-Systemtabellen oder einer Notifier-Eigenschaftsdatei gespeichert ist und Informationen für die Art der Versendung von Nachrichten für die serverinitiierte Synchronisation enthält. Sie sind eine Alternative zu Lightweight-Polling-Modulen und erfordern eine kontinuierliche Netzwerkverbindung.

Eigenschaften von Gateways werden auf einem MobiLink-Server konfiguriert. Sie können mehrere Gateways auf einem einzelnen MobiLink-Server konfigurieren.

Unterstützte Gateways

Folgende Gateways werden auf dem MobiLink-Server unterstützt:

- **SYNC-Gateway** Das SYNC-Gateway ist ein TCP/IP-basiertes Gateway. Push-Benachrichtigungen werden über dasselbe Protokoll wie die MobiLink-Synchronisationen gesendet.

  Das standardmäßige SYNC-Gateway heißt Default-SYNC. Gewöhnlich müssen die Standard-Gateway-Einstellungen nicht geändert werden.

- **UDP-Gateway** Das UDP-Gateway sendet Push-Benachrichtigungen über ein UDP-Gateway.

Das Standard-UDP-Gateway heißt Default-UDP. Gewöhnlich müssen die Standard-Gateway-Einstellungen nicht geändert werden. MobiLink Listener verwenden UDP standardmäßig beim Warten auf Push-Benachrichtigungen.

- **SMTP-Gateway** Das SMTP-Gateway sendet Push-Benachrichtigungen mithilfe eines E-Mail-zu-SMS-Diensts eines Netzbetreibers.

  Das Standard-SMTP-Gateway heißt Default-SMTP.

Device Tracking-Gateway

Zusätzlich zu den unterstützten Gateways können Sie ein Device Tracking-Gateway für die Geräteprotokollierung konfigurieren, das automatisch das am besten geeignete Gateway zum Senden von Push-Benachrichtigungen auswählt. Das standardmäßige Device Tracking-Gateway ist Default-DeviceTracker. Es wird empfohlen, dieses Gateway zu verwenden, wenn Sie kein Lightweight-Polling-Modul verwenden wollen.

Siehe auch

- „MobiLink-Server-Einstellungen für serverinitiierte Synchronisation“ auf Seite 29
- „SYNC-Gateway-Eigenschaften“ auf Seite 52
- „UDP-Gateway-Eigenschaften“ auf Seite 52
- „SMTP-Gateway-Eigenschaften“ auf Seite 50
- „Device Tracking-Gateways“ auf Seite 23

Device Tracking-Gateways

Das Device Tracking (Geräteprotokollierung) gestattet es einem MobiLink-Server, Geräte unter Verwendung der entfernten ID einer Push-Anforderung zu verfolgen. Ein Device Tracking-Gateway verwendet automatisch protokollierte IP-Adressen, Telefonnummern und IDs öffentlicher Mobilfunkanbieter, um Push-Benachrichtigungen über SYNC-, UDP- und SMTP-Gateways bereitzustellen. Das Gateway versucht zunächst, das Gerät mithilfe eines SYNC-Gateways zu verbinden. Falls die Zustellung fehlschlägt, wird ein UDP-Gateway verwendet, gefolgt von einem SMTP-Gateway. Diese Funktion ist nützlich, wenn Sie erwarten, dass die Geräteadresse sich ändert.

Ein Device Tracking-Gateway kann maximal drei untergeordnete Gateways haben: ein SYNC-, ein SMTP- und ein UDP-Gateway. Push-Benachrichtigungen werden basierend auf den vom MobiLink Listener gesendeten Device Tracking-Informationen automatisch an eines der untergeordneten Gateways weitergeleitet. Durch das Aktivieren dieser untergeordneten Gateways werden Änderungen der Geräteadresse automatisch vom MobiLink-Server verwaltet. Wenn sich eine Adresse ändert, wird der MobiLink Listener mit der konsolidierten Datenbank synchronisiert, um die Protokolldaten, die sich in der Systemtabelle ml_device_address befinden, zu aktualisieren.

Die meisten MobiLink Listener der Version 9.0.1 oder später unterstützen Device Tracking. Wenn Sie einen MobiLink Listener verwenden, der kein Device Tracking unterstützt, können Sie ein Device Tracking-Gateway verwenden, indem Sie die Protokollinformationen bereitstellen.

Siehe auch

- „Device Tracking für MobiLink Listener 9.0.0 einrichten“ auf Seite 24
- „Gateways als Alternative zu Lightweight-Polling-Modulen“ auf Seite 22
- „Netzbetreiber und Netzbetreiberkonfiguration“ auf Seite 27

Device Tracking für MobiLink Listener 9.0.0 einrichten

Es sind mehrere Systemprozeduren verfügbar, um das Device Tracking für Mobi Link Listener 9.0.0 manuell einzurichten. Diese Prozeduren aktualisieren die MobiLink-Systemtabellen ml_device, ml_device_address und ml_listening in der konsolidierten Datenbank.

Voraussetzungen

Es gibt keine Voraussetzungen für das Ausführen dieser Aufgabe.

Kontext und Bemerkungen

Sie müssen Device Tracking nur unterstützen, wenn Sie MobiLink Listener verwenden, die unter SQL Anywhere 9.0.0 oder früher ausgeführt werden. Device Tracking wird bereits auf allen anderen MobiLink Listenern für Windows-Geräte unterstützt.

Mithilfe von manuellem Device Tracking können Sie Empfänger anhand des MobiLink-Benutzernamens adressieren, ohne Netzwerkadressinformationen bereitzustellen. Die Informationen können jedoch nicht automatisch von MobiLink aktualisiert werden, wenn sie sich ändern. Dies muss manuell durchgeführt werden. Diese Methode ist insbesondere für SMTP-Gateways nützlich, da sich E-Mail-Adressen nur selten ändern.

Bei UDP-Gateways können Sie nicht auf statische Eingaben zurückgreifen, wenn sich Ihre IP-Adresse bei jeder Verbindung ändert. Sie können dieses Problem lösen, indem Sie den Hostnamen anstatt der IP-Adresse adressieren. Diese Lösung verlangsamt jedoch Aktualisierungen von DNS-Servertabellen und kann Push-Benachrichtigungen fehlleiten. Sie können auch Systemprozeduren einrichten, um die Systemtabellen programmgesteuert zu aktualisieren.

Aufgabe

1. Fügen Sie für jedes Gerät einen Gerätedatensatz in die Systemtabelle ml_device ein. Beispiel:

```
CALL ml_set_device(
    'myWindowsMobile',
    'MobiLink Listeners for myWindowsMobile - 9.0.1',
    '1',
    'not used',
    'y',
    'manually entered by administrator'
);
```

   Der erste Parameter, **myWindowsMobile**, ist ein eindeutiger benutzerdefinierter Gerätename. Der zweite Parameter enthält optionale Angaben zur MobiLink Listener-Version. Der dritte Parameter gibt eine MobiLink Listener-Version an. Verwenden Sie **0** für SQL Anywhere 9.0.0 MobiLink Listener oder **2** für MobiLink Listener für Windows in späteren Versionen als 9.0.0. Der vierte Parameter gibt

optionale Geräteinformationen an. Der fünfte Parameter gibt an, ob das Device Tracking ignoriert werden soll. Der letzte Parameter enthält optionale Anmerkungen zu diesem Eintrag.

2. Fügen Sie für jedes Gerät einen Adressdatensatz in die Systemtabelle ml_device_address ein. Beispiel:

```
CALL ml_set_device_address(
    'myWindowsMobile',
    'ROGERS AT&T',
    '55511234567',
    'y',
    'y',
    'manually entered by administrator'
);
```

   Der erste Parameter, **myWindowsMobile**, ist ein benutzerdefinierter eindeutiger Gerätename. Der zweite Parameter ist eine Netzwerk-Provider-ID und muss mit der Eigenschaft network_provider_id des Netzbetreibers übereinstimmen. Der dritte Parameter ist eine IP-Adresse für UDP. Der vierte Parameter legt fest, ob dieser Eintrag zum Senden von Push-Benachrichtigungen aktiviert werden soll. Der fünfte Parameter gibt an, ob das Device Tracking ignoriert werden soll. Der letzte Parameter enthält optionale Anmerkungen zu diesem Eintrag.

3. Fügen Sie für jede entfernte Datenbank in die Systemtabelle ml_listening einen Empfängerdatensatz für jedes hinzugefügte Gerät ein. Dadurch wird das Gerät dem MobiLink-Benutzernamen zugeordnet. Beispiel:

```
CALL ml_set_listening(
    'myULDB',
    'myWindowsMobile',
    'y',
    'y',
    'manually entered by administrator'
);
```

   Der erste Parameter ist ein MobiLink-Benutzername. Der zweite Parameter ist ein benutzerdefinierter eindeutiger Gerätename. Der dritte Parameter legt fest, ob dieser Eintrag für die Device Tracking-Adressierung aktiviert werden soll. Der vierte Parameter gibt an, ob Device Tracking ignoriert werden soll. Der letzte Parameter enthält optionale Anmerkungen zu diesem Eintrag.

Ergebnisse

Die angegebenen Geräte sind für Device Tracking eingerichtet.

Siehe auch

- „ml_set_device-Systemprozedur“ auf Seite 91
- „ml_set_listening-Systemprozedur“ auf Seite 93
- „ml_set_device_address-Systemprozedur“ auf Seite 92
- „Device Tracking-Gateways“ auf Seite 23
- „Netzbetreibereigenschaften“ auf Seite 53

Schnellstart für die Konfiguration des Device Tracking-Gateways

Die folgende Prozedur gibt einen Überblick darüber, wie Sie Device Tracking-Gateways konfigurieren können.

1. Richten Sie ein SYNC-, UDP- oder SMTP-Gateway ein.

   Wenn Sie den MobiLink-Server starten, sind diese Gateways bereits mit den Standardeinstellungen eingerichtet. Weitere Hinweise zur Konfiguration von Eigenschaften oder zur Erstellung eigener Gateways finden Sie unter „MobiLink-Server-Einstellungen für serverinitiierte Synchronisation“ auf Seite 29.

   Hinweis
   Das SMTP-Gateway erfordert eine Netzbetreiberkonfiguration. Siehe „Netzbetreiber und Netzbetreiberkonfiguration“ auf Seite 27.

2. Erstellen Sie einen neuen Notifier und richten Sie das request_cursor-Ereignis unter Berücksichtigung folgender Bedingungen ein:

   - Der Gateway-Name muss auf den Namen eines Device Tracking-Gateways festgelegt sein, das verwendet werden soll. Das Standard-Gateway heißt Default-DeviceTracker. Dieser Name wird in der ersten Spalte der Ergebnismenge angegeben.
   - Der Adressname muss auf die entfernte ID des Geräts gesetzt werden. Verwenden Sie die dblsn-Option -t+, um die entfernte ID beim MobiLink-Server zu registrieren. Dieser Name wird in der vierten Spalte der Ergebnismenge angegeben.

   Weitere Hinweise zum Einrichten eines request_cursor-Ereignisses finden Sie unter „request_cursor-Ereignis“ auf Seite 40.

3. Fügen Sie den MobiLink Listener-Namen zur Systemtabelle ml_user hinzu.

   Der Standardname des MobiLink Listeners lautet *device_name*-**dblsn**. Dabei gilt: *device_name* ist der Name Ihres Geräts.

   Führen Sie den MobiLink Listener aus, um den Gerätenamen im MobiLink Listener-Meldungsfenster anzuzeigen. Alternativ können Sie den Gerätenamen mithilfe der dblsn-Option -e festlegen oder Sie legen einen anderen MobiLink Listener-Namen mit der dblsn-Option -u fest. Siehe „dblsn-Option -e“ auf Seite 61 und „dblsn-Option -u “ auf Seite 69.

   Weitere Hinweise zum Registrieren von MobiLink-Benutzern finden Sie unter „Erstellen und Registrieren von MobiLink-Benutzern“ [*MobiLink - Clientadministration*].

4. Starten Sie einen MobiLink Listener mit den erforderlichen Optionen. Weitere Hinweise zum Starten eines MobiLink Listeners finden Sie unter „MobiLink Listener-Dienstprogramm für Windows-Geräte (dblsn)“ auf Seite 55.

Netzbetreiber und Netzbetreiberkonfiguration

Ein Netzbetreiber ist ein MobiLink-Objekt, das in MobiLink-Systemtabellen oder einer Notifier-Eigenschaftsdatei gespeichert ist und Informationen über einen öffentlichen Netzbetreiber für die serverinitiierte Synchronisation enthält.

Sie müssen einen Mobilfunkanbieter konfigurieren, um Push-Benachrichtigungen über ein SMTP-Gateway zu senden, da der Notifier gültige E-Mail-Adressen konstruieren muss. Wenn Sie ein Device Tracking-Gateway mit einem aktivierten untergeordneten Gateway verwenden, müssen Sie ebenfalls einen Mobilfunkanbieter konfigurieren.

Eigenschaften des Netzbetreibers, wie z.B. die Netzwerk-Provider-ID und das SMS-E-Mail-Präfix, werden auf einem MobiLink-Server konfiguriert. Um mehrere Netzbetreiber-Dienste zu verwenden, konfigurieren Sie mehrere Netzbetreiber auf dem MobiLink-Server.

Absendersyntax

Wenn eine Push-Benachrichtigung von einem MobiLink Listener empfangen und für die Nachrichtenfilterung vorverarbeitet wird, wird sie in mehrere Schlüsselwörter unterteilt. Das Schlüsselwort **sender** in einer **message** ist eine E-Mail-Adresse, die vom Gerät generiert wird und abhängig vom Mobilfunkanbieter variiert. Weitere Hinweise zur Vorverarbeitung von Nachrichten finden Sie unter „Nachrichtensyntax“ auf Seite 97.

Die **sender**-Syntax hat folgendes Format:

```
sender = sms_email_user_prefix phone-number@sms_email_domain
```

Hinweis
Zwischen *sms_email_user_prefix* und *phone-number* befinden sich keine Leerzeichen.

Die Werte *sms_email_user_prefix* und *sms_email_domain* sind Eigenschaften des Netzbetreibers, die auf dem MobiLink-Server konfiguriert werden müssen. Der Wert *phone-number* wird der Adressspalte der Systemtabelle ml_device_address entnommen.

Um die Absendersyntax zu ermitteln, führen Sie den MobiLink Listener auf einem Gerät aus, das einen Netzbetreiberdienst verwendet. Aktivieren Sie die Nachrichtenprotokollierung und legen Sie die Ausführlichkeitsstufe 2 fest, indem Sie die dblsn-Optionen -m und -v verwenden. Prüfen Sie das Nachrichtenlog nach dem Laden des MobiLink Listeners.

Siehe auch

- „MobiLink-Server-Einstellungen für serverinitiierte Synchronisation“ auf Seite 29
- „Netzbetreibereigenschaften“ auf Seite 53

MobiLink-Server-Einstellungen für serverinitiierte Synchronisation

Serverseitige Einstellungen umfassen Notifier-, Gateway- und Netzbetreiber-Eigenschaften sowie Notifier-Ereignisse. Zur Konfiguration dieser Einstellungen verwenden Sie eine der folgenden Methoden:

- Sybase Central
- Notifier-Konfigurationsdatei
- ml_add_property-Systemprozedur

Die Methoden von Sybase Central und der ml_add_property-Systemprozedur fügen der Systemtabelle ml_property Ereignisse und Einstellungen hinzu.

Hinweis
Änderungen der serverseitigen Einstellungen werden nicht wirksam, solange der MobiLink-Server läuft. Um neue Einstellungen zu übernehmen, müssen Sie den MobiLink-Server herunterfahren und neu starten.

Wenn Sie in der Systemtabelle ml_property bereits serverseitige Einstellungen konfiguriert haben und eine Notifier-Konfigurationsdatei verwenden wollen, werden die Einstellungen der Systemtabelle immer zuerst geladen, gefolgt von den Dateieinstellungen. Die Notifier-Konfigurationsdatei überschreibt vorhandene serverseitige Einstellungen. Die Einstellungen werden jedoch nicht dauerhaft für die konsolidierte Datenbank übernommen.

Serverseitige Einstellungen, die mit der ml_add_property-Systemprozedur konfiguriert wurden

Mit der ml_add_property-Systemprozedur konfigurieren Sie die serverseitigen Einstellungen einer konsolidierten SQL Anywhere-Datenbank. Sie können diese Eigenschaften und Ereignisse mithilfe von Interactive SQL einstellen.

Hinweis
Sie müssen den ANSI-Standard verwenden, wenn Sie Namen für Ihre Notifier, Gateways und Netzbetreiber festlegen.

Syntax allgemeiner Eigenschaften

```
CALL ml_add_property('SIS', '', 'Property', Value);
```

Syntax von Eigenschaften und Ereignissen des Notifiers

```
CALL ml_add_property('SIS', 'Notifier(NotifierName)', 'Event-or-Property',
Value);
```

Syntax von Gateway-Eigenschaften

```
CALL ml_add_property('SIS', 'DeviceTracker(DeviceTrackerName)', 'Property',
Value);

CALL ml_add_property('SIS', 'SMTP(SMTPName)', 'Property', Value);

CALL ml_add_property('SIS', 'UDP(UDPName)', 'Property', Value);

CALL ml_add_property('SIS', 'SYNC(SYNCName)', 'Property', Value);
```

Syntax von Netzbetreiber-Eigenschaften

```
CALL ml_add_property('SIS', 'Carrier(CarrierName)', 'Property', Value);
```

Siehe auch

- „ml_add_property-Systemprozedur“ [*MobiLink - Serveradministration*]

Notifier, Gateways oder Netzbetreiber mit Sybase Central einrichten

Sybase Central stellt eine grafische Benutzeroberfläche zum Ändern von Eigenschaften und Ereignissen bereit. Sie können mithilfe von Sybase Central mehrere Notifier, Gateways und Netzbetreiber konfigurieren.

Voraussetzungen

Es gibt keine Voraussetzungen für das Ausführen dieser Aufgabe.

Kontext und Bemerkungen

Sie müssen den ANSI-Standard verwenden, wenn Sie Namen für Ihre Notifier, Gateways und Netzbetreiber festlegen.

Wenn Sie die serverseitigen Einstellungen über Sybase Central konfigurieren, brauchen Sie bei der Verwendung der mlsrv16-Option -notifier in der Befehlszeile keine Notifier-Konfigurationsdatei anzugeben.

Aufgabe

1. Verwenden Sie in Sybase Central das MobiLink-Plug-In zum Erstellen eines MobiLink-Projekts für Ihre konsolidierte Datenbank, sofern Sie nicht bereits eins erstellt haben.

   Siehe „Erstellen eines MobiLink-Projekts“ [*MobiLink - Erste Orientierung*].

2. Klicken Sie auf **Ansicht** » **Ordner**.

3. Erweitern Sie im linken Fensterausschnitt von **MobiLink 16** Ihren MobiLink-Projektnamen, **Konsolidierte Datenbanken** und den Namen Ihrer konsolidierten Datenbank. Wählen Sie anschließend **Benachrichtigung** aus.

Im rechten Fensterausschnitt werden alle verfügbaren Notifier, Gateways und Netzbetreiber angezeigt.

4. Erstellen Sie neue Notifier, Gateways und Netzbetreiber.

   - Um einen neuen Notifier zu erstellen, klicken Sie im rechten Fensterausschnitt auf die Registerkarte **Notifier** und klicken Sie dann auf **Datei** » **Neu** » **Notifier**.
   - Zum Erstellen eines neuen Gateways klicken Sie im rechten Fensterausschnitt auf die Registerkarte **Gateways** und klicken Sie dann auf **Datei** » **Neu** » **Gateway**.
   - Zum Erstellen eines neuen Netzbetreibers klicken Sie im rechten Fensterausschnitt auf die Registerkarte **Netzbetreiber** und klicken Sie dann auf **Datei** » **Neu** » **Netzbetreiber**.

5. Wählen Sie einen Notifier, ein Gateway oder einen Netzbetreiber für die Konfiguration aus.

   - Zum Einrichten von Eigenschaften oder Ereignissen des Notifiers klicken Sie im rechten Fensterausschnitt auf die Registerkarte **Notifier** und wählen Sie dann den zu konfigurierenden Notifier.
   - Zum Einrichten von Eigenschaften des Gateways klicken Sie im rechten Fensterausschnitt auf die Registerkarte **Gateways** und wählen Sie dann das zu konfigurierende Gateway.
   - Zum Einrichten von Eigenschaften des Netzbetreibers klicken Sie im rechten Fensterausschnitt auf die Registerkarte **Netzbetreiber** und wählen Sie dann den zu konfigurierenden Netzbetreiber.

   Klicken Sie auf **Datei** » **Eigenschaften**.

   Es erscheint ein Fenster, in dem Sie alle zutreffenden Einstellungen für den gewählten Notifier, das gewählte Gateway oder den gewählten Netzbetreiber anpassen können.

6. Klicken Sie auf **OK**.

Ergebnisse

Der Notifier, das Gateway oder der Netzbetreiber ist eingerichtet und zum Einsatz bereit.

Serverseitige Einstellungen aus einer Notifier-Konfigurationsdatei importieren

Verwenden Sie eine Notifier-Konfigurationsdatei, um serverseitige Einstellungen in die Tabelle ml_property zu importieren.

Voraussetzungen

Es gibt keine Voraussetzungen für das Ausführen dieser Aufgabe.

Aufgabe

1. Verwenden Sie in Sybase Central das MobiLink-Plug-In zum Erstellen eines MobiLink-Projekts für Ihre konsolidierte Datenbank, sofern Sie nicht bereits eins erstellt haben.

Siehe „Erstellen eines MobiLink-Projekts“ [*MobiLink - Erste Orientierung*].

2. Klicken Sie auf **Ansicht** » **Ordner**.

3. Erweitern Sie im linken Fensterausschnitt von **MobiLink 16** Ihren MobiLink-Projektnamen, **Konsolidierte Datenbanken** und den Namen Ihrer konsolidierten Datenbank. Wählen Sie anschließend **Benachrichtigung** aus.

4. Klicken Sie auf **Datei** » **Einstellungen importieren** und befolgen Sie die Anweisungen des Assistenten.

Ergebnisse

Die Einstellungen werden aus der Notifier-Konfigurationsdatei in die Tabelle ml_property importiert.

Serverseitige Einstellungen in eine Notifier-Konfigurationsdatei exportieren

Sie können die serverseitigen Einstellungen aus der Tabelle ml_property in eine Notifier-Konfigurationsdatei exportieren. Durch das Exportieren der Einstellungen können Sie mehrere Versionen der serverseitigen Einstellungen erstellen. Mit der mlsrv16-Option -notifier können Sie eine andere Version laden.

Voraussetzungen

Es gibt keine Voraussetzungen für das Ausführen dieser Aufgabe.

Aufgabe

1. Verwenden Sie in Sybase Central das MobiLink-Plug-In zum Erstellen eines MobiLink-Projekts für Ihre konsolidierte Datenbank, sofern Sie nicht bereits eins erstellt haben.

   Siehe „Erstellen eines MobiLink-Projekts“ [*MobiLink - Erste Orientierung*].

2. Klicken Sie auf **Ansicht** » **Ordner**.

3. Erweitern Sie im linken Fensterausschnitt von **MobiLink 16** Ihren MobiLink-Projektnamen, **Konsolidierte Datenbanken** und den Namen Ihrer konsolidierten Datenbank. Wählen Sie anschließend **Benachrichtigung** aus.

4. Klicken Sie auf **Datei** » **Einstellungen exportieren** und befolgen Sie die Anweisungen des Assistenten.

Ergebnisse

Die angegebenen Einstellungen werden in eine Notifier-Konfigurationsdatei exportiert.

Serverseitige Einstellungen, die mit der Notifier-Konfigurationsdatei konfiguriert wurden

Serverseitige Einstellungen können in einer Notifier-Konfigurationsdatei gespeichert werden. Sie können mithilfe dieser Datei mehrere Notifier, Gateways und Netzbetreiber konfigurieren.

Hinweis
Sie müssen den ANSI-Standard verwenden, wenn Sie Namen für Ihre Notifier, Gateways und Netzbetreiber festlegen.

Notifier-Konfigurationsdatei erstellen und konfigurieren

Eine Notifier-Konfigurationsdatei kann mithilfe eines Texteditors erstellt oder mit aus Sybase Central exportierten Eigenschafts- und Ereigniseinstellungen generiert werden. Siehe „Notifier, Gateways oder Netzbetreiber mit Sybase Central einrichten“ auf Seite 30.

Um das Layout einer typischen Notifier-Konfigurationsdatei anzuzeigen, öffnen Sie die Vorlagendatei *%SQLANYSAMP16%\MobiLink\template.Notifier*. Die Vorlagendatei enthält Beispiele für die Konfiguration von serverseitigen Eigenschaften und Ereignissen.

Wenn Sie die erforderlichen Einstellungen konfiguriert haben, speichern Sie die Notifier-Konfigurationsdatei und laden die serverseitigen Eigenschaften und Ereignisse auf den MobiLink-Server.

Syntax allgemeiner Eigenschaften

```
Property = Value
```

Syntax von Notifier-Ereignissen

```
Notifier(NotifierName).Event = \
# Replace this text with SQL script.          \
# Be sure to put a backslash (\) at           \
# the end of every line of code               \
# if your event requires multiple             \
# lines of text.
```

Syntax von Notifier-Eigenschaften

```
Notifier(NotifierName).Property = Value
```

Syntax von Gateway-Eigenschaften

```
# For Device tracking gateways:
DeviceTracker(DeviceTrackerName).Property = Value
# For SMTP gateways:
SMTP(SMTPName).Property = Value
# For SYNC gateways:
SYNC(SYNCName).Property = Value
# For UDP gateways:
UDP(UDPName).Property = Value
```

Syntax von Netzbetreiber-Eigenschaften

```
Carrier(CarrierName).Property = Value
```

Notifier-Konfigurationsdatei laden

Um eine Notifier-Konfigurationsdatei auf den MobiLink-Server zu laden, führen Sie mlsrv16 von der Befehlszeile aus unter Verwendung der Option -notifier aus. Beispiel: Um die serverseitigen Einstellungen aus einer CarDealer.Notifier-Konfigurationsdatei zu verwenden, führen Sie folgenden Befehl aus:

```
mlsrv16 ... -notifier "c:\CarDealer.Notifier"
```

Wenn keine Datei angegeben wird, wird standardmäßig die Datei *config.Notifier* geladen.

Weitere Hinweise zur mlsrv16-Option -notifier finden Sie unter „mlsrv16-Option -notifier “ [*MobiLink - Serveradministration*].

Hinweis
Wenn Sie das Standard-SYNC-Gateway verwenden möchten, dürfen die serverseitigen Einstellungen nicht in einer Notifier-Konfigurationsdatei gespeichert werden. Sie müssen sie mithilfe einer anderen Methode in der Systemtabelle ml_property speichern. Siehe „MobiLink-Server-Einstellungen für serverinitiierte Synchronisation“ auf Seite 29.

Escapesequenzen verwenden

Der Backslash (\) ist das Escapezeichen. Die im Folgenden aufgelisteten üblichen Escapesequenzen können in einer Notifier-Konfigurationsdatei verwendet werden:

Escapesequenz	Beschreibung
\b	Rücktaste
\t	Tabulatortaste
\n	Zeilenvorschub
\r	Wagenrücklauf
\"	Anführungszeichen (")
\'	Apostroph (')
\\	Backslash (\)
\e	Escape

Unicode-Escapesequenzen haben die Form \u*XXXX*, während ASCII-Escapesequenzen die Form \x*XX* haben, wobei jedes *X* ein hexadezimales Zeichen darstellt.

Wenn Sie eine Eigenschaft oder ein Ereignis bearbeiten, das mehrere Zeilen umfasst, müssen Sie jede Zeile mit einem einzelnen Backslash (\) beenden.

Notifier-Ereignisse

Ereignisse werden ausgelöst, wenn ein Notifier einen MobiLink Listener abfragt. Wenn ein Ereignis ausgelöst wird, wird das zugehörige SQL-Skript ausgeführt. Sie können ein SQL-Skript in jedes der in diesem Abschnitt aufgelisteten Notifier-Ereignisse einbeziehen. Wenngleich die Skripterstellung optional ist, müssen Sie für das request_cursor-Polling-Ereignis ein Skript erstellen.

Es gibt drei Klassifizierungen für Notifier-Ereignisse: Polling-Ereignisse, Verbindungsereignisse und asynchrone Ereignisse. Bei jeder Prüfung der konsolidierten Datenbank durch den Notifier werden Polling-Ereignisse ausgelöst. Sie enthalten alle Ereignisse, die zwischen einem begin_poll- und einem end_poll-Ereignis auftreten. Verbindungsereignisse werden während der Datenbankverbindung des Notifiers ausgelöst. Asynchrone Ereignisse können zu jedem Zeitpunkt während des Synchronisationsvorgangs ausgelöst werden.

Sofern nicht anders angegeben, können Notifier-Ereignisse mit allen empfohlenen Methoden konfiguriert werden. Weitere Hinweise zur Konfiguration von Notifier-Ereignissen finden Sie unter „MobiLink-Server-Einstellungen für serverinitiierte Synchronisation“ auf Seite 29.

Wenn ein MobiLink Listener den Notifier abfragt, werden diese Ereignisse in der folgenden Reihenfolge ausgelöst:

```
Fire begin_connection event
For each poll (
    Fire begin_poll event
        Fire shutdown_query event
        Fire request_cursor event
        For all requests expired before required confirmation (
            Fire error_handler event
        )
        Fire request_delete event
    Fire end_poll event
)
Fire end_connection event
```

Ereignisse beim Abruf

Polling-Ereignisse sind eine Klasse von Notifier-Ereignissen, die jedes Mal ausgelöst werden, wenn der Notifier die konsolidierte Datenbank prüft. Diese Ereignisse umfassen alle Ereignisse, die zwischen einem begin_poll-Ereignis und einem end_poll-Ereignis auftreten.

begin_poll-Ereignis

Dieses Polling-Ereignis akzeptiert ein SQL-Skript und wird ausgelöst, bevor der Notifier die konsolidierte Datenbank auf Push-Anforderungen prüft. Der Wert ist standardmäßig NULL, sodass dieses Ereignis nicht ausgelöst wird.

Siehe auch

- „Push-Anforderungen“ auf Seite 7
- „Notifier-Ereignisse“ auf Seite 35
- „MobiLink-Server-Einstellungen für serverinitiierte Synchronisation“ auf Seite 29

Beispiel

In diesem Beispiel wird eine Push-Anforderung für den Notifier A erstellt. Sie verwendet eine SQL-Anweisung, die Zeilen in die Tabelle PushRequest einfügt. Jede Zeile in dieser Tabelle repräsentiert eine Nachricht, die an eine Adresse gesendet werden soll. Die WHERE-Klausel legt fest, welche Push-Anforderungen in die Tabelle PushRequest eingefügt werden.

Mit dem folgenden Befehl können Sie die gespeicherte Prozedur ml_add_property mit einer konsolidierten SQL Anywhere-Datenbank verwenden:

```
ml_add_property(
    'SIS',
    'Notifier(Notifier A)',
    'begin_poll',
    'INSERT INTO PushRequest
        (gateway, mluser, subject, content)
        SELECT ''MyGateway'', DISTINCT mluser, ''sync'',
            stream_param
            FROM MLUserExtra, mluser_union, Dealer
            WHERE MLUserExtra.mluser = mluser_union.name
            AND (push_sync_status = ''waiting for request''
                OR datediff( hour, last_status_change, now() ) > 12 )
            AND ( mluser_union.publication_name is NULL
                OR mluser_union.publication_name =''FullSync'' )
            AND Dealer.last_modified > mluser_union.last_sync_time'
);
```

end_poll-Ereignis

Dieses Polling-Ereignis akzeptiert ein SQL-Skript und wird ausgelöst, nachdem der Notifier die konsolidierte Datenbank auf Push-Anforderungen geprüft hat. Der Wert ist standardmäßig NULL, sodass dieses Ereignis nicht ausgelöst wird.

Sie können dieses Ereignis verwenden, um eine Tabellenbereinigung auszuführen oder um die Ergebnisse eines Pollings zu protokollieren.

Siehe auch

- „Notifier-Ereignisse“ auf Seite 35
- „MobiLink-Server-Einstellungen für serverinitiierte Synchronisation“ auf Seite 29

error_handler-Ereignis

Konfigurieren Sie dieses Ereignis so, dass es darauf hinweist, wenn eine Übertragung fehlgeschlagen ist oder nicht bestätigt wurde. Wenn eine Übertragung fehlschlägt, können Sie mit diesem Ereignis eine Zeile in eine Audit-Tabelle einfügen oder eine Push-Benachrichtigung senden.

In der folgenden Tabelle sind die Parameter aufgelistet, die mit dem error_handler-Ereignis erfasst werden können:

Skriptparameter	Typ	Beschreibung
request_option (out)	Ganzzahl	Regelt, wie der Notifier die Push-Anforderung verarbeitet, nachdem die Fehlerbehandlungsroutine beendet wurde. Die Ausgabe kann einen der folgenden Werte haben: ● **0**: Standardaktion basierend auf dem Fehlercode durchführen und Fehler protokollieren. ● **1**: Nichts tun. ● **2**: Das request_delete-Ereignis ausführen. ● **3**: Zustellung über ein sekundäres Gateway versuchen.
error_code (in)	Ganzzahl	Verwenden Sie einen der folgenden Werte für den Fehlercode: ● **-1**: Zeitüberschreitung der Anforderung mit Erfolgsbestätigung. ● **-8**: Während des Zustellungsversuchs ist ein Fehler aufgetreten.
request_id (in)	Ganzzahl	Identifiziert die Anforderung.
gateway (in)	Varchar	Gibt das mit der Anforderung verbundene Gateway an.
address (in)	Varchar	Gibt die mit der Push-Anforderung verbundene Adresse an. Beim Aufrufen der optionalen Notifier-Fehlerbehandlungsroutine wird aus Sicherheitsgründen ein Sternchen (*) für beliebige Zeichen im Betreff oder Inhalt der Nachricht verwendet, das keines der folgenden ist: ● Alphanumerische Zeichen ● Punkt ● Doppelpunkt ● Minuszeichen ● Pluszeichen ● Unterstrich Der in der Benachrichtigung gesendete Wert ist derselbe wie der ursprüngliche Wert.

Skriptparameter	Typ	Beschreibung
subject (in)	Var-char	Gibt den mit der Push-Anforderung verbundenen Betreff an. Beim Aufrufen der optionalen Notifier-Fehlerbehandlungsroutine wird aus Sicherheitsgründen ein Sternchen (*) für beliebige Zeichen im Betreff oder Inhalt der Nachricht verwendet, das keines der folgenden ist: ● Alphanumerische Zeichen ● Punkt ● Doppelpunkt ● Minuszeichen ● Pluszeichen ● Unterstrich Der in der Benachrichtigung gesendete Wert ist derselbe wie der ursprüngliche Wert.
content (in)	Var-char	Gibt den mit der Push-Anforderung verbundenen Inhalt an. Beim Aufrufen der optionalen Notifier-Fehlerbehandlungsroutine wird aus Sicherheitsgründen ein Sternchen (*) für beliebige Zeichen im Betreff oder Inhalt der Nachricht verwendet, das keines der folgenden ist: ● Alphanumerische Zeichen ● Punkt ● Doppelpunkt ● Minuszeichen ● Pluszeichen ● Unterstrich Der in der Benachrichtigung gesendete Wert ist derselbe wie der ursprüngliche Wert.

Hinweis
Dieses Ereignis erfordert eine Systemprozedur. Sie können dieses Ereignis nicht direkt mit Sybase Central konfigurieren. Siehe „MobiLink-Server-Einstellungen für serverinitiierte Synchronisation“ auf Seite 29.

Siehe auch

- „Notifier-Ereignisse“ auf Seite 35
- „MobiLink-Server-Einstellungen für serverinitiierte Synchronisation“ auf Seite 29

Beispiel

Im folgenden Beispiel erstellen Sie die Tabelle CustomError und protokollieren mit der gespeicherten Prozedur CustomErrorHandler Fehler in der Tabelle. Der Ausgabeparameter Notifier_opcode ist immer 0. Daher wird die Notifier-Standardverarbeitung verwendet.

```
CREATE TABLE CustomError(
    error_code  integer,
    request_id  integer,
    gateway  varchar(255),
    address  varchar(255),
    subject  varchar(255),
    content  varchar(255),
    occurAt  timestamp not null default timestamp
);

CREATE PROCEDURE CustomErrorHandler(
    out @Notifier_opcode integer,
    in @error_code  integer,
    in @request_id  integer,
    in @gateway   varchar(255),
    in @address   varchar(255),
    in @subject   varchar(255),
    in @content   varchar(255)
)

BEGIN
    INSERT INTO CustomError(
        error_code,
        request_id,
        gateway,
        address,
        subject,
        content)
    VALUES(
        @error_code,
        @request_id,
        @gateway,
        @address,
        @subject,
        @content
    );
    SET @Notifier_opcode = 0;
END
```

Mit dem folgenden Befehl können Sie die gespeicherte Prozedur ml_add_property mit einer konsolidierten SQL Anywhere-Datenbank verwenden:

```
call ml_add_property(
    'SIS',
    'Notifier(myNotifier)',
    'error_handler',
    'call CustomErrorHandler(?, ?, ?, ?, ?, ?, ?)');
```

Alternativ dazu können Sie dieses Ereignis auslösen, indem Sie einer Notifier-Konfigurationsdatei folgende Zeile hinzufügen:

```
Notifier(myNotifier).error_handler = call
CustomErrorHandler(?, ?, ?, ?, ?, ?, ?)
```


Führen Sie die Datei mit der mlsrv16-Option -notifier aus. Weitere Hinweise zum Konfigurieren einer Notifier-Konfigurationsdatei finden Sie unter „Serverseitige Einstellungen, die mit der Notifier-Konfigurationsdatei konfiguriert wurden“ auf Seite 33.

request_cursor-Ereignis

Dieses Polling-Ereignis akzeptiert ein SQL-Skript und wird ausgelöst, um Push-Anforderungen zu finden. Sie müssen dieses Ereignis konfigurieren.

Push-Anforderungen bei der Verwendung eines Lightweight-Polling-Moduls abrufen (empfohlen)

Wenn dieses Ereignis bis zu drei Spalten in einer Ergebnismenge enthält, wird vom Notifier bestätigt, dass keine beständige Verbindung zwischen dem Server und dem Gerät besteht und dass ein Gerät den Notifier abfragen muss, bevor Push-Benachrichtigungen gesendet werden können. Der Notifier speichert die Ergebnismenge vor dem Senden von Push-Benachrichtigungen im Cache. Der MobiLink-Server identifiziert das Gerät mithilfe des Polling-Schlüssels, der bei jedem Abruf des Notifiers vom Gerät gesendet wird.

Die Ergebnismenge dieses Ereignisses muss folgende Spalten in der angegebenen Reihenfolge enthalten:

- Polling-Schlüssel
- Betreff (optional)
- Inhalt (optional)

Push-Anforderungen bei der Verwendung eines Gateways abrufen

Wenn dieses Ereignis mehr als drei Spalten in einer Ergebnismenge enthält, bestätigt der Notifier, dass eine beständige Verbindung zwischen dem Server und dem Gerät besteht und sendet dann, wenn Push-Anforderungen erkannt werden, unter Verwendung eines Gateways Push-Benachrichtigungen.

Die Ergebnismenge dieses Ereignisses muss folgende Spalten in der angegebenen Reihenfolge enthalten:

- Anforderungs-ID (optional)
- Gateway
- Betreff
- Inhalt
- Adresse
- Neusendeintervall (optional)
- Restzeit (optional)

Siehe auch

- „Voraussetzungen für Push-Anforderungen“ auf Seite 7
- „Notifier-Ereignisse“ auf Seite 35
- „MobiLink-Server-Einstellungen für serverinitiierte Synchronisation“ auf Seite 29

Beispiel

Im folgenden Beispiel wird mithilfe der ml_add_property-Systemprozedur für einen benutzerdefinierten Notifier namens Simple ein request_cursor-Ereignisskript erstellt. Die Select-Anweisung weist den Notifier an, eine Tabelle namens PushRequest auf Push-Anforderungen zu prüfen.

```
CALL ml_add_property('SIS', 'Notifier(Simple)', 'request_cursor',
    'SELECT poll_key,
        subject,
        content
    FROM PushRequest'
);
```

Es wird empfohlen, eine WHERE-Klausel in das Skript einzubeziehen, um bereits gesendete Anforderungen herauszufiltern. Sie können z.B. eine Spalte für Push-Anforderungen hinzufügen, um den Zeitpunkt zu protokollieren, zu dem Sie eine Anforderung eingefügt haben, und dann eine WHERE-Klausel verwenden, um Anforderungen herauszufiltern, die vor der letzten Synchronisation eingefügt wurden.

request_delete-Ereignis

Dieses Polling-Ereignis akzeptiert ein SQL-Skript. Es wird ausgelöst, um Bereinigungsvorgänge auszuführen, wenn erkannt wird, dass Push-Anforderungen gelöscht werden müssen. Es akzeptiert die Anforderungs-ID als Parameter und wird für jede Anforderungs-ID ausgeführt. Das request_cursor-Ereignis muss eine Spalte für die Anforderungs-ID enthalten, um das request_delete-Ereignis zu verwenden. Sie können die Anforderungs-ID mit einem benannten Parameter referenzieren oder einem Fragezeichen (?). Dieses Ereignis ist optional, wenn Sie bereits einem anderen Prozess oder Ereignis, wie z.B. dem end_poll-Ereignis, Bereinigungsvorgänge zugewiesen haben.

Der Notifier kann mit der DELETE-Anweisung folgende Formen von Push-Anforderungen löschen:

- **Implizit gelöscht** Diese Push-Anforderungen sind früher schon vorgekommen, doch sie kommen nicht in der aktuellen Menge von Anforderungen vor, die aus request_cursor bezogen wurden.
- **Bestätigt** Diese Push-Anforderungen sind als zugestellt bestätigt.
- **Abgelaufen** Diese Push-Anforderungen sind basierend auf ihren Sendewiederholungs-Attributen und der aktuellen Uhrzeit abgelaufen. Anforderungen ohne Attribute zur Sendewiederholung werden als abgelaufen betrachtet, selbst wenn sie in der nachfolgenden Anforderung enthalten sind.

Sie können mit dem request_delete-Ereignis verhindern, dass abgelaufene oder implizit gelöschte Anforderungen gelöscht werden. Beispiel: Das Beispiel CarDealer im Verzeichnis *%SQLANYSAMP16% \MobiLink\SIS_CarDealer* verwendet das request_delete-Ereignis, um das Statusfeld der PushRequest-Tabelle auf 'processed' (verarbeitet) zu setzen.

```
UPDATE PushRequest SET status='processed' WHERE req_id = ?
```


Das begin_poll-Ereignis im Beispiel verwendet die Zeit der letzten Synchronisation, um zu prüfen, ob entfernte Geräte auf dem aktuellen Stand sind, bevor verarbeitete Push-Anforderungen gelöscht werden.

Siehe auch

- „Notifier-Ereignisse“ auf Seite 35
- „MobiLink-Server-Einstellungen für serverinitiierte Synchronisation“ auf Seite 29

shutdown_query-Ereignis

Dieses Polling-Ereignis akzeptiert ein SQL-Skript und wird nach einem begin_poll-Ereignis ausgelöst. Der Rückgabewert gibt den shutdown-Status des Notifiers an. Der Wert ist standardmäßig NULL, sodass dieses Ereignis nicht ausgelöst wird.

Um den Notifier herunterzufahren, richten Sie das SQL-Skript so ein, dass es "yes" zurückgibt. Andernfalls setzen Sie den Wert auf "no". Wenn der Notifier heruntergefahren wird, wird das end_poll-Ereignis nicht ausgelöst.

Wenn der shutdown-Status in einer Tabelle gespeichert wird, verwenden Sie das end_connection-Ereignis, um den shutdown-Status zurückzusetzen.

Siehe auch

- „end_connection-Ereignis“ auf Seite 43
- „Notifier-Ereignisse“ auf Seite 35
- „MobiLink-Server-Einstellungen für serverinitiierte Synchronisation“ auf Seite 29

Beispiel

Im folgenden Beispiel wird mithilfe der ml_add_property-Systemprozedur für einen benutzerdefinierten Notifier namens Simple ein shutdown_query-Ereignisskript erstellt. Die SELECT-Anweisung weist den Notifier an herunterzufahren, wenn die Methode tooManyNotifierErrors TRUE zurückgibt.

```
CALL ml_add_property('SIS', 'Notifier(Simple)', 'shutdown_query',
    'SELECT
        IF tooManyNotifierErrors() THEN
            ''yes''
        ELSE
            ''no''
        ENDIF'
);
```

Verbindungsereignisse

Verbindungsereignisse sind eine Klasse von Notifier-Ereignissen, die während der Verbindung des Notifiers mit der Datenbank ausgelöst werden.

begin_connection-Ereignis

Dieses Ereignis akzeptiert ein SQL-Skript und wird ausgelöst, nachdem der Notifier eine Verbindung mit der konsolidierten Datenbank herstellt, aber bevor er auf Push-Anforderungen prüft. Der Wert ist standardmäßig NULL, sodass dieses Ereignis nicht ausgelöst wird.

Sie können dieses Ereignis verwenden, um temporäre Tabellen oder Variable zu erstellen. Sie sollten dieses Ereignis nicht zum Ändern der Isolationsstufen verwenden. Für die Einstellung der Isolationsstufen steht die Eigenschaft isolation zur Verfügung.

Falls der Notifier die Verbindung zur konsolidierten Datenbank verliert, führt er dieses Ereignis unmittelbar nach der Wiederherstellung der Verbindung erneut aus.

Siehe auch

- „Notifier-Eigenschaften“ auf Seite 47
- „Notifier-Ereignisse“ auf Seite 35
- „MobiLink-Server-Einstellungen für serverinitiierte Synchronisation“ auf Seite 29

end_connection-Ereignis

Dieses Ereignis akzeptiert ein SQL-Skript und wird ausgelöst, unmittelbar bevor der Notifier die Verbindung zur konsolidierten Datenbank trennt. Der Wert ist standardmäßig NULL, sodass dieses Ereignis nicht ausgelöst wird.

Sie können dieses Ereignis für die Bereinigung von temporärem Speicher, wie z.B. SQL-Variablen und temporären Tabellen, verwenden.

Siehe auch

- „Notifier-Ereignisse“ auf Seite 35
- „MobiLink-Server-Einstellungen für serverinitiierte Synchronisation“ auf Seite 29

Asynchrone Ereignisse

Asynchrone Ereignisse sind eine Klasse von Notifier-Ereignissen, die zu jedem beliebigen Zeitpunkt während des Synchronisationsvorgangs ausgelöst werden können.

confirmation_handler-Ereignis

Konfigurieren Sie dieses Ereignis für die Verarbeitung von Informationen zur Zustellungsbestätigung, die von MobiLink Listenern hochgeladen wurden. Wenn der Statusparameter 0 zurückgibt, wurde die durch request_id identifizierte Push-Anforderung erfolgreich vom MobiLink Listener (identifiziert durch den Parameter remote_device) empfangen.

Mit dem Parameter request_option können Sie eine Aktion als Reaktion auf die Zustellungsbestätigung initiieren. Wenn request_option gleich 0 ist, führt das confirmation_handler-Ereignis standardmäßig diese Aktion aus. Das Ereignis request_delete wird ausgeführt, um die ursprüngliche Push-Anforderung zu

löschen. Wenn das Gerät, das die Zustellungsbestätigung sendet, nicht mit dem Gerät übereinstimmt, das durch request_id identifiziert wird, wird standardmäßig die ursprüngliche Push-Anforderung über ein sekundäres Gateway gesendet.

Hinweis
Mit der dblsn-Option -x können MobiLink Listener Informationen zur Zustellungsbestätigung hochladen. Verwenden Sie die dblsn-Option -ni, wenn Sie die Zustellungsbestätigung wünschen, aber kein IP-Tracking. Siehe „MobiLink Listener-Optionen für Windows-Geräte“ auf Seite 57.

Hinweis
Dieses Ereignis erfordert eine Systemprozedur. Sie können dieses Ereignis nicht direkt mit der Sybase Central-Methode konfigurieren. Siehe „MobiLink-Server-Einstellungen für serverinitiierte Synchronisation“ auf Seite 29.

Die folgenden Parameter können mit dem confirmation_handler-Ereignis erfasst werden:

Skriptparameter	Typ	Beschreibung
request_option (out)	Ganzzahl	Regelt, wie der Notifier die Anforderung verarbeitet, nachdem der Handler zurückgegeben wurde. Die folgenden Werte können zurückgegeben werden: • **0**: Standardmäßige Notifier-Aktion basierend auf dem Wert des Statusparameters. Wenn der Status darauf hinweist, dass es sich bei dem antwortenden Gerät um das Zielgerät handelt, löscht der Notifier die Anforderung. Andernfalls versucht der Notifier eine Zustellung über ein sekundäres Gateway. • **1**: Nichts tun. • **2**: Notifier.request_delete ausführen. • **3**: Zustellung über ein sekundäres Gateway versuchen.
status (in)	Ganzzahl	Ein Überblick über die Situation. Der Status kann während der Entwicklung verwendet werden, um Probleme wie beispielsweise falsche Filter und Handler-Attribute zu identifizieren. Die folgenden Werte können zurückgegeben werden: • **0**: Empfangen und bestätigt. • **-2**: Richtige Antwortquelle, Nachricht wurde jedoch zurückgewiesen. • **-3**: Richtige Antwortquelle, Nachricht wurde akzeptiert, aber Aktion ist fehlgeschlagen. • **-4**: Falsche Antwortquelle, Nachricht wurde akzeptiert. • **-5**: Falsche Antwortquelle, Nachricht wurde zurückgewiesen. • **-6**: Falsche Antwortquelle. Die Nachricht wurde akzeptiert und die Aktion war erfolgreich. • **-7**: Falsche Antwortquelle. Die Nachricht wurde akzeptiert, aber die Aktion ist fehlgeschlagen.

Skriptparameter	Typ	Beschreibung
request_id (in)	Ganzzahl	Die Anforderungs-ID. Das request_cursor-Ereignis muss eine Spalte für die Anforderungs-ID enthalten, um das confirmation_handler-Ereignis zu verwenden.
remote_code (in)	Ganzzahl	Das vom MobiLink Listener berichtete Ergebnis. Die folgenden Werte können zurückgegeben werden: • **1**: Nachricht akzeptiert. • **2**: Nachricht zurückgewiesen. • **3**: Nachricht akzeptiert und Aktion erfolgreich. • **4**: Nachricht akzeptiert und Aktion fehlgeschlagen.
remote_device (in)	Varchar	Der Gerätename des antwortenden MobiLink Listeners.
remote_mluser (in)	Varchar	Der MobiLink-Benutzername des antwortenden MobiLink Listeners.
remote_action_return (in)	Varchar	Der Rückgabecode der entfernten Aktion.
remote_action (in)	Varchar	Reserviert für den Aktionsbefehl.
gateway (in)	Varchar	Das der Anforderung zugeordnete Gateway.
address (in)	Varchar	Die der Anforderung zugeordnete Adresse.
subject (in)	Varchar	Der der Anforderung zugeordnete Betreff.
content (in)	Varchar	Der der Anforderung zugeordnete Inhalt.

Siehe auch

- „Notifier-Ereignisse“ auf Seite 35
- „MobiLink-Server-Einstellungen für serverinitiierte Synchronisation“ auf Seite 29
- „Gateway-Eigenschaften“ auf Seite 49

Beispiel

Im folgenden Beispiel erstellen Sie die Tabelle CustomConfirmation und protokollieren dann Bestätigungen mit der gespeicherten Prozedur CustomConfirmationHandler. Der Ausgabeparameter request_option beträgt immer 0. Daher wird die Notifier-Standardverarbeitung verwendet.

```
CREATE TABLE CustomConfirmation(
    error_code   integer,
    request_id   integer,
    remote_code   integer,
    remote_device  varchar(128),
    remote_mluser  varchar(128),
    remote_action_return varchar(128),
    remote_action  varchar(128),
```

```
    gateway   varchar(255),
    address   varchar(255),
    subject   varchar(255),
    content   varchar(255),
    occurAt   timestamp not null default timestamp
);

CREATE PROCEDURE CustomConfirmationHandler(
    out @request_option integer,
    in @error_code  integer,
    in @request_id  integer,
    in @remote_code  integer,
    in @remote_device  varchar(128),
    in @remote_mluser  varchar(128),
    in @remote_action_return varchar(128),
    in @remote_action  varchar(128),
    in @gateway   varchar(255),
    in @address   varchar(255),
    in @subject   varchar(255),
    in @content   varchar(255)
)

BEGIN
    INSERT INTO CustomConfirmation(
        error_code,
        request_id,
        remote_code,
        remote_device,
        remote_mluser,
        remote_action_return,
        remote_action,
        gateway,
        address,
        subject,
        content)
    VALUES (
        @error_code,
        @request_id,
        @remote_code,
        @remote_device,
        @remote_mluser,
        @remote_action_return,
        @remote_action,
        @gateway,
        @address,
        @subject,
        @content
    );
    SET @request_option = 0;
END
```

Mit dem folgenden Befehl können Sie die gespeicherte Prozedur ml_add_property mit einer konsolidierten SQL Anywhere-Datenbank verwenden:

```
call ml_add_property(
    'SIS',
    'Notifier(myNotifier)',
    'confirmation_handler',
    'call CustomConfirmation(?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?)');
```

Alternativ dazu können Sie dieses Ereignis aufrufen, indem Sie einer Notifier-Konfigurationsdatei folgende Zeile hinzufügen:

```
Notifier(myNotifier).confirmation_handler = call
CustomConfirmation(?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?)
```

Führen Sie die Datei mit der mlsrv16-Option -notifier aus. Weitere Hinweise zum Konfigurieren einer Notifier-Konfigurationsdatei finden Sie unter „Serverseitige Einstellungen, die mit der Notifier-Konfigurationsdatei konfiguriert wurden“ auf Seite 33.

Allgemeine Eigenschaften

Allgemeine Eigenschaften werden von Notifiern, Gateways und Netzbetreibern gemeinsam genutzt. Alle allgemeinen Eigenschaften sind optional. Weitere Hinweise zum Einstellen dieser Eigenschaften finden Sie unter „MobiLink-Server-Einstellungen für serverinitiierte Synchronisation“ auf Seite 29.

Eigenschaft	Wert	Beschreibung
verbosity	{ **0** | **1** | **2** | **3** }	Legt die Ausführlichkeitsstufe für Notifier, Gateways und Netzbetreiber fest. Die folgenden Werte können verwendet werden: • **0**: Nicht verfolgen. • **1**: Start-, Stopp- und Eigenschafts-Trace. • **2**: Benachrichtigungen anzeigen. • **3**: Vollständiger Trace. Der Standardwert ist **0**.

Notifier-Eigenschaften

Mit den Eigenschaften des Notifiers können Sie das Verhalten eines Notifiers ändern. Alle Notifier-Eigenschaften sind optional. Weitere Hinweise zum Einstellen dieser Eigenschaften finden Sie unter „MobiLink-Server-Einstellungen für serverinitiierte Synchronisation“ auf Seite 29.

Eigenschaft	Wert	Beschreibung
connect_string	*connection_string*	Setzt das Standardverhalten für die Erstellung von Verbindungen mit Datenbanken außer Kraft. Der Standardwert ist **ianywhere.ml.script.ServerContext**. In diesem Fall wird die in der mlsrv16-Befehlszeile angegebene Verbindungszeichenfolge verwendet. Es kann sinnvoll sein, eine Verbindung zu einer anderen Datenbank herzustellen, wenn Sie wollen, dass Benachrichtigungslogik und Daten von Ihren Synchronisationsdaten getrennt sind. Die meisten Deployments definieren diese Eigenschaft nicht.

Eigenschaft	Wert	Beschreibung
enable	{ **yes** | **no** }	Legt fest, ob der Notifier aktiviert werden soll. Wenn Sie die mlsrv16-Option -notifier ausführen, werden alle aktivierten Notifier gestartet.
gui	{ **yes** | **no** }	Legt fest, ob das Notifier-Fenster angezeigt werden soll, während der Notifier ausgeführt wird. Der Standardwert ist **yes**. Dieses Notifier-Fenster gestattet es dem Benutzer, das Polling-Intervall temporär zu ändern oder den Abruf sofort auszuführen. Damit kann der Notifier auch beendet werden, ohne den MobiLink-Server herunterzufahren. Wenn der Notifier gestoppt wurde, kann er nur neu gestartet werden, indem der MobiLink-Server heruntergefahren und neu gestartet wird.
isolation	{ **0** | **1** | **2** | **3** }	Gibt die Isolationsstufe der Datenbankverbindung des Notifiers an. Die folgenden Werte können verwendet werden: ● **0**: Nicht festgeschriebene Daten lesen. ● **1**: Festgeschriebene Daten lesen. ● **2**: Wiederholbare Lesevorgänge. ● **3**: Serialisierbar. Der Standardwert ist **1**. Höhere Stufen erhöhen Konflikte und können die Performance beeinträchtigen. Die Isolationsstufe **0** gestattet Lesevorgänge von nicht festgeschriebenen Daten, die zurückgesetzt werden können.
poll_every	*number*{ **s** | **m** | **h** }	Gibt an, wie lang der Timeout für Bestätigungen ist. Die im Folgenden aufgelisteten Zeiteinheiten werden akzeptiert: ● **s**: Sekunden. ● **m**: Minuten. ● **h**: Stunden. Der Standardwert ist **1m**. Zeiteinheiten können im Format *HH***h** *MM***m** *SS***s** kombiniert werden. Wenn keine Zeiteinheit angegeben ist, wird die Zeit in Sekunden gemessen.

Eigenschaft	Wert	Beschreibung
shared_database_connection	{ **yes** | **no** }	Gibt an, ob Notifier Datenbankverbindungen gemeinsam nutzen sollen. Der Standardwert ist **no**. Notifier können Verbindungen nur gemeinsam nutzen, wenn sie die gleiche Isolationsstufe haben. Geben Sie **yes** an, um weniger Ressourcen zu verwenden, ohne die Performance dadurch zu beeinträchtigen. In bestimmten Situationen ist eine gemeinsame Nutzung von Verbindungen nicht möglich, z.B., wenn Anwendungen für die verschiedenen Notifier nicht-eindeutige SQL-Variablennamen verwenden.

Gateway-Eigenschaften

Standardmäßig werden beim Start von MobiLink-Server vier vorkonfigurierte Gateways erstellt. Diese werden installiert, wenn Sie MobiLink-Setupskripten für Ihre konsolidierte Datenbank ausführen. Die Standard-Gateways werden folgendermaßen bezeichnet:

- Default-DeviceTracker-Gateway
- Default-SYNC-Gateway
- Default-UDP-Gateway
- Default-SMTP-Gateway

Sie sollten die Standard-Gateways nicht entfernen oder ihre Namen ändern. Es wird empfohlen, zusätzliche Gateways mit anderen Namen zu erstellen.

Es ist nicht erforderlich, die in DefaultSYNC und DefaultUDP definierten Eigenschaften zu ändern, doch Sie müssen dem DefaultSYNC-Gateway SMTP-Serverinformationen bereitstellen. Es sollten die Standard-Gateways verwendet werden, aber Sie können bei Bedarf auch eine alternative Konfiguration verwenden. Dieser Abschnitt beschreibt Verfahren zum Anpassen der Eigenschaften von Gateways.

Eigenschaften von Device Tracking-Gateways

Mit den Eigenschaften des Device Tracking-Gateways können Sie dessen Verhalten ändern. Alle Device Tracking-Gateway-Eigenschaften sind optional. Weitere Hinweise zum Einstellen dieser Eigenschaften finden Sie unter „MobiLink-Server-Einstellungen für serverinitiierte Synchronisation“ auf Seite 29.

Eigenschaft	Wert	Beschreibung
con-firm_action	{ **yes** | **no** }	Gibt an, ob bei der Zustellung über dieses Gateway eine Bestätigung gesendet wird. Der Standardwert ist **no**.
con-firm_deli-very	{ **yes** | **no** }	Legt fest, ob der MobiLink Listener der konsolidierten Datenbank bestätigen soll, dass die Nachricht empfangen wurde. Der Standardwert ist **yes**. Der MobiLink Listener muss mit der MobiLink Listener-Option -x gestartet werden.
description	*description_text*	Beschreibt das Gateway.
enable	{ **yes** | **no** }	Gibt an, ob das Device Tracking-Gateway verwendet werden soll.
smtp_gate-way	*smtp_gateway_name*	Gibt den Namen des untergeordneten SMTP-Gateways an. Der Standardwert ist **DefaultSMTP**. Ein Device Tracking-Gateway kann nur ein einziges SMTP-Gateway verwenden. Das Gateway muss aktiviert sein.
sync_gate-way	*sync_gateway_name*	Gibt den Namen des untergeordneten SYNC-Gateways an. Der Standardwert ist **DefaultSYNC**. Ein Device Tracking-Gateway kann nur ein einziges SYNC-Gateway verwenden. Das Gateway muss aktiviert sein.
udp_gate-way	*udp_gateway_name*	Gibt den Namen des untergeordneten UDP-Gateways an. Der Standardwert ist **DefaultUDP**. Ein Device Tracking-Gateway kann nur ein einziges UDP-Gateway verwenden. Das Gateway muss aktiviert sein.

SMTP-Gateway-Eigenschaften

Mit den Eigenschaften des SMTP-Gateways können Sie dessen Verhalten ändern. Die Servereigenschaft ist erforderlich. Alle anderen SMTP-Gateway-Eigenschaften sind optional. Weitere Hinweise zum Einstellen dieser Eigenschaften finden Sie unter „MobiLink-Server-Einstellungen für serverinitiierte Synchronisation“ auf Seite 29.

Eigenschaft	Wert	Beschreibung
con-firm_action	{ **yes** | **no** }	Gibt an, ob bei der Zustellung über dieses Gateway eine Bestätigung gesendet wird. Der Standardwert ist **no**.

Eigenschaft	Wert	Beschreibung
con-firm_deli-very	{ **yes** | **no** }	Gibt an, ob dieses Gateway Zustellungen bestätigt. Der Standardwert ist **no**.
con-firm_timeout	*number*{ **s** | **m** | **h** }	Gibt an, wie lang der Timeout für Bestätigungen ist. Die im Folgenden aufgelisteten Zeiteinheiten werden akzeptiert: ● **s**: Sekunden. ● **m**: Minuten. ● **h**: Stunden. Der Standardwert ist **1m**. Zeiteinheiten können im Format *HH***h** *MM***m** *SS***s** kombiniert werden. Wenn keine Zeiteinheit angegeben ist, wird die Zeit in Sekunden gemessen.
description	*description_text*	Beschreibt das Gateway.
enable	{ **yes** | **no** }	Gibt an, ob das SYNC-Gateway verwendet werden soll.
Listen-ers_are_900	{ **yes** | **no** }	Gibt an, ob es sich bei allen MobiLink Listenern um SQL Anywhere 9.0.0-Clients handelt. Der Standard-wert ist **no**. Behalten Sie für SQL Anywhere 9.0.1-Clients oder höher den Wert **no** bei.
password	*password*	Gibt das Kennwort des SMTP-Dienstes an. Dies ist für einige Dienste erforderlich.
sender	*SMTP_address*	Gibt die Absenderadresse von SMTP-Push-Benach-richtigungen an. Der Standardwert ist **anonymous**.
server	*IP_address_or_hostname*	Gibt die IP-Adresse oder den Hostnamen des SMTP-Servers an, der verwendet wird, um Nachrichten an einen MobiLink Listener zu senden. Der Standardwert ist **mail**.
user	*username*	Gibt den Benutzernamen des SMTP-Dienstes an. Dies ist für einige Dienste erforderlich.

SYNC-Gateway-Eigenschaften

Mit den Eigenschaften des SYNC-Gateways können Sie dessen Verhalten ändern. Alle SYNC-Gateway-Eigenschaften sind optional. Weitere Hinweise zum Einstellen dieser Eigenschaften finden Sie unter „MobiLink-Server-Einstellungen für serverinitiierte Synchronisation“ auf Seite 29.

Eigenschaft	Wert	Beschreibung
con-firm_action	{ **yes** | **no** }	Gibt an, ob bei der Zustellung über dieses Gateway eine Bestätigung gesendet wird. Der Standardwert ist **no**.
con-firm_deli-very	{ **yes** | **no** }	Gibt an, ob dieses Gateway Zustellungen bestätigt. Der Standardwert ist **no**.
con-firm_timeout	*number*{ **s** | **m** | **h** }	Gibt an, wie lang der Timeout für Bestätigungen ist. Die im Folgenden aufgelisteten Zeiteinheiten werden akzeptiert: • **s**: Sekunden. • **m**: Minuten. • **h**: Stunden. Der Standardwert ist **1m**. Zeiteinheiten können im Format *HH***h** *MM***m** *SS***s** kombiniert werden. Wenn keine Zeiteinheit angegeben ist, wird die Zeit in Sekunden gemessen.
description	*description_text*	Beschreibt das Gateway.
enable	{ **yes** | **no** }	Gibt an, ob das SYNC-Gateway verwendet werden soll.
Listen-ers_are_900	{ **yes** | **no** }	Gibt an, ob es sich bei allen MobiLink Listenern um SQL Anywhere 9.0.0-Clients handelt. Der Standard-wert ist **no**. Lassen Sie den Wert **no** für SQL Anywhe-re 9.0.1-Clients oder höher unverändert.

UDP-Gateway-Eigenschaften

Mit UDP-Gateway-Eigenschaften können Sie das Verhalten eines UDP-Gateways ändern, wie z.B. die IP-Adresse und die Portnummer. Alle UDP-Gateway-Eigenschaften sind optional. Weitere Hinweise zum Einstellen dieser Eigenschaften finden Sie unter „MobiLink-Server-Einstellungen für serverinitiierte Synchronisation“ auf Seite 29.

Eigenschaft	Wert	Beschreibung
confirm_action	{ **yes** | **no** }	Gibt an, ob bei der Zustellung über dieses Gateway eine Bestätigung gesendet wird. Der Standardwert ist **no**.
confirm_delivery	{ **yes** | **no** }	Gibt an, ob dieses Gateway Zustellungen bestätigt. Der Standardwert ist **yes**.
confirm_timeout	*number*{ **s** | **m** | **h** }	Gibt an, wie lang der Timeout für Bestätigungen ist. Die im Folgenden aufgelisteten Zeiteinheiten werden akzeptiert: ● **s**: Sekunden. ● **m**: Minuten. ● **h**: Stunden. Der Standardwert ist **1m**. Zeiteinheiten können im Format *HH***h** *MM***m** *SS***s** kombiniert werden. Wenn keine Zeiteinheit angegeben ist, wird die Zeit in Sekunden gemessen.
description	*description_text*	Beschreibt das Gateway.
enable	{ **yes** | **no** }	Gibt an, ob das UDP-Gateway verwendet werden soll.
Listeners_are_900	{ **yes** | **no** }	Gibt an, ob es sich bei allen MobiLink Listenern um SQL Anywhere 9.0.0-Clients handelt. Der Standardwert ist **no**. Behalten Sie für SQL Anywhere 9.0.1-Clients oder höher den Wert **no** bei.
Listener_port	*port_number*	Gibt den Port an, den die entfernten Geräte verwenden, um UDP-Pakete zu senden. Der Standardwert ist **5001**.
sender	*IP_address_or_hostname*	Wird nur für mehrfach vernetzte Hosts verwendet. Gibt die IP-Adresse oder den Hostnamen des Absenders an. Der Standardwert ist **localhost**.
sender_port	*port_number*	Gibt die Portnummer an, die verwendet wird, um UDP-Pakete zu senden. Standardmäßig wird eine freie Portnummer zufällig vom Betriebssystem zugewiesen.

Netzbetreibereigenschaften

Eigenschaften des Netzbetreibers ermöglichen es Ihnen, das Verhalten einer Mobilfunkbetreiberkonfiguration zu ändern, die Informationen darüber bereitstellt, wie automatisch protokollierte Telefonnummern und Netzbetreiber E-Mail-Adressen zugeordnet werden. Alle Netzbetreiber-Eigenschaften sind optional und nur erforderlich, wenn Sie ein SMTP-Gateway verwenden.

Weitere Hinweise zum Einstellen dieser Eigenschaften finden Sie unter „MobiLink-Server-Einstellungen für serverinitiierte Synchronisation“ auf Seite 29.

Eigenschaft	Wert	Beschreibung
enable	{ **yes** | **no** }	Gibt an, ob der Netzbetreiber verwendet werden soll.
description	*description_text*	Beschreibt den Netzbetreiber.
network_provider_id	*id_text*	Gibt die Netzwerk-Provider-ID an. Um SMS in der Windows Mobile Phone Edition zu verwenden, setzen Sie diese Eigenschaft auf **_generic_**.
sms_email_domain	*domain_name*	Gibt den Domänennamen des Netzbetreibers an.
sms_email_user_prefix	*prefix_name*	Gibt das in den E-Mail-Adressen verwendete Präfix an.

MobiLink Listener-Dienstprogramm für Windows-Geräte (dblsn)

Dieser Abschnitt bietet Informationen über das MobiLink Listener-Dienstprogramm für Windows-Geräte, einschließlich Informationen zu Listener-Bibliotheken, Listener-Optionen und -Schlüsselwörter, Listener-Aktionsbefehlen und -Variablen sowie zur Listener-Konfiguration.

Syntax

dblsn [*options*] **-l** *message-handler* [**-l** *message-handler*...]

message-handler :
[*polling-option*;...] [*filter*;...]*action*; [*option*;...]

polling-option :
[**;poll_connect =** *string*]
[**;poll_notifier =** *string*]
[**;poll_key =** *string*]
[**;poll_every =** *number*]

option :
[**;continue = yes**]
[**;confirm_action = yes**]
[**;confirm_delivery = no**]
[**;maydial = no**]

filter :
[**subject =** *string*]
[**content =** *string*]
[**message =** *string* | **message_start =** *string*]
[**sender =** *string*]

action :
action = *command*[**;altaction =** *command*]

command :
START *program* [*program-arguments*]
| **RUN** *program* [*program-arguments*]
| **POST** *window-message* **TO** { *window-class-name* | *window-title* }
| *tcpip-socket-action*
| **DBLSN FULL SHUTDOWN**

tcpip-socket-action :
SOCKET port=*app-port*
[**;host=***app-host*]
[**;sendText=***text1*]
[**;recvText=** *text2* [**;timeout=***num-sec*]]

window-message : *string* | *message-id*

Bemerkungen

Der Message-Handler -l für dblsn ist ein durch Semikola getrenntes Name-Wert-Paar. Zeichenfolgenwerte in den Name-Wert-Paaren müssen in Apostrophen stehen. Andernfalls kann dblsn nicht gestartet werden, wenn der Zeichenfolgenwert ein Semikolon enthält.

Siehe auch

- „MobiLink Listener-Optionen für Windows-Geräte“ auf Seite 57
- „MobiLink Listener-Schlüsselwörter für Windows-Geräte“ auf Seite 72
- „MobiLink Listener-Aktionsbefehle für Windows-Geräte“ auf Seite 74
- „MobiLink Listener-Aktionsvariablen für Windows-Geräte“ auf Seite 78

Gesichertes Listener-Deployment

Der Listener empfängt externe Benachrichtigungen und kann eine Anwendung aufrufen, während Informationen aus den Benachrichtigungen übergeben werden. Externe Benachrichtigungen könnten theoretisch verwendet werden, um schädliche Daten einzuschleusen und unerwünschte Ergebnisse zu verursachen. Achten Sie sorgfältig darauf, das Listener-Deployment zu sichern.

Es wird empfohlen, dass Sie die folgenden Maßnahmen zum Sichern des Listeners ergreifen:

- Verwenden Sie die dblsn-Option -x und geben Sie ein TLS-basiertes Protokoll an, z.B. HTTPS, um den Server (den Notifier) zu überprüfen und die Netzwerkkommunikation zu sichern.
- Verwenden Sie keine SMS- oder UDP-Listener, weil diese nicht gesichert sind. Sowohl SMS als auch UDP sind standardmäßig deaktiviert.
- Alle mit der dblsn-Option -l konfigurierten Aktionen müssen entweder ungültige Eingaben ablehnen oder es muss gewährleistet sein, dass es keine schädlichen Auswirkungen gibt, wenn beliebige Eingaben empfangen werden.
- Vermeiden Sie es, sehr leistungsfähige oder sehr allgemeine Anwendungen, z.B. *cmd.com* in einer Aktionsspezifikation mit der dblsn-Option -l aufzurufen. Stellen Sie eine sehr spezifische Anwendung bereit und konfigurieren Sie sie so, dass sie genau das tut, was Sie benötigen, aber nicht mehr, und ungültige Eingaben ablehnt.
- Verwenden Sie Nachrichtenfilter, um die Aufrufe von Aktionen zu beschränken.
- Vermeiden Sie die Verwendung von Aktionsvariablen zum Festlegen von Funktionen.

Siehe auch

- „dblsn-Option -x“ auf Seite 71
- „Nachrichtenfilter“ auf Seite 16
- „MobiLink Listener-Schlüsselwörter für Windows-Geräte“ auf Seite 72
- „dblsn-Option -l “ auf Seite 63

Listener-Bibliotheken für Windows

Der MobiLink Listener verfügt über eine UDP-Listener-Bibliothek, *lsn_udp16.dll*, die standardmäßig geladen wird.

Wenn Sie ein SMTP-Gateway verwenden, müssen Sie eine Listener-Bibliothek festlegen. Sie können eine Bibliothek mit der dblsn-Option -d und Bibliotheksoptionen mit der dblsn-Option -a angeben.

UDP (*lsn_udp16.dll*)

Die folgende Liste enthält von der UDP-Listener-Bibliothek unterstützte Optionen:

Option	Beschreibung
Port= *port-number*	Diese Option legt die Portnummer fest, an der auf Nachrichten gewartet werden soll. Der Standardport ist **5001**.
Timeout= *seconds*	Diese Option legt die maximale Blockierungsdauer eines Lesevorgangs am UDP-Listening-Port fest. Dieser Wert muss kleiner als das Polling-Intervall des UDP-Listening-Threads sein. Der Standardwert ist **0**.
ShowSenderPort	Diese Option liefert die Portnummer des Absenders für jedes Auftreten der Aktionsvariablen $sender. Standardmäßig ist die Portnummer ausgeblendet. Wenn diese Option angegeben wird, wird die Portnummer unter Verwendung der Syntax **:***port-number* an das Ende der Adresse des Absenders angehängt.
HideWSAErrorBox	Unterdrückt das Fehlerfenster, in dem Fehler zu Socket-Vorgängen angezeigt werden.
CodePage= *number*	Multibyte-Zeichen werden basierend auf dieser Angabe in Unicode konvertiert. Diese Option gilt nur für Windows Mobile-Geräte.

Siehe auch

- „dblsn-Option -d“ auf Seite 61
- „dblsn-Option -a “ auf Seite 60

MobiLink Listener-Optionen für Windows-Geräte

Der MobiLink Listener kann mit den folgenden Optionen konfiguriert werden:

Option	Beschreibung
@{ *variable* | *filename* }	Wendet MobiLink Listener-Optionen der angegebenen Umgebungsvariablen oder Textdatei an. Siehe „dblsn-Option @data“ auf Seite 60.
-a *value*	Gibt eine einzelne Bibliotheksoption für eine Listener-Bibliothek an. Siehe „dblsn-Option -a “ auf Seite 60.
-d *filename*	Gibt eine Listener-Bibliothek an. Siehe „dblsn-Option -d“ auf Seite 61.
-e *device-name*	Gibt den Gerätenamen an. Siehe „dblsn-Option -e“ auf Seite 61.
-f *string*	Gibt zusätzliche Informationen zum Gerät an. Siehe „dblsn-Option -f “ auf Seite 62.
-gi *seconds*	Legt das IP-Tracking-Polling-Intervall fest. Siehe „dblsn-Option -gi “ auf Seite 62.
-i *seconds*	Gibt das Polling-Intervall für SMTP-Verbindungen an. Siehe „dblsn-Option -i“ auf Seite 62.
-l " *keyword=value;...* "	Legt einen Message-Handler fest und erstellt ihn. Siehe „dblsn-Option -l “ auf Seite 63.
-ls	Aktiviert SMS-Listener. Nur gültig für Windows Mobile. Siehe „dblsn-Option -ls“ auf Seite 63.
-lu	Aktiviert UDP-Listener. Siehe „dblsn-Option -lu“ auf Seite 64.
-m	Aktiviert die Nachrichtenprotokollierung. Siehe „dblsn-Option -m“ auf Seite 64.
-ni	Deaktiviert die IP-Protokollierung. Siehe „dblsn-Option -ni “ auf Seite 64.
-o *prefix*	Schreibt das Protokoll in eine Datei. Siehe „dblsn-Option -o “ auf Seite 65.
-os *bytes*	Legt die maximale Größe der Logdatei fest. Siehe „dblsn-Option -os “ auf Seite 65.
-ot *filename*	Kürzt eine Datei und protokolliert dann die Ausgabe in diese Datei. Siehe „dblsn-Option -ot“ auf Seite 65.

Option	Beschreibung
-p	Gestattet es dem Gerät, bei Inaktivität automatisch herunterzufahren. Siehe „dblsn-Option -p“ auf Seite 66.
-pc { **+** | **-** }	Aktiviert oder deaktiviert beständige Verbindungen. Siehe „dblsn-Option -pc “ auf Seite 66.
-q	Führt den MobiLink Listener im dialogfreien Modus aus. Siehe „dblsn-Option -q“ auf Seite 66.
-qi	Diese Optionseinstellung verhindert, dass das dblsn-Symbol und das Meldungsfenster angezeigt werden. Siehe „dblsn-Option -qi “ auf Seite 67.
-r *filename*	Identifiziert eine entfernte MobiLink-Datenbank, die an der Antwort eines Message-Handlers beteiligt ist. Siehe „dblsn-Option -r“ auf Seite 67.
-sv *script-version*	Legt die Skriptversion für die Authentifizierung fest. Siehe „dblsn-Option -sv “ auf Seite 67.
-t {**+** | **-** } *name*	Registriert die entfernte ID für eine entfernte Datenbank oder hebt die Registrierung auf. Siehe „dblsn-Option -t“ auf Seite 68.
-ts *session-name* (*session-option* = [*option-value*;...])	Richtet eine Protokollierungssitzung für den MobiLink Listener ein Siehe „dblsn-Option -ts“ auf Seite 68.
-u *username*	Gibt einen MobiLink-Benutzernamen an. Siehe „dblsn-Option -u “ auf Seite 69.
-v { **0** | **1** | **2** | **3** }	Legt die Ausführlichkeitsstufe für das Nachrichtenlog fest. Siehe „dblsn-Option -v“ auf Seite 70.
-w *password*	Legt ein MobiLink-Kennwort fest. Siehe „dblsn-Option -w“ auf Seite 71.
-x { **http** | **https** | **tcpip** } [**(** *protocol-option=value*;... **)**]	Legt das Netzwerkprotokoll und die MobiLink-Server-Protokolloptionen fest. Siehe „dblsn-Option -x“ auf Seite 71.
-y *newpassword*	Legt ein neues MobiLink-Kennwort fest. Siehe „dblsn-Option -y“ auf Seite 71.

dblsn-Option @data

Wendet MobiLink Listener-Optionen der angegebenen Umgebungsvariablen oder Textdatei an.

Syntax

dblsn @{ *variable* | *filename* } ...

Bemerkungen

Standardmäßig ist *dblsn.txt* die Argumentdatei, wenn der MobiLink Listener ohne Parameter ausgeführt wird.

Falls eine Datei und eine Umgebungsvariable mit demselben Namen vorhanden sind, wird die Umgebungsvariable verwendet.

Verwenden Sie das Dienstprogramm zum Verschleiern von Dateien, um Kennwörter und andere vertrauliche Daten in der Textdatei zu verschleiern.

Siehe auch

- „Konfigurationsdateien“ [*SQL Anywhere Server - Datenbankadministration*]
- „Dienstprogramm zum Verschleiern von Dateien (dbfhide)“ [*SQL Anywhere Server - Datenbankadministration*]

Beispiel

Eine Beispieldatei befindet sich unter *%SQLANYSAMP16%\MobiLink\SIS_SimpleListener\dblsn.txt*.

Im folgenden Beispiel werden Befehlszeilenoptionen in einer Textdatei namens **dblsnoptions** gespeichert:

```
dblsn @dblsnoptions
```

Im folgenden Beispiel werden Befehlszeilenoptionen in einer vollqualifizierten Textdatei namens **mydblsn.txt** gespeichert:

```
dblsn @mydblsn.txt
```

dblsn-Option -a

Gibt eine einzelne Bibliotheksoption für eine Listener-Bibliothek an.

Syntax

dblsn -a *value* ...

Bemerkungen

Standardmäßig verwendet der MobiLink Listener *lsn_udp16.dll*, wenn keine Bibliothek angegeben wird. Verwenden Sie die Option -d, um alternative oder zusätzliche Bibliotheken anzugeben.

Verwenden Sie den Wert **?**, um alle verfügbaren Bibliotheksoptionen anzuzeigen.

Verwenden Sie die Option -a mehrfach, um zusätzliche Bibliotheksoptionen festzulegen.

Siehe auch

- „Listener-Bibliotheken für Windows“ auf Seite 57
- „dblsn-Option -d“ auf Seite 61

Beispiel

Im folgenden Beispiel werden die Portoption und die Option ShowSenderPort in der Listener-Bibliothek *lsn_udp16.dll* festgelegt:

```
dblsn -d lsn_udp16.dll -a port=1234 -a ShowSenderPort
```

Im folgenden Beispiel wird die Portoption für zwei verschiedene Bibliotheken festgelegt:

```
dblsn -d lsn_udp16.dll -a port=1234 -d maac750.dll -a port=2345
```

Im folgenden Beispiel werden alle verfügbaren Bibliotheksoptionen in der Standardbibliothek angezeigt:

```
dblsn -a ?
```

dblsn-Option -d

Gibt eine Listener-Bibliothek an.

Syntax

dblsn -d *filename* ...

Bemerkungen

Standardmäßig verwendet der MobiLink Listener die *lsn_udp16.dll*-Listener-Bibliothek.

Verwenden Sie die Option -d mehrfach, um die Mehrkanalüberwachung und dadurch das Überwachen mehrerer Medien zu aktivieren.

Siehe auch

- „Listener-Bibliotheken für Windows“ auf Seite 57

Beispiel

Im folgenden Beispiel wird die Listener-Bibliothek *maac750.dll* angegeben:

```
dblsn -d maac750.dll
```

dblsn-Option -e

Gibt den Gerätenamen an.

Syntax

dblsn -e *device-name* ...

Bemerkungen

Standardmäßig wird der Gerätename automatisch vom Betriebssystem generiert.

Wenn eine Verbindung mit dem MobiLink-Server hergestellt wird, müssen Sie sicherstellen, dass alle Gerätenamen eindeutig sind.

Der Gerätename sollte auf Folgendes eingeschränkt werden:

- Alphanumerische Zeichen
- Punkt
- Doppelpunkt
- Minuszeichen
- Pluszeichen
- Unterstrich

Der Gerätename ist im MobiLink Listener-Fenster zu finden.

dblsn-Option -f

Gibt zusätzliche Informationen zum Gerät an.

Syntax

dblsn -f *string* ...

Bemerkungen

Standardmäßig ist diese Information die Versionsnummer des Betriebssystems, das auf dem Gerät läuft.

dblsn-Option -gi

Legt das IP-Tracking-Polling-Intervall fest.

Syntax

dblsn -gi *number* ...

Bemerkungen

Standardmäßig fragt der IP-Tracker alle 60 Sekunden ab.

dblsn-Option -i

Gibt das Polling-Intervall für SMTP-Verbindungen an.

Syntax

dblsn -i *number* ...

Bemerkungen

Die Option -i gibt die Frequenz an, mit der der MobiLink Listener auf Nachrichten prüft.

Bei SMTP-Verbindungen beträgt der Standardwert 30 Sekunden. Bei UDP-Verbindungen wird sofort eine Verbindung des MobiLink Listeners hergestellt.

Die Option -i kann für jede Listener-Bibliothek, die mit der Option -d angegeben wird, einmal verwendet werden.

Siehe auch

- „dblsn-Option -d“ auf Seite 61

Beispiel

Im folgenden Beispiel werden die Polling-Intervalle für zwei verschiedene Bibliotheken festgelegt:

```
dblsn -d lsn_udp16.dll -i 60 -d maac750.dll -i 45
```

dblsn-Option -l

Legt einen Message-Handler fest und erstellt ihn.

Syntax

dblsn -l "*keyword=value;...***"** …

Bemerkungen

Eine Liste der akzeptierten Schlüsselwörter finden Sie unter „MobiLink Listener-Schlüsselwörter für Windows-Geräte“ auf Seite 72.

Verwenden Sie die Option -l mehrfach, um zusätzliche Message-Handler für Push-Benachrichtigungen festzulegen. Message-Handler werden in der Reihenfolge verarbeitet, in der sie angegeben sind.

Siehe auch

- „Message-Handler“ auf Seite 15

dblsn-Option -ls

Aktiviert SMS-Listener. SMS-Listener sind nicht gesichert und sollten vermieden werden. Siehe „Gesichertes Listener-Deployment“ auf Seite 56.

Syntax

dblsn -ls …

Bemerkungen

Eine Liste der akzeptierten Schlüsselwörter finden Sie unter „MobiLink Listener-Schlüsselwörter für Windows-Geräte“ auf Seite 72.

SMS-Listener sind standardmäßig deaktiviert.

Siehe auch

- „dblsn-Option -x“ auf Seite 71

dblsn-Option -lu

Aktiviert UDP-Listener. UDP-Listener sind nicht gesichert und sollten vermieden werden. Siehe „Gesichertes Listener-Deployment“ auf Seite 56.

Syntax

dblsn -lu ...

Bemerkungen

Eine Liste der akzeptierten Schlüsselwörter finden Sie unter „MobiLink Listener-Schlüsselwörter für Windows-Geräte“ auf Seite 72.

UDP-Listener sind standardmäßig deaktiviert.

Siehe auch

- „dblsn-Option -x“ auf Seite 71

dblsn-Option -m

Aktiviert die Nachrichtenprotokollierung.

Syntax

dblsn -m ...

Bemerkungen

Standardmäßig ist die Nachrichtenprotokollierung deaktiviert.

dblsn-Option -ni

Deaktiviert die IP-Protokollierung.

Syntax

dblsn -ni ...

Bemerkungen

Standardmäßig ist die IP-Protokollierung aktiviert.

Diese Option stoppt die Zustellungsbestätigung nicht.

Die Option -ni deaktiviert die UDP-Adressprotokollierung, wenn sie mit der Option -x verwendet wird. Diese Funktion ist nützlich, wenn Sie das Device Tracking (Geräteprotokollierung) verwenden wollen, um UDP-Adressaktualisierungen auszuschließen.

Siehe auch

- „dblsn-Option -x“ auf Seite 71

dblsn-Option -o

Schreibt das Protokoll in eine Datei.

Syntax

dblsn -o *filename* …

Bemerkungen

Standardmäßig wird die Ausgabe im MobiLink Listener-Fenster protokolliert.

Siehe auch

- „dblsn-Option -ot“ auf Seite 65

dblsn-Option -os

Legt die maximale Größe der Logdatei fest.

Syntax

dblsn -os *bytes* …

Bemerkungen

Standardmäßig gibt es keine maximale Größenbegrenzung. Die minimale Größenbegrenzung beträgt 10000.

dblsn-Option -ot

Kürzt eine Datei und protokolliert dann die Ausgabe in diese Datei.

Syntax

dblsn -ot *filename* …

Bemerkungen

Der Dateiinhalt wird vor der Protokollierung der Ausgabe gelöscht.

Siehe auch

- „dblsn-Option -o “ auf Seite 65

dblsn-Option -p

Gestattet es dem Gerät, bei Inaktivität automatisch herunterzufahren.

Syntax

dblsn -p ...

Bemerkungen

Standardmäßig verhindert der MobiLink Listener, dass das Gerät heruntergefahren wird.

Diese Option gilt nur für Windows Mobile-Geräte.

dblsn-Option -pc

Syntax

Aktiviert oder deaktiviert beständige Verbindungen.

dblsn -pc { **+** | **-** } ...

Bemerkungen

Standardmäßig sind beständige Verbindungen aktiviert. Der Parameter **-** deaktiviert beständige Verbindungen. Der Parameter **+** aktiviert sie.

Durch die Deaktivierung beständiger Verbindungen wird verhindert, dass der MobiLink Listener Push-Benachrichtigungen empfängt. Kurzlebige beständige Verbindungen für Device Tracking (Geräteprotokollierung) und Bestätigung sind jedoch gestattet.

Wenn eine beständige Verbindung unterbrochen wird, versucht der MobiLink Listener fortlaufend, die Verbindung wieder aufzubauen.

Beispiel

Im folgenden Beispiel werden beständige Verbindungen deaktiviert:

```
dblsn -pc-
```

dblsn-Option -q

Führt den MobiLink Listener im dialogfreien Modus aus.

Syntax

dblsn -q ...

Bemerkungen

Die Option -q minimiert das MobiLink Listener-Fenster. Standardmäßig wird das MobiLink Listener-Fenster angezeigt.

dblsn-Option -qi

Diese Optionseinstellung verhindert, dass das dblsn-Symbol und das Meldungsfenster angezeigt werden.

Syntax

dblsn -qi ...

Bemerkungen

Bei dieser Option gibt es nach dem Start keinen Hinweis darauf, dass dblsn läuft, abgesehen von eventuellen Fehlermeldungen beim Start. Sie können die Logdateien -o verwenden, um Fehler zu diagnostizieren.

Siehe auch

- „dblsn-Option -o “ auf Seite 65

dblsn-Option -r

Identifiziert eine entfernte MobiLink-Datenbank, die an der Antwort eines Message-Handlers beteiligt ist.

Syntax

dblsn -r *filename* ...

Bemerkungen

Der *filename* muss den vollständigen Pfad einer RID-Datei enthalten. Diese Datei wird von dbmlsync nach der ersten Synchronisation automatisch erstellt. Sie hat denselben Speicherort und Namen wie die Datenbankdatei. Bei UltraLite-Datenbanken sollte der *filename* dem Datenbanknamen entsprechen.

Wenn die Option -r verwendet wird, kann die Aktionsvariable $remote_id in Message-Handlern verwendet werden, um die entfernte ID in der RID-Datei zu referenzieren. Standardmäßig handelt es sich bei entfernten IDs um GUIDs.

Verwenden Sie die Option -r mehrfach, um mehrere Datenbanken festzulegen.

Siehe auch

- „Nachrichtenfilter“ auf Seite 16
- „MobiLink Listener-Aktionsbefehle für Windows-Geräte“ auf Seite 74

dblsn-Option -sv

Legt die Skriptversion für die Authentifizierung fest.

Syntax

dblsn -sv *script-version* ...

Bemerkungen

Standardmäßig verwendet der MobiLink Listener die ml_global-Serverskriptversion, sofern sie definiert ist.

dblsn-Option -t

Registriert die entfernte ID für eine entfernte Datenbank oder hebt die Registrierung auf.

Syntax

dblsn -t { **+** | **-** } *name* ...

Bemerkungen

Der Parameter + registriert eine entfernte ID. Der Parameter - hebt die Registrierung einer ID auf.

Die Registrierung gestattet es dem MobiLink Listener, Push-Benachrichtigungen durch die Referenzierung der betreffenden entfernten ID zu adressieren.

Wenn die Device Tracking-Informationen erfolgreich hochgeladen wurden, werden registrierte IDs in der ml_listening-Systemtabelle auf dem Server beibehalten. Sie müssen die ID nur einmal registrieren.

Verwenden Sie die Option -r mehrfach, um mehrere IDs zu registrieren oder ihre Registrierung aufzuheben. Die Registrierung mehrere IDs ist nützlich für die Adressierung von Push-Benachrichtigungen an mehrere entfernte Datenbanken.

Siehe auch

- „MobiLink Listener-Aktionsbefehle für Windows-Geräte“ auf Seite 74

dblsn-Option -ts

Richtet eine Protokollierungssitzung für den MobiLink Listener ein

Syntax

dblsn -ts *session-name*(*session-option*=[*option-value*;...])

Der Sitzungsname muss **logging** lauten.

Sitzungsoption	Wert der Option
events	Kommagetrennte Liste der System-Trace-Ereignisse. Die unterstützten Ereignisse sind "Info", "Warning" und "Error".
targets	*target-type***(***target-option*=*value***[;...])**, wobei *target-type* nur **file** sein kann.

Die Zieloptionen werden als Name-Wert-Paare angegeben. Die Zieldatei kann die folgenden Optionen aufweisen:

Zieloptionsname	Erwarteter Wert	Beschreibung
filename_prefix	Zeichenfolge	Ein ETD-Dateinamenpräfix mit oder ohne Pfad. Alle ETD-Dateien haben die Erweiterung *.etd*. Dieser Parameter ist erforderlich.
max_size	Integer	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	Integer	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 Listener, 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. Wenn dieser Parameter aktiviert ist, kann die Performance des MobiLink Listener 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_si
ze=10000000;num_files=10;flush_on_write=true}}
```

dblsn-Option -u

Legt einen MobiLink-Benutzernamen für den MobiLink Listener fest.

Syntax

dblsn -u *username* ...

Bemerkungen

Standardmäßig ist der Benutzername *device-name*-**dblsn**, wobei *device-name* der Name Ihres Geräts ist. Sie können einen Gerätenamen mit der Option -e angeben.

Der MobiLink Listener verwendet den Benutzernamen, um eine Verbindung mit dem MobiLink-Server für Device Tracking, Bestätigungen und beständige Verbindungen herzustellen.

Der Benutzername muss ein eindeutiger MobiLink-Benutzername sein, der beim MobiLink-Server registriert wurde. Der Name muss in der Tabelle ml_user in der konsolidierten Datenbank vorhanden sein.

Siehe auch

- „dblsn-Option -e“ auf Seite 61
- „dblsn-Option -w“ auf Seite 71
- „Erstellen und Registrieren von MobiLink-Benutzern“ [*MobiLink - Clientadministration*]

dblsn-Option -v

Legt die Ausführlichkeitsstufe für das Nachrichtenlog fest.

Syntax

dblsn -v { **0** | **1** | **2** | **3** } ...

Bemerkungen

Standardmäßig ist die Ausführlichkeitsstufe auf 0 festgelegt.

In der folgenden Tabelle sind die Werte für die möglichen Ausführlichkeitsstufen angegeben.

Ausführlichkeitsstufe	Beschreibung
0	Ausführlichkeitsstufe ist deaktiviert.
1	Die Nachrichten der Listener-Bibliothek, die grundlegenden Stufen der Aktionsprotokollierung und die Befehlszeilenoptionen werden angezeigt.
2	Die Nachrichten der Ausführlichkeitsstufe 1 und detaillierte Stufen der Aktionsprotokollierung werden angezeigt.
3	Die Nachrichten der Ausführlichkeitsstufe 2, Polling- und Überwachungsstatus werden angezeigt.

Um Push-Benachrichtigungen in das Nachrichtenlog zu schreiben, muss die Option -m angegeben werden.

Siehe auch

- „dblsn-Option -m“ auf Seite 64
- „Datenbankservermeldungen in einer Datei protokollieren“ [*SQL Anywhere Server - Datenbankadministration*]

dblsn-Option -w

Legt ein MobiLink-Kennwort fest.

Syntax

dblsn -w *password* ...

Bemerkungen

Kennwörter müssen beim MobiLink-Server unter dem zugeordneten MobiLink-Benutzernamen registriert werden.

Der MobiLink Listener verwendet das Kennwort, um eine Verbindung mit dem MobiLink-Server für Device Tracking, Bestätigungen und beständige Verbindungen herzustellen.

Siehe auch

- „dblsn-Option -u “ auf Seite 69

dblsn-Option -x

Legt das Netzwerkprotokoll und die MobiLink-Server-Protokolloptionen fest.

Syntax

dblsn -x { **http** | **https** | **tcpip** } [**(***protocol-option***=***value*;...**)**] ...

Bemerkungen

Damit der MobiLink Listener Device Tracking-Informationen und Zustellungsbestätigungen an die konsolidierte Datenbank senden kann, ist eine Verbindung mit dem MobiLink-Server erforderlich. Verwenden Sie die Protokolloption **host**, um den Standort des MobiLink-Servers festzulegen. Siehe „host“ [*MobiLink - Clientadministration*].

Die Option -x gestattet es dem Gerät, die konsolidierte Datenbank zu aktualisieren, wenn die Serveradresse geändert wird.

Siehe auch

- „Netzwerkprotokolloptionen des MobiLink-Clients“ [*MobiLink - Clientadministration*]

dblsn-Option -y

Legt ein neues MobiLink-Kennwort fest.

Syntax

dblsn -y *newpassword* ...

Bemerkungen

Die Option -y kann nicht verwendet werden, wenn das Authentifizierungssystem keine Änderungen der Kennwörter der entfernten Geräte zulässt.

MobiLink Listener-Schlüsselwörter für Windows-Geräte

Die folgenden Schlüsselwörter können verwendet werden, um Message-Handler zu konfigurieren, die mit der dblsn-Option -l erstellt wurden. Weitere Hinweise zur Verwendung von MobiLink Listener-Schlüsselwörtern finden Sie unter „dblsn-Option -l “ auf Seite 63.

Filterschlüsselwörter

Mit den folgenden Schlüsselwörtern können Sie Nachrichten in einer Push-Benachrichtigung filtern:

Schlüsselwortsyntax	Beschreibung
subject=*subject-string*	Filtert eine Nachricht, wenn der Betreff dem Text von *subject-string* entspricht.
content=*content-string*	Filtert eine Nachricht, wenn der Inhalt dem Text von *content-string* entspricht.
message=*message-string*	Filtert eine Nachricht, wenn die gesamte Nachricht dem Text von *message-string* entspricht.
message_start=*message-string*	Filtert eine Nachricht, wenn sie mit *message-string* beginnt.
sender=*sender-string*	Filtert eine Nachricht, wenn sie von *sender-string* gesendet wurde.

Aktionsschlüsselwörter

Mit den folgenden Schlüsselwörtern können Sie eine Aktion initiieren, wenn eine Filterbedingung erfüllt ist:

Schlüsselwortsyntax	Beschreibung
action=*command*	Legt einen Aktionsbefehl fest. Siehe „MobiLink Listener-Aktionsbefehle für Windows-Geräte“ auf Seite 74.

Schlüsselwortsyntax	Beschreibung
altaction=*command*	Legt einen alternativen Aktionsbefehl fest, der initiiert wird, wenn der Aktionsbefehl fehlschlägt. Siehe „MobiLink Listener-Aktionsbefehle für Windows-Geräte“ auf Seite 74.

Polling-Optionen

Mit den folgenden Optionen können Sie ein Lightweight-Polling-Modul konfigurieren :

Schlüsselwortsyntax	Beschreibung
poll_connect={ **http** | **https** | **tcpip** } [**(***protocol-option*=*value*;...**)**]	Legt die Protokolloptionen für das Lightweight-Netzwerk fest, die erforderlich sind, um eine Verbindung mit dem Server herzustellen. Der Standardwert wird von der dblsn-Option -x abgeleitet. Siehe „dblsn-Option -x“ auf Seite 71.
poll_notifier=*Notifier-string*	Legt den Namen des Notifiers fest, der Push-Anforderungen verarbeitet. Erforderlich.
poll_key=*key-string*	Gibt den Namen an, mit dem sich der MobiLink Listener beim Notifier identifiziert. Dieser Wert muss eindeutig sein. Erforderlich.
poll_every=*seconds-number*	Legt fest, wie häufig der MobiLink Listener den Server abfragen soll. Das Intervall wird in Sekunden gemessen. Der Standardwert wird automatisch vom MobiLink-Server abgerufen.

Optionen

Die folgenden Schlüsselwörter können verwendet werden, um das Verhalten von Message-Handlern zu konfigurieren.

Schlüsselwortsyntax	Beschreibung
continue=[**yes** | **no**]	Legt fest, ob der MobiLink Listener mit der Überwachung fortfahren soll, nachdem er eine erste Übereinstimmung gefunden hat. Der Standardwert ist **no**. Der Wert **yes** ist nützlich, wenn mehrere Filter angegeben werden und eine Nachricht mehrere Aktionen initiiert.
confirm_action=[**yes** | **no**]	Legt fest, ob der Filter die Aktion bestätigen soll. Der Standardwert ist **yes**.

Schlüsselwortsyntax	Beschreibung
confirm_delivery=[**yes** | **no**]	Legt fest, ob der Filter die qualifizierte Nachrichtenzustellung bestätigen soll. Der Standardwert ist **yes**, sodass die Zustellungsbestätigung gesendet wird, nachdem der erste Filter die Nachricht akzeptiert. Die Zustellung kann nur bestätigt werden, wenn die Nachricht eine Bestätigung erfordert und der Filter die Nachricht akzeptiert. Eine Nachricht erfordert eine Bestätigung, wenn der Wert des Schlüsselworts confirm_delivery des angegebenen Gateways auf **yes** festgelegt ist. Der Wert **no** kann verwendet werden, wenn mehrere Filter dieselbe Nachricht akzeptieren, sodass Sie eine bessere Kontrolle darüber haben, welcher Filter die Zustellung bestätigen soll. Weitere Informationen zum Verarbeiten von Zustellungsbestätigungen auf dem Server finden Sie unter „confirmation_handler-Ereignis“ auf Seite 43.
maydial=[**yes** | **no**]	Legt fest, ob die Aktion die Berechtigung hat, das Modem zu wählen. Der Standardwert ist **yes**. Der Wert **no** bewirkt, dass der MobiLink Listener das Modem vor der Aktion freigibt.

Siehe auch

- „Message-Handler“ auf Seite 15
- „MobiLink Listener-Aktionsbefehle für Windows-Geräte“ auf Seite 74

MobiLink Listener-Aktionsbefehle für Windows-Geräte

Beim Konfigurieren eines neuen Message-Handlers wird eine Aktion festgelegt. Wenn eine Filterbedingung erfüllt ist, wird eine Aktion initiiert. Wenn die Aktion fehlschlägt, wird eine alternative Aktion initiiert. Aktionen werden mit dem Schlüsselwort **action** festgelegt. Alternative Aktionen werden mit dem Schlüsselwort **altaction** festgelegt.

Die folgende Liste enthält Aktionsbefehle:

Befehl	Beschreibung
START *program arglist*	Startet ein Programm, während der MobiLink Listener im Hintergrund ausgeführt wird. Siehe „START-Aktionsbefehl“ auf Seite 75.

Befehl	Beschreibung
RUN *program arglist*	Unterbricht den MobiLink Listener, um ein Programm auszuführen. Siehe „RUN-Aktionsbefehl“ auf Seite 76.
POST *windowmessage* | *id* to *windowclass* | *windowtitle*	Sendet eine Fensternachricht an eine Fensterklasse. Siehe „POST-Aktionsbefehl“ auf Seite 76.
SOCKET port=*windowname*[;**host**=*hostname*][;**sendText**=*text*][;**recvText**=*text*[;**timeout**=*seconds*]]	Sendet eine Nachricht über eine TCP/IP-Verbindung an eine Anwendung. Siehe „SOCKET-Aktionsbefehl“ auf Seite 77.
DBLSN FULL SHUTDOWN	Zwingt den MobiLink Listener herunterzufahren. Siehe „DBLSN FULL SHUTDOWN-Aktionsbefehl“ auf Seite 78.

Sie können nur eine Aktion pro **action**- oder **altaction**-Schlüsselwort angeben. Wenn eine Aktion mehrere Aufgaben ausführen soll, erstellen Sie eine Batchdatei, die mehrere Befehle enthält, und führen Sie die Datei mit dem Aktionsbefehl **RUN** aus.

Siehe auch

- „Initiierung von Aktionen“ auf Seite 17
- „MobiLink Listener-Schlüsselwörter für Windows-Geräte“ auf Seite 72

START-Aktionsbefehl

Startet ein Programm, während der MobiLink Listener im Hintergrund ausgeführt wird.

Syntax

action='START *program arglist*'

Bemerkungen

Wenn Sie ein Programm starten, fährt der MobiLink Listener damit fort, auf weitere Push-Benachrichtigungen zu prüfen.

Der MobiLink Listener wartet nicht auf das Ausführungsende des Programms, sodass er nur ermitteln kann, ob ein Aktionsbefehl fehlgeschlagen ist oder ob er das angegebene Programm starten kann.

Beispiel

Das folgende Beispiel startet dbmlsync mit einigen Befehlszeilenparametern, die zum Teil mit der Aktionsvariablen $content aus der Nachricht ermittelt werden:

```
dblsn -l "action='start dbmlsync.exe @dbmlsync.txt -n
  $content -wc dbmlsync_$content -e sch=INFINITE';"
```


RUN-Aktionsbefehl

Unterbricht den MobiLink Listener, um ein Programm auszuführen.

Syntax

action='RUN *program arglist*'

Bemerkungen

Der MobiLink Listener wartet auf das Ausführungsende des Programms und fährt dann mit der Überwachung fort.

Wenn ein Programm ausgeführt wird, stellt der MobiLink Listener fest, ob das Programm fehlgeschlagen ist, ob der MobiLink Listener das Programm starten kann oder ob der Rückgabecode des Programms ungleich NULL ist.

Beispiel

Das folgende Beispiel führt dbmlsync mit einigen Befehlszeilenparametern aus, die zum Teil mit der Aktionsvariablen $content aus der Nachricht ermittelt werden:

```
dblsn -l "action='run dbmlsync.exe @dbmlsync.txt -n $content';"
```

POST-Aktionsbefehl

Sendet eine Fensternachricht an eine Fensterklasse.

Syntax

action='POST *windowmessage* | *id* to *windowclass* | *windowtitle*'

Bemerkungen

Der POST-Befehl kann verwendet werden, um Anwendungen Signale zu senden, die Fensternachrichten verwenden.

Sie können die Fensternachricht anhand des Nachrichteninhalts oder der Fensternachrichten-ID, sofern vorhanden, identifizieren.

Sie können die Fensterklasse anhand ihres Klassennamens oder des Fenstertitels identifizieren. Wenn Sie die Fensterklasse anhand ihres Namens identifizieren, können Sie mit der dbmlsync-Option -wc den Namen der Fensterklasse festlegen. Wenn Sie die Fensterklasse anhand des Fenstertitels identifizieren, können Sie sie nur mit dem Fenster auf der obersten Ebene referenzieren.

Wenn die Fensternachricht oder der Fensterklassenname nicht-alphanumerische Zeichen enthält, wie etwa Leerstellen oder Satzzeichen, setzen Sie den Text in Apostrophe ('). Das Escapezeichen ist ebenfalls ein Apostroph, sodass Sie einem Apostroph innerhalb der Fensternachricht oder des Fensterklassennamens einen zweiten Apostroph (") voranstellen müssen.

Folgendes ist bei POST gültig:

- **Per Dezimal-ID senden** Zum Beispiel: `post 999 to <wc|wt>`
- **Per hex-ID senden** Zum Beispiel: `post 0x3E7 to <wc|wt>`
- **Per registriertem Nachrichtennamen senden** Zum Beispiel: `post myRegisteredMsgName to <wc|wt>`

Beispiel

Um die Verwendung von Fenstern und Nachrichten mit Apostrophen zu veranschaulichen, wird im folgenden Beispiel die Fensternachricht mike's_message an die Fensterklasse mike's_class gesendet:

```
dblsn -l "action='post mike''s_message to mike''s_class';"
```

Im folgenden Beispiel wird die Fensternachricht dbas_synchronize an eine dbmlsync-Instanz gesendet, die mit dem Klassennamen dbmlsync_FullSync registriert wurde.

```
dblsn -l "action='post dbas_synchronize to dbmlsync_FullSync';"
```

Siehe auch

- „dbmlsync-Option -wc“ [*MobiLink - Clientadministration*]

SOCKET-Aktionsbefehl

Sendet eine Nachricht über eine TCP/IP-Verbindung an eine Anwendung.

Syntax

action='SOCKET port=*windowname*[;**host=***hostname*][;**sendText=***text*][;**recvText=***text*[;**timeout=***seconds*]]**'**

Bemerkungen

Der SOCKET-Befehl wird verwendet, um dynamische Informationen an eine laufende Anwendung weiterzuleiten und um Nachrichten in Java- und Visual Basic-Anwendungen zu integrieren. Beide Sprachen unterstützen keine Verarbeitung von benutzerdefinierten Fensternachrichten und eMbedded Visual Basic unterstützt keine Befehlszeilenparameter.

Um eine Verbindung mit einem Socket herzustellen, müssen Sie den Port und den Host angeben. Geben Sie Ihre Nachricht mithilfe von sendText ein.

Verwenden Sie recvText, um eine Nachricht anzuzeigen, wenn bestätigt wird, dass sendText erfolgreich von der Anwendung empfangen wurde. Bei der Verwendung von recvText können Sie eine Timeoutgrenze festlegen. Die Aktion schlägt fehl, wenn der MobiLink Listener innerhalb des Timeouts keine Verbindung herstellen, keine Bestätigungen senden oder keine Bestätigungen empfangen kann.

Beispiel

Im folgenden Beispiel wird die Zeichenfolge in $sender=$message an eine lokale Anwendung weitergeleitet, die Port 12345 überwacht. Der MobiLinkListener erwartet, dass die Anwendung innerhalb von 5 Sekunden "beeperAck" als Bestätigung sendet.


```
dblsn -l "action='socket port=12345;
   sendText=$sender=$message;
   recvText=beeperAck;
   timeout=5'"
```

DBLSN FULL SHUTDOWN-Aktionsbefehl

Zwingt den MobiLink Listener herunterzufahren.

Syntax

action='DBLSN FULL SHUTDOWN'

Bemerkungen

Nach dem Herunterfahren bearbeitet der MobiLinkListener keine weiteren Push-Benachrichtigungen mehr und hält die Synchronisation von Device Tracking-Informationen an. Sie müssen den MobiLink Listener neu starten, um mit der serverinitiierten Synchronisation fortzufahren.

MobiLink Listener-Aktionsvariablen für Windows-Geräte

Die folgenden Aktionsvariablen können in einer Aktion oder einem Filter verwendet werden. Der Wert wird in der Aktionsvariablen vor der Initiierung des Message-Handlers eingesetzt.

Aktionsvariablen beginnen mit einem Dollarzeichen ($). Das Dollarzeichen wird auch als Escapezeichen verwendet, sodass Sie zwei Dollarzeichen ($$) angeben müssen, wenn Sie ein einzelnes Dollarzeichen als normalen Text verwenden wollen.

Variable	Beschreibung
$subject	Der Betreff der Nachricht.
$content	Der Inhalt der Nachricht.
$message	Eine gesamte Nachricht, einschließlich Betreff, Inhalt und Absender.
$message_start	Der erste Teil einer Nachricht, der vom Filterschlüsselwort message_start angegeben wird. Diese Variable ist nur verfügbar, wenn Sie das Filterschlüsselwort message_start angegeben haben. Siehe „MobiLink Listener-Schlüsselwörter für Windows-Geräte“ auf Seite 72.
$message_end	Der Teil einer Nachricht, der nicht im Filterschlüsselwort message_start enthalten ist. Diese Variable ist nur verfügbar, wenn Sie das Filterschlüsselwort message_start angegeben haben. Siehe „MobiLink Listener-Schlüsselwörter für Windows-Geräte“ auf Seite 72.

Variable	Beschreibung
$ml_connect	Die MobiLink-Netzwerk-Protokolloptionen, wie durch die dblsn-Option -x festgelegt. Standardwert ist eine leere Zeichenfolge. Siehe „dblsn-Option -x“ auf Seite 71.
$ml_user	Der MobiLink-Benutzername, wie durch die dblsn-Option -u festgelegt. Der Standardname lautet *device-name*-**dblsn**.
$ml_password	Das MobiLink-Kennwort, wie durch die dblsn-Option -w festgelegt, oder das neue MobiLink-Kennwort, falls -y verwendet wird.
$priority	Die Bedeutung dieser Variablen ist von der Netzbetreiberbibliothek abhängig.
$request_id	Die Anforderungs-ID, die in einer Push-Anforderung angegeben wurde. Siehe „Push-Anforderungen“ auf Seite 7.
$remote_id	Die entfernte ID. Diese Variable kann nur verwendet werden, wenn Option dblsn -r angegeben wurde. Siehe „Entfernte ID als Filter“ auf Seite 19.
$sender	Der Absender der Nachricht.
$type	Die Bedeutung dieser Variablen ist von der Netzbetreiberbibliothek abhängig.
$year	Die Bedeutung dieser Variablen ist von der Netzbetreiberbibliothek abhängig.
$month	Die Bedeutung dieser Variablen ist von der Netzbetreiberbibliothek abhängig. Werte können zwischen 1 und 12 liegen.
$day	Die Bedeutung dieser Variablen ist von der Netzbetreiberbibliothek abhängig. Werte können zwischen 1 und 31 liegen.
$hour	Die Bedeutung dieser Variablen ist von der Netzbetreiberbibliothek abhängig. Werte können zwischen 0 und 23 liegen.
$Minute	Die Bedeutung dieser Variablen ist von der Netzbetreiberbibliothek abhängig. Werte können zwischen 0 und 59 liegen.
$second	Die Bedeutung dieser Variablen ist von der Netzbetreiberbibliothek abhängig. Werte können zwischen 0 und 59 liegen.
$best_adapter_mac	Die MAC-Adresse des besten NIC zum Erreichen des MobiLink-Servers, der mit der dblsn-Option -x angegeben wurde. Wenn die beste Route nicht über einen NIC führt, ist der Wert eine leere Zeichenfolge.

Variable	Beschreibung
$best_adapter_name	Der Adaptername des besten NIC zum Erreichen des MobiLink-Servers, der mit der dblsn-Option -x angegeben wurde. Wenn die beste Route nicht über einen NIC führt, ist der Wert eine leere Zeichenfolge.
$best_ip	Die IP-Adresse der besten IP-Schnittstelle zum Erreichen des MobiLink-Servers, der mit der dblsn-Option -x angegeben wurde. Wenn dieser Server nicht erreichbar ist, ist der Wert 0.0.0.0.
$best_network_name	Der RAS- oder Einwähl-Profilname des besten Profils zum Erreichen des MobiLink-Servers, der mit der dblsn-Option -x angegeben wurde. Wenn die beste Route nicht über eine RAS-/Einwählverbindung führt, ist der Wert eine leere Zeichenfolge.
$adapters	Eine Liste der Namen aktiver Netzwerkadapter, die jeweils durch einen senkrechten Strich (|) voneinander getrennt sind.
$network_names	Eine Liste der Namen verbundener RAS-Einträge, die jeweils durch einen senkrechten Strich (|) voneinander getrennt sind. Die Namen von RAS-Einträgen werden manchmal auch Einwähleintragnamen oder Einwählnetzwerk (Dial-Up Network, DUN) genannt.
$poll_connect	Die MobiLink-Netzwerk-Protokolloptionen, wie durch das Polling-Schlüsselwort poll_connect festgelegt. Standardwert ist eine leere Zeichenfolge. Siehe „MobiLink Listener-Schlüsselwörter für Windows-Geräte“ auf Seite 72.
$poll_notifier	Der Name des Notifiers, wie durch das Polling-Schlüsselwort poll_connect festgelegt. Siehe „MobiLink Listener-Schlüsselwörter für Windows-Geräte“ auf Seite 72.
$poll_key	Der Polling-Schlüssel, wie durch das Polling-Schlüsselwort poll_key festgelegt. Siehe „MobiLink Listener-Schlüsselwörter für Windows-Geräte“ auf Seite 72.
$poll_every	Die Polling-Häufigkeit, wie durch das Polling-Schlüsselwort poll_every festgelegt. Siehe „MobiLink Listener-Schlüsselwörter für Windows-Geräte“ auf Seite 72.

Siehe auch

- „dblsn-Option -l “ auf Seite 63
- „MobiLink Listener-Schlüsselwörter für Windows-Geräte“ auf Seite 72
- „MobiLink Listener-Aktionsbefehle für Windows-Geräte“ auf Seite 74

Beispiel

In den folgenden Beispielen wird die Aktionsvariablen $message_end verwendet, um zu ermitteln, welche Publikation synchronisiert werden soll:

```
dblsn -l "message_start=start-of-message;action='run dbmlsync.exe -c ... -n
$message_end'"
```


Lightweight-Polling-API

Die Lightweight-Polling-API ist eine Programmierschnittstelle, die Sie in die Geräteanwendung integrieren können. Sie enthält die für das Polling eines Servers erforderlichen Methoden.

Erforderliche Dateien

Alle Verzeichnisse sind relativ zum Verzeichnis *%SQLANY16%*. Die im Folgenden aufgelisteten Dateien sind erforderlich, um die Lightweight-Polling-API zu kompilieren:

Dateiname und Speicherort	Beschreibung
Bin32\mllplib16.dll	Dynamische Laufzeitbibliothek der Lightweight-Polling-API
SDK\Lib\x86\mllplib16.lib und *SDK\Lib\x64\mllplib16.lib*	Import-Laufzeitbibliothek der Lightweight-Polling-API
SDK\Include\mllplib.h	Header-Datei der Lightweight-Polling-API

Die in der Programmiersprache C geschriebene Beispielanwendung SIS_CarDealer_LP wird in *%SQLANYSAMP16%\MobiLink\SIS_CarDealer_LP_API* bereitgestellt.

API-Mitglieder

Methode	Beschreibung
„MLLightPoller-Klasse“	Stellt ein Objekt eines Lightweight-Polling-Moduls dar.
„MLLPCreatePoller-Methode“	Erstellt eine Instanz von MLLightPoller.
„MLLPDestroyPoller-Methode“	Vernichtet eine Instanz von MLLightPoller.

MLLightPoller-Klasse

Stellt ein Objekt eines Lightweight-Polling-Moduls dar.

Syntax

```
public class MLLightPoller
```

Mitglieder

Name	Beschreibung
„Poll-Methode“	Ruft den Server ab und fordert einen Notifier auf, einen Cache auf Push-Anforderungen zu prüfen.

Name	Beschreibung
„SetConnectInfo-Methode“	Richtet einen Datenstromtyp und Netzwerk-Protokolloptionen für den MobiLink-Client ein.
„auth_status-Enumeration“	Gibt mögliche Codes für den Authentifizierungsstatus an.
„return_code-Enumeration“	Gibt mögliche Rückgabecodes an.

Beispiel

```
MLLightPoller * poller = MLLCreatePoller();
```

Poll-Methode

Ruft den Server ab und fordert einen Notifier auf, einen Cache auf Push-Anforderungen zu prüfen.

Syntax

```
public virtual return_code MLLightPoller::Poll(
    const char * notifier,
    const char * key,
    char * subject = 0,
    size_t * subjectSize = 0,
    char * content = 0,
    size_t * contentSize = 0
)
```

Parameter

- **notifier** Der Name des Notifiers.
- **key** Der Name des Polling-Schlüssels, der den MobiLink Listener identifiziert.
- **subject** Der Puffer, der den Betreff einer Nachricht empfangen soll. (mit NULL abgeschlossen).
- **subjectSize** IN: Die Größe des Betreffpuffers.

  OUT: Die Größe des empfangenen Betreffs, einschließlich eines Nullabschlusszeichens, wobei Null einen leeren Betreff anzeigt.
- **content** Der Puffer, der den Inhalt einer Nachricht empfangen soll (mit NULL abgeschlossen).
- **contentSize** IN: Die Größe des Inhaltspuffers.

  OUT: Die Größe des empfangenen Inhalts, einschließlich eines Nullabschlusszeichens, wobei Null einen leeren Inhalt anzeigt.

Rückgabe

Einer der in der return_code-Enumeration angegebenen Codes. Siehe „return_code-Enumeration“ auf Seite 86.

Bemerkungen

Der MobiLink Listener stellt eine Verbindung mit dem Notifier her und trennt die Verbindung dann, nachdem der Notifier seinen Cache auf eine Push-Benachrichtigung für den angegebenen Polling-Schlüssel geprüft hat.

SetConnectInfo-Methode

Richtet einen Datenstromtyp und Netzwerk-Protokolloptionen für den MobiLink-Client ein.

Syntax

```
public virtual return_code MLLightPoller::SetConnectInfo(
    const char * streamName,
    const char * streamParams
);
```

Parameter

- **streamName** Das zu verwendende Netzwerkprotokoll. Akzeptierte Werte sind: **tcpip**, **http**, **https** und **tls**.
- **streamParams** Eine Zeichenfolge von Protokolloptionen in einer durch Semikola getrennten Liste. Eine vollständige Liste der Optionen finden Sie unter „Netzwerkprotokolloptionen des MobiLink-Clients“ [*MobiLink - Clientadministration*].

Rückgabe

Einer der in der return_code-Enumeration angegebenen Codes. Siehe „return_code-Enumeration“ auf Seite 86.

Beispiel

```
poller_ret = poller->SetConnectInfo("http", "host=localhost;port=80;")
```

auth_status-Enumeration

Gibt mögliche Codes für den Authentifizierungsstatus an.

Syntax

```
public typedef enum MLLightPoller::auth_status;
```

Mitglieder

Mitgliedsname	Beschreibung
AUTH_EXPIRED	Authentifizierung ist abgelaufen.
AUTH_IN_USE	Der Benutzername wurde bereits authentifiziert.
AUTH_INVALID	Die Authentifizierung ist ungültig.

Mitgliedsname	Beschreibung
AUTH_UNKNOWN	Die Authentifizierung ist unbekannt.
AUTH_VALID	Die Authentifizierung ist gültig.
AUTH_VALID_BUT_EXPIRES_SOON	Die Authentifizierung ist gültig, läuft aber demnächst ab.

return_code-Enumeration

Gibt mögliche Rückgabecodes an.

Syntax

```
public typedef enum MLLightPoller::return_code;
```

Mitglieder

Name	Beschreibung
AUTH_FAILED	Die Verbindung mit dem MobiLink-Server konnte nicht authentifiziert werden.
BAD_STREAM_NAME	Nicht erkannter Datenstromname.
BAD_STREAM_PARAM	Syntaktische Analyse eines Datenstromparameters ist fehlgeschlagen.
COMMUNICATION_ERROR	Aufgrund eines Verbindungsfehlers fehlgeschlagen.
CONNECT_FAILED	Verbindung mit dem MobiLink-Server konnte nicht hergestellt werden.
CONTENT_OVERFLOW	Der Inhaltspuffer ist zu klein.
KEY_NOT_FOUND	Es ist keine Push-Benachrichtigung für den angegebenen Schlüssel vom angegebenen Notifier vorhanden.
NYI	Wird nur intern verwendet.
OK	Methode wurde erfolgreich ausgeführt.
SUBJECT_OVERFLOW	Der Betreffpuffer ist zu klein.

MLLPCreatePoller-Methode

Erstellt eine Instanz von MLLightPoller.

Syntax

```
extern MLLightPoller * MLLPCreatePoller()
```

Rückgabe

Ein neues MLLightPoller-Objekt.

Siehe auch

- „MLLPDestroyPoller-Methode“ auf Seite 87

Beispiel

```
poller = MLLPCreatePoller();
```

MLLPDestroyPoller-Methode

Vernichtet eine Instanz von MLLightPoller.

Syntax

```
extern void MLLPDestroyPoller(
    MLLightPoller * poller
)
```

Parameter

- **poller** Der MLLightPoller, der zerstört werden soll.

Bemerkungen

Diese Methode akzeptiert NULL-MLLightPoller-Objekte.

Siehe auch

- „MLLPCreatePoller-Methode“ auf Seite 87

Beispiel

```
MLLPDestroyPoller(poller);
```


Systemprozeduren für die serverinitiierte Synchronisation

Mit Systemprozeduren für die serverinitiierte Synchronisation werden in den MobiLink-Systemtabellen Zeilen hinzugefügt und gelöscht.

Hinweis
Diese Systemprozeduren werden für das Device Tracking verwendet. Wenn Sie entfernte Geräte verwenden, die das automatische Device Tracking unterstützen, benötigen Sie diese Systemprozeduren nicht. Wenn Sie entfernte Geräte verwenden, die das automatische Device Tracking nicht unterstützen, können Sie unter Verwendung dieser Systemprozeduren manuell das Device Tracking konfigurieren.

Siehe auch

- „Device Tracking-Gateways“ auf Seite 23
- „Device Tracking für MobiLink Listener 9.0.0 einrichten“ auf Seite 24
- „Systemtabellen des MobiLink-Servers“ [*MobiLink - Serveradministration*]
- „Systemprozeduren des MobiLink-Servers“ [*MobiLink - Serveradministration*]

ml_delete_device-Systemprozedur

Verwenden Sie diese Systemprozedur, um alle Informationen über ein entferntes Gerät zu löschen, wenn Sie das Device Tracking manuell einrichten.

Parameter

Element	Parameter	Beschreibung
1	@device	VARCHAR(255). Gerätename

Bemerkungen

Diese Funktion ist nur dann sinnvoll, wenn Sie das Device Tracking manuell einrichten.

Siehe auch

- „Device Tracking für MobiLink Listener 9.0.0 einrichten“ auf Seite 24

Beispiel

Den Gerätedatensatz und alle zugeordneten Datensätze löschen, die diesen Gerätedatensatz referenzieren:

```
CALL ml_delete_device('myOldDevice');
```

ml_delete_device_address-Systemprozedur

Verwenden Sie diese Systemprozedur, um eine Geräteadresse zu löschen, wenn Sie das Device Tracking manuell einrichten.

Parameter

Element	Parameter	Beschreibung
1	@device	VARCHAR(255)
2	@medium	VARCHAR(255)

Bemerkungen

Diese Systemprozedur ist nur dann sinnvoll, wenn Sie das Device Tracking manuell einrichten.

Siehe auch

- „Device Tracking für MobiLink Listener 9.0.0 einrichten“ auf Seite 24

Beispiel

Löschen Sie einen Adressdatensatz:

```
CALL ml_delete_device_address('myWindowsMobile', 'ROGERS AT&T');
```

ml_delete_listening-Systemprozedur

Verwenden Sie diese Systemprozedur, um Zuordnungen zwischen einem MobiLink-Benutzer und entfernten Geräten zu löschen, wenn Sie manuell das Device Tracking einrichten.

Parameter

Element	Parameter	Beschreibung
1	@name	VARCHAR(128)

Bemerkungen

Diese Systemprozedur ist nur dann sinnvoll, wenn Sie das Device Tracking manuell einrichten.

Siehe auch

- „Device Tracking für MobiLink Listener 9.0.0 einrichten“ auf Seite 24

Beispiel

Löschen Sie einen Empfängerdatensatz:

```
CALL ml_delete_listening('myULDB');
```

ml_set_device-Systemprozedur

Benutzen Sie diese Systemprozedur, um Informationen zu entfernten Geräten hinzuzufügen bzw. zu ändern, wenn Sie das Device Tracking manuell einrichten. Sie fügt eine Zeile in die ml_device-Tabelle ein bzw. aktualisiert sie.

Parameter

Element	Parameter	Beschreibung
1	@device	VARCHAR(255). Benutzerdefinierter eindeutiger Gerätename
2	@listener_version	VARCHAR(128). Optionale Anmerkungen zur MobiLink Listener-Version.
3	@listener_protocol	INTEGER. Verwenden Sie **0** für Version 9.0.0 oder **2** für spätere Versionen des MobiLink Listeners für Windows-Geräte.
4	@info	VARCHAR(255). Optionale Geräteinformation
5	@ignore_tracking	VARCHAR(1). Auf **y** eingestellt, wird die Protokollierung ignoriert und das Überschreiben von manuell eingegebenen Daten verhindert.
6	@source	VARCHAR(255). Optionale Anmerkungen über die Quelle dieses Datensatzes

Bemerkungen

Mithilfe der Systemprozeduren ml_set_device, ml_set_device_address und ml_set_listening können Sie das automatische Device Tracking aufheben, indem Informationen in den MobiLink-Systemtabellen ml_device, ml_device_address und ml_listening geändert werden.

Diese Systemprozedur ist nur dann sinnvoll, wenn Sie das Device Tracking manuell einrichten.

Siehe auch

- „Device Tracking für MobiLink Listener 9.0.0 einrichten“ auf Seite 24
- „ml_set_device_address-Systemprozedur“ auf Seite 92
- „ml_set_listening-Systemprozedur“ auf Seite 93

Beispiel

Fügen Sie für jedes Gerät einen Gerätedatensatz hinzu:

```
CALL ml_set_device(
    'myWindowsMobile',
    'MobiLink Listeners for myWindowsMobile - 9.0.1',
    '1',
    'not used',
    'y',
```



```
        'manually entered by administrator'
    );
```

ml_set_device_address-Systemprozedur

Verwenden Sie diese Systemprozedur, um Informationen über Adressen von entfernten Geräten hinzuzufügen oder zu ändern, wenn Sie das Device Tracking manuell einrichten. Sie fügt eine Zeile in die ml_device_address-Tabelle ein bzw. aktualisiert sie.

Parameter

Element	Parameter	Beschreibung
1	@device	VARCHAR(255). Vorhandener Gerätename
2	@medium	VARCHAR(255). Netzwerk-Provider-ID (muss der Eigenschaft network_provider_id eines Netzbetreibers entsprechen)
3	@address	VARCHAR(255). Telefonnummer eines SMS-fähigen Geräts
4	@active	VARCHAR(1). Legen Sie **y** fest, um diesen Datensatz für die Verwendung beim Senden von Push-Benachrichtigungen zu aktivieren.
5	@ignore_tracking	VARCHAR(1). Auf **y** eingestellt, wird die Protokollierung ignoriert und das Überschreiben von manuell eingegebenen Daten verhindert.
6	@source	VARCHAR(255). Optionale Anmerkungen über die Quelle dieses Datensatzes

Bemerkungen

Mithilfe der Systemprozeduren ml_set_device, ml_set_device_address und ml_set_listening können Sie das automatische Device Tracking aufheben, indem Informationen in den MobiLink-Systemtabellen ml_device, ml_device_address und ml_listening geändert werden.

Diese Systemprozedur ist nur dann sinnvoll, wenn Sie das Device Tracking manuell einrichten.

Siehe auch

- „Device Tracking für MobiLink Listener 9.0.0 einrichten“ auf Seite 24
- „ml_set_device-Systemprozedur“ auf Seite 91
- „ml_set_listening-Systemprozedur“ auf Seite 93

Beispiel

Fügen Sie für jedes Gerät einen Gerätedatensatz hinzu:

```
CALL ml_set_device_address(
    'myWindowsMobile',
    'ROGERS AT&T',
    '3211234567',
    'y',
    'y',
    'manually entered by administrator'
);
```

ml_set_listening-Systemprozedur

Benutzen Sie diese Systemprozedur, um Verknüpfungen zwischen MobiLink-Benutzern und entfernten Geräten hinzuzufügen bzw. zu ändern, wenn Sie das Device Tracking manuell einrichten. Sie fügt eine Zeile in die ml_listening-Tabelle ein bzw. aktualisiert sie.

Parameter

Element	Parameter	Beschreibung
1	@name	VARCHAR(128). MobiLink-Benutzername.
2	@device	VARCHAR(255). Vorhandener Gerätename
3	@listening	VARCHAR(1). Auf **y** eingestellt, wird dieser Datensatz für die Verwendung bei der DeviceTracker-Adressierung aktiviert.
4	@ignore_tracking	VARCHAR(1). Auf **y** eingestellt, wird die Protokollierung ignoriert und das Überschreiben von manuell eingegebenen Daten verhindert.
5	@source	VARCHAR(255). Optionale Anmerkungen über die Quelle dieses Datensatzes

Bemerkungen

Mithilfe der Systemprozeduren ml_set_device, ml_set_device_address und ml_set_listening können Sie das automatische Device Tracking aufheben, indem Informationen in den MobiLink-Systemtabellen ml_device, ml_device_address und ml_listening geändert werden.

Diese Systemprozedur ist nur dann sinnvoll, wenn Sie das Device Tracking manuell einrichten.

Siehe auch

- „Device Tracking für MobiLink Listener 9.0.0 einrichten“ auf Seite 24
- „ml_set_device-Systemprozedur“ auf Seite 91
- „ml_set_device_address-Systemprozedur“ auf Seite 92

Beispiel

Fügen Sie für jede entfernte Datenbank einen Empfängerdatensatz für ein Gerät hinzu. Dadurch wird das Gerät dem MobiLink-Benutzernamen zugeordnet.

```
CALL ml_set_listening(
    'myULDB',
    'myWindowsMobile',
    'y',
    'y',
    'manually entered by administrator'
);
```

ml_set_sis_sync_state-Systemprozedur

Mit dieser Prozedur können Sie den Zustand der MobiLink-Synchronisation in der ml_sis_sync_state-Systemtabelle aufzeichnen.

Parameter

Element	Parameter	Beschreibung
1	@remote_id	VARCHAR(128)
2	@subscripti-on_id	VARCHAR(128)
3	@publicati-on_name	VARCHAR(128)
4	@user_name	VARCHAR(128)
5	@last_upload	TIMESTAMP
6	@last_downlo-ad	TIMESTAMP

Bemerkungen

Rufen Sie im Ereignis publication_nonblocking_download_ack die Systemprozedur ml_set_sis_sync_state auf, um Benutzern zu gestatten, ein request_cursor-Ereignis zu erstellen, das die ml_sis_sync_state-Tabelle referenziert.

Siehe auch

- „publication_nonblocking_download_ack (Verbindungsereignis)“ [*MobiLink - Serveradministration*]

Beispiel

Geben Sie mit folgendem Skript zur Aufzeichnung des Synchronisationsstatus ein publication_nonblocking_download_ack-Ereignisskript an:

```
CALL ml_set_sis_sync_state(
    {ml s.remote_id},
    NULL,
    {ml s.publication_name},
    {ml s.username},
    NULL,
```

```
        {ml s.last_publication_download}
    );
```

Serverinitiierte Synchronisation - erweiterte Themen

In den folgenden Abschnitten finden Sie Informationen zu erweiterten Themen im Zusammenhang mit der serverinitiierten Synchronisation.

Nachrichtensyntax

Die folgende Nachrichtensyntax gilt für den Lightweight-Polling-Modus (Standard), UDP-Gateways und SYNC-Gateways:

message = **[subject]content**

Die Syntax von Nachrichten, die mithilfe des SMTP-Gateways gesendet werden, hat eine der folgenden Strukturen:

- **message** = *sender*[*subject*]*content*
- **message** = *sender*(*subject*)*content*
- **message** = *sender*{*subject*}*content*
- **message** = *sender*<*subject*>*content*
- **message** = *sender*'*subject*'*content*
- **message** = *sender*"*subject*"*content*

Die richtige Nachrichtensyntax und die Syntax der *sender*-E-Mail-Adresse hängen vom jeweiligen Mobilfunkbetreiber ab. Um die Nachrichtensyntax zu bestimmen, führen Sie den MobiLink Listener mit aktivierter Nachrichtenprotokollierung und Ausführlichkeitsstufe 2 aus, unter Verwendung der dblsn-Optionen -m und -v. Das Nachrichtenlog enthält die geeignete Syntax, wenn Sie den MobiLink Listener erstmals ausgeführt haben.

Bei der Verwendung eines Device Tracking-Gateways, hängt die Nachrichtensyntax vom untergeordneten Gateway ab, das zum Senden der Nachricht verwenden wird. Wenn Sie ein untergeordnetes SMTP-Gateway verwenden, hängt die Syntax vom öffentlichen Mobilnetzbetreiber ab.

Bemerkungen

Geschweifte, runde und eckige Klammern, Anführungszeichen und Apostrophe sind für die interne Verwendung reserviert und dürfen in **subject** nicht verwendet werden. Weitere Hinweise zu Einschränkungen für Nachrichten finden Sie unter „Mit Push-Anforderungen arbeiten“ auf Seite 9.

Siehe auch

- „MobiLink Listener-Schlüsselwörter für Windows-Geräte“ auf Seite 72
- „Gateways und Netzbetreiber“ auf Seite 22
- „dblsn-Option -m“ auf Seite 64
- „dblsn-Option -v“ auf Seite 70

Senden einer Push-Benachrichtigung mithilfe der sa_send_udp-Systemprozedur

Eine konsolidierte SQL Anywhere-Datenbank kann die sa_send_upd-Systemprozedur verwenden, um über ein UDP-Gateway Push-Benachrichtigungen an ein Gerät zu senden. Diese Methode ist eine Alternative zum Senden von Push-Benachrichtigungen mithilfe von Notifiern.

Voraussetzungen

- Ein MobiLink Listener ist auf einem Gerät eingerichtet und wartet auf Push-Benachrichtigungen
- Internet Explorer ist auf dem Gerät installiert
- Der folgende Befehl wurde auf dem Gerät ausgeführt:

  ```
  dblsn -l "message=RunBrowser;action='START iexplore.exe http://
  www.sap.com';"
  ```

- Eine konsolidierte SQL Anywhere-Datenbank wird auf dem MobiLink-Server ausgeführt

Kontext und Bemerkungen

Indem Sie an die ursprüngliche Nachricht **1** anhängen und anschließend diese Nachricht im msg-Argument einer sa_send_upd-Systemprozedur verwenden, senden Sie die ursprüngliche Nachricht an einen MobiLink Listener.

Aufgabe

1. Führen Sie Interactive SQL aus und stellen Sie eine Verbindung zu Ihrer konsolidierten Datenbank mit einem Befehl ähnlich dem unten gezeigten her, wobei Sie *consdb-source-name* durch den ODBC-Namen der konsolidierten Datenbank ersetzen.

   ```
   dbisql -c "dsn=consdb-source-name"
   ```

2. Führen Sie die folgende Anweisung aus, um die Push-Benachrichtigung zu senden:

   ```
   CALL sa_send_udp('device-ip-address', 5001, 'RunBrowser1')
   ```

   Das erste Argument stellt sicher, dass die Push-Benachrichtigung an das richtige Gerät gesendet wird. Ersetzen Sie *device-ip-address* durch die IP-Adresse des Geräts. Wenn der MobiLink Listener auf demselben Computer wie der MobiLink-Server ausgeführt wird, verwenden Sie **localhost**.

   Das zweite Argument ist die Portnummer. Standardmäßig verwenden MobiLink Listener Port 5001, um auf UDP-Nachrichten zu warten.

Das dritte Argument ist die Nachricht, die mit **1** als Suffix gesendet wird. Durch das Anhängen von 1, wobei es sich um ein reserviertes serverinitiiertes Synchronisationsprotokoll handelt, wird die **RunBrowser**-Nachricht über ein UDP-Gateway an das Gerät gesendet.

Ergebnisse

Beim Ausführen des Systemaufrufs wird die **RunBrowser**-Nachricht an das Gerät gesendet. Daraufhin startet das Gerät den Internet Explorer und lädt die SAP-Startseite.

Siehe auch

- „sa_send_udp-Systemfunktion“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]

Praktische Einführung in die serverinitiierte Synchronisation

Nutzen Sie die folgenden praktischen Einführungen, um besser zu verstehen, wie die serverinitiierte Synchronisation verwendet wird.

Praktische Einführung: Konfigurieren der serverinitiierten Synchronisation unter Verwendung von Lightweight-Polling

Diese praktische Einführung zeigt, wie eine konsolidierte SQL Anywhere-Datenbank und eine entfernte Datenbank für die serverinitiierte Synchronisation konfiguriert werden. Sie basiert auf dem Beispielcode unter *%SQLANYSAMP16%\MobiLink\SIS_CarDealer_LP_DBLSN*.

Beispielimplementierungen serverinitiierter Synchronisationen befinden sich unter *%SQLANYSAMP16%\MobiLink*. Die Namen aller Verzeichnisse der Beispiele für die serverinitiierte Synchronisation beginnen mit dem Präfix SIS_.

Hinweis
Sie können mit Sybase Central entfernte Datenbanken verwalten und anschließend serverinitiierte entfernte Aufgaben (SIRT) als Alternative zur serverinitiierten Synchronisation mithilfe des Lightweight-Polling-Moduls verwenden. Weitere Hinweise finden Sie unter „Serverinitiierte entfernte Aufgaben (SIRT)“ [*MobiLink - Serveradministration*] und „Praktische Einführung: Zentrale Administration von entfernten Datenbanken“ [*MobiLink - Erste Orientierung*].

Erforderliche Software

- SQL Anywhere 16

Kenntnisse und Erfahrungen

- Grundkenntnisse über MobiLink-Ereignisskripten.

Privilegien

Sie müssen die folgenden Rollen und Privilegien für die konsolidierte Datenbank haben:

- SYS_AUTH_RESOURCE_ROLE-Kompatibilitätsrolle
- MONITOR-Systemprivileg

Sie müssen die folgenden Rollen und Privilegien für die entfernte Datenbank haben:

- SYS_REPLICATION_ADMIN_ROLE-Systemrolle
- SYS_RUN_REPLICATION_ROLE-Systemrolle

Ziele

- Eine konsolidierte SQL Anywhere-Datenbank für die serverinitiierte Synchronisation einrichten
- Serverseitige Eigenschaften konfigurieren
- Push-Anforderungen zum Anstoßen einer serverinitiierten Synchronisation ausgeben

Empfohlene Hintergrundlektüre

- „Serverinitiierte Synchronisation“ auf Seite 1

Lektion 1: Einrichten der konsolidierten Datenbank

In dieser Lektion erstellen Sie mithilfe des Dienstprogramms dbinit eine konsolidierte Datenbank namens **SIS_CarDealer_LP_DBLSN_CONDB** mit den für die Synchronisation erforderlichen Skripten. Verwenden Sie dann den SQL Anywhere 16-Treiber, um eine ODBC-Datenquelle für die **SIS_CarDealer_LP_DBLSN_CONDB**-Datenbank zu definieren.

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: Konfigurieren der serverinitiierten Synchronisation unter Verwendung von Lightweight-Polling“ auf Seite 101.

Aufgabe

1. Erstellen Sie ein Arbeitsverzeichnis, um die konsolidierte Datenbank zu speichern.

   In dieser praktischen Einführung wird davon ausgegangen, dass *c:\MLsis* das Arbeitsverzeichnis ist.

2. Erstellen Sie die konsolidierte SQL Anywhere-Datenbank mithilfe des Dienstprogramms dbinit. Setzen Sie die DBA-Benutzer-ID auf DBA und das Kennwort auf "sql".

   Wechseln Sie zum Verzeichnis *c:\MLsis* und führen Sie den folgenden Befehl aus:

   ```
   dbinit -dba DBA,sql SIS_CarDealer_LP_DBLSN_CONDB
   ```

3. Starten Sie die konsolidierte Datenbank.

   Führen Sie den folgenden Befehl aus:

   ```
   dbsrv16 SIS_CarDealer_LP_DBLSN_CONDB
   ```

4. Wählen Sie **Start** » **Programme** » **SQL Anywhere 16** » **Administrationstools** » **ODBC-Datenquellen-Administrator**.

5. Klicken Sie auf der Registerkarte **Benutzer-DSN** auf **Hinzufügen**.

6. Klicken Sie im Fenster **Neue Datenquelle erstellen** auf **SQL Anywhere 16** und auf **Fertig stellen**.

7. Führen Sie im Fenster **ODBC-Konfiguration für SQL Anywhere** folgende Aufgaben aus:
   a. Klicken Sie auf die Registerkarte **ODBC**.
   b. Im Feld **Datenquellenname** geben Sie **SIS_CarDealer_LP_DBLSN_CONDB** ein.
   c. Klicken Sie auf die Registerkarte **Login**.
   d. Im Feld **Benutzer-ID** geben Sie **DBA** ein.
   e. Im Feld **Kennwort** geben Sie **sql** ein.
   f. Wählen Sie in der Dropdown-Liste **Aktion** die Option **Mit einer laufenden Datenbank auf diesem Computer verbinden**.
   g. Im Feld **Servername** geben Sie **SIS_CarDealer_LP_DBLSN_CONDB** ein.
   h. Klicken Sie auf **OK**.
8. Schließen Sie den ODBC-Datenquellenadministrator.

   Klicken Sie auf **OK** im Fenster **ODBC-Datenquellenadministrator**.

Ergebnisse

Die konsolidierte Datenbank wird erstellt und eine ODBC-Datenquelle wird definiert.

Nächste Schritte

Gehen Sie weiter zu „Lektion 2: Generieren eines Datenbankschemas“ auf Seite 103.

Siehe auch

- „ODBC-Datenquellen“ [*SQL Anywhere Server - Datenbankadministration*]

Lektion 2: Generieren eines Datenbankschemas

In dieser Lektion generieren Sie ein Datenbankschema, das die Tabelle Dealer, eine non_sync_request-Tabelle und ein download_cursor-Synchronisationsskript enthält. Dieses Datenbankschema erfüllt die Anforderungen für das Generieren von Push-Anforderungen.

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: Konfigurieren der serverinitiierten Synchronisation unter Verwendung von Lightweight-Polling“ auf Seite 101.

Aufgabe

1. Klicken Sie auf **Start** » **Programme** » **SQL Anywhere 16** » **Administrationstools** » **Sybase Central**.
2. Führen Sie die folgenden Aufgaben durch, um eine Verbindung zur konsolidierten Datenbank herzustellen.

a. Klicken Sie auf **Verbindungen** » **Verbinden mit SQL Anywhere 16**.
b. Wählen Sie in der Dropdown-Liste **Aktion** die Option **Mit einer ODBC-Datenquelle verbinden**.
c. Klicken Sie auf **ODBC-Datenquellenname** und anschließend auf **Durchsuchen**.
d. Wählen Sie **SIS_CarDealer_LP_DBLSN_CONDB** und klicken Sie anschließend auf **OK**.
e. Klicken Sie auf **Verbinden**.

3. Verwenden Sie Interactive SQL, um sich mit Ihrer Datenbank zu verbinden.

   Sie können Interactive SQL über Sybase Central oder eine Eingabeaufforderung starten.

   - Um Interactive SQL aus Sybase Central zu starten, rechtsklicken Sie auf die **SIS_CarDealer_LP_DBLSN_CONDB - DBA**-Datenbank und auf **Interactive SQL öffnen**.
   - Um Interactive SQL an einer Eingabeaufforderung zu starten, führen Sie folgenden Befehl aus:

     ```
     dbisql -c "dsn=SIS_CarDealer_LP_DBLSN_CONDB"
     ```

4. Führen Sie die folgenden SQL Anweisungen aus, um die Tabellen Dealer und non_sync_request einzurichten:

   ```
   CREATE TABLE Dealer (
       name           VARCHAR(10) NOT NULL PRIMARY KEY,
       rating         VARCHAR(5),
       last_modified  TIMESTAMP DEFAULT TIMESTAMP
   )

   CREATE TABLE non_sync_request(
       poll_key       VARCHAR(128)
   )
   ```

5. Fügen Sie die Daten in die Tabelle Dealer mithilfe der folgenden Anweisungen ein:

   ```
   INSERT INTO Dealer(name, rating) VALUES ('Audi', 'a');
   INSERT INTO Dealer(name, rating) VALUES ('Buick', 'b');
   INSERT INTO Dealer(name, rating) VALUES ('Chrysler', 'c');
   INSERT INTO Dealer(name, rating) VALUES ('Dodge', 'd');
   INSERT INTO Dealer(name, rating) VALUES ('Eagle', 'e');
   INSERT INTO Dealer(name, rating) VALUES ('Ford', 'f');
   INSERT INTO Dealer(name, rating) VALUES ('Geo', 'g');
   INSERT INTO Dealer(name, rating) VALUES ('Honda', 'h');
   INSERT INTO Dealer(name, rating) VALUES ('Isuzu', 'I');
   COMMIT;
   ```

6. Führen Sie die folgende SQL-Anweisung aus, um die MobiLink-Systemtabellen und die gespeicherten Prozeduren zu erstellen. Ersetzen Sie *C:\Program Files\SQL Anywhere 16* durch den Pfad Ihrer SQL Anywhere 16-Installation.

   ```
   READ "C:\Program Files\SQL Anywhere 16\MobiLink\setup\syncsa.sql"
   ```

7. Führen Sie das folgende SQL-Skript aus, um ein download_cursor-Synchronisationsskript anzugeben und den Synchronisationsstatus in der ml_sis_sync_state-Systemtabelle aufzuzeichnen.

   ```
   CALL ml_add_table_script(
       'CarDealer',
   ```

```
        'Dealer',
        'download_cursor',
        'SELECT * FROM Dealer WHERE last_modified >= ?'
    );

    CALL ml_add_connection_script(
        'CarDealer',
        'publication_nonblocking_download_ack',
        'CALL ml_set_sis_sync_state(
            {ml s.remote_id},
            NULL,
            {ml s.publication_name},
            {ml s.username},
            NULL,
            {ml s.last_publication_download}
        )'
    );

    CALL ml_add_table_script(
        'CarDealer', 'Dealer', 'download_delete_cursor', '--{ml_ignore}'
    );

    COMMIT;
```

Dieses Skript legt ml_sis_sync_state fest, um die Synchronisation mit reinem Download aufzuzeichnen. Die Aufzeichnung des Synchronisationsstatus gestattet es, die ml_sis_sync_state-Systemtabelle vom request_cursor-Ereignis aus zu referenzieren. Das request_cursor-Ereignis legen Sie in der nächsten Lektion fest.

8. Schließen Sie Interactive SQL.

Ergebnisse

Das Datenbankschema wird erstellt.

Nächste Schritte

Gehen Sie weiter zu „Lektion 3: Erstellen eines MobiLink-Projekts“ auf Seite 105.

Siehe auch

- „Syntax des SQL Anywhere-Datenbankservers“ [*SQL Anywhere Server - Datenbankadministration*]
- „CREATE TABLE-Anweisung“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- „Schreiben von Synchronisationsskripten“ [*MobiLink - Serveradministration*]
- „download_cursor (Tabellenereignis)“ [*MobiLink - Serveradministration*]
- „publication_nonblocking_download_ack (Verbindungsereignis)“ [*MobiLink - Serveradministration*]

Lektion 3: Erstellen eines MobiLink-Projekts

In dieser Lektion verbinden Sie sich mit der konsolidierten Datenbank, indem Sie ein neues MobiLink-Projekt erstellen.

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:

Konfigurieren der serverinitiierten Synchronisation unter Verwendung von Lightweight-Polling“ auf Seite 101.

Erstellen eines neuen MobiLink-Projekts

1. Klicken Sie auf **Start** » **Programme** » **SQL Anywhere 16** » **Administrationstools** » **Sybase Central**.
2. Klicken Sie auf **Extras** » **MobiLink 16** » **Neues Projekt**.
3. Im Feld **Name** geben Sie **SIS_CarDealer_LP_DBLSN_CONDB_project** ein.
4. Im Feld **Speicherort** geben Sie *C:\MLsis* ein und klicken dann auf **Weiter**.
5. Im Feld **Anzeigename der Datenbank** geben Sie **SIS_CarDealer_LP_DBLSN_CONDB** ein.
6. Klicken Sie auf **Bearbeiten**. Das Fenster **Mit einer allgemeinen ODBC-Datenbank verbinden** erscheint.
7. Im Feld **Benutzer-ID** geben Sie **DBA** ein.
8. Im Feld **Kennwort** geben Sie **sql** ein.
9. Klicken Sie im Feld **ODBC-Datenquellenname** auf **Durchsuchen** und wählen Sie dann **SIS_CarDealer_LP_DBLSN_CONDB**.
10. Klicken Sie auf **OK** und dann auf **Speichern**.
11. Aktivieren Sie die Option **Kennwort speichern** und klicken Sie auf **Weiter**.
12. Wählen Sie auf der Seite **Neues entferntes Datenbankschema** nur die Tabelle **Dealer** zum Synchronisieren. Klicken Sie auf **Next (Weiter)**.
13. Klicken Sie auf **Weiter** und anschließend auf **Fertig stellen**. Klicken Sie auf **OK**.

Ergebnisse

Das MobiLink-Projekt wird erstellt.

Nächste Schritte

Gehen Sie weiter zu „Lektion 4: Konfigurieren des Notifiers“ auf Seite 106.

Lektion 4: Konfigurieren des Notifiers

In dieser Lektion konfigurieren Sie ein Notifier-Ereignis, das festlegt, wie der Notifier Push-Anforderungen erstellt und Push-Benachrichtigungen an Geräte sendet.

Voraussetzungen

In dieser Lektion wird davon ausgegangen, dass Sie bereits alle vorherigen Lektionen abgeschlossen haben. Siehe „Lektion 1: Einrichten der konsolidierten Datenbank“ auf Seite 102.

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: Konfigurieren der serverinitiierten Synchronisation unter Verwendung von Lightweight-Polling“ auf Seite 101.

Kontext und Bemerkungen

Das request_cursor-Ereignisskript erkennt Push-Anforderungen. Jede Push-Anforderung legt fest, welche Informationen gesendet werden und welches Gerät die Informationen empfangen soll.

Aufgabe

1. Erweitern Sie im linken Fensterausschnitt von Sybase Central unter **MobiLink 16** **SIS_CarDealer_LP_DBLSN_CONDB_project**, **Konsolidierte Datenbanken** und dann **SIS_CarDealer_LP_DBLSN_CONDB - DBA**.

2. Rechtsklicken Sie auf **Benachrichtigung** und klicken Sie dann auf **Neu** » **Notifier**.

3. Im Feld **Wie soll der Name des neuen Notifiers lauten?** geben Sie **CarDealerNotifier** ein.

4. Klicken Sie auf **Fertig stellen**.

5. Wählen Sie im rechten Fensterausschnitt **CarDealerNotifier** aus und klicken Sie dann auf **Eigenschaften**.

6. Klicken Sie auf die Registerkarte **Ereignisse** und klicken Sie auf **request_cursor** in der Liste **Ereignisse**.

7. Geben Sie die folgende SQL-Anweisung in das angezeigte Textfeld ein:

```
SELECT ml_sis_sync_state.remote_id + '.sync' FROM ml_sis_sync_state
WHERE
(
    EXISTS (SELECT 1 FROM Dealer
        WHERE last_modified >= ml_sis_sync_state.last_download)
    AND EXISTS (SELECT poll_key FROM non_sync_request)
)
```

8. Klicken Sie auf **OK**, um das das Notifier-Ereignis zu speichern.

Ergebnisse

Ein Notifier wird erstellt und konfiguriert.

Nächste Schritte

Gehen Sie weiter zu „Lektion 5: Starten des MobiLink-Servers“ auf Seite 108.

Siehe auch

- „request_cursor-Ereignis“ auf Seite 40
- „ml_set_sis_sync_state-Systemprozedur“ auf Seite 94

Lektion 5: Starten des MobiLink-Servers

In dieser Lektion starten Sie den MobiLink-Server mit dem Notifier, sodass Push-Benachrichtigungen an Geräte gesendet werden können.

Voraussetzungen

In dieser Lektion wird davon ausgegangen, dass Sie bereits alle vorherigen Lektionen abgeschlossen haben. Siehe „Lektion 1: Einrichten der konsolidierten Datenbank“ auf Seite 102.

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: Konfigurieren der serverinitiierten Synchronisation unter Verwendung von Lightweight-Polling“ auf Seite 101.

Aufgabe

- Stellen Sie eine Verbindung mit Ihrer konsolidierten Datenbank her.

  Führen Sie den folgenden Befehl aus:

  ```
  mlsrv16 -notifier -c "dsn=SIS_CarDealer_LP_DBLSN_CONDB" -o serverOut.txt -
  v+ -dl -zu+ -x tcpip
  ```

  Die folgende Tabelle beschreibt die in dieser Lektion verwendeten mlsrv16-Optionen. Die Optionen -o - und -v bieten Informationen zur Fehlersuche und -behebung. Die Verwendung dieser Protokollierungsoptionen empfiehlt sich in einer Entwicklungsumgebung. Aus Gründen der Performance wird -v bei der Produktion normalerweise nicht eingesetzt.

Option	Beschreibung
-notifier	Startet alle aktivierten Notifier für die serverinitiierte Synchronisation. Siehe „mlsrv16-Option -notifier “ [*MobiLink - Serveradministration*].
-c	Legt die Verbindungszeichenfolge fest. Siehe „mlsrv16-Option -c “ [*MobiLink - Serveradministration*].
-o	Legt die Meldungslogdatei *serverOut.txt* fest Siehe „mlsrv16-Option -o“ [*MobiLink - Serveradministration*].

Option	Beschreibung
-v+	Gibt an, welche Informationen protokolliert werden. Mit -v+ wird die maximale ausführliche Protokollierung aktiviert. Siehe „mlsrv16-Option -v “ [*MobiLink - Serveradministration*].
-zu+	Fügt neue Benutzer automatisch hinzu. Siehe „mlsrv16-Option -zu “ [*MobiLink - Serveradministration*].
-x	Legt das Kommunikationsprotokoll und Protokolloptionen für MobiLink-Clients fest. Siehe „mlsrv16-Option -x“ [*MobiLink - Serveradministration*].

Ergebnisse

Das MobiLink-Server-Meldungsfenster wird eingeblendet. Der Notifier gibt an, dass er für den Empfang von Push-Benachrichtigung von Geräten bereit ist.

Nächste Schritte

Gehen Sie weiter zu „Lektion 6: Einrichten einer entfernten Datenbank“ auf Seite 109.

Siehe auch

- „MobiLink-Server“ [*MobiLink - Serveradministration*]
- „MobiLink-Serveroptionen“ [*MobiLink - Serveradministration*]

Lektion 6: Einrichten einer entfernten Datenbank

In dieser Lektion erstellen Sie eine entfernte SQL Anywhere-Datenbank, eine Synchronisationspublikation, einen Benutzer und eine Subskription.

Voraussetzungen

In dieser Lektion wird davon ausgegangen, dass Sie bereits alle vorherigen Lektionen abgeschlossen haben. Siehe „Lektion 1: Einrichten der konsolidierten Datenbank“ auf Seite 102.

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: Konfigurieren der serverinitiierten Synchronisation unter Verwendung von Lightweight-Polling“ auf Seite 101.

Aufgabe

1. Erstellen Sie Ihre MobiLink Clientdatenbank mit dem Befehlszeilen-Dienstprogramm dbinit.

   Führen Sie im Verzeichnis *c:\MLsis* den folgenden Befehl aus:


```
dbinit -dba DBA,sql SIS_CarDealer_LP_DBLSN_REM
```

2. Starten Sie Ihre MobiLink-Clientdatenbank mit dem Befehlszeilen-Dienstprogramm dbsrv16.

   Führen Sie den folgenden Befehl aus:

   ```
   dbsrv16 SIS_CarDealer_LP_DBLSN_REM
   ```

3. Verbinden Sie sich über Interactive SQL mit Ihrem MobiLink-Client.

   Führen Sie den folgenden Befehl aus:

   ```
   dbisql -c "SERVER=SIS_CarDealer_LP_DBLSN_REM;UID=DBA;PWD=sql"
   ```

4. Erstellen Sie die Tabelle "Dealer" in der entfernten Datenbank.

   Führen Sie hierzu folgende SQL-Anweisung in Interactive SQL aus:

   ```
   CREATE TABLE Dealer (
       name           VARCHAR(10) NOT NULL PRIMARY KEY,
       rating         VARCHAR(5),
       last_modified  TIMESTAMP DEFAULT TIMESTAMP
   )
   COMMIT;
   ```

5. Erstellen Sie Ihren MobiLink-Synchronisationsbenutzer sowie die Publikation und Subskription.

   Führen Sie hierzu folgende SQL-Anweisung in Interactive SQL aus:

   ```
   CREATE PUBLICATION CarDealer(TABLE DEALER WHERE 0=1)
   CREATE SYNCHRONIZATION USER test_mluser OPTION ScriptVersion='CarDealer'
   CREATE SYNCHRONIZATION SUBSCRIPTION TO CarDealer FOR test_mluser
   SET OPTION public.ml_remote_id = remote_id;
   COMMIT;
   ```

Ergebnisse

Eine entfernte SQL Anywhere-Datenbank, eine Synchronisationspublikation, ein Benutzer und eine Subskription werden erstellt.

Nächste Schritte

Gehen Sie weiter zu „Lektion 7: Konfigurieren des MobiLink Listeners“ auf Seite 111.

Siehe auch

- „MobiLink-Clients“ [*MobiLink - Clientadministration*]
- „CREATE TABLE-Anweisung“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- „Publikationen“ [*MobiLink - Clientadministration*]
- „CREATE PUBLICATION-Anweisung [MobiLink] [SQL Remote]“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- „CREATE SYNCHRONIZATION USER-Anweisung [MobiLink]“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- „CREATE SYNCHRONIZATION SUBSCRIPTION-Anweisung [MobiLink]“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- „Skriptversionen“ [*MobiLink - Serveradministration*]

Lektion 7: Konfigurieren des MobiLink Listeners

In dieser Lektion konfigurieren Sie den MobiLink Listener, indem Sie die MobiLink Listener-Optionen in einer Textdatei speichern und dann dblsn unter Angabe des Dateinamens in der Befehlszeile ausführen.

Voraussetzungen

In dieser Lektion wird davon ausgegangen, dass Sie bereits alle vorherigen Lektionen abgeschlossen haben. Siehe „Lektion 1: Einrichten der konsolidierten Datenbank“ auf Seite 102.

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: Konfigurieren der serverinitiierten Synchronisation unter Verwendung von Lightweight-Polling“ auf Seite 101.

Aufgabe

1. Führen Sie den folgenden Befehl aus, um eine Synchronisation mit dem MobiLink-Server auszuführen und die Datei *SIS_CarDealer_LP_DBLSN__REM.rid* zu erstellen:

   ```
   dbmlsync -c "SERVER=SIS_CarDealer_LP_DBLSN_REM;UID=DBA;PWD=sql" -e sa=on -
   o rem1.txt -v+
   ```

   Der MobiLink Listener kann mithilfe der Aktionsvariablen $remote_id einen Polling-Schlüssel definieren, der vom MobiLink-Server verwendet wird, um das Gerät zu identifizieren. Diese Variable wird von der entfernten ID-Datei *SIS_CarDealer_LP_DBLSN__REM.rid* abgerufen, die während der ersten Synchronisation mit dem MobiLink-Server erstellt wird. Sie müssen eine Synchronisation mit dem MobiLink-Server ausführen, um die entfernte ID-Datei verwenden zu können.

2. Klicken Sie im Fenster des SQL Anywhere MobiLink-Clients auf **Herunterfahren**.

3. Erstellen Sie eine MobiLink Listener-Befehlsdatei, indem Sie eine Textdatei mit folgendem Inhalt erstellen:

   ```
   # Verbosity level
   -v2

   # Show notification messages in console and log
   ```

```
-m

# Truncate, then write output to dblsn.txt
-ot dblsn.txt

# Remote ID file (defining the scope of $remote_id)
-r SIS_CarDealer_LP_DBLSN_REM.rid

# Message handlers

# Watch for a notification without action
-l "poll_connect='tcpip(host=localhost)';
    poll_notifier=CarDealerNotifier;
    poll_key=$remote_id.no_action;"

# Signal dbmlsync to launch, sync and then shutdown
-l "poll_connect='tcpip(host=localhost)';
    poll_notifier=CarDealerNotifier;
    poll_key=$remote_id.sync;
    action='run dbmlsync.exe -c
SERVER=SIS_CarDealer_LP_DBLSN_REM;UID=DBA;PWD=sql -e sa=on -o rem1.txt -v
+';"

# Shutdown the MobiLink Listener
-l "poll_connect='tcpip(host=localhost)';
    poll_notifier=CarDealerNotifier;
    poll_key=$remote_id.shutdown;
    action='DBLSN FULL SHUTDOWN';"
```

4. In dieser praktischen Einführung wird davon ausgegangen, dass *c:\MLsis* das Arbeitsverzeichnis für serverseitige Komponenten ist. Speichern Sie die Textdatei als *mydblsn.txt* in diesem Verzeichnis.

5. Starten Sie den MobiLink Listener.

   Navigieren Sie an einer Eingabeaufforderung zu *c:\MLsis* oder zu dem Verzeichnis, in dem Sie die MobiLink Listener-Befehlsdatei gespeichert haben.

   Starten Sie den MobiLink Listener mit dem folgenden Befehl:

   ```
   dblsn @mydblsn.txt
   ```

   Das Fenster **MobiLink Listener für Windows** erscheint mit der Anzeige, dass der MobiLink Listener im Ruhezustand ist.

Ergebnisse

Der MobiLink Listener ist konfiguriert.

Nächste Schritte

Gehen Sie weiter zu „Lektion 8: Ausgeben von Push-Anforderungen“ auf Seite 113.

Siehe auch

- „Listener“ auf Seite 14
- „MobiLink Listener-Dienstprogramm für Windows-Geräte (dblsn)“ auf Seite 55
- „dblsn-Option @data“ auf Seite 60
- „Aktionsvariablen“ auf Seite 18

Lektion 8: Ausgeben von Push-Anforderungen

In dieser Lektion führen Sie Änderungen an der Tabelle Dealer in der konsolidierten Datenbank durch, sodass die Informationen in die entfernte Datenbank heruntergeladen werden können, wenn der MobiLink Listener Push-Benachrichtigungen abruft. Sie stoßen dann eine serverinitiierte Synchronisation an, indem Sie einen Polling-Schlüsselwert in die konsolidierte Datenbank eingeben. Der Notifier führt das request_cursor-Ereignis aus, findet den Polling-Schlüssel in der Tabelle non_sync_request und sendet dann eine Push-Benachrichtigung an den MobiLink Listener. Wenn der MobiLink Listener die Push-Benachrichtigung empfängt, führt er eine Synchronisation mit der MobiLink-Datenbank durch und aktualisiert die entfernte Datenbank.

Voraussetzungen

In dieser Lektion wird davon ausgegangen, dass Sie bereits alle vorherigen Lektionen abgeschlossen haben. Siehe „Lektion 1: Einrichten der konsolidierten Datenbank“ auf Seite 102.

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: Konfigurieren der serverinitiierten Synchronisation unter Verwendung von Lightweight-Polling“ auf Seite 101.

Aufgabe

1. Verbinden Sie sich über Interactive SQL mit Ihrer konsolidierten Datenbank, falls dies nicht schon geschehen ist.

   Führen Sie den folgenden Befehl aus:

   ```
   dbisql -c "dsn=SIS_CarDealer_LP_DBLSN_CONDB"
   ```

2. Führen Sie die folgenden SQL-Anweisungen aus:

   ```
   UPDATE Dealer
       SET RATING = 'B' WHERE name = 'Geo';
   COMMIT;
   ```

3. Geben Sie Push-Anforderungen aus, indem Sie die Tabelle non_sync_request direkt füllen. Die Polling-Schlüsselspalte legt fest, welches Gerät Push-Benachrichtigungen empfängt.

   Führen Sie die folgenden SQL-Anweisungen aus:

   ```
   INSERT INTO non_sync_request(poll_key) VALUES ('%remote_id%.no_action');
   COMMIT;
   ```

4. Warten Sie ein paar Sekunden, bis die Synchronisation ausgeführt wird.

   Der MobiLink Listener sollte die konsolidierte Datenbank abfragen, die Push-Benachrichtigung herunterladen und die Tabelle Dealer in der entfernten Datenbank aktualisieren.

5. Stoppen Sie die serverinitiierte Synchronisation mit einem Gerät, indem Sie den Abrufschlüsselwert aus der Tabelle non_sync_request in der konsolidierten Datenbank löschen.

   Führen Sie die folgenden SQL-Anweisungen aus:

   ```
   DELETE FROM non_sync_request WHERE poll_key = '%remote_id%.no_action';
   COMMIT;
   ```

6. Überprüfen Sie, ob die Tabelle "Dealer" in der entfernten Datenbank aktualisiert wurde.

   Führen Sie hierzu die folgende SQL-Anweisung aus:

   ```
   SELECT * FROM Dealer
   ```

   Die Bewertung für **Geo** sollte nun **B** sein.

Ergebnisse

In der konsolidierten Datenbank wird eine Änderung vorgenommen und die serverinitiierte Synchronisation wird initiiert.

Nächste Schritte

Gehen Sie weiter zu „Aufräumen“ auf Seite 114.

Siehe auch

- Push-Anforderungen generieren auf Seite 9
- „INSERT-Anweisung“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- „UPDATE-Anweisung“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- „DELETE-Anweisung“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]

Aufräumen

Entfernen Sie die Daten der praktischen Einführung von Ihrem Computer.

Voraussetzungen

In dieser Lektion wird davon ausgegangen, dass Sie bereits alle vorherigen Lektionen abgeschlossen haben. Siehe „Lektion 1: Einrichten der konsolidierten Datenbank“ auf Seite 102.

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: Konfigurieren der serverinitiierten Synchronisation unter Verwendung von Lightweight-Polling“ auf Seite 101.

Aufgabe

1. Schließen Sie Interactive SQL.
2. Schließen Sie SQL Anywhere, MobiLink und den Synchronisationsclient.
3. Löschen Sie alle ODBC-Datenquellen im Zusammenhang mit der praktischen Einführung.
   a. Starten Sie den ODBC-Datenquellen-Administrator.
      Geben Sie an einer Eingabeaufforderung folgenden Befehl ein:

      ```
      odbcad32
      ```

   b. Entfernen Sie die Datenquelle **SIS_CarDealer_LP_DBLSN_CONDB**.
4. Wechseln Sie in das Verzeichnis *c:\MLsis* mit der konsolidierten und der entfernten Datenbank und löschen Sie alle Dateien.

Ergebnisse

Die Daten der praktischen Einführung werden von Ihrem Computer entfernt.

Nächste Schritte

Keine.

Praktische Einführung: Konfigurieren der serverinitiierten Synchronisation unter Verwendung von Gateways

Diese praktische Einführung zeigt, wie eine konsolidierte SQL Anywhere-Datenbank und eine entfernte Datenbank für die serverinitiierte Synchronisation konfiguriert werden. Sie basiert auf dem Beispielcode unter *%SQLANYSAMP16%\MobiLink\SIS_CarDealer*.

Mehrere Beispielimplementierungen serverinitiierter Synchronisationen befinden sich unter *%SQLANYSAMP16%\MobiLink*. Die Namen aller Verzeichnisse der Beispiele für die serverinitiierte Synchronisation beginnen mit dem Präfix SIS_.

Erforderliche Software

- SQL Anywhere 16

Kenntnisse und Erfahrungen

- Grundkenntnisse über MobiLink-Ereignisskripten.

Privilegien

Sie müssen die folgenden Rollen und Privilegien für die konsolidierte Datenbank haben:

- SYS_AUTH_RESOURCE_ROLE-Kompatibilitätsrolle
- MONITOR-Systemprivileg

Sie müssen die folgenden Rollen und Privilegien für die entfernte Datenbank haben:

- SYS_REPLICATION_ADMIN_ROLE-Systemrolle
- SYS_RUN_REPLICATION_ROLE-Systemrolle

Ziele

- Eine konsolidierte SQL Anywhere-Datenbank für die serverinitiierte Synchronisation einrichten
- Serverseitige Eigenschaften konfigurieren
- Push-Anforderungen zum Anstoßen einer serverinitiierten Synchronisation ausgeben

Empfohlene Hintergrundlektüre

- „Serverinitiierte Synchronisation“ auf Seite 1

Lektion 1: Einrichten der konsolidierten Datenbank

In dieser Lektion erstellen Sie mithilfe des Dienstprogramms dbinit eine konsolidierte Datenbank namens **MLconsolidated** mit den für die Synchronisation erforderlichen Skripten. Anschließend definieren Sie eine ODBC-Datenquelle für die Datenbank.

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: Konfigurieren der serverinitiierten Synchronisation unter Verwendung von Gateways“ auf Seite 115.

Aufgabe

1. Erstellen Sie ein Arbeitsverzeichnis, um die konsolidierte Datenbank zu speichern.

   In dieser praktischen Einführung wird davon ausgegangen, dass *c:\MLsis* das Arbeitsverzeichnis ist.

2. Erstellen Sie die konsolidierte SQL Anywhere-Datenbank mithilfe des Dienstprogramms dbinit.

3. Führen Sie den folgenden Befehl aus:

   ```
   dbinit -dba DBA,sql MLconsolidated
   ```

4. Starten Sie die konsolidierte Datenbank mithilfe des Dienstprogramms dbsrv16.

   Führen Sie den folgenden Befehl aus:

   ```
   dbsrv16 MLconsolidated
   ```

5. Wählen Sie **Start** » **Programme** » **SQL Anywhere 16** » **Administrationstools** » **ODBC-Datenquellen-Administrator**.

6. Klicken Sie auf der Registerkarte **Benutzer-DSN** auf **Hinzufügen**.

7. Klicken Sie im Fenster **Neue Datenquelle erstellen** auf **SQL Anywhere 16** und auf **Fertig stellen**.

8. Führen Sie im Fenster **ODBC-Konfiguration für SQL Anywhere** folgende Aufgaben aus:
   a. Klicken Sie auf die Registerkarte **ODBC**.
   b. Im Feld **Datenquellenname** geben Sie **sis_cons** ein.
   c. Klicken Sie auf die Registerkarte **Login**.
   d. Im Feld **Benutzer-ID** geben Sie **DBA** ein.
   e. Im Feld **Kennwort** geben Sie **sql** ein.
   f. Wählen Sie in der Dropdown-Liste **Aktion** die Option **Mit einer laufenden Datenbank auf diesem Computer verbinden**.
   g. Im Feld **Servername** geben Sie **MLconsolidated** ein.
   h. Klicken Sie auf **OK**.

9. Schließen Sie den ODBC-Datenquellenadministrator.

   Klicken Sie auf **OK** im Fenster **ODBC-Datenquellenadministrator**.

Ergebnisse

Die konsolidierte Datenbank wird erstellt und eine ODBC-Datenquelle wird definiert.

Nächste Schritte

Gehen Sie weiter zu „Lektion 2: Generieren eines Datenbankschemas“ auf Seite 117.

Siehe auch

- „ODBC-Datenquellen“ [*SQL Anywhere Server - Datenbankadministration*]

Lektion 2: Generieren eines Datenbankschemas

In dieser Lektion generieren Sie ein Datenbankschema, das die Tabelle Dealer und ein download_cursor-Synchronisationsskript enthält. Eine Tabelle und eine gespeicherte Prozedur werden verwendet, um serverinitiierte Synchronisations-Push-Anforderungen zu generieren.

Voraussetzungen

In dieser Lektion wird davon ausgegangen, dass Sie bereits alle vorherigen Lektionen abgeschlossen haben. Siehe „Lektion 1: Einrichten der konsolidierten Datenbank“ auf Seite 116.

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: Konfigurieren der serverinitiierten Synchronisation unter Verwendung von Gateways“ auf Seite 115.

Aufgabe

1. Klicken Sie auf **Start** » **Programme** » **SQL Anywhere 16** » **Administrationstools** » **Sybase Central**.

2. Führen Sie die folgenden Aufgaben durch, um eine Verbindung zur konsolidierten Datenbank herzustellen.
   a. Klicken Sie auf **Verbindungen** » **Verbinden mit SQL Anywhere 16**.
   b. Im Feld **Benutzer-ID** geben Sie **DBA** ein.
   c. Im Feld **Kennwort** geben Sie **sql** ein.
   d. Klicken Sie in der Dropdown-Liste **Aktion** auf **Mit einer ODBC-Datenquelle verbinden**.
   e. Klicken Sie auf **ODBC-Datenquellenname** und anschließend auf **Durchsuchen**.
   f. Wählen Sie **sis_cons** und klicken Sie dann auf **OK**.
   g. Klicken Sie auf **Verbinden**.

3. Verwenden Sie Interactive SQL, um sich mit Ihrer Datenbank zu verbinden.

   Sie können Interactive SQL über Sybase Central oder eine Eingabeaufforderung starten.

   - Um Interactive SQL aus Sybase Central zu starten, rechtsklicken Sie auf die **MLconsolidated - DBA**-Datenbank und auf **Interactive SQL öffnen**.
   - Um Interactive SQL an einer Eingabeaufforderung zu starten, führen Sie folgenden Befehl aus:

     ```
     dbisql -c "dsn=sis_cons"
     ```

4. Führen Sie die folgende SQL-Anweisung aus, um die Tabelle Dealer zu erstellen und einzurichten:

   ```
   CREATE TABLE Dealer (
       name VARCHAR(10) NOT NULL PRIMARY KEY,
       rating VARCHAR(5),
       last_modified TIMESTAMP DEFAULT TIMESTAMP
   )
   ```

5. Fügen Sie die Daten in die Tabelle Dealer mithilfe der folgenden Anweisungen ein:

   ```
   INSERT INTO Dealer(name, rating) VALUES ('Audi', 'a');
   INSERT INTO Dealer(name, rating) VALUES ('Buick', 'b');
   INSERT INTO Dealer(name, rating) VALUES ('Chrysler', 'c');
   INSERT INTO Dealer(name, rating) VALUES ('Dodge', 'd');
   INSERT INTO Dealer(name, rating) VALUES ('Eagle', 'e');
   INSERT INTO Dealer(name, rating) VALUES ('Ford', 'f');
   INSERT INTO Dealer(name, rating) VALUES ('Geo', 'g');
   INSERT INTO Dealer(name, rating) VALUES ('Honda', 'h');
   INSERT INTO Dealer(name, rating) VALUES ('Isuzu', 'I');
   COMMIT;
   ```

6. Führen Sie das folgende SQL-Skript aus, um die MobiLink-Systemtabellen und die gespeicherten Prozeduren zu erstellen. Ersetzen Sie *C:\Program Files\SQL Anywhere 16* durch den Pfad Ihrer SQL Anywhere 16-Installation.

   ```
   READ "C:\Program Files\SQL Anywhere 16\MobiLink\setup\syncsa.sql"
   ```

7. Führen Sie das folgende SQL-Skript aus, um ein download_cursor-Synchronisationsskript anzugeben und um die Synchronisation aufzuzeichnen:

   ```
   CALL ml_add_table_script(
       'sis_ver1',
       'Dealer',
       'download_cursor',
       'SELECT * FROM Dealer WHERE last_modified >= ?'
   );

   CALL ml_add_table_script(
       'sis_ver1', 'Dealer', 'download_delete_cursor', '--{ml_ignore}'
   );

   COMMIT
   ```

   Schließen Sie Interactive SQL nicht.

Ergebnisse

Das Datenbankschema wird generiert und umfasst eine Tabelle "Dealer" und ein download_cursor-Synchronisationsskript. MobiLink-Systemtabellen und gespeicherte Prozeduren werden installiert.

Nächste Schritte

Gehen Sie weiter zu „Lektion 3: Erstellen einer Tabelle zum Speichern von Push-Anforderungen“ auf Seite 119.

Siehe auch

- „Syntax des SQL Anywhere-Datenbankservers“ [*SQL Anywhere Server - Datenbankadministration*]
- „CREATE TABLE-Anweisung“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- „Schreiben von Synchronisationsskripten“ [*MobiLink - Serveradministration*]
- „download_cursor (Tabellenereignis)“ [*MobiLink - Serveradministration*]

Lektion 3: Erstellen einer Tabelle zum Speichern von Push-Anforderungen

In dieser Lektion erstellen Sie eine Push-Anforderungstabelle, um Push-Anforderungen zu speichern. Wenn der Notifier eine Push-Anforderung entdeckt, sendet er eine Nachricht an ein Gerät.

Voraussetzungen

In dieser Lektion wird davon ausgegangen, dass Sie bereits alle vorherigen Lektionen abgeschlossen haben. Siehe „Lektion 1: Einrichten der konsolidierten Datenbank“ auf Seite 116.

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: Konfigurieren der serverinitiierten Synchronisation unter Verwendung von Gateways“ auf Seite 115.

Aufgabe

1. Sie sollten aus der vorherigen Lektion in Interactive SQL mit Ihrer Datenbank verbunden sein.

   Wenn Sie nicht mit der Datenbank verbunden sind, können Sie Interactive SQL in Sybase Central oder an einer Eingabeaufforderung starten.

   - Um Interactive SQL aus Sybase Central zu starten, rechtsklicken Sie auf die **MLconsolidated - DBA**-Datenbank und auf **Interactive SQL öffnen**.
   - Um Interactive SQL an einer Eingabeaufforderung zu starten, führen Sie folgenden Befehl aus:

     ```
     dbisql -c "dsn=sis_cons"
     ```

2. Führen Sie hierzu folgende SQL-Anweisungen in Interactive SQL aus:

   ```
   CREATE TABLE PushRequest (
       req_id INTEGER DEFAULT AUTOINCREMENT PRIMARY KEY,
       mluser VARCHAR(128),
       subject VARCHAR(128),
       content VARCHAR(128),
       resend_interval VARCHAR(30) DEFAULT '20s',
       time_to_live VARCHAR(30) DEFAULT '1m',
       status VARCHAR(128) DEFAULT 'created'
   )
   COMMIT;
   ```

3. Schließen Sie Interactive SQL.

Ergebnisse

Eine Push-Anforderungstabelle wird zum Speichern von Push-Anforderungen erstellt.

Nächste Schritte

Gehen Sie weiter zu „Lektion 4: Erstellen eines MobiLink-Projekts“ auf Seite 120.

Siehe auch

- „Push-Anforderungen“ auf Seite 7
- „Serverinitiierte Synchronisation“ auf Seite 1
- „Komponenten der serverinitiierten Synchronisation“ auf Seite 2

Lektion 4: Erstellen eines MobiLink-Projekts

In dieser Lektion verbinden Sie sich mit der konsolidierten Datenbank, indem Sie ein neues MobiLink-Projekt erstellen.

Voraussetzungen

In dieser Lektion wird davon ausgegangen, dass Sie bereits alle vorherigen Lektionen abgeschlossen haben. Siehe „Lektion 1: Einrichten der konsolidierten Datenbank“ auf Seite 116.

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: Konfigurieren der serverinitiierten Synchronisation unter Verwendung von Gateways“ auf Seite 115.

Erstellen eines neuen MobiLink-Projekts

1. Klicken Sie auf **Start** » **Programme** » **SQL Anywhere 16** » **Administrationstools** » **Sybase Central**.
2. Klicken Sie auf **Extras** » **MobiLink 16** » **Neues Projekt**.
3. Im Feld **Name** geben Sie **sis_cons_project** ein.
4. Im Feld **Speicherort** geben Sie *C:\MLsis* ein und klicken dann auf **Weiter**.
5. Im Feld **Anzeigename der Datenbank** geben Sie **sis_cons** ein.
6. Klicken Sie auf **Bearbeiten**. Das Fenster **Mit einer allgemeinen ODBC-Datenbank verbinden** erscheint.
7. Im Feld **Benutzer-ID** geben Sie **DBA** ein.
8. Im Feld **Kennwort** geben Sie **sql** ein.
9. Klicken Sie im Feld ODBC-Datenquellenname auf **Durchsuchen** und wählen Sie dann **sis_cons**.
10. Klicken Sie auf **OK** und dann auf **Speichern**.
11. Aktivieren Sie die Option **Kennwort speichern** und klicken Sie auf **Weiter**.
12. Akzeptieren Sie die Standardwerte auf der Seite **Neues entferntes Datenbankschema** und klicken Sie auf **Weiter**.
13. Klicken Sie auf **Weiter** und auf **Fertig stellen**. Klicken Sie auf **OK**.

Ergebnisse

Das MobiLink-Projekt wird erstellt.

Nächste Schritte

Gehen Sie weiter zu „Lektion 5: Konfigurieren des Notifiers“ auf Seite 122.

Lektion 5: Konfigurieren des Notifiers

In dieser Lektion konfigurieren Sie drei Notifier-Ereignisse, um festzulegen, wie der Notifier Push-Anforderungen erstellt, die Anforderungen an den MobiLink Listener sendet und abgelaufene Anforderungen löscht.

Voraussetzungen

In dieser Lektion wird davon ausgegangen, dass Sie bereits alle vorherigen Lektionen abgeschlossen haben. Siehe „Lektion 1: Einrichten der konsolidierten Datenbank“ auf Seite 116.

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: Konfigurieren der serverinitiierten Synchronisation unter Verwendung von Gateways“ auf Seite 115.

Kontext und Bemerkungen

Der Notifier erkennt Änderungen in der konsolidierten Datenbank und erstellt mithilfe des begin_poll-Ereignisses Push-Anforderungen. In diesem Fall füllt das begin_poll-Skript die PushRequest-Tabelle, sofern in der Tabelle Dealer Änderungen vorhanden sind und eine entfernte Datenbank nicht mehr auf dem neuesten Stand ist.

Das request_cursor-Skript ruft Push-Anforderungen ab. Jede Push-Anforderung legt fest, welche Informationen in der Nachricht gesendet werden und welche entfernte Datenbank die Informationen empfängt.

Das Notifier-Ereignis request_delete legt Bereinigungsvorgänge fest. Mit diesem Skript kann der Notifier automatisch implizit gelöschte und abgelaufene Anforderungen entfernen.

Aufgabe

1. Erweitern Sie im linken Fensterausschnitt von Sybase Central unter **MobiLink 16 sis_cons_project**, **Konsolidierte Datenbanken** und dann **sis_cons**.
2. Rechtsklicken Sie auf **Benachrichtigung** und klicken Sie dann auf **Neu** » **Notifier**.
3. Im Feld **Wie soll der Name des neuen Notifiers lauten?** geben Sie **CarDealerNotifier** ein.
4. Klicken Sie auf **Fertig stellen**.
5. Geben Sie das begin_poll-Ereignisskript ein.
   a. Wählen Sie im rechten Fensterausschnitt **CarDealerNotifier** und klicken Sie dann auf **Datei** » **Eigenschaften**.
   b. Klicken Sie auf die Registerkarte **Ereignisse**.
   c. Wählen Sie **begin_poll** aus der Liste **Ereignisse**.
   d. Geben Sie die folgenden SQL-Anweisungen in das angezeigte Textfeld ein:

```
--
-- Insert the last consolidated database
```

```
-- modification date into @last_modified
--
DECLARE @last_modified timestamp;
SELECT MAX(last_modified) INTO @last_modified FROM Dealer;

--
-- Delete processed requests if the mluser is up-to-date
--
DELETE FROM PushRequest
    FROM PushRequest AS p, ml_user AS u, ml_subscription AS s
    WHERE p.status = 'processed'
        AND u.name = p.mluser
        AND u.user_id = s.user_id
        AND @last_modified <= GREATER(s.last_upload_time,
s.last_download_time);

--
-- Insert new requests when a device is not up-to-date
--
INSERT INTO PushRequest(mluser, subject, content)
SELECT u.name, 'sync', 'ignored'
    FROM ml_user as u, ml_subscription as s
    WHERE u.name IN (SELECT name FROM ml_listening WHERE listening =
'y')
        AND u.user_id = s.user_id
        AND @last_modified > greater(s.last_upload_time,
s.last_download_time)
        AND u.name NOT LIKE '%-dblsn'
        AND NOT EXISTS(SELECT * FROM PushRequest
            WHERE PushRequest.mluser = u.name
                AND PushRequest.subject = 'sync')
```

Im ersten größeren Abschnitt des begin_poll-Skripts werden verarbeitete Anforderungen aus der PushRequest-Tabelle entfernt, wenn ein Gerät aktualisiert wurde:

```
@last_modified <= GREATER(s.last_upload_time, s.last_download_time)
```

@last_modified ist das maximale Änderungsdatum in der Tabelle Dealer der konsolidierten Datenbank. Der Ausdruck 'greater(s.last_upload_time, s.last_download_time)' stellt den letzten Synchronisationszeitpunkt für eine entfernte Datenbank dar.

Sie können Push-Anforderungen mit dem request_delete-Ereignis auch direkt löschen. Das begin_poll-Ereignis stellt in diesem Fall jedoch sicher, dass abgelaufene oder implizit gelöschte Anforderungen erst entfernt werden, nachdem eine entfernte Datenbank synchronisiert wurde.

Der nächste Codeabschnitt prüft auf Änderungen in der Spalte last_modified der Tabelle Dealer und gibt Push-Anforderungen für alle aktiven MobiLink Listener aus (aufgelistet in der Tabelle ml_listening), die nicht aktualisiert wurden:

```
@last_modified > GREATER(s.last_upload_time, s.last_download_time)
```

Beim Füllen der Tabelle PushRequest setzt das begin_poll-Skript den Betreff auf 'sync'.

6. Geben Sie das request_cursor-Skript ein.
   a. Klicken Sie auf **request_cursor** aus der Liste **Ereignisse**.
   b. Geben Sie die folgende SQL-Anweisung in das angezeigte Textfeld ein:

```
SELECT
    p.req_id,
```

```
        'Default-DeviceTracker',
        p.subject,
        p.content,
        p.mluser,
        p.resend_interval,
        p.time_to_live
        FROM PushRequest AS p
```

Die Tabelle PushRequest stellt dem request_cursor-Skript Zeilen bereit.

Die Reihenfolge und die Werte in der request_cursor-Ergebnismenge sind wichtig. Der zweite Parameter legt z.B. das Standard-Gateway Default-DeviceTracker fest. Ein Device Tracking-Gateway protokolliert, wie Benutzer erreicht werden, und wählt automatisch UDP oder SMTP für die Verbindung mit entfernten Geräten aus.

7. Geben Sie das request_delete-Skript ein.
   a. Klicken Sie auf **request_delete** aus der Liste **Ereignisse**.
   b. Geben Sie die folgende SQL-Anweisung in das angezeigte Textfeld ein:

```
UPDATE PushRequest SET status='processed' WHERE req_id = ?
```

   Anstatt die Zeile zu löschen, aktualisiert dieses request_delete-Skript den Zustand einer Zeile in der PushRequest-Tabelle auf 'processed'.

8. Klicken Sie auf **OK**, um die Notifier-Ereignisse zu speichern.

Ergebnisse

Drei Notifier-Ereignisse werden definiert.

Nächste Schritte

Gehen Sie weiter zu „Lektion 6: Konfigurieren von Gateways und Netzbetreibern“ auf Seite 124.

Siehe auch

- „request_delete-Ereignis“ auf Seite 41
- „begin_poll-Ereignis“ auf Seite 35
- „Device Tracking-Gateways“ auf Seite 23
- „request_cursor-Ereignis“ auf Seite 40
- „request_delete-Ereignis“ auf Seite 41

Lektion 6: Konfigurieren von Gateways und Netzbetreibern

In dieser Lektion wird davon ausgegangen, dass Sie bereits alle vorherigen Lektionen abgeschlossen haben. Siehe „Lektion 1: Einrichten der konsolidierten Datenbank“ auf Seite 116.

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: Konfigurieren der serverinitiierten Synchronisation unter Verwendung von Gateways“ auf Seite 115.

Gateways sind Mechanismen zum Versenden von Nachrichten. Sie können entweder ein unterstütztes Gateway oder ein Device Tracking-Gateway für die Geräteprotokollierung definieren. Der MobiLink-Server protokolliert, wie Clients zu erreichen sind, wenn Sie ein Device Tracking-Gateway angeben, und wählt automatisch das am besten geeignete Gateway.

Sie verwenden für die Zwecke dieser praktischen Einführung ein Standard-Gateway für die Geräteprotokollierung, sodass eine Konfiguration nicht erforderlich ist.

Gehen Sie weiter zu „Lektion 7: Starten des MobiLink-Servers“ auf Seite 125.

Siehe auch

- „Gateways als Alternative zu Lightweight-Polling-Modulen“ auf Seite 22
- „Gateways und Netzbetreiber“ auf Seite 22
- „Eigenschaften von Device Tracking-Gateways“ auf Seite 49

Lektion 7: Starten des MobiLink-Servers

In dieser Lektion starten Sie den MobiLink-Server mit dem Notifier, sodass Push-Benachrichtigungen an Geräte gesendet werden können.

Voraussetzungen

In dieser Lektion wird davon ausgegangen, dass Sie bereits alle vorherigen Lektionen abgeschlossen haben. Siehe „Lektion 1: Einrichten der konsolidierten Datenbank“ auf Seite 116.

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: Konfigurieren der serverinitiierten Synchronisation unter Verwendung von Gateways“ auf Seite 115.

Aufgabe

- Stellen Sie eine Verbindung mit Ihrer konsolidierten Datenbank her.

  Führen Sie den folgenden Befehl aus:

  ```
  mlsrv16 -notifier -c "dsn=sis_cons" -o serverOut.txt -v+ -dl -zu+ -x tcpip
  ```

  Die folgende Tabelle beschreibt die in dieser Lektion verwendeten mlsrv16-Optionen. Die Optionen -o - und -v bieten Informationen zur Fehlersuche und -behebung. Die Verwendung dieser Protokollierungsoptionen empfiehlt sich in einer Entwicklungsumgebung. Aus Gründen der Performance wird -v bei der Produktion normalerweise nicht eingesetzt.

Option	Beschreibung
-notifier	Startet alle aktivierten Notifier für die serverinitiierte Synchronisation. Siehe „mlsrv16-Option -notifier “ [*MobiLink - Serveradministration*].

Option	Beschreibung
-c	Legt die Verbindungszeichenfolge fest. Siehe „mlsrv16-Option -c “ [*MobiLink - Serveradministration*].
-o	Legt die Meldungslogdatei *serverOut.txt* fest Siehe „mlsrv16-Option -o“ [*MobiLink - Serveradministration*].
-v+	Gibt an, welche Informationen protokolliert werden. Mit -v+ wird die maximale ausführliche Protokollierung aktiviert. Siehe „mlsrv16-Option -v “ [*MobiLink - Serveradministration*].
-zu+	Fügt neue Benutzer automatisch hinzu. Siehe „mlsrv16-Option -zu “ [*MobiLink - Serveradministration*].
-x	Legt das Kommunikationsprotokoll und Protokolloptionen für MobiLink-Clients fest. Siehe „mlsrv16-Option -x“ [*MobiLink - Serveradministration*].

Ergebnisse

Das MobiLink-Server-Meldungsfenster wird eingeblendet. Der Notifier gibt an, dass er für den Empfang von Push-Benachrichtigung von Geräten bereit ist.

Nächste Schritte

Gehen Sie weiter zu „Lektion 8: Einrichten einer entfernten Datenbank“ auf Seite 126.

Siehe auch

Weitere Hinweise zu den Themen dieser Lektion finden Sie unter:

- „MobiLink-Server“ [*MobiLink - Serveradministration*]
- „MobiLink-Serveroptionen“ [*MobiLink - Serveradministration*]

Lektion 8: Einrichten einer entfernten Datenbank

In dieser Lektion erstellen Sie eine entfernte SQL Anywhere-Datenbank, eine Synchronisationspublikation, einen Benutzer und eine Subskription.

Voraussetzungen

In dieser Lektion wird davon ausgegangen, dass Sie bereits alle vorherigen Lektionen abgeschlossen haben. Siehe „Lektion 1: Einrichten der konsolidierten Datenbank“ auf Seite 116.

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: Konfigurieren der serverinitiierten Synchronisation unter Verwendung von Gateways“ auf Seite 115.

Aufgabe

1. Erstellen Sie Ihre MobiLink Clientdatenbank mit dem Befehlszeilen-Dienstprogramm dbinit.

   Führen Sie den folgenden Befehl aus:

   ```
   dbinit -dba DBA,sql remote1
   ```

2. Starten Sie Ihre MobiLink-Clientdatenbank mit dem Befehlszeilen-Dienstprogramm dbsrv16.

   Führen Sie den folgenden Befehl aus:

   ```
   dbsrv16 remote1
   ```

3. Verbinden Sie sich über Interactive SQL mit Ihrem MobiLink-Client.

   Führen Sie den folgenden Befehl aus:

   ```
   dbisql -c "SERVER=remote1;UID=DBA;PWD=sql"
   ```

4. Erstellen Sie die Dealer-Tabelle.

   Führen Sie hierzu folgende SQL-Anweisung in Interactive SQL aus:

   ```
   CREATE TABLE Dealer (
       name            VARCHAR(10) NOT NULL PRIMARY KEY,
       rating          VARCHAR(5),
       last_modified   TIMESTAMP DEFAULT TIMESTAMP
   )
   COMMIT;
   ```

5. Erstellen Sie Ihren MobiLink-Synchronisationsbenutzer sowie die Publikation und Subskription.

   Führen Sie hierzu folgende SQL-Anweisung in Interactive SQL aus:

   ```
   CREATE PUBLICATION car_dealer_pub (table Dealer);
   CREATE SYNCHRONIZATION USER sis_user1;
   CREATE SYNCHRONIZATION SUBSCRIPTION
       TO car_dealer_pub
       FOR sis_user1
       OPTION scriptversion='sis_ver1';
   COMMIT;
   ```

Ergebnisse

Eine entfernte SQL Anywhere-Datenbank, eine Synchronisationspublikation, ein Benutzer und eine Subskription werden erstellt.

Nächste Schritte

Gehen Sie weiter zu „Lektion 9: Konfigurieren des MobiLink Listeners“ auf Seite 128.

Siehe auch

- „MobiLink-Clients“ [*MobiLink - Clientadministration*]
- „CREATE TABLE-Anweisung“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- „Publikationen“ [*MobiLink - Clientadministration*]
- „CREATE PUBLICATION-Anweisung [MobiLink] [SQL Remote]“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- „CREATE SYNCHRONIZATION USER-Anweisung [MobiLink]“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- „CREATE SYNCHRONIZATION SUBSCRIPTION-Anweisung [MobiLink]“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- „Skriptversionen“ [*MobiLink - Serveradministration*]

Lektion 9: Konfigurieren des MobiLink Listeners

In dieser Lektion konfigurieren Sie den MobiLink Listener, indem Sie die MobiLink Listener-Optionen in einer Textdatei speichern und dann dblsn unter Angabe des Dateinamens in der Befehlszeile ausführen.

Voraussetzungen

In dieser Lektion wird davon ausgegangen, dass Sie bereits alle vorherigen Lektionen abgeschlossen haben. Siehe „Lektion 1: Einrichten der konsolidierten Datenbank“ auf Seite 116.

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: Konfigurieren der serverinitiierten Synchronisation unter Verwendung von Gateways“ auf Seite 115.

Aufgabe

1. MobiLink Listener-Befehlsdatei

   Erstellen Sie eine MobiLink Listener-Befehlsdatei, indem Sie eine Textdatei mit folgendem Inhalt erstellen:

```
#----------------------------------
# Verbosity level
-v2

# Show notification messages in console and log
-m

# Polling interval, in seconds
-i 3

# Truncate, then write output to dblsn.txt
-ot dblsn.txt

# MobiLink address and connect parameter for dblsn
-x "host=localhost"

# Enable device tracking and specify the MobiLink user name.
-t+ sis_user1

# Message handlers
```

```
# Synchronize using dbmlsync
-l "subject=sync;
action='start dbmlsync.exe
 -c SERVER=remote1;UID=DBA;PWD=sql
 -o dbmlsyncOut.txt
';"
```

2. In dieser praktischen Einführung wird davon ausgegangen, dass *c:\MLsis* das Arbeitsverzeichnis für serverseitige Komponenten ist. Speichern Sie die Textdatei als *mydblsn.txt* in diesem Verzeichnis.

3. Starten Sie den MobiLink Listener.

   Navigieren Sie an einer Eingabeaufforderung zu *c:\MLsis* oder zu dem Verzeichnis, in dem Sie die MobiLink Listener-Befehlsdatei gespeichert haben.

   Starten Sie den MobiLink Listener mit dem folgenden Befehl:

   ```
   dblsn @mydblsn.txt
   ```

Ergebnisse

Das Fenster **MobiLink Listener für Windows** erscheint mit der Anzeige, dass der MobiLink Listener im Ruhezustand ist.

Wenn Protokollinformationen in die konsolidierte Datenbank hochgeladen werden, wird im MobiLink-Servermeldungsfenster ein neuer Eintrag angezeigt. Diese Informationen zeigen die erfolgreiche Kommunikation zwischen dem MobiLink Listener und dem MobiLink-Server an.

Nächste Schritte

Gehen Sie weiter zu „Lektion 10: Ausgeben von Push-Anforderungen“ auf Seite 129.

Siehe auch

- „Listener“ auf Seite 14
- „MobiLink Listener-Dienstprogramm für Windows-Geräte (dblsn)“ auf Seite 55
- „dblsn-Option @data“ auf Seite 60

Lektion 10: Ausgeben von Push-Anforderungen

Bei der serverinitiierten Synchronisation können Sie Push-Anforderungen ausgeben, indem Sie die PushRequest-Tabelle direkt füllen oder eine Änderung in der Tabelle Dealer durchführen. In zweiten Fall erkennt das Notifier-Skript begin_poll die Änderung in der Tabelle Dealer und füllt die PushRequest-Tabelle. In beiden Fällen liefert die Tabelle PushRequest Zeilen für das Notifier-Skript request_cursor, das festlegt, wie Nachrichten von entfernten Geräten empfangen werden.

Voraussetzungen

In dieser Lektion wird davon ausgegangen, dass Sie bereits alle vorherigen Lektionen abgeschlossen haben. Siehe „Lektion 1: Einrichten der konsolidierten Datenbank“ auf Seite 116.

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: Konfigurieren der serverinitiierten Synchronisation unter Verwendung von Gateways“ auf Seite 115.

Direktes Einfügen einer Push-Anforderung in die PushRequest-Tabelle und Anstoßen der serverinitiierten Synchronisation

1. Verbinden Sie sich über Interactive SQL mit Ihrer konsolidierten Datenbank, falls dies nicht schon geschehen ist.

   Führen Sie den folgenden Befehl aus:

   ```
   dbisql -c "dsn=sis_cons"
   ```

2. Führen Sie die folgenden SQL-Anweisungen aus:

   ```
   INSERT INTO PushRequest(mluser, subject, content)
       VALUES ('sis_user1', 'sync', 'not used');
   COMMIT;
   ```

3. Warten Sie ein paar Sekunden, bis die Synchronisation ausgeführt wird.

   Wenn die Tabelle PushRequest gefüllt wurde, stellt sie dem request_cursor-Skript des Notifiers Zeilen bereit. Das request_cursor-Skript legt fest, welche Informationen in der Nachricht gesendet werden und welche entfernten Geräte die Informationen empfangen.

4. Führen Sie die folgenden SQL-Anweisungen aus, um eine Änderung in der **Dealer**-Tabelle der konsolidierten Datenbank vorzunehmen und die serverinitiierte Synchronisation auszulösen:

   ```
   UPDATE Dealer
       SET RATING = 'B' WHERE name = 'Geo';
   COMMIT;
   ```

5. Warten Sie ein paar Sekunden, bis die Synchronisation ausgeführt wird.

   In diesem Fall erkennt das Notifier-Skript begin_poll die Änderungen in der Tabelle Dealer und füllt die PushRequest-Tabelle entsprechend. Wie zuvor legt das Notifier-Skript request_cursor nach dem Füllen der PushRequest-Tabelle fest, welche Informationen in der Nachricht gesendet werden und welche entfernten Geräte die Informationen empfangen.

6. Überprüfen Sie, ob die Tabelle "Dealer" in der entfernten Datenbank aktualisiert wurde.

   Führen Sie hierzu die folgende SQL-Anweisung aus:

   ```
   SELECT * FROM Dealer
   ```

   Die Bewertung für **Geo** sollte nun **B** sein.

Ergebnisse

Eine Push-Anforderung wird direkt in die Tabelle PushRequest eingefügt, wodurch die serverinitiierte Synchronisation ausgelöst wird.

Nächste Schritte

Gehen Sie weiter zu „Aufräumen“ auf Seite 131.

Siehe auch

- Push-Anforderungen generieren auf Seite 9
- „INSERT-Anweisung“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- „UPDATE-Anweisung“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]

Aufräumen

Entfernen Sie die Daten der praktischen Einführung von Ihrem Computer.

Voraussetzungen

In dieser Lektion wird davon ausgegangen, dass Sie bereits alle vorherigen Lektionen abgeschlossen haben. Siehe „Lektion 1: Einrichten der konsolidierten Datenbank“ auf Seite 116.

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: Konfigurieren der serverinitiierten Synchronisation unter Verwendung von Gateways“ auf Seite 115.

Aufgabe

1. Schließen Sie Interactive SQL.
2. Schließen Sie SQL Anywhere, MobiLink und den Synchronisationsclient.
3. Löschen Sie alle ODBC-Datenquellen im Zusammenhang mit der praktischen Einführung.
   a. Starten Sie den ODBC-Datenquellen-Administrator.

      Geben Sie an einer Eingabeaufforderung folgenden Befehl ein:

      ```
      odbcad32
      ```

   b. Entfernen Sie die Datenquelle **sis_cons**.
4. Wechseln Sie in das Verzeichnis *c:\MLsis* mit der konsolidierten und der entfernten Datenbank und löschen Sie alle Dateien.

Ergebnisse

Die Daten der praktischen Einführung werden von Ihrem Computer entfernt.

Nächste Schritte

Keine.

Index

Symbole

A

B

C

D

E

F

G

H

O

P

R

S

U

V

Z