iAnywhere SOLUTIONS™
A SYBASE COMPANY

Mobile Link クライアント

パート番号：DC00205-01-0902-01
改訂：2005 年 3 月

版権

Copyright © 2005 iAnywhere Solutions, Inc., Sybase, Inc. All rights reserved.

ここに記載されている内容を iAnywhere Solutions, Inc.、Sybase, Inc. またはその関連会社の書面による事前許可を得ずに電子的、機械的、手作業、光学的、またはその他のいかなる手段によっても複製、転載、翻訳することを禁じます。

Sybase、SYBASE のロゴ、Adaptive Server、AnswerBase、Anywhere、EIP、Embedded SQL、Enterprise Connect、Enterprise Portal、GainMomentum、iAnywhere、jConnect MASS DEPLOYMENT、Netimpact、ObjectConnect、ObjectCycle、OmniConnect、Open ClientConnect、Open ServerConnect、PowerBuilder、PowerDynamo、Powersoft、Quickstart Datamart、Replication Agent、Replication Driver、SQL Anywhere、SQL Central、SQL Remote、Support Plus、SWAT、Sybase IQ、Sybase System 11、Sybase WAREHOUSE、SyBooks、XA-Library は米国法人 Sybase, Inc. の登録商標です。Backup Server、Client-Library、jConnect for JDBC、MainframeConnect、Net-Gateway、Net-Library、Open Client、Open Client/Server、S-Designor、SQL Advantage、SQL Debug、SQL Server、SQL Server Manager、Sybase Central、Watcom、Web.SQL、XP Server は米国法人 Sybase, Inc. の商標です。

Certicom、MobileTrust および、SSL Plus は Certicom Corp. の商標です。Security Builder は Certicom Corp. の登録商標です。

ここに記載されている上記以外の社名および製品名は、各社の商標または登録商標の場合があります。

目次

はじめに vii
SQL Anywhere Studio のマニュアル viii
表記の規則 xii
CustDB サンプル・データベース xv
詳細情報の検索／フィードバックの提供 xvi

1 Mobile Link クライアントの紹介 3
Adaptive Server Anywhere クライアント 4
Ultra Light クライアント 5
クライアントの通信プロトコルの指定 6
Mobile Link ユーザ 7

2 Mobile Link ユーザの認証 9
Mobile Link ユーザの概要 10
ユーザ認証メカニズムの選択 14
ユーザ認証アーキテクチャ 16
ユーザの最初のパスワードを設定する 20
新しいユーザからの同期 22
エンド・ユーザに対するパスワード入力の要求 24
パスワードの変更 25
カスタム・ユーザ認証 26

3 ユーティリティ 33
ActiveSync プロバイダ・インストール・ユーティリティ 34
Mobile Link クライアント・データベース抽出ユーティリティ (旧式) 38

4 Mobile Link クライアントのネットワーク・プロトコル・オプション 43
プロトコル・オプション 44

5 Adaptive Server Anywhere クライアント 75
リモート・データベースの作成.....76
データのパブリッシュ 82
Mobile Link ユーザの作成 92
Mobile Link 同期ユーザのサブスクリプションの作成 98
同期の開始 102
ActiveSync 同期の使用 108
削除同期の一時停止.....114
同期のスケジュール.....115
Adaptive Server Anywhere バージョン 7 Mobile Link クライアント.....117

6 Adaptive Server Anywhere クライアントの同期パラメータ 123
Mobile Link 同期クライアント 124
dbmlsync オプション 129

7 Mobile Link SQL 文..... 199
ALTER PUBLICATION 文 200
ALTER SYNCHRONIZATION SUBSCRIPTION 文 [Mobile Link].....202
ALTER SYNCHRONIZATION USER 文 [Mobile Link] 205
CREATE PUBLICATION 文.....207
CREATE SYNCHRONIZATION SUBSCRIPTION 文 [Mobile Link] 210
CREATE SYNCHRONIZATION USER 文 [Mobile Link].....213
DROP PUBLICATION 文 216
DROP SYNCHRONIZATION SUBSCRIPTION 文 [Mobile Link].....217
DROP SYNCHRONIZATION USER 文 [Mobile Link] 219
START SYNCHRONIZATION DELETE 文 [Mobile Link].....220
STOP SYNCHRONIZATION DELETE 文 [Mobile Link].....222

8 dbmlsync クライアント・イベント・フック 225
クライアント同期処理のカスタマイズ.....226
sp_hook_dbmlsync_abort.....234
sp_hook_dbmlsync_begin 237
sp_hook_dbmlsync_connect_failed.....239
sp_hook_dbmlsync_delay 243
sp_hook_dbmlsync_download_begin 246
sp_hook_dbmlsync_download_com_error 248
sp_hook_dbmlsync_download_end.....250
sp_hook_dbmlsync_download_fatal_sql_error 252

sp_hook_dbmlsync_download_log_ri_violation 254
sp_hook_dbmlsync_download_ri_violation.....257
sp_hook_dbmlsync_download_sql_error 260
sp_hook_dbmlsync_download_table_begin.....263
sp_hook_dbmlsync_download_table_end 265
sp_hook_dbmlsync_end 268
sp_hook_dbmlsync_log_rescan.....271
sp_hook_dbmlsync_logscan_begin.....273
sp_hook_dbmlsync_logscan_end 275
sp_hook_dbmlsync_process_return_code.....278
sp_hook_dbmlsync_schema_upgrade 281
sp_hook_dbmlsync_set_extended_options 284
sp_hook_dbmlsync_upload_begin.....286
sp_hook_dbmlsync_upload_end.....288
sp_hook_dbmlsync_validate_download_file 293

9 dbmlsync 統合コンポーネント.....297
概要 298
dbmlsync 統合コンポーネントの設定 299
dbmlsync 統合コンポーネントのメソッド.....304
dbmlsync 統合コンポーネントのプロパティ 306
dbmlsync 統合コンポーネントのイベント.....312
IRowTransferData インタフェース 328

10 dbmlsync の DBTools インタフェース 333
概要 334
dbmlsync の DBTools インタフェースの設定.....336

11 Ultra Light クライアント.....347
概要 348
Ultra Light アプリケーションへの同期の追加 349
同期させるデータの選択.....351
外部キー循環.....358
Ultra Light ネットワーク・プロトコル 360
プライマリ・キーの一意性の管理 365
Palm Computing Platform での Ultra Light データベースの同期.....372
Windows CE での Ultra Light データベースの同期.....391

12 Ultra Light 同期パラメータ 397
同期パラメータ 398
Ultra Light 同期クライアントのネットワーク・プロトコルのオプション.....433

索引 447

はじめに

このマニュアルの内容

このマニュアルでは、セッションベースのリレーショナル・データベース同期システムである Mobile Link について説明します。Mobile Link テクノロジは、双方向レプリケーションを可能にし、モバイル・コンピューティング環境に非常に適しています。

対象読者

このマニュアルは、使用している情報システムに同期またはレプリケーションを追加したいと考えている Adaptive Server Anywhere ユーザと他のリレーショナル・データベース・システムのユーザを対象としています。

始める前に

Mobile Link と他の同期／レプリケーション・テクノロジの比較については、『SQL Anywhere Studio の紹介』> 「レプリケーション・テクノロジの概要」を参照してください。

SQL Anywhere Studio のマニュアル

このマニュアルは、SQL Anywhere のマニュアル・セットの一部です。この項では、マニュアル・セットに含まれる各マニュアルと使用法について説明します。

SQL Anywhere Studio のマニュアル

SQL Anywhere Studio のマニュアルは、各マニュアルを 1 つの大きなヘルプ・ファイルにまとめたオンライン形式、マニュアル別の PDF ファイル、および有料の製本版マニュアルで提供されます。SQL Anywhere Studio のマニュアルは、次の分冊マニュアルで構成されています。

- **『SQL Anywhere Studio の紹介』** このマニュアルでは、SQL Anywhere Studio のデータベース管理と同期テクノロジの概要について説明します。また、SQL Anywhere Studio を構成する各部分について説明するチュートリアルも含まれています。
- **『SQL Anywhere Studio 新機能ガイド』** このマニュアルは、SQL Anywhere Studio のこれまでのリリースのユーザを対象としています。ここでは、製品の今回のリリースと以前のリリースで導入された新機能をリストし、アップグレード手順を説明しています。
- **『Adaptive Server Anywhere データベース管理ガイド』** このマニュアルでは、データベースおよびデータベース・サーバの実行、管理、設定について説明しています。
- **『Adaptive Server Anywhere SQL ユーザーズ・ガイド』** このマニュアルでは、データベースの設計と作成の方法、データのインポート・エクスポート・変更の方法、データの検索方法、ストアド・プロシージャとトリガの構築方法について説明します。
- **『Adaptive Server Anywhere SQL リファレンス・マニュアル』** このマニュアルは、Adaptive Server Anywhere で使用する SQL 言語の完全なリファレンスです。また、Adaptive Server Anywhere のシステム・テーブルとシステム・プロシージャについても説明しています。
- **『Adaptive Server Anywhere プログラミング・ガイド』** このマニュアルでは、C、C++、Java プログラミング言語を使用してデータベース・アプリケーションを構築、配備する方法につい

て説明します。Visual Basic や PowerBuilder などのツールのユーザは、それらのツールのプログラミング・インタフェースを使用できます。また、Adaptive Server Anywhere ADO.NET データ・プロバイダについても説明します。

- **『Adaptive Server Anywhere SNMP Extension Agent ユーザーズ・ガイド』** このマニュアルでは、SNMP 管理アプリケーションとともに使用するように Adaptive Server Anywhere SNMP Extension Agent を設定して、Adaptive Server Anywhere データベースを管理する方法を説明します。

- **『Adaptive Server Anywhere エラー・メッセージ』** このマニュアルでは、Adaptive Server Anywhere エラー・メッセージの完全なリストを、その診断情報とともに説明します。

- **『SQL Anywhere Studio セキュリティ・ガイド』** このマニュアルでは、Adaptive Server Anywhere データベースのセキュリティ機能について説明します。Adaptive Server Anywhere 7.0 は、米国政府から TCSEC (Trusted Computer System Evaluation Criteria) の C2 セキュリティ評価を授与されています。このマニュアルには、Adaptive Server Anywhere の現在のバージョンを、C2 基準を満たした環境と同等の方法で実行することを望んでいるユーザにとって役に立つ情報が含まれています。

- **『Mobile Link 管理ガイド』** このマニュアルでは、モバイル・コンピューティング用の Mobile Link データ同期システムについてあらゆる角度から説明します。このシステムによって、Oracle、Sybase、Microsoft、IBM の単一データベースと、Adaptive Server Anywhere や Ultra Light の複数データベースの間でのデータ共有が可能になります。

- **『Mobile Link サーバ起動同期ユーザーズ・ガイド』** このマニュアルでは、Mobile Link のサーバによって開始される同期について説明します。サーバによって開始される同期とは、統合データベースから同期の開始を可能にする Mobile Link の機能です。

- **『Mobile Link クライアント』** このマニュアルでは、Adaptive Server Anywhere リモート・データベースと Ultra Light リモート・データベースの設定を行い、これらを同期させる方法について説明します。

- **『Mobile Link チュートリアル』** このマニュアルには、Mobile Link アプリケーションの設定と実行を行う方法を説明するチュートリアルがいくつか用意されています。

- **『QAnywhere ユーザーズ・ガイド』** このマニュアルでは、Mobile Link QAnywhere について説明します。Mobile Link QAnywhere は、従来のデスクトップ・クライアントやラップトップ・クライアントだけでなく、モバイル・クライアントや無線クライアント用のメッセージング・アプリケーションの開発と展開を可能にするメッセージング・プラットフォームです。

- **『Mobile Link およびリモート・データ・アクセスの ODBC ドライバ』** このマニュアルでは、Mobile Link 同期サーバから、または Adaptive Server Anywhere リモート・データ・アクセスによって、Adaptive Server Anywhere 以外の統合データベースにアクセスするための ODBC ドライバの設定方法について説明します。

- **『SQL Remote ユーザーズ・ガイド』** このマニュアルでは、モバイル・コンピューティング用の SQL Remote データ・レプリケーション・システムについて、あらゆる角度から説明します。このシステムによって、Adaptive Server Anywhere または Adaptive Server Enterprise の単一データベースと Adaptive Server Anywhere の複数データベースの間で、電子メールやファイル転送などの間接的リンクを使用したデータ共有が可能になります。

- **『SQL Anywhere Studio ヘルプ』** このマニュアルには、Sybase Central や Interactive SQL、その他のグラフィカル・ツールに関するコンテキスト別のヘルプが含まれています。これは、製本版マニュアル・セットには含まれていません。

- **『Ultra Light データベース・ユーザーズ・ガイド』** このマニュアルは、Ultra Light 開発者を対象としています。ここでは、Ultra Light データベース・システムの概要について説明します。また、すべての Ultra Light プログラミング・インタフェースに共通する情報を提供します。

- **Ultra Light のインタフェースに関するマニュアル** 各 Ultra Light プログラミング・インタフェースには、それぞれに対応するマニュアルを用意しています。これらのインタフェースは、RAD

(ラピッド・アプリケーション開発) 用の Ultra Light コンポーネントとして提供されているものと、C、C++、Java 開発用の静的インタフェースとして提供されているものがあります。

このマニュアル・セットの他に、PowerDesigner と InfoMaker には、独自のオンライン・マニュアル (英語版) がそれぞれ用意されています。

マニュアルの形式

SQL Anywhere Studio のマニュアルは、次の形式で提供されています。

- **オンライン・マニュアル**　オンライン・マニュアルには、SQL Anywhere Studio の完全なマニュアルがあり、SQL Anywhere ツールに関する印刷マニュアルとコンテキスト別のヘルプの両方が含まれています。オンライン・マニュアルは、製品のメンテナンス・リリースごとに更新されます。これは、最新の情報を含む最も完全なマニュアルです。

  Windows オペレーティング・システムでオンライン・マニュアルにアクセスするには、[スタート] － [プログラム] － [SQL Anywhere 9] － [オンライン・マニュアル] を選択します。オンライン・マニュアルをナビゲートするには、左ウィンドウ枠で HTML ヘルプの目次、索引、検索機能を使用し、右ウィンドウ枠でリンク情報とメニューを使用します。

  UNIX オペレーティング・システムでオンライン・マニュアルにアクセスするには、SQL Anywhere のインストール・ディレクトリに保存されている HTML マニュアルを参照してください。

- **PDF 版マニュアル**　SQL Anywhere の各マニュアルは、Adobe Acrobat Reader で表示できる PDF ファイルで提供されています。

  PDF 版マニュアルは、オンライン・マニュアルまたは Windows の [スタート] メニューから利用できます。

- **製本版マニュアル**　製本版マニュアルをご希望の方は、ご購入いただいた販売代理店または弊社営業担当までご連絡ください。

表記の規則

この項では、このマニュアルで使用されている書体およびグラフィック表現の規則について説明します。

SQL 構文の表記規則

SQL 構文の表記には、次の規則が適用されます。

- **キーワード**　SQL キーワードはすべて次の例に示す ALTER TABLE のように大文字で表記します。

  ALTER TABLE [*owner*.]*table-name*

- **プレースホルダ**　適切な識別子または式で置き換えられる項目は、次の例に示す *owner* や *table-name* のように表記します。

  ALTER TABLE [*owner*.]*table-name*

- **繰り返し項目**　繰り返し項目のリストは、次の例に示す *column-constraint* のように、リストの要素の後ろに省略記号 (ピリオド 3 つ …) を付けて表します。

  ADD *column-definition* [*column-constraint*, …]

  複数の要素を指定できます。複数の要素を指定する場合は、各要素間をカンマで区切る必要があります。

- **オプション部分**　文のオプション部分は角カッコで囲みます。

  RELEASE SAVEPOINT [*savepoint-name*]

  この例では、角カッコで囲まれた *savepoint-name* がオプション部分です。角カッコは入力しないでください。

- **オプション**　項目リストから 1 つだけ選択するか、何も選択しなくてもよい場合は、項目間を縦線で区切り、リスト全体を角カッコで囲みます。

  [**ASC** | **DESC**]

  この例では、ASC と DESC のどちらか 1 つを選択しても、どちらも選択しなくてもかまいません。角カッコは入力しないでください。

- **選択肢**　オプションの中の 1 つを必ず選択しなければならない場合は、選択肢を中カッコで囲み、縦棒で区切ります。

  [**QUOTES** { **ON** | **OFF** }]

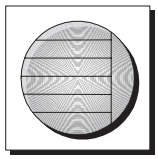
  QUOTES オプションを使用する場合は、ON または OFF のどちらかを選択する必要があります。角カッコと中カッコは入力しないでください。

グラフィック・アイコン

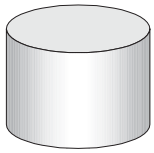
このマニュアルでは、次のアイコンを使用します。

- クライアント・アプリケーション

- Sybase Adaptive Server Anywhere などのデータベース・サーバ



- データベース。高度な図では、データベースとデータベースを管理するデータ・サーバの両方をこのアイコンで表します。



- レプリケーションまたは同期のミドルウェア。ソフトウェアのこれらの部分は、データベース間のデータ共有を支援します。たとえば、Mobile Link 同期サーバ、SQL Remote Message Agent などがあげられます。

- プログラミング・インタフェース

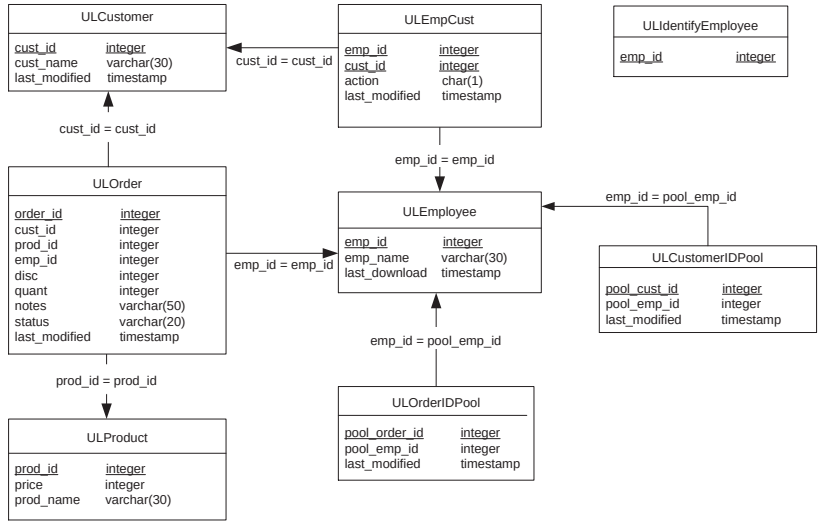
CustDB サンプル・データベース

Mobile Link と Ultra Light のマニュアルでは、多くの例で Ultra Light のサンプル・データベースが使用されています。

Ultra Light サンプル・データベースのリファレンス・データベースは、*custdb.db* という名前のファイルに保存され、SQL Anywhere ディレクトリのサブディレクトリ *Samples¥UltraLite¥CustDB* に置かれています。全面的にこのデータベースを使用して構築したアプリケーションも提供されています。

サンプル・データベースは、あるハードウェア販売会社の販売管理データベースです。データベースには、この販売会社の顧客、製品、営業戦力に関する情報が入っています。

次の図は、CustDB データベース内のテーブルと、各テーブル間の関係を示しています。



詳細情報の検索／フィードバックの提供

詳細情報の検索

詳しい情報やリソース (コード交換など) については、iAnywhere Developer Network (http://www.ianywhere.com/developer/) を参照してください。

ご質問がある場合や支援が必要な場合は、次に示す iAnywhere Solutions ニュースグループのいずれかにメッセージをお寄せください。

ニュースグループにメッセージをお送りいただく際には、ご使用の SQL Anywhere Studio バージョンのビルド番号を明記し、現在発生している問題について詳しくお知らせくださいますようお願いいたします。バージョン情報は、コマンド・プロンプトで **dbeng9 -v** と入力して確認できます。

ニュースグループは、ニュース・サーバ *forums.sybase.com* にあります (ニュースグループにおけるサービスは英語でのみの提供となります)。以下のニュースグループがあります。

- sybase.public.sqlanywhere.general
- sybase.public.sqlanywhere.linux
- sybase.public.sqlanywhere.mobilink
- sybase.public.sqlanywhere.product_futures_discussion
- sybase.public.sqlanywhere.replication
- sybase.public.sqlanywhere.ultralite
- ianywhere.public.sqlanywhere.qanywhere

ニュースグループに関するお断り

iAnywhere Solutions は、ニュースグループ上に解決策、情報、または意見を提供する義務を負うものではありません。また、システム・オペレータ以外のスタッフにこのサービスを監視させて、操作状況や可用性を保証する義務もありません。

iAnywhere Solutions のテクニカル・アドバイザとその他のスタッフは、時間のある場合にかぎりニュースグループでの支援を行います。こうした支援は基本的にボランティアで行われるため、解決策や情報を定期的に提供できるとはかぎりません。支援できるかどうかは、スタッフの仕事量に左右されます。

フィードバック

このマニュアルに関するご意見、ご提案、フィードバックをお寄せください。

マニュアルに関するご意見、ご提案は、SQL Anywhere ドキュメンテーション・チームの iasdoc@ianywhere.com 宛てに電子メールでお寄せください。このアドレスに送信された電子メールに返信はいたしませんが、お寄せいただいたご意見、ご提案は必ず読ませていただきます。

マニュアルまたはソフトウェアについてのフィードバックは、上記のニュースグループを通してお寄せいただいてもかまいません。

第 1 部　Mobile Link クライアントの紹介

第 1 部では、Mobile Link 同期に使用できるクライアントについて紹介するとともに、あらゆる種類の Mobile Link クライアントに共通する情報を示します。

第 1 章

Mobile Link クライアントの紹介

この章の内容

この章では、Mobile Link リモート・データベースについて説明します。

Mobile Link は、2 種類のリモート・データベース、Adaptive Server Anywhere と Ultra Light をサポートします。

Adaptive Server Anywhere クライアント

dbmlsync というコマンド・ライン・ユーティリティを実行すると、同期を開始します。このユーティリティは、リモート・データベースに接続し、リモート・データベースのトランザクション・ログに含まれる情報を使用してアップロード・ストリームを準備します。次に、同期パブリケーションと同期サブスクリプションに保管された情報を使用して Mobile Link 同期サーバに接続し、データを交換します。

Adaptive Server Anywhere クライアントの詳細については、『Mobile Link クライアント』> 「Adaptive Server Anywhere クライアント」を参照してください。

dbmlsync コマンド・ライン・オプションの詳細については、「Adaptive Server Anywhere クライアントの同期パラメータ」123 ページを参照してください。

Ultra Light クライアント

SQL Anywhere Studio で使用可能な Ultra Light テクノロジで構築されたアプリケーションは、適切な Mobile Link 同期機能の呼び出しが含まれていると自動的に Mobile Link が有効になります。SQL Anywhere Studio に含まれる Ultra Light 開発ツールで Ultra Light アプリケーションを構築すると、同期論理が自動的に組み込まれます。

Ultra Light のアプリケーションとライブラリは、アプリケーション側での同期アクションを処理します。Ultra Light アプリケーションは、同期をほとんど考慮しないで記述できます。Ultra Light ランタイムは、前回の同期以後に加えられた変更を追跡します。

TCP/IP、HTTP、HTTPS、または ActiveSync を使用している場合には、同期関数を 1 回呼び出すと、アプリケーションから同期が開始されます。

HotSync 用のインタフェースは、わずかに異なります。同期がアプリケーションまたは HotSync から開始されると、Mobile Link 同期サーバと Ultra Light ランタイムが同期中の動作を制御します。

Ultra Light クライアントの詳細については、『Mobile Link クライアント』> 「Ultra Light クライアント」を参照してください。

同期の開始についての詳細は、『Mobile Link クライアント』> 「Ultra Light 同期パラメータ」を参照してください。

クライアントの通信プロトコルの指定

Mobile Link 同期サーバでは、-x コマンド・ライン・オプションを使用して、同期クライアントが Mobile Link サーバに接続するための 1 つ以上のネットワーク・プロトコルを指定します。クライアントが使用する同期プロトコルと一致するネットワーク・プロトコルを選択してください。このコマンド・ライン・オプションの構文は次のとおりです。

dbmlsrv9 -c "*connection-string***" -x** *protocol***(** *options* **)**

次の例では、TCP/IP プロトコルが選択されますが、プロトコル・オプションが指定されていません。

```
dbmlsrv9 -c "dsn=ASA 9.0 Sample" -x tcpip
```

プロトコルは、次の形式のオプションを使用して設定できます。

(*keyword***=***value***;…)**

次に例を示します。

```
dbmlsrv9 -c "dsn=ASA 9.0 Sample" -x tcpip(
    host=localhost;port=2439)
```

ネットワーク・プロトコルとプロトコル・オプションの詳細については、『Mobile Link 管理ガイド』> 「-x オプション」を参照してください。

Mobile Link ユーザ

Mobile Link システムでは、リモート・データベースごとにユニークな Mobile Link ユーザ名を 1 つ設定する必要があります。ユーザ名は、各 Mobile Link リモート・データベースをユニークに識別します。

統合データベースに置かれている ml_user Mobile Link システム・テーブルには、Mobile Link ユーザ名のリストが保管されています。各ユーザの同期のステータスは commit_state カラムまたは progress カラムに記録されています。同期が中断されたときは、この情報によって適切なリカバリが確保されます。

Mobile Link ユーザの詳細については、次の項目を参照してください。

- 『Mobile Link クライアント』> 「Mobile Link ユーザの概要」
- 『Mobile Link クライアント』> 「Mobile Link ユーザの作成」
- 『Mobile Link 管理ガイド』> 「ml_user」

第 2 章

Mobile Link ユーザの認証

この章の内容

Mobile Link ユーザは、同期のためにリモート・データベースをユニークに識別する名前です。

この章では、パスワード管理と認証のためのメカニズムなど、Mobile Link ユーザを管理する方法について説明します。

セキュリティ・オプションの詳細については、『Mobile Link 管理ガイド』> 「Mobile Link トランスポート・レイヤ・セキュリティ」を参照してください。

Mobile Link ユーザの概要

「**Mobile Link ユーザ**」とは、リモート・データベースに割り当てられる名前で、「**同期ユーザ**」とも呼ばれます。それぞれの Mobile Link ユーザ名は、同期システム内でユニークでなければなりません。

Mobile Link ユーザの名前とパスワードは、データベース・ユーザの名前とパスワードとは異なります。Mobile Link ユーザ名とパスワードは、リモート・データベースをユニークに識別するために使用されます。また、オプションでリモート・データベースから Mobile Link 同期サーバへの接続を認証します。

Mobile Link ユーザ名を使用して、同期サーバの動作を制御することもできます。そのためには、同期スクリプト内でユーザ名を使用します。たとえば、複数のリモート・データベースに対して、そのユーザ名ごとに異なるローを送信できます。

Mobile Link ユーザ名は、統合データベースの ml_user Mobile Link システム・テーブルに格納されます。

命名規則

大規模な配備の場合、Mobile Link ユーザ名に命名規則を採用すると便利です。たとえば、Mobile Link 名が *user:application* として作成される命名規則を採用できます。ここで、*user* は人を識別し、*application* は `EmpA01:HRApp` などの特定のアプリケーションを識別します。または、*user:application:device* という規則を採用できます。ここで *device* はリモート・デバイスを識別します。

Ultra Light ユーザ認証

Ultra Light と Mobile Link のユーザ認証スキームは異なりますが、Ultra Light ユーザ ID の値を Mobile Link ユーザ名と共有にして簡素化できます。このように簡素化できるのは、Ultra Light アプリケーションを単一ユーザが使用している場合のみです。

Ultra Light ユーザ認証の詳細については、『Ultra Light データベース・ユーザーズ・ガイド』> 「Ultra Light のユーザ認証」を参照してください。

Mobile Link ユーザの作成

警告
Mobile Link ユーザ名によって、リモート・データベースがユニークに識別されます。したがって、リモート・データベースが 2 つ存在する場合、それぞれ異なる Mobile Link ユーザ名が必要です。

リモート・データベースのユーザの作成

リモート・データベース側にユーザを追加する場合、次のオプションがあります。

- Adaptive Server Anywhere リモートの場合は、Sybase Central または CREATE SYNCHRONIZATION USER 文を使用します。

  詳細については、「リモート・データベースへの Mobile Link ユーザの追加」92 ページを参照してください。

- Ultra Light リモートの場合は、ul_synch_info 構造体の user_name フィールドを使用できます。Java では、同期する前に ULSynchInfo クラスの SetUserName() メソッドを使用します。

  詳細については、「User Name 同期パラメータ」430 ページと「Password 同期パラメータ」410 ページを参照してください。

統合データベースへの Mobile Link ユーザ名の追加

リモート・データベースでユーザ名が作成された後、次の方法のいずれかを使用して、統合データベースにユーザ名を登録できます。

- dbmluser ユーティリティを使用する。

  詳細については、『Mobile Link 管理ガイド』> 「Mobile Link ユーザ認証ユーティリティ」を参照してください。

- Sybase Central を使用する。

- authenticate_user イベント用または authenticate_user_hashed イベント用のスクリプトを実装する。これらのスクリプトのどちらかを起動すると、Mobile Link 同期サーバによって、認証が正常に行われるユーザが自動的に追加されます。

- dbmlsrv9 で -zu+ コマンド・ライン・オプションを指定する。この場合、最初に同期するときに、統合データベースに追加されていない既存の Mobile Link ユーザが追加されます。

  詳細については、『Mobile Link 管理ガイド』> 「-zu オプション」を参照してください。

Mobile Link ユーザ名の共有

Mobile Link ユーザ名は、同期システム内でユニークでなければなりません。複数のリモート・データベース (Ultra Light または Adaptive Server Anywhere) で同じ Mobile Link ユーザ名を共有する場合は、ベース名にユニークなサフィックスの付いた Mobile Link ユーザ名を作成できます。

通常、この方法を使用するのは複数のリモート・データベースを使用するユーザです。各リモート・データベースには、ユニークな Mobile Link ユーザ名がなければなりませんが、同じベース名を共有することが可能です。

例

次の例では、102 の後にコロンとユニバーサル・ユニーク ID が続く Mobile Link ユーザ名が作成されます。

```
BEGIN
 EXECUTE IMMEDIATE 'CREATE SYNCHRONIZATION USER "102' +
':' +
 NEWID() + '"';
 END;
```

この場合、102:b23fdbed-bead-418a-9d53-917e774c2f4f などの Mobile Link ユーザ名が作成されます。

Mobile Link によって、各 Mobile Link スクリプトにユーザ名が指定されるようにする必要もあります。こうするには、modify_user という Mobile Link イベントを使用します。このイベントは入力として Mobile Link ユーザを取ります。また、Mobile Link ユーザを修正することもできます。修正された値は、すべてのダウンロード・イベントに渡されます。次に例を示します。

```
CALL sp_ML_modify_user( ? )
```

この結果、次の download_cursor は 102:b23fdbed-bead-418a-9d53-917e774c2f4f ではなく、102 の値に基づいています。

```
Select emp_id, emp_name
 From ULEmployee
 Where last_modified >= ?
  And emp_id = ?
```

次に、Adaptive Server Anywhere の構文を使用して記述されたプロシージャを示します。これは、その他の RDBMS に対応するように簡単に変換できます。

```
CREATE PROCEDURE sp_ML_modify_user( INOUT @ml_user_name
VARCHAR(255) )
 BEGIN
     DECLARE @colon_at INT;
     SET @colon_at = LOCATE( @ml_user_name, ':' );
     IF( @colon_at > 0 ) THEN
         -- Message statements are displayed in the
minimized engine
         -- window, this is useful for debugging



         MESSAGE 'UUID: ' +
                 RIGHT( @ml_user_name,
                   (LENGTH(@ml_user_name)-@colon_at) );
         SET @ml_user_name = LEFT( @ml_user_name,
(@colon_at-1) );
         MESSAGE 'New MobiLink User: ' + @ml_user_name;
     ELSE
         MESSAGE 'No change to MobiLink User: ' +
@ml_user_name;
     END IF;
 END;
```

ユーザ認証メカニズムの選択

ユーザの認証は、データを保護するためのセキュリティ・システムの一部です。

Mobile Link では、ユーザ認証メカニズムを選択することができます。インストール環境全体にわたって 1 つのメカニズムを使用する必要はありません。Mobile Link には、インストール環境内の各スクリプト・バージョンが異なる認証メカニズムを使用できるという柔軟性があります。

- **Mobile Link ユーザ認証なし**　パスワード保護が必要でないデータの場合は、インストール環境でユーザ認証を使用しないように選択できます。この場合、Mobile Link ユーザ名は ml_user テーブルに含まれていなければなりませんが、hashed_password カラムは NULL です。

- **組み込みの Mobile Link ユーザ認証**　Mobile Link では、ml_user Mobile Link システム・テーブルに格納されているユーザ名とパスワードを使用して、認証が実行されます。

  組み込みのメカニズムについては、次の項で説明します。

- **カスタム認証**　Mobile Link スクリプトの authenticate_user を使用して、組み込みの Mobile Link ユーザ認証システムを、独自の別のシステムに置き換えることができます。たとえば、使用する統合データベースの管理システムによって、Mobile Link システムの代わりにデータベースのユーザ認証を使用することができます。

  カスタム・ユーザ認証メカニズムの詳細については、「カスタム・ユーザ認証」26 ページを参照してください。

Mobile Link と関連製品の他のセキュリティ関連機能については、次を参照してください。

- 『Mobile Link 管理ガイド』> 「Mobile Link トランスポート・レイヤ・セキュリティ」

- 『Ultra Light データベース・ユーザーズ・ガイド』> 「Ultra Light データベースの暗号化」

- 『SQL Anywhere Studio セキュリティ・ガイド』> 「安全なデータの管理」

ユーザ認証アーキテクチャ

Mobile Link のユーザ認証システムは、ユーザ名とパスワードに依存します。組み込みのメカニズムを使用して、Mobile Link 同期サーバに対してユーザ名とパスワードを認証させることも、独自のカスタム・ユーザ認証メカニズムを実装することもできます。

組み込みの認証システムでは、ユーザ名とパスワードの両方が、統合データベース内の ml_user Mobile Link システム・テーブルに格納されます。パスワードは、ハッシュされた状態で格納されます。これは、Mobile Link 同期サーバ以外のアプリケーションからは、ml_user テーブルを読み込んでオリジナルのフォームのパスワードを再構成できないようにするためです。統合データベースにユーザ名とパスワードを追加するには、Sybase Central または dbmluser ユーティリティを使用するか、Mobile Link 同期サーバの起動時に -zu+ オプションを指定します。

詳細については、「Mobile Link ユーザの作成」11 ページを参照してください。

Mobile Link クライアントは、Mobile Link 同期サーバに接続するときに、次の値を提供します。

- **ユーザ名**　Mobile Link ユーザ名。必須です。同期を実行するには、ユーザ名を ml_user システム・テーブルに格納する必要があります。または、Mobile Link 同期サーバを -zu+ オプションを使用して起動し、ml_user テーブルに新しいユーザを追加する必要があります。
- **パスワード**　Mobile Link パスワード。この値は、ユーザが不明な場合、または ml_user Mobile Link システム・テーブル内の対応するパスワードが NULL の場合にのみオプションになります。
- **新しいパスワード**　新しい Mobile Link パスワード。この値はオプションであり、Mobile Link ユーザはこの値を設定することでパスワードを変更できます。

カスタム認証

オプションで、ユーザ認証メカニズムを独自のものに置き換えることができます。

詳細については、「カスタム・ユーザ認証」26 ページを参照してください。

認証処理

次に、認証中に発生するイベントの順序を説明します。

1. Mobile Link ユーザ名を使用し、オプションでパスワードと新しいパスワードを使用して、同期要求を開始します。Mobile Link 同期サーバは新しいトランザクションを開始し、begin_connection イベントをトリガします。

2. Mobile Link は、指定した Mobile Link ユーザ名が現在同期を実行中でないことを検証します。

3. authenticate_user スクリプトを定義している場合は、次のイベントが発生します。

   a. authenticate_user スクリプトが呼び出されます。このスクリプトは、指定された Mobile Link ユーザ名で auth_status フィールドを設定し、オプションでパスワードと新しいパスワードを設定します。

   b. authenticate_user スクリプトで例外が発生した場合、同期処理は停止します。

   c. authenticate_user スクリプトが SQL 文を返した場合、文が実行されます。SQL 文は、2 つから 4 つまでの引数をとるストアド・プロシージャへの呼び出しでなければなりません。手順 3.a で設定した auth_status 値が最初のパラメータとして渡され、このストアド・プロシージャによって更新されます。

4. エラーが発生した場合、認証はエラーで停止します。

5. 正常に実行された場合、authenticate_user_hashed スクリプトが存在すれば、次のイベントが発生します。

   a. パスワードが指定されている場合、ハッシュされた値が計算されます。新しいパスワードが指定されている場合、ハッシュされた値が計算されます。

b. authenticate_user_hashed スクリプトが、auth_status の現在値とハッシュされたパスワードで呼び出されます。authenticate_user_hashed スクリプトが手順 3.a で設定されている場合は、auth_status の値をより大きな値 (より重大度が高い値) で上書きできることを除き、動作は手順 3.b および 3.c と同じです。authenticate_user スクリプトが定義されていないときは、どのような場合でも authenticate_user_hashed によって設定された auth_status 値が使用されます。

6. Mobile Link 同期サーバは、指定された Mobile Link ユーザ名を ml_user テーブルで探します。

   a. カスタム・スクリプト authenticate_user または authenticate_user_hashed のいずれかが呼び出されますが、指定された Mobile Link ユーザ名が ml_user テーブルになく、auth_status が有効な場合 (1000 または 2000) は、Mobile Link ユーザ名が Mobile Link システム・テーブルの ml_user に追加されます。auth_status が有効でない場合、ml_user は更新されず、エラーが発生します。

   b. カスタム・スクリプトが呼び出されず、指定された Mobile Link ユーザ名も ml_user テーブルにない場合は、Mobile Link 同期サーバを -zu オプションで起動していれば、指定した Mobile Link ユーザ名が ml_user に追加されます。それ以外の場合は、エラーが発生し、それに応じて auth_status が設定されます。

   c. カスタム・スクリプトが呼び出され、指定された Mobile Link ユーザ名が ml_user テーブルに存在する場合は、何も実行されません。

   d. カスタム・スクリプトが呼び出されず、指定された Mobile Link ユーザ名が ml_user テーブルに存在する場合、ml_user テーブルにある値に対してパスワードがチェックされ、それに応じて auth_status が設定されます。

7. Mobile Link 同期サーバは、指定された Mobile Link ユーザ名を、指定されたパスワードを使用して認証します。

8. authenticate_user と authenticate_user_hashed のいずれのスクリプトも呼び出されなかった場合に新しいパスワードを指定すると、パスワードが指定したものに変更されます。

9. authenticate_parameters スクリプトを定義しており、auth_status が有効である場合 (1000 または 2000)、次のイベントが発生します。

   a. パラメータが authenticate_parameters スクリプトに渡されます。

   b. authenticate_parameters スクリプトが現在の auth_status より大きい auth_status 値を返す場合、新しい auth_status は古い値を上書きします。

10. auth_status が有効でない場合、同期はアボートされます。

11. modify_user スクリプトを定義している場合はそのスクリプトが呼び出され、指定していた Mobile Link ユーザ名が、新しい Mobile Link ユーザ名に置き換えられます。

12. Mobile Link 同期サーバは必要に応じて同期をコミットまたはロールバックし、同期を継続します。

ユーザの最初のパスワードを設定する

各ユーザのパスワードは、ユーザ名とともに ml_user テーブルに格納されます。最初のパスワードは、Sybase Central から指定するか、または dbmluser コマンド・ライン・ユーティリティを使用して指定できます。

Sybase Central は、個々のユーザとパスワードを追加する場合に便利です。dbmluser ユーティリティは、バッチで追加する場合に便利です。

パスワードを指定しないでユーザを作成すると、Mobile Link はそのユーザに対してユーザ認証を実行しません。つまり、そのユーザは、パスワードを入力せずに接続や同期ができます。

❖ 1 ユーザに最初の Mobile Link パスワードを設定するには、次の手順に従います (Sybase Central の場合)。

1. Sybase Central から、Mobile Link プラグインを使用して統合データベースに接続します。
2. [ユーザ] フォルダを開きます。
3. [ユーザの追加] をダブルクリックします。[ユーザ追加] ウィザードが表示されます。
4. ユーザ名と任意のパスワードを入力します。
5. [完了] をクリックして、タスクを完了します。

❖ 最初の Mobile Link パスワードを設定するには、次の手順に従います (コマンド・ラインの場合)。

1. 各行に 1 人分ずつ、ユーザ名とパスワードを空白スペースで区切って入力したファイルを作成します。
2. コマンド・プロンプトを開き、dbmluser コマンド・ライン・ユーティリティを実行します。次に例を示します。

```
dbmluser -c "dsn=my_dsn" -f password-file
```

このコマンド・ラインでは、-c オプションで、統合データベースへの ODBC 接続を指定します。-f オプションで、ユーザ名とパスワードを含むファイルを指定します。

dbmluser の詳細については、『Mobile Link 管理ガイド』> 「Mobile Link ユーザ認証ユーティリティ」を参照してください。

新しいユーザからの同期

通常、Mobile Link 同期サーバに接続するには、各 Mobile Link クライアントが有効な Mobile Link ユーザ名とパスワードを指定します。

起動時に -zu+ オプションを設定すると、Mobile Link 同期サーバは次の規則に従って新しいユーザ名を ml_user テーブルに自動的に追加できます。

このオプションを設定すると、新しいユーザは独自の Mobile Link アカウントの作成を許可されるため、新しいユーザの管理が簡単になります。この設定は、サーバとクライアントがすべてファイアウォール内で動作している場合に便利です。

Mobile Link クライアントが、現在の ml_user テーブルにないユーザ名で同期を実行すると、Mobile Link はデフォルトで次のアクションをとります。

- **パスワードを持たない新しいユーザ**　ユーザがパスワードを入力しない場合、デフォルトでは、ml_user テーブルに NULL パスワードでユーザ名が追加されます。この動作は、ユーザ認証機能がない以前のリリースの Mobile Link との互換性を提供します。

  詳細については、『Mobile Link 管理ガイド』> 「-zu オプション」を参照してください。

- **パスワードを持つ新しいユーザ**　ユーザがパスワードを入力すると、ユーザ名とパスワードの両方が ml_user テーブルに追加され、Mobile Link システム内で新しいユーザ名が認識されるようになります。

- **新しいパスワードを持つ新しいユーザ**　新しいユーザは、[パスワード] フィールドの代わりに [新しいパスワード] フィールドに情報を入力するか、両方のフィールドに入力します。いずれの場合も、新しいパスワード設定が古いパスワードに上書きされ、新しいパスワードを使用して、新しいユーザが Mobile Link システムに追加されます。

未知のユーザによる同期の防止

-zu オプションを使用して Mobile Link 同期サーバを起動すると、デフォルトの動作を変更できます。この場合、Mobile Link 同期サーバは、ml_user テーブルにないユーザ名からの同期を拒否します。

この設定には 2 つの利点があります。第 1 に、Mobile Link 同期サーバへの不正なアクセスによるリスクが減少します。第 2 に、すでに登録されているユーザが間違ったユーザ名やスペルミスのあるユーザ名で誤って接続するのを防ぐことができます。Mobile Link システムに予期しない動作が発生する危険性があるため、誤って接続するような事態は避けてください。

エンド・ユーザに対するパスワード入力の要求

Mobile Link 同期サーバでユーザ認証を無効にするように選択しないかぎり、Mobile Link クライアントから同期を行うすべてのエンド・ユーザは、毎回 Mobile Link ユーザ名とパスワードを入力しなければなりません。

❖ エンド・ユーザに Mobile Link パスワードの入力を要求するには、次の手順に従います。

- ユーザ名とパスワードを入力するメカニズムは、Ultra Light クライアントと Adaptive Server Anywhere クライアントで異なります。

  - **Ultra Light**　Ultra Light クライアントでは同期時に、同期構造体 (C/C++) またはオブジェクト (Java) の password フィールドに有効な値を指定します。組み込みの Mobile Link 同期の場合、有効なパスワードとは、ml_user Mobile Link システム・テーブルにある値と一致するものです。

    アプリケーションでは、エンド・ユーザに対して Mobile Link ユーザ名とパスワードの入力を要求してから、同期を行ってください。

    詳細については、「Ultra Light 同期パラメータ」397 ページを参照してください。

  - **Adaptive Server Anywhere**　dbmlsync コマンド・ラインで有効なパスワードを入力できます。ただし、コマンド・ラインに入力しないと、dbmlsync 接続ダイアログに入力するように要求されます。コマンド・ラインは、同じコンピュータで実行中の他のプロセスから参照できるため、ダイアログで入力する方が安全です。

    認証が失敗すると、ユーザ名とパスワードを再入力するように要求されます。

    詳細については、「-c オプション」132 ページを参照してください。

パスワードの変更

Mobile Link では、エンド・ユーザが自身のパスワードを変更するメカニズムが用意されています。インタフェースは、Ultra Light クライアントと Adaptive Server Anywhere クライアントで異なります。

❖ エンド・ユーザに Mobile Link パスワードを入力させるには、次の手順に従います。

- ユーザ名とパスワードを入力するメカニズムは、Ultra Light クライアントと Adaptive Server Anywhere クライアントで異なります。
  - **Ultra Light**　アプリケーションでは、同期構造体の password フィールドに既存のパスワードを、new_password フィールドに新しいパスワードを同期時に指定します。

    詳細については、「Password 同期パラメータ」410 ページと 「New Password 同期パラメータ」406 ページを参照してください。
  - **Adaptive Server Anywhere**　dbmlsync コマンド・ラインまたは dbmlsync 接続ダイアログ (コマンド・ライン・パラメータを指定しない場合) で、有効な既存のパスワードと新しいパスワードを入力します。

    詳細については、「-mp オプション」182 ページと 「-mn オプション」182 ページを参照してください。

新しいパスワードは、次の同期まで確認されません。dbmlsync ユーティリティの場合や、同期時に Ultra Light アプリケーションで要求した場合は、新しいパスワードの確認は即座に行われます。

最初のパスワードは、統合サーバで、または最初の同期で設定できます。詳細については、「ユーザの最初のパスワードを設定する」20 ページと「新しいユーザからの同期」22 ページを参照してください。

パスワードをいったん割り当てると、クライアント側でパスワードを NULL にリセットすることはできません。

カスタム・ユーザ認証

組み込みの Mobile Link メカニズム以外のユーザ認証メカニズムを使用するように選択することもできます。カスタム・ユーザ認証メカニズムを使用する理由として、既存の DBMS ユーザ認証スキームまたは外部認証メカニズムとの統合があり、組み込みの Mobile Link メカニズムにはないパスワードの最小長や有効期限といったカスタム機能の提供もあげられます。

次の 3 つのカスタム認証ツールがあります。

- dbmlsrv9 -zu オプション
- authenticate_user スクリプト
- authenticate_parameters スクリプト

dbmlsrv9 -zu オプションを使用すると、ユーザの自動追加処理を制御できます。たとえば、-zu+ オプションを指定すると、認識されなかった Mobile Link ユーザ名が最初の同期時に ml_user テーブルに追加されます。-zu オプションを必要とするのは、組み込みの Mobile Link 認証だけです。

authenticate_user スクリプトと authenticate_parameters スクリプトは、いずれもデフォルトの Mobile Link ユーザ認証メカニズムを無効にします。正常に認証が行われるユーザは、自動的に ml_user テーブルに追加されます。

Mobile Link は、authenticate_user イベント用に事前に定義されたスクリプトをいくつかインストールします。これらのスクリプトは、LDAP、POP3、IMAP サーバを使用した認証を簡単に行えるようにします。詳細については、「外部サーバに対する認証」28 ページを参照してください。

ユーザ ID とパスワードのカスタム認証を作成するには、authenticate_user を使用します。このスクリプトがあると、組み込みのパスワード比較の代わりにそのメカニズムが実行されます。このスクリプトでは、認証の成功または失敗を示すエラー・コードを返さなければなりません。

ユーザ ID とパスワード以外の値によるカスタム認証を作成するには、authenticate_parameters を使用します。

詳細については、次の項を参照してください。

- 『Mobile Link 管理ガイド』> 「-zu オプション」
- 『Mobile Link 管理ガイド』> 「authenticate_user 接続イベント」
- 『Mobile Link 管理ガイド』> 「authenticate_parameters 接続イベント」

Java と .NET のユーザ認証

Java クラスと .NET クラスではアプリケーション・サーバなどのコンピューティング環境で使用されるユーザ名とパスワードの他のソースにアクセスできるため、ユーザ認証は Java と .NET の同期論理で本来使用されているものです。

簡単な例は、ディレクトリ *Samples¥MobiLink¥JavaAuthentication* にあります。*Samples¥MobiLink¥JavaAuthentication¥CustEmpScripts.java* に格納されているサンプル・コードは、単純なユーザ認証システムを実装します。最初の同期時に、Mobile Link ユーザ名が login_added テーブルに追加されます。それ以降の同期時には、login_audit テーブルにローが 1 つ追加されます。このサンプルでは、login_added テーブルにユーザ ID を追加する前のテストは行いません。

ユーザ認証を説明する .NET サンプルについては、『Mobile Link 管理ガイド』> 「.NET 同期のサンプル」を参照してください。

SQL ユーザ認証

一般的な authenticate_user SQL スクリプトは、パラメータ auth_status、ml_username、user_password、user_new_password を使用するストアド・プロシージャの呼び出しです。呼び出しでのパラメータの順序

は、これと一致する必要があります。たとえば、Adaptive Server Anywhere 統合データベースでは、フォーマットは次のようになります。

```
call my_authentication( ?, ?, ?, ? )
```

ここで、最初の引数は認証コードです。認証コードは integer 型で、その他のパラメータは VARCHAR(128) です。

Transact-SQL のフォーマットは次のとおりです。

```
execute ? = my_authentication( ?, ?, ? )
```

ここで、認証コードは左側のパラメータです。

詳細については、『Mobile Link 管理ガイド』> 「authenticate_user 接続イベント」を参照してください。

外部サーバに対する認証

Mobile Link には、authenticate_user イベントを使用して外部サーバに対する認証を簡単に行えるようにする、事前に定義された Java 同期スクリプトが用意されています。現在、事前に定義されたスクリプトは、次の認証サーバで使用できます。

- JavaMail 1.2 API を使用している POP3 または IMAP サーバ
- Java Naming および Directory Interface (JNDI) を使用している LDAP サーバ

これらのスクリプトをどのように使用するかは、Mobile Link ユーザ名を外部認証システムのユーザ ID に直接マッピングしているかどうかによって決まります。

Mobile Link ユーザ名をユーザ ID に直接マッピングしている場合

Mobile Link ユーザ名を認証システムの有効なユーザ ID に直接マッピングしている単純なケースでは、authenticate_user 接続イベントに対する応答で、定義済みのスクリプトを直接使用できます。認証コードは、ml_property テーブルに格納されているプロパティに基づいてそのコード自体を初期化します。

❖ 事前に定義されたスクリプトを authenticate_user で直接使用するには、次の手順に従います。

1. 事前に定義された Java 同期スクリプトを Mobile Link の ml_scripts システム・テーブルに追加します。追加するには、ストアド・プロシージャを使用するか、Sybase Central を使用します。

   - ml_add_java_connection_script ストアド・プロシージャを使用するには、次のコマンド・プロンプトを入力します。

     ```
     call ml_add_java_connection_script(
        'MyVersion'
        'authenticate_user'

     'ianywhere.ml.authentication.ServerType.authenti
     cate' )
     ```

     ここで、*MyVersion* はスクリプト・バージョンの名前で、*ServerType* は **LDAP**、**POP3**、または **IMAP** です。

   - Sybase Central の [接続スクリプトを追加] ウィザードを使用するには、スクリプト・タイプとして authenticate_user を選択し、コード・エディタで次のコードを入力します。

     ```
     ianywhere.ml.authentication.ServerType.authentic
     ate
     ```

     ここで、*ServerType* は **LDAP**、**POP3**、または **IMAP** です。

2. この認証サーバのプロパティを追加します。

   設定が必要な各プロパティに対し、ml_add_property ストアド・プロシージャを使用します。

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

ここで、*MyVersion* はスクリプト・バージョンの名前で、*property_name* は認証サーバによって決まります。また、*property_value* は使用しているアプリケーションに適切な値です。この呼び出しを、設定の必要な各プロパティに対して繰り返し行います。

詳細については、「外部認証識別符号プロパティ」31 ページを参照してください。

Mobile Link ユーザ名をユーザ ID に直接マッピングしていない場合

Mobile Link ユーザ名がユーザ ID と同じでない場合は、コードを間接的に呼び出す必要があり、ユーザ ID を ml_user 値から抽出するか、マッピングしなければなりません。これを行うには、Java クラスを作成します。

Java クラス作成の詳細については、『Mobile Link 管理ガイド』>「Java による同期スクリプトの作成」を参照してください。

次に、簡単な例を示します。extractUserID メソッド内のコードは、ml_user 値をユーザ ID にマッピングする方法によって異なるため、この例では省きます。すべての作業は、認証クラスの "authenticate" メソッドで行われます。

```
package com.mycompany.mycode;

 import ianywhere.ml.authentication.*;
 import ianywhere.ml.script.*;

 public class MLEvents
 {
     private DBConnectionContext _context;
     private POP3 _pop3;

     public MLEvents( DBConnectionContext context )
     {
         _context = context;
         _pop3 = new POP3( context );
     }



     public void authenticateUser(
       InOutInteger status,
       String userID,
       String password,
```

```
        String newPassword )
      {
          String realUserID = extractUserID( userID );
          _pop3.authenticate( status, realUserID,
 password, newPassword );
      }

      private String extractUserID( String userID )
      {
          // code here to map ml_user to a "real" POP3
 user
      }
  }
```

この例では、初期化プロパティを検出できるようにするために、POP3 オブジェクトを DBConnectContext オブジェクトで初期化する必要があります。この方法でオブジェクトを初期化しない場合は、コードにプロパティを設定する必要があります。次に例を示します。

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

これはどのような認証クラスにも適用されますが、プロパティはクラスによって異なります。

外部認証識別符号プロパティ

Mobile Link では、特に LDAP の場合において、可能な限り適切なデフォルトを用意しています。設定できるプロパティはさまざまですが、次に基本的なプロパティを説明します。

POP3 認証識別符号

mail.pop3.host	サーバのホスト名
mail.pop3.port	ポート番号 (デフォルトの 110 を使用する場合は省略可能)

詳細については、http://java.sun.com/products/javamail/javadocs/com/sun/mail/pop3/package-summary.html を参照してください。

IMAP 認証識別符号

mail.imap.host	サーバのホスト名
mail.imap.port	ポート番号 (デフォルトの 143 を使用する場合は省略可能)

詳細については、http://java.sun.com/products/javamail/javadocs/com/sun/mail/imap/package-summary.html を参照してください。

LDAP 認証識別符号

java.naming.provider.url	`ldap://ops-yourLocation/dn=sybase,dn=com` などの LDAP サーバの URL

詳細については、JNDI のマニュアルを参照してください。

第 3 章

ユーティリティ

この章の内容

この章では、Mobile Link のクライアント・ユーティリティについて説明します。

Mobile Link 同期サーバ・ユーティリティについては、『Mobile Link 管理ガイド』> 「ユーティリティ」を参照してください。

他の Adaptive Server Anywhere ユーティリティについては、『ASA データベース管理ガイド』> 「データベース管理ユーティリティ」を参照してください。

ActiveSync プロバイダ・インストール・ユーティリティ

ActiveSync 用 Mobile Link プロバイダをインストールするか、Windows CE デバイス上で Ultra Light アプリケーションを登録してインストールします。

構文

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

オプション	説明
-d	作成時にアプリケーションを無効にする
-k *path*	デスクトップ・プロバイダ *dbasdesk.dll* のロケーションを指定する
-l *filename*	指定したファイルにアクティビティ・ログを書き込む。filename のパスまたは相対パスが指定されていない場合、ログ・ファイルは ActiveSync のインストール・ディレクトリまたはこのディレクトリを基準とした相対ディレクトリに配置される。
-n	アプリケーションを登録するが、デバイスにはコピーしない
-u	Mobile Link ActiveSync プロバイダをアンインストールする
-v *path*	デバイス・プロバイダ *dbasdev.dll* のロケーションを指定する

引数	説明
src	アプリケーションのソース・ファイル名とパス
dst	アプリケーションのインストール先ファイル名とパス
name	アプリケーション名
class	アプリケーションの登録済み Windows クラス名
args	ActiveSync でアプリケーションの起動時に使用するコマンド・ライン引数

説明

このユーティリティによって、ActiveSync 用の Mobile Link プロバイダがインストールされます。プロバイダには、デスクトップ上で実行されるコンポーネント (*dbasdesk.dll*) と、Windows CE デバイスに展開されるコンポーネント (*dbasdev.dll*) の両方が含まれています。dbasinst ユーティリティは、デスクトップ・プロバイダの現在のロケーションを示すレジストリ・エントリを作成し、デバイス・プロバイダをデバイスにコピーします。

また、*dbasinst* ユーティリティに追加の引数を指定すると、Ultra Light アプリケーションを Windows CE デバイスに登録してインストールできます。さらに、ActiveSync ソフトウェアを使用して Ultra Light アプリケーションの登録とインストールを行う方法もあります。

ライセンス要件に応じて、このアプリケーションをデスクトップ・コンポーネントやデバイス・コンポーネントとともにエンド・ユーザに提供できる場合があります。この場合、エンド・ユーザは、ActiveSync で使用できるようにアプリケーションのコピーを作成できます。

ActiveSync プロバイダをインストールするには、リモート・デバイスに接続してください。

ActiveSync プロバイダ・インストール・ユーティリティの使用方法の詳細は、次の項を参照してください。

- Adaptive Server Anywhere：「ActiveSync 用 Mobile Link プロバイダのインストール」110 ページ
- Ultra Light：「ActiveSync 用 Mobile Link プロバイダのインストール」391 ページ

オプション

-d　デフォルトでは、dbasinst によって登録されたアプリケーションは有効になります。つまり、ActiveSync が同期を開始するときに自動的に同期されます。-d オプションを指定すると、アプリケーションは登録されますが、[ActiveSync Mobile Link の設定] ダイアログでは有効になっていません。

-l　ActiveSync プロバイダが、指定されたファイルにそのアクティビティのログを取ります。filename のパスまたは相対パスが指定されていない場合、ログ・ファイルは ActiveSync のインストール・ディレクトリまたはこのディレクトリを基準とした相対ディレクトリに配置されます。

-k デスクトップ・プロバイダ *dbasdesk.dll* へのパス。デフォルトでは、このファイルは SQL Anywhere ディレクトリの *win32* サブディレクトリ内で検索されます。エンド・ユーザ (通常は SQL Anywhere 全体がインストールされていないユーザ) は、Mobile Link ActiveSync プロバイダのインストール時に、-k の指定が必要になる場合があります。

-n Mobile Link ActiveSync プロバイダをインストールするだけでなく、アプリケーションの登録も行いますが、デバイスにはコピーしません。アプリケーションに複数のファイルがある場合 (静的ライブラリではなく Ultra Light ランタイム・ライブラリ DLL を使用するようにコンパイルされている場合など)、または他の方法でアプリケーションをデバイスにコピーする場合は、このスイッチを指定します。

-u Mobile Link ActiveSync プロバイダ用に登録されたアプリケーションの登録をすべて解除し、Mobile Link ActiveSync プロバイダをアンインストールします。この操作によってデスクトップ・マシンやデバイスからファイルが削除されることはありません。デバイスがデスクトップに接続されていない場合は、エラーが返されます。

-v デスクトップ・プロバイダ *dbasdev.dll* へのパス。デフォルトでは、このファイルは SQL Anywhere ディレクトリの *CE* サブディレクトリにある、プラットフォーム固有のディレクトリ内で検索されます。エンド・ユーザ (通常は SQL Anywhere 全体がインストールされていないユーザ) は、Mobile Link ActiveSync プロバイダのインストール時に、-v の指定が必要になる場合があります。

引数

src アプリケーションをデバイスにコピーするためのソース・ファイル名とパス。このパラメータを指定するのは、アプリケーションを登録してデバイスにコピーする場合のみです。-n オプションを使用する場合は、このパラメータを指定しないでください。

dst デバイス上でアプリケーションに使用するコピー先ファイル名とパス。

name アプリケーション名。これは、ActiveSync がアプリケーションを参照するときに使用する名前です。

class アプリケーションの登録済み Windows クラス名。

args　ActiveSync でアプリケーションの起動時に使用するコマンド・ライン引数。

例

次のコマンドは、デフォルト引数を使用して、ActiveSync 用の Mobile Link プロバイダをインストールします。アプリケーションの登録は行いません。正常にインストールするには、デバイスをデスクトップ・マシンに接続してください。

```
dbasinst
```

次のコマンドは、ActiveSync 用の Mobile Link プロバイダをアンインストールします。正常にアンインストールするには、デバイスをデスクトップ・マシンに接続してください。

```
dbasinst -u
```

次のコマンドは、ActiveSync 用の Mobile Link プロバイダがまだインストールされていない場合はインストールし、アプリケーション *myapp.exe* を登録します。また、*c:¥My Files¥myapp.exe* ファイルをデバイス上の *¥Program Files¥myapp.exe* にコピーします。-p -x 引数は、ActiveSync によって起動されるときの myapp.exe に対するコマンド・ライン・オプションです。このコマンドは、1 行に入力してください。

```
dbasinst "C:¥My Files¥myapp.exe" "¥Program
Files¥myapp.exe"
    "My Application" MYAPP -p -x
```

参照

- 「ActiveSync 同期の使用」108 ページ
- 「Ultra Light 同期パラメータ」397 ページ

Mobile Link クライアント・データベース抽出ユーティリティ (旧式)

他の Adaptive Server Anywhere データベースをテンプレートとして使用して、Adaptive Server Anywhere クライアント・データベースを作成します。このユーティリティは使用されなくなりました。クライアント・データベースを作成する別の方法については、「リモート・データベースの作成」76 ページを参照してください。

構文

mlxtract [*additional-options*] *directory site-name*

オプション	説明
-ac *"keyword=value; …"*	接続文字列で指定したデータベースに接続して、再ロードする
-al *filename*	この新しいデータベースのログ・ファイル名
-an *filename*	アンロードするデータベースと同じ設定でデータベース・ファイルを作成し、そのデータベース・ファイルを自動的に再ロードする
-c *"keyword=value; …"*	データベース接続パラメータを指定する
-id	スキーマ定義とデータを抽出する
-it	トリガを抽出する
-j *count*	ビュー作成文の対話回数
-l *level*	指定した独立性レベルですべての抽出オペレーションを実行する
-o *file*	メッセージをファイルに出力する
-p *character*	エスケープ文字
-q	クワイエット・モードで処理を実行し、メッセージ出力またはウィンドウ表示を行わない
-r *file*	生成された再ロード Interactive SQL コマンド・ファイルの名前を指定する (デフォルトは "*reload.sql*")

オプション	説明
-s7	Adaptive Server Anywhere バージョン 7 の構文を使用して同期定義を作成する
-u	データを順序付けしない
-v	冗長メッセージ
-x	外部テーブル・ロードを使用する
-xh	プロシージャ・フックを除外する
-xf	外部キーを除外する
-xp	ストアド・プロシージャを除外する
-xv	ビューを除外する
-y	確認メッセージを表示しないでコマンド・ファイルを上書きする
directory	ファイルが書き込まれたディレクトリ。`-an` または `-ac` を使用する場合、このオプションは不要。
site-name	生成するクライアント・データベースを指定する

説明

mlxtract は、Adaptive Server Anywhere クライアント・データベースの Mobile Link 抽出ユーティリティです。このユーティリティを Adaptive Server Anywhere リファレンス・データベースに対して実行すると、選択したオプションに応じて、新しいクライアント・データベースまたは Adaptive Server Anywhere クライアント・データベース用のコマンド・ファイルが作成されます。

コマンド・ライン抽出ユーティリティは、コマンド・ファイルと、一連の関連データ・ファイルを作成します。このコマンド・ファイルを新しく初期化された Adaptive Server Anywhere データベースに対して実行すると、データベース・オブジェクトを作成し、クライアント・データベース用のデータをロードすることができます。

デフォルトのコマンド・ファイル名は *reload.sql* です。

オプション

既存データベースへのデータの再ロード (-ac)　このオプションを使用すると、データベースの抽出処理と、既存データベースへの結果の再ロード処理を組み合わせることができます。

たとえば、次のコマンド (すべて 1 行に入力) は、field_user サブスクライバのデータのコピーを既存のデータベース・ファイル *newdemo.db* にロードします。

```
mlxtract -c "uid=DBA;pwd=SQL;dbf=asademo.db" -ac
"uid=DBA;pwd=SQL;dbf=newdemo.db" field_user
```

このオプションを使用した場合、データのコピーはディスク上に作成されないため、コマンド・ラインでアンロード用ディレクトリを指定する必要はありません。これによりデータのセキュリティは高まりますが、パフォーマンスは多少低下します。

新規データベースへのデータの再ロード (-an)　このオプションを使用すると、データベースの抽出処理、新規データベースの作成処理、データのロード処理を組み合わせることができます。

たとえば、次のコマンド (1 行で入力) は、新規のデータベース・ファイル *asacopy.db* を作成し、そこに *asademo.db* の field_user サブスクライバのスキーマとデータをコピーします。

```
mlxtract -c "uid=DBA;pwd=SQL;dbf=asademo.db" -an
asacopy.db field_user
```

このオプションを使用した場合、データのコピーはディスク上に作成されないため、コマンド・ラインでアンロード用ディレクトリを指定する必要はありません。これによりデータのセキュリティは高まりますが、パフォーマンスは多少低下します。

接続パラメータ (-c)　一連の接続パラメータの文字列です。

- **mlxtract 接続パラメータ**　ユーザはデータベースの全テーブル上にパーミッションを持っている必要があるため、user ID には DBA 権限を持つユーザ ID を指定してください。

  たとえば、次の文 (1 行で入力) は、ユーザ ID が DBA、パスワードが SQL で接続している sample_server サーバ上で実行している ASADemo データベースから、Mobile Link ユーザ名 joe_remote のデータベースを抽出します。データは *c:¥extract* ディレクトリへアンロードされます。

  ```
  mlxtract -c "eng=sample_server;dbn=sademo;
   uid=DBA;pwd=SQL" c:¥extract joe_remote
  ```

スキーマ定義とデータの抽出 (-id)　デフォルトでは、スキーマだけが抽出されます。このようなデータベースは、Mobile Link 同期サーバへの最初の接続時にデータを使って初期化できます。このオプションを使用すると、リファレンス・データベースから最初のデータ・セットを抽出するかどうかを選択できます。

トリガの抽出 (-it)　デフォルトでは、トリガは抽出されません。このオプションを使用すると、リファレンス・データベース内のトリガが抽出されます。

ビューの対話回数 (-j)　統合データベースにネストされたビューがある場合は、このオプションを使用してビューの抽出時の最大対話回数を指定します。

指定された独立性レベルでの抽出の実行 (-l)　デフォルト設定では、独立性レベルは 0 です。アクティブなサーバからデータベースを抽出する場合は、独立性レベル 3 で実行し、抽出されたデータベース内のデータがサーバ上のデータと一致するようにします。独立性レベルを大きくすると、抽出ユーティリティが多数のロックを使用することになり、他のユーザによるデータベースの使用が制限される可能性があります。

メッセージをファイルに出力 (-o)　抽出処理時のメッセージを、後で検討できるようにファイルに出力します。

エスケープ文字 (-p)　このオプションを使用して、デフォルトのエスケープ文字 (\) を別の文字に置き換えることができます。

クワイエット・モードで処理を実行 (-q)　エラー以外のメッセージを表示しません。

ファイル名の再ロード (-r)　再ロードしたコマンド・ファイルのデフォルト名は、現在のディレクトリの *reload.sql* です。このオプションを使用して異なるファイル名を指定できます。

ASA v7 構文を使用 (-s7)　このオプションは、Adaptive Server Anywhere バージョン 8 以降の統合データベースと Adaptive Server Anywhere バージョン 7 リモート・データベースの組み合わせで使用している場合に役立ちます。たとえば、バージョン 9 統合データベースを作成し、-s7 オプションを使用してリモート・データベースを抽出し、reload.sql ファイルをリモートに配備します。

データを順序付けせずに出力 (-u) デフォルトでは、各テーブルのデータはプライマリ・キーを基準に順序付けられます。-u オプションを使用するとアンロード処理は高速になりますが、クライアント・データベースへのデータのロード処理は遅くなります。

冗長モード (-v) アンロードされているテーブル名とアンロードされたロー数が表示されます。使用された SELECT 文も表示されます。

外部ロードを使用 (-x) 再ロード・スクリプトでは、データベースにデータをロードするとき、デフォルトで LOAD TABLE 文が使用されます。外部ロードの使用を選択すると、代わりに Interactive SQL の INPUT 文が使用されます。LOAD TABLE 文の方が INPUT 文よりも高速です。

データ・ファイルのパスは、INPUT 文ではクライアントからの相対パスを、LOAD TABLE 文ではサーバからの相対パスを使用します。

外部キー定義を除外 (-xf) クライアント・データベースに統合データベース・スキーマのサブセットがあり、いくつかの外部キー参照がない場合に、このオプションを使用できます。

ストアド・プロシージャを除外 (-xp) データベースからストアド・プロシージャを抽出しません。

ビューを除外 (-xv) データベースからビューを抽出しません。

確認メッセージを表示することなく処理を実行 (-y) このオプションを指定しないと、既存の再ロードしたコマンド・ファイルを置き換えるときに、確認メッセージが表示されます。

第 4 章

Mobile Link クライアントのネットワーク・プロトコル・オプション

この章の内容

この章では、リモート・データベースから Mobile Link 同期サーバに接続するために使用するネットワーク・プロトコル・オプションについて説明します。

プロトコル・オプション

この章では、Mobile Link クライアントを Mobile Link 同期サーバに接続する場合に使用できるネットワーク・プロトコル・オプションについて説明します。

これらのオプションを dbmlsync で使用する方法については、「CommunicationAddress (adr) 拡張オプション」138 ページを参照してください。

これらのオプションを Ultra Light で使用する方法については、次の項を参照してください。

- 「TCP/IP プロトコル・オプション」438 ページ
- 「HTTP プロトコル・オプション」439 ページ
- 「HTTPS プロトコル・オプション」441 ページ
- 「UlSecureRSASocketStream 同期パラメータ」443 ページ
- 「UlSecureSocketStream 同期パラメータ」444 ページ

Mobile Link 同期サーバの接続オプションを設定する方法については、『Mobile Link 管理ガイド』> 「-x オプション」を参照してください。

buffer_size

機能

内容が固定長であるメッセージの HTTP 本文の最大サイズ (バイト単位) を指定します。

構文

buffer_size=*number*

プロトコル

- dbmlsync - HTTP、HTTPS、HTTPS_FIPS
- Ultra Light - HTTP、HTTPS

デフォルト

- Ultra Light - 1024
- PocketPC における dbmlsync - 1024
- 他のデバイスにおける dbmlsync - 64000

説明

一般に、HTTP 本文のサイズが大きくなるほど、HTTP 要求応答のサイクルの数は減りますが、本文をバッファリングするために必要なメモリ量は増えます。

HTTPS_FIPS を使用できるのは Windows 上だけです。

dbmlsync を使用してネットワーク・プロトコル・オプションを設定する方法については、「CommunicationAddress (adr) 拡張オプション」138 ページを参照してください。

Ultra Light を使用してネットワーク・プロトコル・オプションを設定する方法については、「Ultra Light 同期クライアントのネットワーク・プロトコルのオプション」433 ページを参照してください。

certificate_company

機能

このオプションを指定した場合、証明書に記されている組織フィールドがこの値と一致する場合にだけ、アプリケーションはサーバ証明書を受け入れます。

別途ライセンスを取得できるオプションが必要
トランスポート・レイヤ・セキュリティを使用するには、別途ライセンスを取得できる SQL Anywhere Studio セキュリティ・オプションを入手する必要があります。このセキュリティ・オプションは、輸出規制対象品目です。

このコンポーネントのご注文については、『SQL Anywhere Studio の紹介』> 「別途ライセンスが入手可能なコンポーネント」を参照してください。

構文

certificate_company=*organization*

プロトコル

- dbmlsync - TCP/IP (セキュリティ・オプションを使用)、HTTPS、HTTPS_FIPS
- Ultra Light - TCP/IP (セキュリティ・パラメータを使用)、HTTPS、UlSecureRSASocketStream、UlSecureSocketStream

デフォルト

なし

説明

Mobile Link クライアントは認証局が署名した証明書をすべて信用するため、同じ認証局が他の会社用に発行した証明書も信用してしまうことがあります。識別方法がないままだと、クライアントは競争相手

の Mobile Link 同期サーバを自分の会社のものだと勘違いし、誤って機密性の高い情報を送信してしまう可能性があります。このオプションによって追加の検証が指定され、証明書の識別情報部分にある組織フィールドが、指定した特定の値と照合されます。

TCP/IP の場合は、このオプションをセキュリティ・オプションの一部として設定してください。詳細については、「security」64 ページを参照してください。

HTTPS_FIPS を使用できるのは Windows 上だけです。

dbmlsync を使用してネットワーク・プロトコル・オプションを設定する方法については、「CommunicationAddress (adr) 拡張オプション」138 ページを参照してください。

Ultra Light を使用してネットワーク・プロトコル・オプションを設定する方法については、「Ultra Light 同期クライアントのネットワーク・プロトコルのオプション」433 ページを参照してください。

参照

- 『Mobile Link 管理ガイド』> 「Mobile Link トランスポート・レイヤ・セキュリティ」
- 『Mobile Link 管理ガイド』> 「証明書フィールドの確認」
- 『Mobile Link 管理ガイド』> 「-x オプション」
- 「security」64 ページ
- 「trusted_certificates」68 ページ
- 「certificate_name」47 ページ
- 「certificate_unit」49 ページ

例

3 つの識別情報フィールドすべてを検査し、指定した値だけを受け入れるように Adaptive Server Anywhere クライアントに指示する例を示します。この例は 3 つのフィールドすべてを確認します。フィールドを 1 つまたは 2 つだけ確認するように選択することもできます。

たとえば、Adaptive Server Anywhere クライアントが存在する場合、証明書の検証をサブスクリプションで次のように設定できます。

```
CREATE SYNCHRONIZATION SUBSCRIPTION
 FOR 'user01'
 TO test_pub
 ADDRESS 'port=3333;security=ecc_tls(
  trusted_certificates=certicom.crt;
      certificate_company=Sybase, Inc.;
     certificate_unit=iAnywhere;certificate_name=sample
)'
```

Ultra Light クライアントの場合、正確な構文はアプリケーションの構築に使用したインタフェースによって異なります。次の C コードのフラグメントは、C または C++ で Embedded SQL を使用して Ultra Light アプリケーションを開発した場合、同じタスクを行います。

```
ul_synch_info info;
 info.stream = ULSocketStream();
 info.stream_parms = TEXT("port=9999");
 info.security_stream = ULSecureRSATLSStream();
 info.security_parms =
     UL_TEXT ( "certificate_company=Sybase, Inc." )
     UL_TEXT ( ";" )
     UL_TEXT ( "certificate_unit=iAnywhere" )
     UL_TEXT ( ";" )
     UL_TEXT ( "certificate_name=sample" );
 . . .
 ULSynchronize( &info );
```

certificate_name

機能

このオプションを指定した場合、証明書に記されている共通名フィールドがこの値と一致する場合にだけ、アプリケーションはサーバ証明書を受け入れます。

別途ライセンスを取得できるオプションが必要

トランスポート・レイヤ・セキュリティを使用するには、別途ライセンスを取得できる SQL Anywhere Studio セキュリティ・オプションを入手する必要があります。このセキュリティ・オプションは、輸出規制対象品目です。

このコンポーネントのご注文については、『SQL Anywhere Studio の紹介』> 「別途ライセンスが入手可能なコンポーネント」を参照してください。

構文

certificate_name=*common-name*

プロトコル

- dbmlsync - TCP/IP (セキュリティ・オプションを使用)、HTTPS、HTTPS_FIPS
- Ultra Light - TCP/IP (セキュリティ・パラメータを使用)、HTTPS、UlSecureRSASocketStream、UlSecureSocketStream

デフォルト

なし

説明

TCP/IP の場合は、このオプションをセキュリティ・パラメータの一部として設定してください。詳細については、「security」64 ページを参照してください。

HTTPS_FIPS を使用できるのは Windows 上だけです。

dbmlsync を使用してネットワーク・プロトコル・オプションを設定する方法については、「CommunicationAddress (adr) 拡張オプション」138 ページを参照してください。

Ultra Light を使用してネットワーク・プロトコル・オプションを設定する方法については、「Ultra Light 同期クライアントのネットワーク・プロトコルのオプション」433 ページを参照してください。

参照

- 『Mobile Link 管理ガイド』> 「Mobile Link トランスポート・レイヤ・セキュリティ」
- 『Mobile Link 管理ガイド』> 「証明書フィールドの確認」
- 『Mobile Link 管理ガイド』> 「-x オプション」
- 「security」64 ページ
- 「trusted_certificates」68 ページ
- 「certificate_company」45 ページ
- 「certificate_unit」49 ページ

例

HTTPS プロトコルの RSA 暗号化を設定する例を示します。これは、サーバとクライアント両方での設定が必要です。各コマンドは、1 行に入力する必要があります。

```
dbmlsrv9
    -c "dsn=asa90sample;uid=DBA;pwd=SQL"
    -x https(
      port=9999;
      certificate=c:¥asa90¥win32¥rsaserver.crt;
      certificate_password=test)
```

Adaptive Server Anywhere クライアントでは、実装は次のようになります。

```
dbmlsync
    -c "dsn=mydb;uid=DBA;pwd=SQL"
    -e "ctp=https;
        adr='port=9999;

trusted_certificates=c:¥asa90¥win32¥rsaroot.crt;
          certificate_name=RSA Server'"
```

Ultra Light クライアントでは、同等の実装は次のようになります。

```
    info.stream = ULHTTPSStream();
     info.stream_parms = TEXT(
       "port=9999;
        trusted_certificates=¥rsaroot.crt;
        certificate_name=RSA Server");
     info.security_stream = NULL;
     info.security_parms = NULL;
```

certificate_unit

機能

このオプションを指定した場合、証明書に記されている組織単位フィールドがこの値と一致する場合にだけ、アプリケーションはサーバ証明書を受け入れます。

別途ライセンスを取得できるオプションが必要
トランスポート・レイヤ・セキュリティを使用するには、別途ライセンスを取得できる SQL Anywhere Studio セキュリティ・オプションを入手する必要があります。このセキュリティ・オプションは、輸出規制対象品目です。

このコンポーネントのご注文については、『SQL Anywhere Studio の紹介』> 「別途ライセンスが入手可能なコンポーネント」を参照してください。

構文

certificate_unit=*organization-unit*

プロトコル

- dbmlsync - TCP/IP (セキュリティ・オプションを使用)、HTTPS、HTTPS_FIPS
- Ultra Light - TCP/IP (セキュリティ・パラメータを使用)、HTTPS、UlSecureRSASocketStream、UlSecureSocketStream

デフォルト

なし

説明

TCP/IP の場合は、このオプションをセキュリティ・オプションの一部として設定してください。詳細については、「security」64 ページを参照してください。

HTTPS_FIPS を使用できるのは Windows 上だけです。

dbmlsync を使用してネットワーク・プロトコル・オプションを設定する方法については、「CommunicationAddress (adr) 拡張オプション」138 ページを参照してください。

Ultra Light を使用してネットワーク・プロトコル・オプションを設定する方法については、「Ultra Light 同期クライアントのネットワーク・プロトコルのオプション」433 ページを参照してください。

参照

- 『Mobile Link 管理ガイド』> 「Mobile Link トランスポート・レイヤ・セキュリティ」
- 『Mobile Link 管理ガイド』> 「証明書フィールドの確認」
- 『Mobile Link 管理ガイド』> 「-x オプション」
- 「security」64 ページ
- 「trusted_certificates」68 ページ
- 「certificate_company」45 ページ
- 「certificate_name」47 ページ

例

HTTPS セキュリティの例は、「certificate_name」47 ページ と 「trusted_certificates」68 ページ を参照してください。

client_port

機能　通信に使用するクライアント・ポートの範囲を指定します。

構文　**client_port=***nnnnn*[**-***mmmmm*]

プロトコル

- dbmlsync - TCP/IP、HTTP、HTTPS、HTTPS_FIPS
- Ultra Light - TCP/IP、HTTP、HTTPS、UlSecureRSASocketStream、UlSecureSocketStream

デフォルト　なし

説明

可能なポート番号の範囲を示す開始値と終了値を指定します。クライアントを特定のポート番号に制限するには、*nnnnn* と *mmmmm* に同じ番号を指定します。値を 1 つだけ指定すると、範囲の上限値は初期値より 100 大きくなり、ポート数の合計は 101 になります。

このオプションは、ファイアウォール内のクライアントがファイアウォール外の Mobile Link 同期サーバと通信する場合に役立ちます。

HTTPS_FIPS を使用できるのは Windows 上だけです。

dbmlsync を使用してネットワーク・プロトコル・オプションを設定する方法については、「CommunicationAddress (adr) 拡張オプション」138 ページを参照してください。

Ultra Light を使用してネットワーク・プロトコル・オプションを設定する方法については、「Ultra Light 同期クライアントのネットワーク・プロトコルのオプション」433 ページを参照してください。

custom_header

機能　カスタム HTTP ヘッダを指定します。

構文　**custom_header=***header*

HTTP ヘッダの形式は、*header_name*: *header_value* です。

プロトコル

- dbmlsync - HTTP、HTTPS、HTTPS_FIPS

- Ultra Light - HTTP と HTTPS (ただし、Java Ultra Light ではサポートされない)

デフォルト

なし

説明

カスタム HTTP ヘッダを指定すると、クライアントは HTTP 要求を送信するごとにそのヘッダを含めます。複数のカスタム・ヘッダを指定するには、custom_header を複数回使用してください。

カスタム・ヘッダは、カスタム・ヘッダが必要なサードパーティ・ツールとの対話を同期クライアントが行う場合に便利です。

HTTPS_FIPS を使用できるのは Windows 上だけです。

dbmlsync を使用してネットワーク・プロトコル・オプションを設定する方法については、「CommunicationAddress (adr) 拡張オプション」138 ページを参照してください。

Ultra Light を使用してネットワーク・プロトコル・オプションを設定する方法については、「Ultra Light 同期クライアントのネットワーク・プロトコルのオプション」433 ページを参照してください。

例

一部の HTTP プロキシは、すべての要求に対して特殊なヘッダを含めること要求します。次の例では、Embedded SQL または C++ の Ultra Light アプリケーション内の値 ProxyUser に MyProxyHdr というカスタム HTTP ヘッダを設定します。

```
info.stream = ULHTTPStream();
 info.stream_parms = TEXT(
   "host=www.myhost.com;proxy_host=www.myproxy.com;
   custom_header=MyProxyHdr:ProxyUser");
```

host

機能

Mobile Link 同期サーバを実行中のマシン、または、Web サーバを介して同期する場合は Web サーバを実行中のマシンのホスト名または IP 番号を指定します。

構文

host=*hostname-or-ip*

プロトコル

- dbmlsync - TCP/IP、HTTP、HTTPS、HTTPS_FIPS
- Ultra Light - TCP/IP、HTTP、HTTPS、UlSecureRSASocketStream、UlSecureSocketStream

デフォルト

- Windows CE - デフォルト値は、CE デバイス・クレードルが接続されているデスクトップ・マシンから取得されます (レジストリ・フォルダ *Comm¥Tcpip¥Hosts¥ppp_peer* の *ipaddr* エントリとして格納される)。
- 他のすべてのデバイス - デフォルトは **localhost** です。

説明

Windows CE では、localhost を使用しないでください。これはリモート・デバイス自体を示します。デフォルト値を使用すると、Windows CE デバイスのクレードルが接続されているデスクトップ・マシン上の Mobile Link 同期サーバに Windows CE デバイスを接続できます。

Palm Computing Platform の場合は、localhost のデフォルト値がデバイスを指します。デスクトップ・マシンに接続するには、明示的なホスト名または IP アドレスを指定してください。

HTTPS_FIPS を使用できるのは Windows 上だけです。

dbmlsync を使用してネットワーク・プロトコル・オプションを設定する方法については、「CommunicationAddress (adr) 拡張オプション」138 ページを参照してください。

Ultra Light を使用してネットワーク・プロトコル・オプションを設定する方法については、「Ultra Light 同期クライアントのネットワーク・プロトコルのオプション」433 ページを参照してください。

http_password

機能

RFC 2617 の基本認証またはダイジェスト認証を使用してサードパーティの HTTP サーバとゲートウェイに対する認証を行います。

構文

http_password=*password*

プロトコル

- dbmlsync - HTTP、HTTPS、HTTPS_FIPS
- Ultra Light - HTTP、HTTPS

デフォルト

なし

説明

この機能は、RFC 2617 に記述されている基本認証とダイジェスト認証をサポートします。

基本認証ではパスワードはクリア・テキストで HTTP ヘッダに含められますが、HTTPS を使用すると、ヘッダを暗号化してパスワードを保護できます。ダイジェスト認証では、ヘッダはクリア・テキストでは送信されず、ハッシュされます。

このオプションは、http_userid と併用する必要があります。

HTTPS_FIPS を使用できるのは Windows 上だけです。

dbmlsync を使用してネットワーク・プロトコル・オプションを設定する方法については、「CommunicationAddress (adr) 拡張オプション」138 ページを参照してください。

Ultra Light を使用してネットワーク・プロトコル・オプションを設定する方法については、「Ultra Light 同期クライアントのネットワーク・プロトコルのオプション」433 ページを参照してください。

参照

- 「http_userid」57 ページ
- 「http_proxy_password」54 ページ
- 「http_proxy_userid」56 ページ

例

次に示す Embedded SQL または C++ の Ultra Light アプリケーションの例は、Web サーバに対する基本認証のユーザ ID とパスワードを提供します。

```
synch_info.stream = ULHTTPSStream();
      synch_info.stream_parms =
TEXT("http_userid=user;http_password=pwd");
```

http_proxy_password

機能

RFC 2617 の基本認証またはダイジェスト認証を使用してサードパーティの HTTP プロキシに対する認証を行います。

構文

http_proxy_password=*password*

プロトコル

- dbmlsync - HTTP、HTTPS、HTTPS_FIPS
- Ultra Light - HTTP、HTTPS

デフォルト

なし

説明

この機能は、RFC 2617 に記述されている基本認証とダイジェスト認証をサポートします。

基本認証では、パスワードはクリア・テキストで HTTP ヘッダに含められ、HTTPS を使用できます。ただし、プロキシに対する最初の接続は HTTP を通して行われるため、このパスワードはクリア・テキストです。ダイジェスト認証では、ヘッダはクリア・テキストでは送信されず、ハッシュされます。

このオプションは、http_proxy_userid と併用する必要があります。

HTTPS_FIPS を使用できるのは Windows 上だけです。

dbmlsync を使用してネットワーク・プロトコル・オプションを設定する方法については、「CommunicationAddress (adr) 拡張オプション」138 ページを参照してください。

Ultra Light を使用してネットワーク・プロトコル・オプションを設定する方法については、「Ultra Light 同期クライアントのネットワーク・プロトコルのオプション」433 ページを参照してください。

参照

- 「http_password」53 ページ
- 「http_userid」57 ページ
- 「http_proxy_userid」56 ページ

例

次に示す Embedded SQL または C++ の Ultra Light アプリケーションの例は、Web プロキシに対する基本認証のユーザ ID とパスワードを提供します。

```
synch_info.stream = ULHTTPSStream();
      synch_info.stream_parms =
TEXT("http_proxy_userid=user;http_proxy_password=pwd");
```

http_proxy_userid

機能

RFC 2617 の基本認証またはダイジェスト認証を使用してサードパーティの HTTP プロキシに対する認証を行います。

構文

http_proxy_userid=*userid*

プロトコル

- dbmlsync - HTTP、HTTPS、HTTPS_FIPS
- Ultra Light - HTTP、HTTPS

デフォルト

なし

説明

この機能は、RFC 2617 に記述されている基本認証とダイジェスト認証をサポートします。

基本認証では、パスワードはクリア・テキストで HTTP ヘッダに含められ、HTTPS を使用できます。ただし、プロキシに対する最初の接続は HTTP を通して行われるため、このパスワードはクリア・テキストです。ダイジェスト認証では、ヘッダはクリア・テキストでは送信されず、ハッシュされます。

このオプションは、http_proxy_password と併用する必要があります。

HTTPS_FIPS を使用できるのは Windows 上だけです。

dbmlsync を使用してネットワーク・プロトコル・オプションを設定する方法については、「CommunicationAddress (adr) 拡張オプション」138 ページを参照してください。

Ultra Light を使用してネットワーク・プロトコル・オプションを設定する方法については、「Ultra Light 同期クライアントのネットワーク・プロトコルのオプション」433 ページを参照してください。

参照

- 「http_password」53 ページ
- 「http_userid」57 ページ
- 「http_proxy_password」54 ページ

例

次に示す Embedded SQL または C++ の Ultra Light アプリケーションの例は、Web プロキシに対する基本認証のユーザ ID とパスワードを提供します。

```
synch_info.stream = ULHTTPSStream();
      synch_info.stream_parms =
TEXT("http_proxy_userid=user;http_proxy_password=pwd");
```

http_userid

機能

RFC 2617 の基本認証またはダイジェスト認証を使用してサードパーティの HTTP サーバとゲートウェイに対する認証を行います。

構文

http_userid=*userid*

プロトコル

- dbmlsync - HTTP、HTTPS、HTTPS_FIPS
- Ultra Light - HTTP、HTTPS

デフォルト

なし

説明

この機能は、RFC 2617 に記述されている基本認証とダイジェスト認証をサポートします。

基本認証ではパスワードはクリア・テキストで HTTP ヘッダに含められますが、HTTPS を使用すると、ヘッダを暗号化してパスワードを保護できます。ダイジェスト認証では、ヘッダはクリア・テキストでは送信されず、ハッシュされます。

このオプションは、http_password と併用する必要があります。

HTTPS_FIPS を使用できるのは Windows 上だけです。

dbmlsync を使用してネットワーク・プロトコル・オプションを設定する方法については、「CommunicationAddress (adr) 拡張オプション」138 ページを参照してください。

Ultra Light を使用してネットワーク・プロトコル・オプションを設定する方法については、「Ultra Light 同期クライアントのネットワーク・プロトコルのオプション」433 ページを参照してください。

参照

- 「http_password」53 ページ
- 「http_proxy_password」54 ページ
- 「http_proxy_userid」56 ページ

例

次に示す Embedded SQL または C++ の Ultra Light アプリケーションの例は、Web サーバに対する基本認証のユーザ ID とパスワードを提供します。

```
synch_info.stream = ULHTTPSStream();
      synch_info.stream_parms =
TEXT("http_userid=user;http_password=pwd");
```

liveness_timeout

機能

Mobile Link がクライアントとの接続を遮断したとみなして同期をアボートするまでの、クライアントとの最後の通信からの経過時間 (秒単位) を指定します。

構文

liveness_timeout=*n*

プロトコル

- dbmlsync - TCP/IP
- Ultra Light - TCP/IP、UlSecureRSASocketStream、UlSecureSocketStream

デフォルト

120 秒

説明

値 0 はタイムアウトがないことを意味します。

このオプションは、クライアントのダウンロード確認が off (デフォルト) に設定されている場合に便利です。ダウンロード確認が on の場合は通常、このオプションを使用しません。ダウンロードの適用のときに指定の n 秒を超えると、Mobile Link 同期サーバが同期をアボートするためです。これを避けるために liveness_timeout 値を大きくしては、タイムアウトの目的に合いません。

dbmlsync を使用してネットワーク・プロトコル・オプションを設定する方法については、「CommunicationAddress (adr) 拡張オプション」138 ページを参照してください。

Ultra Light を使用してネットワーク・プロトコル・オプションを設定する方法については、「Ultra Light 同期クライアントのネットワーク・プロトコルのオプション」433 ページを参照してください。

network_connect_timeout

機能

同期クライアントが接続の試行を中止するまでの時間 (秒単位) を指定します。

構文

network_connect_timeout=*seconds*

プロトコル

- dbmlsync - TCP/IP、HTTP、HTTPS、HTTPS_FIPS
- Ultra Light - TCP/IP、HTTP、HTTPS

デフォルト

120 秒

説明

このオプションを使用するには、network_name を指定する必要があります。

この機能は、Pocket PC 2002 でのみ使用できます。Windows では、特定の network_name の接続プロファイルを設定してこの機能を制御します。

HTTPS_FIPS を使用できるのは Windows 上だけです。

dbmlsync を使用してネットワーク・プロトコル・オプションを設定する方法については、「CommunicationAddress (adr) 拡張オプション」138 ページを参照してください。

Ultra Light を使用してネットワーク・プロトコル・オプションを設定する方法については、「Ultra Light 同期クライアントのネットワーク・プロトコルのオプション」433 ページを参照してください。

network_leave_open

機能

network_name を指定すると、オプションとして、同期が終了した後にネットワーク接続を開いたままにするように指定できます (1)。

構文

network_leave_open={ 0 | 1 }

プロトコル

- dbmlsync - TCP/IP、HTTP、HTTPS、HTTPS_FIPS
- Ultra Light - TCP/IP、HTTP、HTTPS

デフォルト

ネットワーク接続は、同期の後に切断されます (**0**)。

説明

このオプションを使用するには、network_name を指定する必要があります。

このオプションを 1 に設定すると、同期が終了した後もネットワーク接続は開いたままになります。

HTTPS_FIPS を使用できるのは Windows 上だけです。

dbmlsync を使用してネットワーク・プロトコル・オプションを設定する方法については、「CommunicationAddress (adr) 拡張オプション」138 ページを参照してください。

Ultra Light を使用してネットワーク・プロトコル・オプションを設定する方法については、「Ultra Light 同期クライアントのネットワーク・プロトコルのオプション」433 ページを参照してください。

参照

「network_name」60 ページ

network_name

機能

ネットワークにまだ接続していない場合に開始するネットワーク名を指定します。

構文

network_name=*name*

プロトコル

- dbmlsync - TCP/IP、HTTP、HTTPS、HTTPS_FIPS
- Ultra Light - TCP/IP、HTTP、HTTPS

デフォルト

なし

説明

ネットワーク名を指定して、Mobile Link の自動ダイヤル機能を使用できるようにします。これによって、手動でダイヤルすることなく Pocket PC 2002 または Windows デスクトップ・コンピュータから接続できます。

スケジュールと組み合わせて使用すると、リモート・データベースを無人で同期できます。スケジュールと組み合わせなくても、手動でダイヤルして接続せずに dbmlsync を実行できます。この名前は、

[設定] － [接続] － [接続] の順にクリックすると表示されるドロップダウン・リスト (Pocket PC の場合)、または [ネットワークとダイヤルアップ接続] (Windows の場合) で指定したネットワーク名でなければなりません。

HTTPS_FIPS を使用できるのは Windows 上だけです。

dbmlsync を使用してネットワーク・プロトコル・オプションを設定する方法については、「CommunicationAddress (adr) 拡張オプション」138 ページを参照してください。

Ultra Light を使用してネットワーク・プロトコル・オプションを設定する方法については、「Ultra Light 同期クライアントのネットワーク・プロトコルのオプション」433 ページを参照してください。

参照

- 『Mobile Link クライアント』> 「同期のスケジュール」
- 「network_connect_timeout」59 ページ
- 「network_leave_open」59 ページ

persistent

機能

同期のすべての HTTP 要求に単一の TCP/IP 接続を使用します。

構文

persistent={ 0 | 1 }

プロトコル

- dbmlsync - HTTP、HTTPS、HTTPS_FIPS
- Ultra Light - HTTP、HTTPS

HTTP、HTTPS

デフォルト

- Palm - **0**
- 他の全デバイス - **1**

説明

1 は、クライアントが同期のすべての HTTP 要求に対して同じ TCP/IP 接続を使用することを意味しています。設定を 0 にすると、通常、中間エージェントとの互換性が高まります。

Palm デバイスを除き、Mobile Link に直接接続している場合は、persistent を 1 に設定してください。プロキシやリダイレクタなどの中間エージェントを通じて接続している場合は、永続的な接続によって問題が発生することがあります。

HTTPS_FIPS を使用できるのは Windows 上だけです。

dbmlsync を使用してネットワーク・プロトコル・オプションを設定する方法については、「CommunicationAddress (adr) 拡張オプション」138 ページを参照してください。

Ultra Light を使用してネットワーク・プロトコル・オプションを設定する方法については、「Ultra Light 同期クライアントのネットワーク・プロトコルのオプション」433 ページを参照してください。

port

機能

ソケットのポート番号を指定します。

構文

port=*port-number*

プロトコル

- dbmlsync - TCP/IP、HTTP、HTTPS、HTTPS_FIPS
- Ultra Light - TCP/IP、HTTP、HTTPS、UlSecureRSASocketStream、UlSecureSocketStream

デフォルト

TCP/IP のデフォルトは **2439** です。これは、Mobile Link 同期サーバの IANA 登録ポート番号です。

HTTP のデフォルト値は **80** です。

HTTPS と HTTPS_FIPS のデフォルトは **443** です。

説明

ポート番号は 10 進数で、Mobile Link 同期サーバが受信するように設定されているポートと一致させます。

Web サーバを介して同期する場合は、HTTP 要求または HTTPS 要求を受け付ける Web サーバ・ポートを指定してください。

HTTPS_FIPS を使用できるのは Windows 上だけです。

dbmlsync を使用してネットワーク・プロトコル・オプションを設定する方法については、「CommunicationAddress (adr) 拡張オプション」138 ページを参照してください。

Ultra Light を使用してネットワーク・プロトコル・オプションを設定する方法については、「Ultra Light 同期クライアントのネットワーク・プロトコルのオプション」433 ページを参照してください。

proxy_host

機能

プロキシ・サーバのホスト名または IP アドレスを指定します。

構文

proxy_host=*proxy_hostname_or_ip*

プロトコル

- dbmlsync - HTTP、HTTPS、HTTPS_FIPS
- Ultra Light - HTTP、HTTPS

HTTP、HTTPS

デフォルト

なし

説明

プロキシを通す場合だけ使用します。

HTTPS_FIPS を使用できるのは Windows 上だけです。

dbmlsync を使用してネットワーク・プロトコル・オプションを設定する方法については、「CommunicationAddress (adr) 拡張オプション」138 ページを参照してください。

Ultra Light を使用してネットワーク・プロトコル・オプションを設定する方法については、「Ultra Light 同期クライアントのネットワーク・プロトコルのオプション」433 ページを参照してください。

proxy_port

機能

プロキシ・サーバのポート番号を指定します。

構文

proxy_port=*proxy_port_number*

プロトコル

- dbmlsync - HTTP、HTTPS、HTTPS_FIPS
- Ultra Light - HTTP、HTTPS

デフォルト

なし

説明

プロキシを通す場合だけ使用します。

HTTPS_FIPS を使用できるのは Windows 上だけです。

dbmlsync を使用してネットワーク・プロトコル・オプションを設定する方法については、「CommunicationAddress (adr) 拡張オプション」138 ページを参照してください。

Ultra Light を使用してネットワーク・プロトコル・オプションを設定する方法については、「Ultra Light 同期クライアントのネットワーク・プロトコルのオプション」433 ページを参照してください。

security

機能

同期用の暗号を解く鍵と暗号化オプションを指定します。

別途ライセンスを取得できるオプションが必要
トランスポート・レイヤ・セキュリティを使用するには、別途ライセンスを取得できる SQL Anywhere Studio セキュリティ・オプションを入手する必要があります。このセキュリティ・オプションは、輸出規制対象品目です。

このコンポーネントのご注文については、『SQL Anywhere Studio の紹介』> 「別途ライセンスが入手可能なコンポーネント」を参照してください。

構文

security=*cipher***(keyword=***value***;…)**

プロトコル

dbmlsync - TCP/IP

デフォルト

なし

説明

この同期のための通信はすべて、指定された暗号を使用して暗号化されます。暗号は次のいずれかを指定してください。

- **ecc_tls** 楕円曲線暗号化です。下位互換性を保つために、**ecc_tls** を **certicom_tls** と指定することもできます。
- **rsa_tls** RSA 暗号化です。
- **rsa_tls_fips** FIPA によって認可された RSA 暗号化です。rsa_tls_fips 暗号は、Certicom が提供している別の FIPS 140-2 準拠ソフトウェアを使用します。rsa_tls を使用するクライアントは rsa_tls_fips を使用するサーバと互換性があり、rsa_tls_fips を使用するクライアントは rsa_tls を使用するサーバと互換性があります。rsa_tls_fips は、Windows 上の Adaptive Server Anywhere データベースにしか使用できません。

詳細については、『Mobile Link 管理ガイド』> 「トランスポート・レイヤ・セキュリティを使用する Mobile Link クライアントの設定」を参照してください。

以下のセキュリティ・キーワードがサポートされています。

- **certificate_company=***organization*
- **certificate_name=***common_name*
- **certificate_unit=***organization_unit*
- **trusted_certificates=***filename*

HTTPS で dbmlsync を使用する場合、このセキュリティ・オプションは設定しませんが、これらの 4 つのセキュリティ・キーワードを直接設定します。セキュリティ・オプションの設定方法の詳細は、各セキュリティ・キーワードを参照してください。

Ultra Light については、「Security 同期パラメータ」415 ページを参照してください。

dbmlsync を使用してネットワーク・プロトコル・オプションを設定する方法については、「CommunicationAddress (adr) 拡張オプション」138 ページを参照してください。

Ultra Light を使用してネットワーク・プロトコル・オプションを設定する方法については、「Ultra Light 同期クライアントのネットワーク・プロトコルのオプション」433 ページを参照してください。

参照

- 『Mobile Link 管理ガイド』> 「Mobile Link トランスポート・レイヤ・セキュリティ」
- 『Mobile Link 管理ガイド』> 「-x オプション」
- 「certificate_company」45 ページ
- 「certificate_name」47 ページ
- 「certificate_unit」49 ページ
- 「trusted_certificates」68 ページ

例

dbmlsync TCP/IP プロトコルの RSA 暗号化を設定する例を示します。これは、サーバとクライアント両方での設定が必要です。各コマンドは、1 行に入力する必要があります。

```
dbmlsrv9
   -c "dsn=asa90sample;uid=DBA;pwd=SQL"
   -x tcpip(
     port=9999;
     security=rsa_tls(
       certificate=c:¥asa90¥win32¥rsaserver.crt;
       certificate_password=test))

dbmlsync
   -c "dsn=mydb;uid=DBA;pwd=SQL"
   -e "ctp=tcpip;
       adr='port=9999;
          security=rsa_tls(

trusted_certificates=c:¥asa90¥win32¥rsaroot.crt;
            certificate_name=RSA Server)'"
```

Ultra Light クライアントの場合、クライアント側の実装が多少異なります。次に、Embedded SQL または C++ の Ultra Light アプリケーションにおける同等のクライアント実装を示します。

```
     info.stream = ULSocketStream();
      info.stream_parms = TEXT("port=9999");
      info.security_stream = ULSecureRSATLSStream();
      info.security_parms =
TEXT("trusted_certificates=¥rsaroot.crt;certificate_nam
e=RSA Server");
```

set_cookie

機能

同期中に使用される HTTP 要求内に設定するカスタム HTTP cookie を指定します。

構文

set_cookie=*cookie_name***=***cookie_value* [**,***cookie_name***=***cookie_value*, ...]

プロトコル

- dbmlsync - HTTP、HTTPS、および HTTPS_FIPS
- Ultra Light - HTTP と HTTPS (ただし、Java Ultra Light ではサポートされない)

デフォルト

なし

説明

カスタム HTTP cookie は、同期クライアントがサードパーティ・ツール (セッションを識別するために cookie を使用する認証ツールなど) と対話をする場合に便利です。たとえば、ユーザ・エージェントが Web サーバ、プロキシ、またはゲートウェイに接続し、それ自体の認証を行うシステムが存在するとします。正常に動作する場合、エージェントはサーバから 1 つ以上の cookie を受け取ります。続いてエージェントは同期を開始し、set_cookie オプションを通してそのセッション cookie を渡します。

HTTPS_FIPS を使用できるのは Windows 上だけです。

dbmlsync を使用してネットワーク・プロトコル・オプションを設定する方法については、「CommunicationAddress (adr) 拡張オプション」138 ページを参照してください。

Ultra Light を使用してネットワーク・プロトコル・オプションを設定する方法については、「Ultra Light 同期クライアントのネットワーク・プロトコルのオプション」433 ページを参照してください。

例

Embedded SQL または C++ の Ultra Light アプリケーションでカスタム HTTP cookie を設定する例を示します。

```
info.stream = ULHTTPStream();
 info.stream_parms = TEXT(
   "host=www.myhost.com;
   set_cookie=MySessionID=12345, enabled=yes;");
```

trusted_certificates

機能

安全な同期に使用される信用されたルート証明書のリストを含むファイルを指定します。

別途ライセンスを取得できるオプションが必要
トランスポート・レイヤ・セキュリティを使用するには、別途ライセンスを取得できる SQL Anywhere Studio セキュリティ・オプションを入手する必要があります。このセキュリティ・オプションは、輸出規制対象品目です。

このコンポーネントのご注文については、『SQL Anywhere Studio の紹介』> 「別途ライセンスが入手可能なコンポーネント」を参照してください。

構文

trusted_certificates=*filename*

プロトコル

- dbmlsync - TCP/IP (セキュリティ・オプションを使用)、HTTPS、HTTP_FIPS
- Ultra Light - TCP/IP (セキュリティ・パラメータを使用)、HTTPS。Palm OS ではサポートされません。

デフォルト

なし

説明

Certicom TLS 同期ストリームを介して同期が発生すると、Mobile Link 同期サーバは、証明書をクライアントに送信します。送信する証明書はエンティティの署名付き証明書だけでなく、自己署名ルート証明書にまで及びます。

クライアントは、連鎖が有効で連鎖内のルート証明書が信用できることを確認します。この機能を使用すると、信用するルート証明書を指定できます。

このオプションは、dbmlsync に暗号化を使用する場合に必要です。

このオプションは、Palm プラットフォーム上と Ultra Light 静的型 Java 環境では使用できません。

TCP/IP の場合は、このオプションをセキュリティ・オプションの一部として設定してください。詳細については、「security」64 ページを参照してください。

HTTPS_FIPS を使用できるのは Windows 上だけです。

dbmlsync を使用してネットワーク・プロトコル・オプションを設定する方法については、「CommunicationAddress (adr) 拡張オプション」138 ページを参照してください。

Ultra Light を使用してネットワーク・プロトコル・オプションを設定する方法については、「Ultra Light 同期クライアントのネットワーク・プロトコルのオプション」433 ページを参照してください。

参照

- 『Mobile Link 管理ガイド』> 「Mobile Link トランスポート・レイヤ・セキュリティ」
- 『Mobile Link 管理ガイド』> 「-x オプション」
- 「security」64 ページ
- 「certificate_company」45 ページ
- 「certificate_name」47 ページ
- 「certificate_unit」49 ページ

例

HTTPS プロトコルの RSA 暗号化を設定する例を示します。これは、サーバとクライアント両方での設定が必要です。各コマンドは、1 行に入力する必要があります。

```
dbmlsrv9
    -c "dsn=asa90sample;uid=DBA;pwd=SQL"
    -x https(
      port=9999;
      certificate=c:¥asa90¥win32¥rsaserver.crt;
      certificate_password=test)
```

Adaptive Server Anywhere クライアントでは、実装は次のようになります。

```
dbmlsync
    -c "dsn=mydb;uid=DBA;pwd=SQL"
    -e "ctp=https;
        adr='port=9999;

trusted_certificates=c:¥asa90¥win32¥rsaroot.crt;
          certificate_name=RSA Server'"
```

Ultra Light クライアントでの同等の実装は次のようになります。

```
info.stream = ULHTTPSStream();
 info.stream_parms = TEXT(
   "port=9999;
    trusted_certificates=¥rsaroot.crt;
    certificate_name=RSA Server");
 info.security_stream = NULL;
 info.security_parms = NULL;
```

Palm OS に対応する Ultra Light クライアントの場合、Ultra Light ジェネレータによって生成されたコード内に次のコマンドを使用して信用されたルート証明書を埋め込むことができます。

```
ulgen.exe -c "dsn=asa90sample;uid=DBA;pwd=SQL"  -r
c:¥asa90¥win32¥rsaroot.crt test test.c
```

あるいは、次のように Ultra Light の初期化ユーティリティを使用することもできます。

```
ulinit.exe -c "dsn=asa90sample;uid=DBA;pwd=SQL" -t
c:¥asa90¥win32¥rsaroot.crt -f test.usm -n test_pub
```

url_suffix

機能

同期中に送信される各 HTTP 要求の 1 行目の URL に追加するサフィックスを指定します。

構文

url_suffix=*suffix*

プロトコル

- dbmlsync - HTTP、HTTPS、HTTPS_FIPS
- Ultra Light - HTTP、HTTPS

HTTP、HTTPS

デフォルト

デフォルト値は *MobiLink¥* です。

説明

プロキシ・サーバまたは Web サーバを介して同期する場合、Mobile Link 同期サーバを検索するために url_suffix が必要な場合があります。

リダイレクタを使用する場合にこのオプションを設定する方法については、『Mobile Link 管理ガイド』> 「Mobile Link クライアントとサーバのリダイレクタ設定」を参照してください。

HTTPS_FIPS を使用できるのは Windows 上だけです。

dbmlsync を使用してネットワーク・プロトコル・オプションを設定する方法については、「CommunicationAddress (adr) 拡張オプション」138 ページを参照してください。

Ultra Light を使用してネットワーク・プロトコル・オプションを設定する方法については、「Ultra Light 同期クライアントのネットワーク・プロトコルのオプション」433 ページを参照してください。

version

機能

同期に使用する HTTP のバージョンを指定します。

構文

version=*HTTP_version_number*

プロトコル

- dbmlsync - HTTP、HTTPS、HTTPS_FIPS
- Ultra Light - HTTP、HTTPS

デフォルト

デフォルト値は **1.1** です。

説明

このオプションは、HTTP インフラが特定の HTTP バージョンを必要とする場合に便利です。値は、**1.0** または **1.1** です。

HTTPS_FIPS を使用できるのは Windows 上だけです。

dbmlsync を使用してネットワーク・プロトコル・オプションを設定する方法については、「CommunicationAddress (adr) 拡張オプション」138 ページを参照してください。

Ultra Light を使用してネットワーク・プロトコル・オプションを設定する方法については、「Ultra Light 同期クライアントのネットワーク・プロトコルのオプション」433 ページを参照してください。

第 2 部　Mobile Link 用の Adaptive Server Anywhere クライアント

第 2 部では、Mobile Link 同期のために Adaptive Server Anywhere クライアントを設定し実行する方法について説明します。

第 5 章

Adaptive Server Anywhere クライアント

この章の内容

この章では、Adaptive Server Anywhere データベースを Mobile Link クライアントとして使用する方法について説明します。

この章で紹介している概念の一部について説明したチュートリアルは、『Mobile Link チュートリアル』> 「チュートリアル：Mobile Link の概要」を参照してください。

Mobile Link クライアントとしての Ultra Light データベースについては、「Ultra Light クライアント」347 ページを参照してください。

リモート・データベースの作成

すべての Adaptive Server Anywhere データベースは、Mobile Link インストール環境でリモート・データベースとして使用できるように変換できます。必要な作業は、パブリケーションを作成し、Mobile Link ユーザを作成し、Mobile Link ユーザをパブリケーションにサブスクライブすることだけです。

❖ **Adaptive Server Anywhere リモート・データベースを作成するには、次の手順に従います。**

1. 既存の Adaptive Server Anywhere データベースを指定して起動するか、新しいデータベースを作成してテーブルを追加します。
2. 新しいデータベース内で 1 つまたは複数のパブリケーションを作成します。

   「データのパブリッシュ」82 ページを参照してください。

3. Mobile Link ユーザを作成します。

   「Mobile Link ユーザの作成」92 ページを参照してください。

4. 1 つまたは複数のパブリケーションに対する Mobile Link ユーザのサブスクリプションを作成します。

   「Mobile Link 同期ユーザのサブスクリプションの作成」98 ページを参照してください。

リモート・データベースの配備

Adaptive Server Anywhere リモート・データベースを配備するには、データベースを作成し、適切なパブリケーションおよびサブスクリプションを追加する必要があります。これを行うには、プロトタイプのリモート・データベースをカスタマイズします。

❖ プロトタイプをカスタマイズして Mobile Link リモート・データベースを配備するには、次の手順に従います。

1. プロトタイプとなるリモート・データベースを作成します。

   プロトタイプ・データベースには、必要なテーブルとパブリケーションをすべて入れますが、各データベースに固有の情報を入れる必要はありません。通常、この個別情報は次のとおりです。

   - Mobile Link ユーザ名
   - 同期サブスクリプション
   - グローバル・オートインクリメント・キー値の始点を提供する GLOBAL_DATABASE_ID オプション

2. リモート・データベースごとに、次の操作を実行します。

   - リモート・データベースを保持するディレクトリを作成します。
   - そのディレクトリにプロトタイプのリモート・データベースをコピーします。

     トランザクション・ログがリモート・データベースと同じディレクトリに保持されている場合、ログ・ファイル名を変更する必要はありません。

   - 個々の情報をデータベースに追加する SQL スクリプトを実行します。

     この SQL スクリプトは、パラメータ化されたスクリプトにすることができます。パラメータ化されたスクリプトの詳細については、『ASA SQL リファレンス・マニュアル』> 「PARAMETERS 文 [Interactive SQL]」と『ASA SQL ユーザーズ・ガイド』> 「SQL コマンド・ファイルの使用」を参照してください。

例

次の SQL スクリプトは、Contact の例から抜粋したものです。これは、*Samples¥MobiLink¥Contact¥customize.sql* に入っています。

```
PARAMETERS ml_userid, db_id;
 go
 SET OPTION PUBLIC.GLOBAL_DATABASE_ID = {db_id}
 go

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

次のコマンド・ラインは、データ・ソース **dsn_remote_1** を指定し、リモート・データベースに対してスクリプトを実行します。

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

リモート・データベース間でのデータ分割

通常、リモート・データベースはそれぞれの要件に従って別々のカテゴリに分割されます。販売アプリケーションを例にとってみましょう。ある地域のすべての営業要員は特定のデータ・セットにアクセスする必要があっても、担当外の地域に関する情報へのアクセスは不要です。他の部署の従業員は、まったく異なる性質のデータが必要かもしれません。マネージャは、自分の部下がデータにアクセスできないようにすることが必要な場合があります。

通常、パブリケーションは基本的に異なるデータ・セットを指定するために使用します。たとえば、営業スタッフ用に 1 つ、テクニカル・サポート要員用にもう 1 つのパブリケーションを作成できます。

パブリケーションで WHERE 句を使えば、特定のリモート・データベースが受け取るデータをさらに詳細に指定できます。この機能は、リモート・データベースが同種の情報を要求しているとき役に立ちます。たとえば、この機能を使って、営業担当者に担当地域に関連する情報のみを提供できます。

詳細については、『Mobile Link 管理ガイド』> 「リモート・データベース間でローを分割する」を参照してください。

リモート・データベースのアップグレード

新しい Adaptive Server Anywhere リモート・データベースをインストールして古いバージョンを上書きすると、統合データベース内の同期進行状況情報が正しくなくなります。

この問題を解決するには、このユーザの ml_user テーブルの progress カラムを 0 (ゼロ) に設定します。これは、Mobile Link システム・テーブルを直接修正する必要がある例外的なケースです。これ以外の場合は、Mobile Link システム・テーブルには直接アクセスしないでください。

詳細については、『SQL Anywhere Studio 新機能ガイド』> 「Adaptive Server Anywhere Mobile Link クライアントのアップグレード」を参照してください。

進行オフセット

オフセット (進行やステータスとも呼ばれる) は、リモート・データベースのトランザクション・ログの位置を示します。これは、サブスクリプションに対するすべての操作がアップロードされ確認された点を示します。dbmlsync は、オフセットを使用してどのデータをアップロードするか決定します。リモート・データベースでは、オフセットは SYS.SYSSYNC システム・テーブルの progress カラムに格納されます。統合データベースでは、バージョン 7.x データベースのオフセットは ml_user テーブルの progress カラムに格納され、バージョン 8.0 以降のデータベースのオフセットは ml_subscription テーブルの progress カラムに格納されます。

リモートごとに、リモート・データベースと統合データベースが各サブスクリプションに対するオフセットを管理します。ユーザが同期を行うと、そのユーザに関連するすべてのサブスクリプションに対してオフセットが確認されます。これは、その時点でサブスクリプションの同期がとられていない場合でも同様です。このように処理されるのは、各パブリケーションが同じデータを同期できるためです。唯一の例外として、dbmlsync は、アップロードを試みるまでサブスクリプションの進行オフセットをチェックしません。

リモート・データベースのオフセットと統合データベースのオフセットが一致しない場合、デフォルトの動作はリモートのオフセットを統合データベースの値で更新し、そのオフセットに基づいて新しいアップロードを送信します。ほとんどの場合、このデフォルト動作が適切です。たとえば、統合データベースがバックアップからリストアされ、リモート・トランザクション・ログが変更されていないとき、またはアップロードは成功したが通信エラーによりアップロードの確認が送信されないときに、広く有効です。

進行オフセットが一致しない場合の大部分は、統合進行値を使用することで自動的に解決されます。進行オフセットの問題の修正が必要になることがまれにありますが、この場合は dbmlsync -r オプションを使用できます。

詳細については、「-r オプション」188 ページを参照してください。

最初の同期は常に行われる

リモート・データベースの削除および再作成後、初めて同期する場合、統合データベース上の進行値と一致しなくても、進行オフセットの不一致にはなりません。これは、最初の同期であることを dbmlsync が検出し、統合データベースのオフセットをチェックする代わりにリモート・データベースの進行オフセットを使用するためです。

リモート・データベース・システム・テーブル SYS.SYSSYNC 内の **progress** カラムの値が **created** カラムの値と同じであり、**log_sent** カラムの値が NULL の場合、dmlsync は最初の同期を検出します。

ただし、最初の同期で同じアップロード・ストリーム内の複数のサブスクリプションを同期し、そのサブスクリプションのいずれかが初めての同期でない場合、初めて同期されるサブスクリプションも含めて、同期対象のすべてのサブスクリプションに関して進行オフセットがチェックされます。たとえば、2 つのパブリケーション (-n pub1,pub2) に関して dbmlsync -n オプションを指定する場合で、pub1

は以前に同期され、pub2 は同期されていないとします。この場合、リモート・データベースを削除および再作成すると、統合データベースとの間で進行オフセット不一致が発生する可能性があります。

詳細については、次の項を参照してください。

- 『ASA SQL リファレンス・マニュアル』> 「SYSSYNC システム・テーブル」
- 『Mobile Link 管理ガイド』> 「ml_user」
- 『Mobile Link 管理ガイド』> 「ml_subscription」

データのパブリッシュ

パブリケーションとは、同期されるデータを識別するデータベース・オブジェクトです。パブリケーションは、テーブルのカラム、ロー、あるいはその両方のサブセットであるアーティクルで構成されています。各パブリケーションに、1 つまたは複数のテーブル全体、または選択したローとカラムからなるテーブルの一部を含めることができます。1 つのパブリケーションでは、1 つのテーブルを複数のアーティクルに含めることはできません。

Sybase Central または CREATE PUBLICATION 文を使用して、パブリケーションを作成します。

Sybase Central では、[パブリケーション] フォルダにすべてのパブリケーションとアーティクルがあります。

パブリケーションについての注意

- パブリケーションの作成と削除には DBA 権限が必要です。
- 複数のパブリケーションにサブスクライブするときに同期するためには、同じテーブルの同じカラム・サブセットがそのパブリケーションに含まれる必要があります。
- パブリケーションはどのカラムが選択されているかは確認しますが、それらが送信される順序は確認しません。カラムは、CREATE TABLE 文で定義された順に常に送信されます。
- パブリケーションは、それらが参照するテーブルのプライマリ・キー内のカラムをすべて含んでいる必要があります。
- 1 つのパブリケーションで、テーブル・セットのカラム・サブセットをパブリッシュし、WHERE 句を使用して、レプリケートするロー・セットを選択できます。
- パブリケーションにビューとストアド・プロシージャを入れることはできません。
- パブリケーションとサブスクリプションは、Sybase のメッセージベースのレプリケーション・テクノロジである SQL Remote でも使用されます。SQL Remote の場合は、統合データベースとリモート・データベースの両方にパブリケーションとサブスクリプションが必要です。これに対して、Mobile Link では、パブ

リケーションは Adaptive Server Anywhere リモート・データベースにのみ存在します。Mobile Link 統合データベースは、同期スクリプトを使用して設定されます。

テーブル全体のパブリッシュ

作成できる最も簡単なパブリケーションは、単一のアーティクルで構成されます。アーティクルは、1 つまたは複数のテーブルに含まれるすべてのローとカラムで構成されます。これらのテーブルをあらかじめ用意してください。

❖ 1 つ以上のテーブル全体をパブリッシュするには、次の手順に従います (Sybase Central の場合)。

1. Adaptive Server Anywhere プラグインを使用して、DBA 権限のあるユーザとしてリモート・データベースに接続します。
2. [パブリケーション] フォルダを開きます。
3. [ファイル] メニューから [新規] － [パブリケーション] を選択します。[新しいパブリケーションの作成] ウィザードが表示されます。
4. 新しく作成するパブリケーションの名前を入力します。[次へ] をクリックします。
5. [テーブル] タブで、[使用可能なテーブル] のリストからテーブルを 1 つ選択します。[追加] をクリックします。選択したテーブルが、右側の [選択したテーブル] リストに表示されます。
6. オプションで、さらにテーブルを追加することもできます。テーブルの順序は重要ではありません。
7. [完了] をクリックします。

❖ **1 つまたは複数のテーブル全体をパブリッシュするには、次の手順に従います (SQL の場合)。**

1. DBA 権限のあるユーザとしてリモート・データベースに接続します。
2. CREATE PUBLICATION 文を実行して、新しく作成するパブリケーション名とパブリッシュするテーブルを指定します。

例

次の文は、customer テーブル全体をパブリッシュするパブリケーションを作成します。

```
CREATE PUBLICATION pub_customer (
     TABLE customer
 )
```

次の文は、Adaptive Server Anywhere のサンプル・データベースから、テーブル・セットの各テーブルのすべてのカラムとローを含むパブリケーションを作成します。

```
CREATE PUBLICATION sales (
     TABLE customer,
     TABLE sales_order,
     TABLE sales_order_items,
     TABLE product
 )
```

詳細については、『ASA SQL リファレンス・マニュアル』> 「CREATE PUBLICATION 文」を参照してください。

テーブル内の一部のカラムだけをパブリッシュする

Sybase Central では、テーブルのすべてのローと、一部のカラムだけを含むパブリケーションを作成できます。また、CREATE PUBLICATION 文でカラムのリストを指定しても、同様に作成できます。

注意

異なるカラムのサブセットを持つ同じテーブルが含まれたパブリケーションを 2 つ作成すると、両方のパブリケーションをサブスクライブするユーザは同期することができません。

❖ テーブル内の一部のカラムだけをパブリッシュするには、次の手順に従います (Sybase Central の場合)。

1. Adaptive Server Anywhere プラグインを使用して、DBA 権限のあるユーザとしてリモート・データベースに接続します。
2. [パブリケーション] フォルダを開きます。
3. [ファイル] メニューから [新規] － [パブリケーション] を選択します。[新しいパブリケーションの作成] ウィザードが表示されます。
4. 新しく作成するパブリケーションの名前を入力します。[次へ] をクリックします。
5. [テーブル] タブで、[使用可能なテーブル] のリストからテーブルを 1 つ選択します。[追加] をクリックします。選択したテーブルが右側の [選択したテーブル] のリストに追加されます。
6. [カラム] タブで、テーブルのアイコンをダブルクリックし、[使用できるカラム] のリストを展開します。パブリッシュするカラムを 1 つずつ選択し、[追加] をクリックします。右側に選択したカラムが表示されます。
7. [完了] をクリックします。

❖ テーブル内の一部のカラムだけをパブリッシュするには、次の手順に従います (SQL の場合)。

1. DBA 権限のあるユーザとしてリモート・データベースに接続します。
2. CREATE PUBLICATION 文を実行して、パブリケーション名とテーブル名を指定します。テーブル名の後ろにあるカッコの中に、パブリッシュするカラムをリストします。

例

次の文は、customer テーブルのカラムである id、company_name、city のすべてのローをパブリッシュするパブリケーションを作成します。

```
CREATE PUBLICATION pub_customer (
     TABLE customer (id, company_name,
         city )
 )
```

詳細については、『ASA SQL リファレンス・マニュアル』>「CREATE PUBLICATION 文」を参照してください。

テーブル内の一部のローだけをパブリッシュする

パブリケーションを作成するときには、テーブルの一部またはすべてのカラムと一部のローだけを含むものを作成できます。これを行うには、パブリッシュするローにのみ一致する探索条件を指定します。

テーブル内の一部のローのみをパブリッシュする方法は、Sybase Central と SQL 言語から 2 とおり提供されますが、Mobile Link との互換性があるのは 1 つだけです。

- **WHERE 句**　Mobile Link との互換性があります。WHERE 句を使用すると、アーティクルにローのサブセットを入れることができます。
- **サブスクリプション式**　Mobile Link では無視されます。

Mobile Link では、WHERE 句を使用して、あるパブリケーションのすべてのサブスクリプションから同一セットのローを除外できます。パブリケーションのすべてのサブスクライバは、探索条件を満たすローのすべての変更をアップロードします。

❖ WHERE 句を使用してパブリケーションを作成するには、次の手順に従います (Sybase Central の場合)。

1. Adaptive Server Anywhere プラグインを使用して、DBA 権限のあるユーザとしてリモート・データベースに接続します。
2. [パブリケーション] フォルダを開きます。
3. [ファイル] メニューから [新規] － [パブリケーション] を選択します。[新しいパブリケーションの作成] ウィザードが表示されます。

4 新しく作成するパブリケーションの名前を入力します。[次へ] をクリックします。

5 [テーブル] タブで、[使用可能なテーブル] のリストからテーブルを 1 つ選択します。[追加] をクリックします。選択したテーブルが右側の [選択したテーブル] のリストに追加されます。

6 [WHERE 句] タブでテーブルを選択し、下側のボックスに探索条件を入力します。

7 [完了] をクリックします。

❖ WHERE 句を使用してパブリケーションを作成するには、次の手順に従います (SQL の場合)。

1 DBA 権限のあるユーザとしてリモート・データベースに接続します。

2 パブリケーション対象のテーブルと WHERE 条件を含む CREATE PUBLICATION 文を実行します。

例

次の文は、status カラムの中で active とマークされた顧客を対象として、customer テーブルの id、company_name、city、state カラムをパブリッシュするパブリケーションを作成します。

```
CREATE PUBLICATION pub_customer (
     TABLE customer (
         id,
         company_name,
         city,
         state )
     WHERE status = 'active'
 )
```

この場合、status カラム自体はパブリッシュされません。パブリッシュ対象外のすべてのローにはデフォルト値を設定しておきます。設定しないと、統合データベースから挿入用にダウンロードするときに、エラーが発生します。

次の例は、単一のアーティクルで構成され、営業担当者 ID 番号 856 の受注情報が含まれるパブリケーションを作成します。

```
CREATE PUBLICATION pub_orders_samuel_singer (
     TABLE sales_order WHERE sales_rep = 856
 )
```

詳細については、『ASA SQL リファレンス・マニュアル』>「CREATE PUBLICATION 文」を参照してください。CREATE PUBLICATION 文では、SUBSCRIBE BY 句を使用できます。SQL Remote を使用して、選択したローのみをパブリッシュするときに、この句を使用できます。ただし、Mobile Link 同期では無視されます。

既存のパブリケーションの変更

パブリケーションを作成してから、アーティクルの追加、修正、削除などの変更を加えたり、パブリケーションの名前を変更したりできます。アーティクルを修正する場合は、そのアーティクル全体の仕様を入力してください。

Sybase Central を使用するか、ALTER PUBLICATION 文によって、これらのタスクを実行できます。

注意

- DBA 権限を持つユーザか、パブリケーションの所有者だけがパブリケーションを変更できます。
- 変更には十分注意してください。Mobile Link 設定の実行中にパブリケーションを変更すると、エラーが発生し、データが失われることがあります。

❖ 既存のパブリケーションまたはアーティクルのプロパティを修正するには、次の手順に従います (Sybase Central の場合)。

1. パブリケーションを所有するユーザ、または DBA 権限を持つユーザとしてリモート・データベースに接続します。
2. 左ウィンドウ枠で、パブリケーションまたはアーティクルをクリックします。プロパティは、右ウィンドウ枠に表示されます。
3. 必要なプロパティを設定します。

❖ アーティクルを追加するには、次の手順に従います (Sybase Central の場合)。

1. Adaptive Server Anywhere プラグインを使用して、パブリケーションを所有するユーザ、または DBA 権限を持つユーザとしてリモート・データベースに接続します。
2. [パブリケーション] フォルダを開きます。
3. パブリケーションを選択します。
4. [ファイル] メニューから [新規] － [アーティクル] を選択します。[新しいアーティクルの作成] ウィザードが表示されます。
5. [アーティクル作成] ウィザードで、次の作業を実行します。
   - 最初のページでテーブルを選択する。
   - 次のページでカラム数を選択する。
   - 最後のページで WHERE 句を入力する (必要な場合)。
6. [完了] をクリックすると、アーティクルが作成されます。

❖ アーティクルを削除するには、次の手順に従います (Sybase Central の場合)。

1. Adaptive Server Anywhere プラグインを使用して、パブリケーションを所有するユーザ、または DBA 権限を持つユーザとしてデータベースに接続します。
2. [パブリケーション] フォルダを開きます。
3. パブリケーションをクリックします。
4. 右ウィンドウ枠で、削除するアーティクルを右クリックし、ポップアップ・メニューから [削除] を選択します。

❖ 既存のパブリケーションを修正するには、次の手順に従います (SQL の場合)。

1. パブリケーションを所有するユーザ、または DBA 権限を持つユーザとしてリモート・データベースに接続します。
2. DBA 権限を使用してデータベースに接続します。
3. ALTER PUBLICATION 文を実行します。

例

- 次の文は、customer テーブルを pub_contact パブリケーションに追加します。

```
ALTER PUBLICATION pub_contact (
     ADD TABLE customer
  )
```

『ASA SQL リファレンス・マニュアル』> 「ALTER PUBLICATION 文」も参照してください。

パブリケーションの削除

Sybase Central または DROP PUBLICATION 文のいずれかを使用して、パブリケーションを削除できます。パブリケーションに接続されたサブスクリプションをすべて削除してから、パブリケーションを削除してください。

パブリケーションを削除するには、DBA 権限が必要です。

❖ パブリケーションを削除するには、次の手順に従います (Sybase Central の場合)。

1. Adaptive Server Anywhere プラグインを使用して、DBA 権限のあるユーザとしてリモート・データベースに接続します。
2. [パブリケーション] フォルダを開きます。
3. 対象のパブリケーションを右クリックし、ポップアップ・メニューから [削除] を選択します。

❖ パブリケーションを削除するには、次の手順に従います (SQL の場合)。

1. DBA 権限のあるユーザとしてリモート・データベースに接続します。
2. DROP PUBLICATION 文を実行します。

例

次の文は、パブリケーション pub_orders を削除します。

```
DROP PUBLICATION pub_orders
```

『ASA SQL リファレンス・マニュアル』> 「DROP PUBLICATION 文」も参照してください。

Mobile Link ユーザの作成

Mobile Link ユーザ名によって、リモート・データベースがユニークに識別されます。つまり、Mobile Link 同期サーバにアクセスしようとするクライアントを識別し、オプションで認証するために使用されます。

Mobile Link ユーザは、データベース・ユーザとは異なります。データベース・ユーザ名と一致する Mobile Link ユーザ名を作成することはできますが、Mobile Link も Adaptive Server Anywhere も、名前の一致の影響は受けません。

統合データベースに Mobile Link ユーザを追加する方法については、「Mobile Link ユーザの概要」10 ページを参照してください。

リモート・データベースへの Mobile Link ユーザの追加

この項では、リモート・データベースに Mobile Link ユーザ名を追加する方法について説明します。パスワードなど、Mobile Link ユーザのプロパティを指定する方法については、「Mobile Link ユーザのプロパティの設定」93 ページを参照してください。

❖ Mobile Link ユーザをリモート・データベースに追加するには、次の手順に従います (Sybase Central の場合)。

1. Adaptive Server Anywhere プラグインから、DBA 権限のあるユーザとしてデータベースに接続します。
2. [Mobile Link ユーザ] フォルダをクリックします。
3. [ファイル] メニューから [新規] － [Mobile Link ユーザ] を選択します。[新しい Mobile Link ユーザの作成] ウィザードが表示されます。
4. Mobile Link ユーザ名を入力します。このユーザ名は、同期中に Mobile Link 同期サーバに提供されます。

Mobile Link ユーザ名はリモート・データベースをユニークに識別します。このため、Mobile Link ユーザ名は同期システム内でユニークである必要があります。

5　[完了] をクリックします。

❖ Mobile Link ユーザをリモート・データベースに追加するには、次の手順に従います (SQL の場合)。

1　DBA 権限のあるユーザとしてデータベースに接続します。

2　CREATE SYNCHRONIZATION USER 文を実行します。Mobile Link ユーザ名はリモート・データベースをユニークに識別します。このため、Mobile Link ユーザ名は同期システム内でユニークである必要があります。

次は、Mobile Link ユーザ SSinger を追加する例です。

```
CREATE SYNCHRONIZATION USER SSinger
```

Mobile Link ユーザのプロパティは、CREATE SYNCHRONIZATION USER 文の一部として指定したり、別個に ALTER SYNCHRONIZATION USER 文で指定したりします。

詳細については、『ASA SQL リファレンス・マニュアル』>「CREATE SYNCHRONIZATION USER 文 [Mobile Link]」を参照してください。

Mobile Link ユーザのプロパティの設定

リモート・データベースでは、それぞれの Mobile Link ユーザにプロパティを指定できます。

プロパティには、次の 2 種類があります。

- **オプション**　dbmlsync コマンド・ラインで指定できます。dbmlsync コマンド・ライン・オプションの完全なリストは、「Adaptive Server Anywhere クライアントの同期パラメータ」123 ページを参照してください。

- **拡張オプション**　拡張オプションは、コマンド・ラインで指定、データベースに格納、sp_hook_dbmlsync_set_extended_options イベント・フックで指定できます。

  拡張オプションのリストは、「dbmlsync 拡張オプション」135 ページを参照してください。

❖ データベースに Mobile Link 拡張オプションを保存するには、次の手順に従います (Sybase Central の場合)。

1. Adaptive Server Anywhere プラグインから、DBA 権限のあるユーザとしてデータベースに接続します。
2. [Mobile Link ユーザ] フォルダを開きます。
3. Mobile Link ユーザ名を右クリックし、ポップアップ・メニューから [プロパティ] を選択します。
4. 必要に応じてプロパティを変更します。

❖ データベースに Mobile Link 拡張オプションを保存するには、次の手順に従います (SQL の場合)。

1. DBA 権限のあるユーザとしてデータベースに接続します。
2. ALTER SYNCHRONIZATION SUBSCRIPTION 文を実行します。

   次は、Mobile Link ユーザ SSinger の拡張オプションをデフォルト値に変更する例です。

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

   詳細については、『ASA SQL リファレンス・マニュアル』>「ALTER SYNCHRONIZATION USER 文 [Mobile Link]」を参照してください。

   Mobile Link ユーザ名を作成するときに、プロパティを指定することも可能です。

詳細については、『ASA SQL リファレンス・マニュアル』>「CREATE SYNCHRONIZATION USER 文 [Mobile Link]」を参照してください。

❖ コマンド・ラインで Mobile Link ユーザ・プロパティを設定するには、次の手順に従います。

- dbmlsync の起動時に dbmlsync オプションを指定できます。

  詳細については、「Adaptive Server Anywhere クライアントの同期パラメータ」123 ページを参照してください。

❖ クライアント・イベント・フックを使用して Mobile Link ユーザ・プロパティを指定するには、次の手順に従います。

- 次の同期の動作はプログラムを使用してカスタマイズできます。

  詳細については、「sp_hook_dbmlsync_set_extended_options」284 ページを参照してください。

拡張オプションと接続パラメータの優先順位

CREATE/ALTER SYNCHRONIZATION USER 文と CREATE/ALTER SYNCHRONIZATION SUBSCRIPTION 文を使用すると、拡張オプションと接続パラメータをデータベースに格納し、サブスクリプション、ユーザ、またはパブリケーションに関連付けることができます。dbmlsync ユーティリティは、この情報をデータベースから読み込みます。

*注意：*パブリケーションのオプションを指定するには、CREATE SYNCHRONIZATION SUBSCRIPTION 文で FOR 句を省略します。

データベースとコマンド・ラインの両方で拡張オプションが指定されている場合は、オプション文字列が結合されます。オプションが競合して指定された場合、dbmlsync は次に示すリストに従って競合を解決します。このリストで先に指定されたオプションが、後に指定されたオプションより優先されます。

1. sp_hook_dbmlsync_set_extended_options イベント・フックで指定されているオプション。
2. 拡張オプションではないコマンド・ラインで指定されたオプション (たとえば、`-ds` は `-e "ds=off"` を上書きします)。
3. コマンド・ラインで dbmlsync -eu オプションを使用して指定されているオプション。
4. コマンド・ラインで dbmlsync -e オプションを使用して指定されているオプション。
5. (SQL 文または Sybase Central を使用して) サブスクリプション定義で保存されているオプション。
6. (SQL 文または Sybase Central を使用して) ユーザ定義で保存されているオプション。
7. (SQL 文または Sybase Central を使用して) パブリケーション定義で保存されているオプション。

接続の TYPE または ADDRESS が複数箇所に指定されている場合は、上のリストに従って、最上位の優先順位を持つオプション指定によって他の指定が上書きされます。

dbmlsync 接続パラメータ

dbmlsync 接続情報には、サーバとの通信に使用するプロトコル、Mobile Link 同期サーバのアドレスなどの接続パラメータが含まれます。

詳細については、次の項を参照してください。

- 「CommunicationType (ctp) 拡張オプション」143 ページ
- 「CommunicationAddress (adr) 拡張オプション」138 ページ
- 「-c オプション」132 ページ

Mobile Link ユーザの削除

Mobile Link ユーザは、そのユーザ用のサブスクリプションをすべて削除した後で、リモート・データベースから削除してください。

❖ Mobile Link ユーザをリモート・データベースから削除するには、次の手順に従います (Sybase Central の場合)。

1. Adaptive Server Anywhere プラグインから、DBA 権限のあるユーザとしてデータベースに接続します。
2. [Mobile Link ユーザ] フォルダで、対象の Mobile Link ユーザを探します。
3. Mobile Link ユーザを右クリックし、ポップアップ・メニューから [削除] を選択します。

❖ Mobile Link ユーザをリモート・データベースから削除するには、次の手順に従います (SQL の場合)。

1. DBA 権限のあるユーザとしてデータベースに接続します。
2. DROP SYNCHRONIZATION SUBSCRIPTION 文を実行します。

   次は、データベースから Mobile Link ユーザ SSinger を削除する例です。

   ```
   DROP SYNCHRONIZATION USER SSinger
   ```

   詳細については、『ASA SQL リファレンス・マニュアル』>「DROP SYNCHRONIZATION USER 文 [Mobile Link]」を参照してください。

Mobile Link 同期ユーザのサブスクリプションの作成

設定を完了するには、1 つまたは複数の既存のパブリケーションに対して、少なくとも 1 人の Mobile Link ユーザのサブスクリプションを作成します。

パブリケーションの作成については、「データのパブリッシュ」82 ページを参照してください。Mobile Link ユーザの作成については、「Mobile Link ユーザの作成」92 ページを参照してください。

サブスクリプションと同期サブスクリプションの違い

サブスクリプション (CREATE SUBSCRIPTION 文) を、同期サブスクリプション (CREATE SYNCHRONIZATION SUBSCRIPTION 文) と混同しないでください。サブスクリプションは SQL Remote にのみ機能します。サブスクリプションによって、パブリケーションと、リモート権限が付与されているデータベース・ユーザとの関係が作成されます。同期サブスクリプションは Mobile Link で使用され、パブリケーションと *Mobile Link* ユーザとの関係を作成します。

同期サブスクリプションは、特定の Mobile Link ユーザをパブリケーションとリンクします。また、同期に必要なその他の情報も指定できます。たとえば、Mobile Link サーバのアドレスや、同期サブスクリプションに必要な他のオプションを指定できます。特定の同期サブスクリプションの値によって、Mobile Link ユーザに設定された値が上書きされます。

同期サブスクリプションは、Mobile Link Adaptive Server Anywhere リモート・データベース内でのみ必要です。サーバ論理は、統合データベース内の Mobile Link システム・テーブルに格納されている同期スクリプトによって実装されます。

単一の Adaptive Server Anywhere データベースは複数の Mobile Link 同期サーバと同期できます。複数のサーバとの同期を行うには、サーバごとに異なる Mobile Link ユーザを作成します。

例

Adaptive Server Anywhere サンプル・データベース内の customer テーブルと sales_order テーブルの同期を行うには、次の文を使用します。

1. 最初に、customer テーブルと sales_order テーブルをパブリッシュします。パブリケーション名として testpub を指定します。

```
CREATE PUBLICATION testpub
     (TABLE customer, TABLE sales_order)
```

2. 次に Mobile Link ユーザを作成します。この場合、Mobile Link ユーザは demo_ml_user です。

```
CREATE SYNCHRONIZATION USER demo_ml_user
```

3. 処理を完了するために、パブリケーションに対するユーザのサブスクリプションを作成します。

```
CREATE SYNCHRONIZATION SUBSCRIPTION TO testpub
     FOR demo_ml_user
     TYPE tcpip
     ADDRESS 'host=localhost;port=2439;'
     OPTION sv='version1'
```

Mobile Link サブスクリプションの変更

Sybase Central を使用するか、ALTER SYNCHRONIZATION SUBSCRIPTION 文を使用すると、同期サブスクリプションを変更できます。構文は CREATE SYNCHRONIZATION SUBSCRIPTION 文に似ていますが、より簡単に追加、修正、削除できるように拡張オプションが用意されています。

❖ 同期サブスクリプションを変更するには、次の手順に従います (Sybase Central の場合)。

1. DBA 権限のあるユーザとしてデータベースに接続します。
2. [Mobile Link ユーザ] フォルダを開きます。
3. 対象のユーザをクリックします。プロパティが右ウィンドウ枠に表示されます。
4. 右ウィンドウ枠で、[同期サブスクリプション] タブをクリックします。変更するサブスクリプションを右クリックし、ポップアップ・メニューから [プロパティ] を選択します。

5 必要に応じてプロパティを変更します。

❖ 同期サブスクリプションを変更するには、次の手順に従います (SQL の場合)。

1 DBA 権限のあるユーザとしてデータベースに接続します。

2 ALTER SYNCHRONIZATION SUBSCRIPTION 文を実行します。

詳細については、『ASA SQL リファレンス・マニュアル』> 「ALTER SYNCHRONIZATION USER 文 [Mobile Link]」を参照してください。

Mobile Link サブスクリプションの削除

Sybase Central または DROP SYNCHRONIZATION SUBSCRIPTION 文のいずれかを使用して、同期サブスクリプションを削除できます。

同期サブスクリプションを削除するには、DBA 権限が必要です。

❖ 同期サブスクリプションを削除するには、次の手順に従います (Sybase Central の場合)。

1 DBA 権限のあるユーザとしてデータベースに接続します。

2 [Mobile Link ユーザ] フォルダを開きます。

3 Mobile Link ユーザを選択します。

4 対象のサブスクリプションを右クリックし、ポップアップ・メニューから [削除] を選択します。

❖ 同期サブスクリプションを削除するには、次の手順に従います (SQL の場合)。

1 DBA 権限のあるユーザとしてデータベースに接続します。

2 DROP SYNCHRONIZATION SUBSCRIPTION 文を実行します。

例

次の文は、パブリケーション pub_orders に対する Mobile Link ユーザ jsmith の同期サブスクリプションを削除します。

```
DROP SYNCHRONIZATION SUBSCRIPTION
 FOR jsmith TO pub_orders
```

『ASA SQL リファレンス・マニュアル』> 「DROP SYNCHRONIZATION SUBSCRIPTION 文 [Mobile Link]」も参照してください。

同期の開始

Mobile Link 同期を開始するのは、常にクライアントです。Adaptive Server Anywhere クライアントの場合は、dbmlsync ユーティリティを実行することで同期処理が開始されます。このユーティリティは Adaptive Server Anywhere リモート・データベースへの接続と同期を行います。

dbmlsync コマンド・ラインで -c オプションを使用して接続パラメータを指定できます。これらのパラメータは、リモート・データベース用です。接続パラメータを指定しないと、[接続] ダイアログが表示され、必要な接続パラメータと起動オプションを指定するように要求されます。

リモート・データベース内の同期サブスクリプション内で設定されている接続パラメータによって、適切な Mobile Link 同期サーバが検索されます。

dbmlsync のパーミッション

dbmlsync でデータベースに接続するときは、処理中のすべての変更を適用するパーミッションが必要です。dbmlsync コマンド・ラインには、この接続用のパスワードが含まれます。このため、セキュリティ上の問題が発生する可能性があります。

セキュリティの問題を回避するには、ユーザ (DBA 以外) に REMOTE DBA 権限を付与し、このユーザ ID を dbmlsync 接続文字列に使用します。REMOTE DBA 権限を付与されたユーザ ID が DBA 権限を持つのは、dbmlsync ユーティリティから接続が確立された場合のみです。同じユーザ ID を使用する他の接続には、特別な権限は付与されません。

複数の Mobile Link 同期ユーザ

通常、各リモート・データベース内の Mobile Link 同期ユーザは 1 人だけです。この場合、dbmlsync コマンド・ラインで Mobile Link ユーザ名を指定する必要はありません。ただし、リモート・データベースに複数の Mobile Link ユーザが含まれている場合は、-u コマンド・ライン・オプションを使用して、どの Mobile Link 同期ユーザを同期するかを指定してください。

```
dbmlsync -c "dsn=remote;uid=syncuser" -u mluser
```

同様に、-mp オプションでユーザのパスワードを指定したり、-mn オプションで新規パスワードを指定してパスワードを変更したりできます。これらは、Mobile Link 同期サーバに使用されるユーザ ID とパスワードであり、リモート・データベースへの接続に使用されるユーザ ID とパスワードとは異なる場合があります。

同期のカスタマイズ

Mobile Link には、同期処理をカスタマイズするための拡張オプションがいくつか用意されています。拡張オプションは、パブリケーション、ユーザ、またはサブスクリプションに設定できます。また、拡張オプションの値は、dbmlsync コマンド・ラインでオプションを使用して上書きできます。

拡張オプションの完全なリストは、「dbmlsync 拡張オプション」135 ページを参照してください。

❖ dbmlsync コマンド・ラインで拡張オプションを上書きするには、次の手順に従います。

- -e または -eu dbmlsync オプションで、dbmlsync の拡張オプションの値を *option-name=value* の形式で指定します。次に例を示します。

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

❖ サブスクリプション、パブリケーション、またはユーザの拡張オプションを設定するには、次の手順に従います。

- Adaptive Server Anywhere リモート・データベース内で、CREATE SYNCHRONIZATION SUBSCRIPTION 文または CREATE SYNCHRONIZATION USER 文にオプションを追加します。

パブリケーションに拡張オプションを追加する場合は、少し異なります。パブリケーションに拡張オプションを追加するには、ALTER/CREATE SYNCHRONIZATION SUBSCRIPTION 文で FOR 句を省略します。

例

次の文は、拡張オプションを使用する同期サブスクリプションを作成します。拡張オプションによって、アップロード・ストリームを準備するキャッシュ・サイズを 3 MB に設定し、アップロードのインクリメント・サイズを 3 KB に設定します。

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

オプション値は一重引用符で囲むことができますが、オプション名は引用符で囲まないでください。

トランザクション・ログ・ファイル

アップロード・ストリームの準備において、dbmlsync ユーティリティは、正常に行われた最新の同期よりも後に書き込まれたすべてのトランザクション・ログにアクセスする必要があります。ただし、通常、ログ・ファイルは、定期的なデータベース管理作業の中でトランケートされ、名前が変更されます。その場合は、記述されている変更内容がすべて正常に同期されるまで、古いログ・ファイルの名前を変更し、別のディレクトリに保存してください。

dbmlsync コマンド・ラインで、名前が変更されたログ・ファイルが格納されているディレクトリを指定できます。前回の同期の後で作業ログ・ファイルのトランケートと名前の変更が行われていない場合、または名前が変更されたログ・ファイルがあるディレクトリから dbmlsync を実行する場合は、このパラメータを省略できます。

詳細については、『ASA データベース管理ガイド』> 「バックアップとデータ・リカバリ」を参照してください。

例

古いログ・ファイルがディレクトリ *c:¥oldlogs* に格納されているとします。次のコマンドを使用してリモート・データベースを同期することができます。

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

古いログ・ディレクトリへのパスは、コマンド・ラインに最後の引数として指定してください。

同期中の同時実行性

同期の整合性を確保するために、dbmlsync ではアップロード・ストリームが構築されてからダウンロードが適用されるまでの間に、ダウンロード・ストリームのローが修正されないようにする必要があります。

Windows CE 以外のプラットフォームでは、デフォルトの動作として dbmlsync は同期中のパブリケーションで指定されているすべてのテーブルの共有ロックを取得します。Windows CE では、デフォルトの動作として dbmlsync は排他ロックを取得します。dbmlsync は、アップロード・ストリームの構築を開始する前にロックを取得し、ダウンロードが適用されるまでそのロックを保持します。

ロックの詳細については、『ASA SQL ユーザーズ・ガイド』> 「ロックの種類」を参照してください。

このロック動作は、次のオプションを使用してカスタマイズできます。

- -d オプション
- LockTables オプション

-d オプション

このロック・メカニズムを使用しているときに、データベースに別の接続が存在し、その接続に同期テーブルに対するロックがある場合は、同期が失敗します。別のロックが存在しても、同期がすぐに行われるようにする場合は、dbmlsync で -d オプションを使用します。このオプションを指定すると、同期に影響するロックのある接続はデータベースによって削除されるため、同期を進行できます。削除された接続のコミットされていない変更は、ロールバックされます。

詳細については、「-d オプション」133 ページを参照してください。

LockTables オプション

データの整合性を保持する別の方法は、LockTables 拡張オプションを OFF に設定することです。OFF にすると、アーティクルのテーブルがロックされるのを防ぎます。こうすると、dbmlsync はアップロード・

ストリームの構築後に修正されたローをすべて追跡します。ダウンロードを受信しても、そのローが修正されている場合はダウンロードは適用されません。この場合、dbmlsync は同期のリトライを行います。リトライは、新しいダウンロードの競合が検出されない限り正常に実行されます。

詳細については、「LockTables (lt) 拡張オプション」157 ページを参照してください。

競合が検出された場合は、ダウンロード・フェーズがキャンセルされ、新しい変更が上書きされないようにダウンロード操作がロールバックされます。次に、dbmlsync ユーティリティはアップロード手順を含む同期を再試行します。今回はローが同期処理の最初に処理されており、アップロード・ストリームにこのローが含まれているため、このローを失うことはありません。

デフォルトでは、dbmlsync は正常に実行されるまで同期のリトライを行います。リトライの回数を制限するには、拡張オプション ConflictRetries を使用します。ConflictRetries を -1 に設定すると、正常に実行されるまで dbmlsync によってリトライが実行されます。これを正の整数に設定すると、dbmlsync は指定した回数以内でリトライを実行します。

詳細については、「ConflictRetries (cr) 拡張オプション」144 ページを参照してください。

アプリケーションからの同期の開始

リモート・ユーザに別個の実行ファイルを提供するのではなく、アプリケーションに dbmlsync の機能を組み込むことができます。

このようにするには、次の 2 つの方法があります。

- dbmlsync 統合コンポーネントを使用します。

  詳細については、「dbmlsync 統合コンポーネント」297 ページを参照してください。

- DLL を呼び出せる言語で開発し、C または C# でプログラミングを行っている場合は、SQL Anywhere ディレクトリの *h* サブディレクトリにある *dbtools.h* ヘッダ・ファイルをインクルードでき

ます。このファイルには、a_sync_db 構造体と DBSynchronizeLog 関数の記述があり、dbmlsync 機能をアプリケーションに追加するときに使用します。この解決方法は、Windows、UNIX、Linux、Macintosh など、サポート対象となっているすべてのプラットフォームに使用できます。

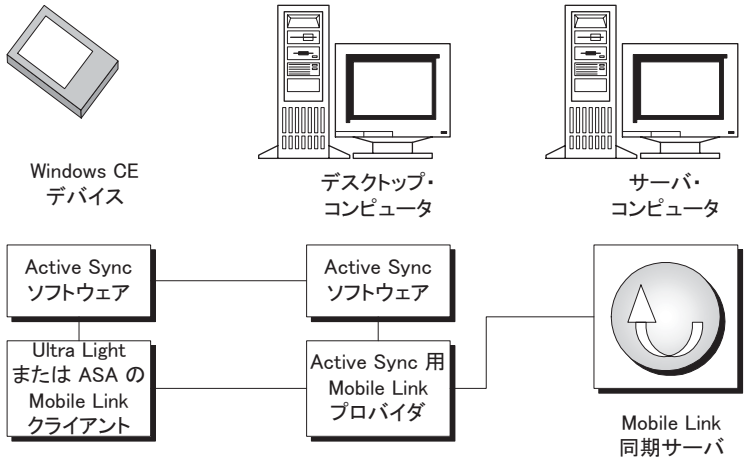
詳細については、次の項を参照してください。

- 「dbmlsync の DBTools インタフェース」333 ページ
- 『ASA プログラミング・ガイド』> 「DBSynchronizeLog 関数」
- 『ASA プログラミング・ガイド』> 「a_sync_db 構造体」

ActiveSync 同期の使用

ActiveSync は、Microsoft Windows CE ハンドヘルド・デバイス用の同期ソフトウェアです。Adaptive Server Anywhere Mobile Link クライアントでは、ActiveSync バージョン 3.1 または 3.5 を使用できます。

ActiveSync は、Windows CE デバイスとデスクトップ・コンピュータ間の同期を制御します。ActiveSync 用の Mobile Link プロバイダは、次の図のように Mobile Link 同期サーバへの同期を制御します。



Adaptive Server Anywhere クライアント用の ActiveSync 同期を設定するには、次の手順に従います。

- Adaptive Server Anywhere リモート・データベースを ActiveSync 同期用に設定する。

  「ActiveSync 用の Adaptive Server Anywhere リモート・データベースの設定」109 ページを参照してください。

- ActiveSync 用 Mobile Link プロバイダをインストールする。

  「ActiveSync 用 Mobile Link プロバイダのインストール」110 ページを参照してください。

- Adaptive Server Anywhere クライアントを、ActiveSync で使用できるように登録する。

  「ActiveSync 用 Adaptive Server Anywhere クライアントの登録」112 ページを参照してください。

ActiveSync 同期を使用する場合は、ActiveSync ソフトウェアから同期を開始します。ActiveSync 用の Mobile Link プロバイダは、dbmlsync を起動するか、スケジュール文字列でのスケジュールに従ってスリープ中の dbmlsync をウェイクアップできます。

リモート・データベース内で遅延フックを使用して dbmlsync をスリープ・モードに設定することもできますが、ActiveSync 用の Mobile Link プロバイダは、このステータスからは同期を開始できません。

同期スケジュールの詳細については、「同期のスケジュール」115 ページを参照してください。

ActiveSync 用の Adaptive Server Anywhere リモート・データベースの設定

❖ Adaptive Server Anywhere リモート・データベースを ActiveSync 用に設定するには、次の手順に従います。

1. 同期タイプとして ActiveSync を選択します。

   同期タイプは、同期パブリケーション、同期ユーザ、または同期サブスクリプション用に設定できます。それぞれの設定方法は似ています。ここでは、典型的な CREATE SYNCHRONIZATION USER 文の一部を示します。

   ```
   CREATE SYNCHRONIZATION USER SSinger
    TYPE ActiveSync
    ...
   ```

2. ADDRESS 句を使用して、ActiveSync 用 Mobile Link プロバイダと Mobile Link 同期サーバ間の通信を指定します。

HTTP または TCP/IP 同期の場合は、CREATE SYNCHRONIZATION USER または CREATE SYNCHRONIZATION SUBSCRIPTION 文の ADDRESS 句によって、Mobile Link クライアントとサーバ間の通信を指定します。ActiveSync の場合、通信は 2 段階で発生します。つまり、デバイス上の dbmlsync ユーティリティからデスクトップ・マシン上の ActiveSync 用 Mobile Link プロバイダへの通信が発生してから、デスクトップ・マシンから Mobile Link 同期サーバへの通信が発生します。ADDRESS 句では、ActiveSync 用 Mobile Link プロバイダと Mobile Link 同期サーバ間の通信を指定します。

次の文は、マシン kangaroo 上の Mobile Link 同期サーバへの TCP/IP 通信を指定します。

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

詳細については、『ASA SQL リファレンス・マニュアル』>「CREATE SYNCHRONIZATION USER 文 [Mobile Link]」を参照してください。

ActiveSync 用 Mobile Link プロバイダのインストール

インストール・ユーティリティ (*dbasinst.exe*) を使用して、ActiveSync 用 Mobile Link プロバイダをインストールしてから、Adaptive Server Anywhere Mobile Link クライアントを ActiveSync 用に登録します。

Adaptive Server Anywhere for Windows CE セットアップ・プログラムによって、ActiveSync 用 Mobile Link プロバイダがインストールされます。Adaptive Server Anywhere for Windows CE をインストールする場合、この項で説明する手順を実行する必要はありません。

ActiveSync 用 Mobile Link プロバイダをインストールしたら、各アプリケーションを個別に登録してください。作業手順については、「ActiveSync 用 Adaptive Server Anywhere クライアントの登録」112 ページを参照してください。

❖ ActiveSync 用 Mobile Link プロバイダをインストールするには、次の手順に従います。

1. 使用中のマシンに ActiveSync ソフトウェアがインストールしてあり、Windows CE デバイスが接続されていることを確認します。

2. 次のコマンドを入力して、Mobile Link プロバイダをインストールします。

   `dbasinst -k` *desk-path* `-v` *dev-path*

   desk-path はプロバイダのデスクトップ・コンポーネント (*dbasdesk.dll*) のロケーション、*dev-path* はデバイス・コンポーネント (*dbasdev.dll*) のロケーションです。

   使用しているコンピュータに SQL Anywhere がインストールされている場合、*dbasdesk.dll* は SQL Anywhere ディレクトリの *win32* または *win64* サブディレクトリにあります。また、*dbasdev.dll* は *CE* サブディレクトリのプラットフォーム固有のディレクトリにあります。-v または -k を省略すると、デフォルトでこれらのディレクトリが検索されます。

   リモート・プロバイダが開けないというメッセージが表示された場合は、デバイスのソフト・リセットを実行し、コマンドを繰り返します。

   詳細については、「ActiveSync プロバイダ・インストール・ユーティリティ」34 ページを参照してください。

3. マシンを再起動します。

   マシンを再起動すると、ActiveSync で新しいプロバイダが認識されます。

4. Mobile Link プロバイダを有効にします。

   - [ActiveSync] ウィンドウで [オプション] をクリックします。

- リストにある [Mobile Link] 項目を有効にして [OK] をクリックし、Mobile Link プロバイダをアクティブにします。
- 登録されたアプリケーションのリストを表示するには、もう一度 [オプション] をクリックし、Mobile Link プロバイダを選択して [設定] をクリックします。

  アプリケーションの登録についての詳細は、「ActiveSync 用 Adaptive Server Anywhere クライアントの登録」112 ページを参照してください。

ActiveSync 用 Adaptive Server Anywhere クライアントの登録

ActiveSync で使用するアプリケーションを登録するには、ActiveSync プロバイダのインストール・ユーティリティを使用する方法と、ActiveSync ソフトウェア自体を使用する方法があります。この項では、ActiveSync ソフトウェアを使用する方法について説明します。

もう 1 つの方法については、「ActiveSync プロバイダ・インストール・ユーティリティ」34 ページを参照してください。

❖ Adaptive Server Anywhere クライアントを ActiveSync 用に登録するには、次の手順に従います。

1. ActiveSync 用 Mobile Link プロバイダがインストールされていることを確認します。

   詳細は、「ActiveSync 用 Mobile Link プロバイダのインストール」110 ページを参照してください。

2. デスクトップ・マシンで、ActiveSync ソフトウェアを起動します。
3. [ActiveSync] ウィンドウで [オプション] を選択します。
4. 情報タイプのリストから、[Mobile Link] を選択し、[設定] をクリックします。

5　[Mobile Link 同期] ダイアログで [新規] をクリックします。[プロパティ] ダイアログが表示されます。

6　アプリケーションについて次の情報を入力します。

- **[アプリケーション名]**　ActiveSync ユーザ・インタフェースに表示されるアプリケーションを識別する名前。
- **[クラス名]**　-wc オプションを使用して設定した、dbmlsync クライアントのクラス名。

  詳細については、「-wc オプション」195 ページを参照してください。
- **[パス]**　デバイス上の dbmlsync アプリケーションのロケーション。
- **[引数]**　ActiveSync が dbmlsync の起動時に使用するコマンド・ライン引数。

  dbmlsync は、2 つのモードのうちの 1 つを使用して開始します。

  - スケジューリング・オプションを指定すると、dbmlsync は停止モードに入ります。この場合、クラス名の設定と一致する値を指定した dbmlsync -wc オプションを使用します。

    詳細については、「-wc オプション」195 ページと「同期のスケジュール」115 ページを参照してください。
  - このように指定しないと、dbmlsync は停止モードに入りません。この場合、-k を使用して dbmlsync を停止します。

    詳細については、「-k オプション」181 ページを参照してください。

7　[OK] をクリックしてアプリケーションを登録します。

削除同期の一時停止

通常、Adaptive Server Anywhere は同期サブスクリプションのあるパブリケーションに属するテーブルまたはカラムへの変更を自動的に記録します。これらの変更は次の同期時に統合データベースにアップロードされます。

しかし、同期対象のデータからローを削除しても、変更がアップロードされないようにする必要がある場合もあります。この機能は、特別な修正のために使用できますが、自動同期機能の一部が無効化されるので、注意して使用してください。この方法は、download_delete_cursor スクリプトを使用して必要なローを削除する処理に代わる、実用的な代替手段です。

STOP SYNCHRONIZATION DELETE 文を実行すると、その接続に対してそれ以降に実施された削除操作は同期されません。この効果は、START SYNCHRONIZATION DELETE 文が実行されるまで継続します。この効果はネストしません。つまり、最初の STOP SYNCHRONIZATION DELETE 文の後に同じ文を実行してもさらなる効果はないということです。

❖ 接続を介して実行された削除のアップロードを一時停止するには、次の手順に従います。

1. 次の文を発行して削除の自動ロギングを停止します。

   ```
   STOP SYNCHRONIZATION DELETE
   ```

2. 必要に応じて、DELETE 文を使用して同期対象のデータからローを削除します。これまでの変更内容をコミットします。

3. 次の文を使用して削除のログを再開します。

   ```
   START SYNCHRONIZATION DELETE
   ```

削除されたローは Mobile Link 同期サーバに送られないため、統合データベースからは削除されません。

同期のスケジュール

dbmlsync をバッチ形式で実行して、同期終了後に停止するように設定する代わりに、dbmlsync を継続的に実行してあらかじめ決められた時間に同期を行うように、Adaptive Server Anywhere クライアントを設定することもできます。

同期スケジュールは、拡張オプションとして指定します。同期スケジュールを dbmlsync コマンド・ライン上で指定するか、同期ユーザ、同期サブスクリプション、または同期パブリケーション用にデータベースに同期スケジュールを格納できます。

拡張オプションについては、「dbmlsync 拡張オプション」135 ページまたは 「-eu オプション」180 ページを参照してください。スケジュールの設定方法についての詳細は、「Schedule (sch) 拡張オプション」164 ページを参照してください。

❖ 同期サブスクリプションにスケジュールを追加するには、次の手順に従います。

- 同期サブスクリプション内で Schedule 拡張オプションを設定します。次に例を示します。

```
CREATE SYNCHRONIZATION SUBSCRIPTION TO mypub
 FOR mluser
 ADDRESS 'host=localhost'
 OPTION schedule='weekday@11:30am-12:30pm'
```

  スケジュール構文については、「Schedule (sch) 拡張オプション」164 ページを参照してください。

  dbmlsync -is オプションを使用すると、スケジュールの指示を無効にし、直ちに同期を行うことができます。-is オプションを使用すると、dbmlsync はすべてのスケジュール情報を無視します。詳細については、「-is オプション」181 ページを参照してください。

❖ **dbmlsync コマンド・ラインでスケジュールを追加するには、次の手順に従います。**

- スケジュール拡張オプションを設定します。拡張オプションは -e または -eu で設定します。次に例を示します。

```
dbmlsync -e sch=weekday@11:30am-12:30pm ...
```

同期スケジュールがこのどちらかで指定された場合、dbmlsync は同期終了後も停止せず、継続して実行します。

停止

スケジュール・オプションを指定すると、dbmlsync は停止モードになります。停止とは、ログのスキャンに使用される時間を少なくする機能です。停止を実行してパフォーマンスを向上させるには、dbmlsync 拡張オプション HoverRescanThreshold を設定するか、dbmlsync ストアド・プロシージャ sp_hook_dbmlsync_log_rescan を使用します。

詳細については、「HoverRescanThreshold (hrt) 拡張オプション」153 ページと 「sp_hook_dbmlsync_log_rescan」271 ページ を参照してください。

Adaptive Server Anywhere バージョン 7 Mobile Link クライアント

Adaptive Server Anywhere 7.0 の Mobile Link クライアントは、旧式の SQL 文を使用して設定されていました。特に、パブリケーションとサブスクリプションの代わりに、同期定義を使用していました。以前の文は、サポートされなくなっています。これらの文には、次のいくつかの欠点があります。

1. 同期定義は、1 つのパブリケーションとそれに対する 1 つのサブスクリプションに相当します。複数のパブリケーションに対するサブスクリプションはサポートされていません。これに対して現在は、1 人の Mobile Link ユーザが、複数のパブリケーションに対するサブスクリプションを作成できるようになりました。これによって、すべてのデータを同期しなくても、その一部を同期することができます。

2. 一部のユーザにとって、旧バージョンの用語はわかりにくいものでした。たとえば、旧バージョンの場合、Adaptive Server Anywhere クライアントのコンテキストでは Mobile Link ユーザ名がサイトと呼ばれていました。現在は、Mobile Link ユーザを Mobile Link ユーザまたは同期ユーザと呼びます。

3. 新しい文は、Sybase のメッセージベースのレプリケーション・テクノロジである SQL Remote で使用されている文に似ています。

同期定義による、バージョン 7 のリモート・データベースにアップロードするデータの識別

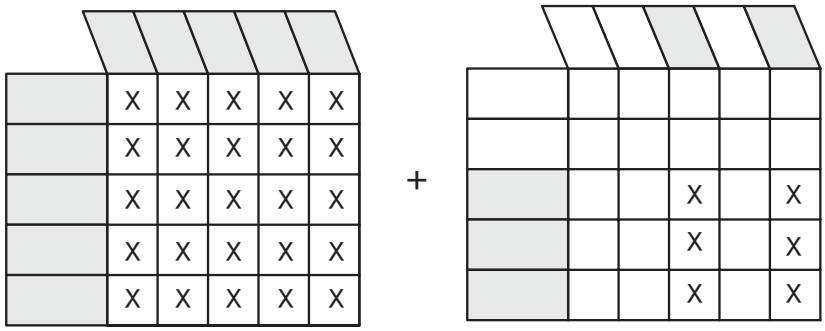
Adaptive Server Anywhere クライアント・データベース内のデータは、すべて同期するか、一部を同期するかを選択できます。テーブル全体または特定のカラムやローの同期を選択できます。

Adaptive Server Anywhere クライアント・データベースにある同期定義には、レプリケートするデータと適切な Mobile Link 同期サーバのロケーションが記述されています。

統合データベースに格納されている同期スクリプトは、アップロードされたローの処理方法と、リモート・データベースにダウンロードされるローを制御します。これらのスクリプトはリモート・データベースのタイプには依存しません。

1 つの同期定義は、複数のデータベース・テーブルからのデータを含むことがあります。各テーブルから同期定義に提供されるデータを、*アーティクル*と呼びます。各アーティクルは、1 つのテーブル全体か、1 つのテーブル内のローとカラムのサブセットで構成されます。

2 つのテーブルの同期定義



アーティクル 1:
テーブル A 全体

アーティクル 2:テーブル B
の一部のローとカラム

リモート・データベースの同期

リモート・データベースを設定した後は、2 つのデータベースを定期的に処理し、両方に同じ情報が含まれている状態にしてください。この同期処理は、dbmlsync コマンド・ライン・ユーティリティを使用して実行します。

同期されるテーブルの変更

いったん同期定義に追加した後は、テーブルを変更しないでください。テーブルを変更すると、同期処理が妨げられます。変更が必要な場合は、同期の実行直後に行ってください。

ALTER 文を確実に同期の直後に実行する唯一の方法は、この文をスクリプトに挿入し、そのスクリプトを実行するときに *dbmlsync* コマンド・ライン・ユーティリティで `-I` オプションを使用することです。

Ultra Light クライアントとの比較

Ultra Light アプリケーションを Mobile Link クライアントとして使えるように開発していれば、次の情報が役に立つ場合があります。同期定義の要素の多くは、Ultra Light にも対応します。

Adaptive Server Anywhere 9.0 クライアント	Adaptive Server Anywhere 7.0 クライアント	Ultra Light クライアント	Mobile Link 同期サーバ
Mobile Link 同期ユーザ	サイト	ユーザ名	Mobile Link ユーザ
タイプ	タイプ	ストリーム	接続タイプ
アドレス	アドレス	接続パラメータ	サーバのアドレス
スクリプト・バージョン	スクリプト・バージョン	バージョン	スクリプト・バージョン
パブリケーション	リモート・データベース内の定義の一部、またはリファレンス・データベース内のテンプレートの一部	*なし* - すべてのテーブルが同期される	パブリケーション
サブスクリプション	リモート・データベース内の定義の一部、またはリファレンス・データベース内のサイトの一部	*なし*	*なし*

同期定義の作成

同期定義は、バージョン 7.0 データベースのオブジェクトであり、特定の Mobile Link 同期サーバと同期される Adaptive Server Anywhere リモート・データベース内のデータを記述しています。Adaptive Server Anywhere 9.0 以降を使用している場合は、代わりにパブリケーションと同期サブスクリプションを使用してください。

詳細については、「リモート・データベースの作成」76 ページを参照してください。

同期定義は、Adaptive Server Anywhere 7.0 リモート・データベース内でのみ使用してください。Mobile Link 統合サーバはスクリプトを使って設定します。

同期定義には次のような情報を定義します。

- **名前**　リモート・データベース内でのみ認識される同期定義名。
- **サイト**　特定の Mobile Link クライアントをユニークに識別する名前。
- **タイプ**　Mobile Link 同期サーバとの通信に使用するストリームのタイプ。
- **アドレス**　Mobile Link 同期サーバへの接続に必要なパラメータ。
- **スクリプト・バージョン**　このクライアントと同期するときに Mobile Link 同期サーバが使用する同期スクリプトのバージョン。
- **アーティクル**　同期するデータの説明。同期対象としては、テーブル全体または特定のカラムやローだけを選択できます。

次の文は、サイト demo_sync_site と同期するデータを定義する同期定義 testpub を作成します。

```
CREATE SYNCHRONIZATION DEFINITION testpub
     SITE 'demo_sync_site'
     TYPE 'tcpip'
     ADDRESS 'host=localhost;port=2439;'
     OPTION sv='version1'
     (table People( person_id, fname, lname ),table
Pets);
```

この文では、以下のとおりになります。

- この同期定義の名前は testpub です。この名前はリモート・データベース内でのみ認識されます。
- 名前 demo_sync_site によって、Mobile Link 同期サーバがこのクライアントをユニークに識別します。この名前は、統合データベースにある ml_user Mobile Link システム・テーブルに入っていなければなりません。
- 同期は TCP/IP 接続を介して行われます。接続パラメータは ADDRESS 句の文字列に表示されます。

  TCP/IP 接続パラメータは、Mobile Link 同期サーバが現在のマシンのポート 2439 で受信していることを示します。同期されるのは、People テーブルのリストされたカラムのみです。option 句を

追加して、このクライアントからのデータを処理するときに、Mobile Link 同期サーバが *version1* の同期スクリプトを使うよう指定しています。このパラメータのデフォルト値は *default* です。カラムのリストもカッコで囲まれていることに注意してください。

- Mobile Link 同期サーバは、このクライアントと同期するときに、version1 の名前で識別される同期スクリプト・セットを使用します。このスクリプトのバージョン名は、統合データベースにある ml_script_version Mobile Link システム・テーブルに入っていなければなりません。
- *Pets* テーブルのすべてのカラムやローと *People* テーブルのリストされたカラムが同期されます。

複数のサーバとの同期

リモート・データベースを複数の Mobile Link 同期サーバと同期するには、複数の同期定義をリモート・データベース内に作成します。各同期定義にはユニークなサイト名を付けてください。これは、Mobile Link 同期サーバから見て、各定義は個別の論理クライアントだからです。

1 つのリモート・データベース内の同じデータを複数の Mobile Link 同期サーバと同期する処理は現在のところサポートされていません。

バージョン 8 以降で使用するための同期定義の書き換え

Adaptive Server Anywhere 7 データベースを Mobile Link クライアントとして使用するには、同期定義を使用して、アップロードするデータを識別します。バージョン 8.0 以降では、パブリケーションと同期サブスクリプションに書き換えることをおすすめします。

例

サンプル・データベースのテーブル Customer と Sales_Order を同期するとします。以前は、次のように同期定義を作成できました。

```
CREATE SYNCHRONIZATION DEFINITION testpub
     SITE 'demo_ml_user'
     TYPE 'tcpip'
     ADDRESS 'host=localhost;port=2439;'
     OPTION sv='version1'
     (TABLE Customer, TABLE Sales_Order);
```

今後は、代わりに次の手順を実行してください。

1. 最初に、Customer テーブルと Sales_Order テーブルをパブリッシュします。

```
CREATE PUBLICATION testpub
     (TABLE Customer, TABLE Sales_Order);
```

2. 次に、Mobile Link ユーザ用に、このパブリケーションに対するサブスクリプションを作成します。この場合、Mobile Link ユーザは demo_ml_user です。同じ名前のデータベース・ユーザが存在していなくてもかまいません。Mobile Link ユーザとデータベース・ユーザは、独立しています。

```
CREATE SYNCHRONIZATION SUBSCRIPTION TO testpub
     FOR demo_ml_user
     TYPE 'tcpip'
     ADDRESS 'host=localhost;port=2439;'
     OPTION sv='version1'
```

情報は以前と同じですが、長い文を 1 つ使用する代わりに、短い 2 つの文に分かれています。

同期定義の SITE 句では、この Mobile Link クライアントが Mobile Link ユーザ名 demo_sync_site を使用して同期するように指定しています。同期は TCP/IP 接続を介して発生します。同期サーバは、同期スクリプトの version1 バージョンを使用してクライアントと対話します。

2 番目の例では、同期テーブルがパブリッシュされてから、demo_sync_site Mobile Link ユーザ用のサブスクリプションが作成されます。TYPE、ADDRESS、OPTION の各句の構文は同じです。

第 6 章

Adaptive Server Anywhere クライアントの同期パラメータ

この章の内容

この章では、Mobile Link 同期クライアント (dbmlsync) に設定できるすべてのオプションについて詳しく説明します。dbmlsync を使用すると、Adaptive Server Anywhere リモート・データベースと統合データベースを同期することができます。

dbmlsync ユーティリティは、Adaptive Server Anywhere リモート・データベースでのみ有効です。Ultra Light リモートデータベースを同期させる場合は、「Ultra Light 同期パラメータ」397 ページを参照してください。

Mobile Link 同期クライアント

dbmlsync ユーティリティを使用して、Adaptive Server Anywhere リモート・データベースと統合データベースの同期を行います。

構文

dbmlsync [*options*] [*transaction-logs-directory*]

オプション	説明
@data	指定された環境変数または設定ファイルからオプションを読み込む。「@data オプション」129 ページを参照してください。
-a	エラー時に再入力のプロンプトを表示しない。「-a オプション」129 ページを参照してください。
-ap	認証パラメータを指定する。「-ap オプション」129 ページを参照してください。
-ba *filename*	ダウンロード・ファイルを適用する。「-ba オプション」130 ページを参照してください。
-bc *filename*	ダウンロード・ファイルを作成する。「-bc オプション」130 ページを参照してください。
-be *string*	ダウンロード・ファイルを作成するときに文字列を追加する。「-be オプション」131 ページを参照してください。
-bg	ダウンロード・ファイルを作成するときに、そのファイルを新しいリモートに適合するようにする。「-bg オプション」131 ページを参照してください。
-c *connection-string*	*parm1=value1*; *parm2=value2,…* の形式でデータベース接続パラメータを指定する。このオプションを指定しないと、ダイアログが表示されます。このダイアログで接続情報を指定します。「-c オプション」132 ページを参照してください。

オプション	説明
-d	ロックされているデータベースへの接続のうち、同期されたアーティクルと競合しているものをすべて削除する。「-d オプション」133 ページを参照してください。
-dc	再起動可能なダウンロードを有効にする。「-dc オプション」133 ページを参照してください。
-dl	コンソールにログ・メッセージを表示する。「-dl オプション」134 ページを参照してください。
-ds	ダウンロード専用の同期を指定する。「-ds オプション」134 ページを参照してください。
-e "*option=value*"…	拡張オプションを指定する。「dbmlsync 拡張オプション」135 ページを参照してください。
-eh	フック関数で発生したエラーを無視する。
-ek *key*	暗号化キーを指定する。「-ek オプション」180 ページを参照してください。
-ep	暗号化キーを入力するよう要求する。「-ep オプション」180 ページを参照してください。
-eu	最新の -n オプションで定義されたアップロードに対して拡張オプションを指定する。「-eu オプション」180 ページを参照してください。
-is	スケジュールを無視する。「-is オプション」181 ページを参照してください。
-k	完了時にウィンドウを閉じる。「-k オプション」181 ページを参照してください。
-l	使用可能な拡張オプションをリストする。「-l オプション」181 ページを参照してください。
-mn *password*	新しい Mobile Link パスワードを指定する。「-mn オプション」182 ページを参照してください。

オプション	説明
-mp *password*	Mobile Link パスワードを指定する。「-mp オプション」182 ページを参照してください。
-n *name*	同期パブリケーション名を指定する。「-n オプション」183 ページを参照してください。
-o *logfile*	このファイルに出力メッセージのログを取る。「-o オプション」183 ページを参照してください。
-os *size*	出力ログの最大サイズを指定する (このサイズに達するとログの名前が変更される)。「-os オプション」184 ページを参照してください。
-ot *logfile*	ファイルをトランケートし、ファイルに出力メッセージのログを取る。「-ot オプション」184 ページを参照してください。
-p	ログスキャンのポーリングを無効にする。「-p オプション」185 ページを参照してください。
-pd *dllname*;...	Windows CE 用の指定された dll をプリロードする。「-pd オプション」186 ページを参照してください。
-pi	Mobile Link に接続できるかどうかをテストする。「-pi オプション」186 ページを参照してください。
-pp *number*	ログスキャンのポーリング期間を設定する。「-pp オプション」187 ページを参照してください。
-q	最小化ウィンドウで実行する。「-q オプション」188 ページを参照してください。
-r[**a** | **b**]	クライアントの進行時にアップロードをリトライする。「-r オプション」188 ページを参照してください。
-sc	各同期の前にスキーマ情報を再ロードする。「-sc オプション」190 ページを参照してください。

オプション	説明
-tu	アップロード時に、リモート・トランザクションを保持する。「-tu オプション」190 ページを参照してください。
-u *ml_username*	同期対象となる Mobile Link ユーザを指定する。「-u オプション」192 ページを参照してください。
-uo	同期はアップロード専用 (ダウンロードはなし)。「-uo オプション」193 ページを参照してください。
-urc *row-estimate*	アップロードされるロー数の推定値を指定する。「-urc オプション」193 ページを参照してください。
-v[*levels*]	冗長オペレーション。「-v オプション」194 ページを参照してください。
-wc *classname*	ウィンドウ・クラス名を指定する。「-wc オプション」195 ページを参照してください。
-x	トランザクション・ログの名前を変更して再起動する。「-x オプション」196 ページを参照してください。
transaction-logs-directory	トランザクション・ログのロケーションを指定する。下のトランザクション・ログ・ファイルを参照してください。

説明

コマンド・ラインで dbmlsync を実行して、Adaptive Server Anywhere リモート・データベースと統合データベースの同期を行います。

Mobile Link 同期サーバを検出して接続するために、dbmlsync はパブリケーション、同期ユーザ、同期サブスクリプション、またはコマンド・ラインの情報を使用します。

トランザクション・ログ・ファイル　*transaction-logs-directory* には、Adaptive Server Anywhere リモート・データベースのトランザクション・ログが格納されているディレクトリを指定します。アクティブなトランザクション・ログ・ファイルとトランザクション・ログ・アー

カイブ・ファイルがあります。dbmlsync がアップロードするデータを判別するには、この両方が必要です。次の条件を満たす場合、このパラメータを指定してください。

- 前回の同期の後で、作業ログ・ファイルのトランケートと名前の変更が行われている場合
- 名前が変更されたログ・ファイルが格納されているディレクトリ以外のディレクトリから、dbmlsync ユーティリティを実行する場合

詳細については、『Mobile Link クライアント』> 「トランザクション・ログ・ファイル」を参照してください。

dbmlsync イベント・フック　同期処理のカスタマイズに役立つ dbmlsync クライアント・ストアド・プロシージャもあります。詳細については、『Mobile Link クライアント』> 「クライアント同期処理のカスタマイズ」と 「dbmlsync クライアント・イベント・フック」225 ページを参照してください。

dbmlsync の使用　dbmlsync の使用についての詳細は、『Mobile Link クライアント』> 「同期の開始」を参照してください。

dbmlsync オプション

この項では、Mobile Link 同期クライアントのコマンド・ライン・オプションについて説明します。

@data オプション

機能

指定された環境変数または設定ファイルからオプションを読み込みます。

構文

dbmlsync @*data* …

説明

このオプションを指定すると、環境変数または設定ファイル内にコマンド・ライン・オプションを記述できます。指定された名前に環境変数と設定ファイルの両方が存在する場合、環境変数が使用されます。

設定ファイルの詳細については、『ASA データベース管理ガイド』> 「設定ファイルの使用」を参照してください。

設定ファイル内のパスワードなどの情報を保護する場合は、ファイル非表示ユーティリティを使用して、設定ファイルの内容を難読化できます。

『ASA データベース管理ガイド』> 「dbfhide コマンド・ライン・ユーティリティを使用してファイル内容を隠す」を参照してください。

-a オプション

機能

dbmlsync がエラー時に再入力のプロンプトを表示しないように指定します。

構文

dbmlsync -a …

-ap オプション

機能

認証用のパラメータを指定します。

構文 **dbmlsync -ap "***parameters,...***"** …

説明 authenticate_parameters 接続スクリプトを使用するときに使用します。次に例を示します。

```
dbmlsync -ap "parm1,parm2,parm3"
```

詳細については、『Mobile Link 管理ガイド』> 「authenticate_parameters 接続イベント」を参照してください。

-ba オプション

機能 ダウンロード・ファイルを適用します。

構文 **dbmlsync -ba "***filename***"** …

説明 既存のダウンロード・ファイルの名前を指定します。オプションでパスを指定できます。パスを指定しない場合、デフォルトのロケーションは dbmlsync の現在の作業ディレクトリ (dbmlsync が起動されたディレクトリ) です。

詳細については、『Mobile Link 管理ガイド』> 「ファイルベースのダウンロード」を参照してください。

-bc オプション

機能 ダウンロード・ファイルを作成します。

構文 **dbmlsync -bc "***filename***"** …

説明 指定された名前でダウンロード・ファイルを作成します。ダウンロード・ファイルにはファイル拡張子 .df を使用してください。

オプションでパスを指定できます。パスを指定しない場合、デフォルトのロケーションは dbmlsync の現在の作業ディレクトリ、つまり dbmlsync が起動されたディレクトリです。

オプションで、ダウンロード・ファイルを作成したときと同じ dbmlsync コマンド・ラインで、-be オプションを使用してリモート・データベースで検証できる文字列を指定したり、-bg オプションを使用して新しいリモート・データベースのダウンロード・ファイルを作成したりできます。

参照

『Mobile Link 管理ガイド』> 「ファイルベースのダウンロード」

「-be オプション」131 ページ

「-bg オプション」131 ページ

-be オプション

機能

ダウンロード・ファイルを作成するとき、このオプションはファイルに含まれる追加の文字列を指定します。

構文

dbmlsync -bc "*filename***" -be "***string***"** …

説明

文字列は、認証や他の目的に使用できます。この文字列は、sp_hook_dbmlsync_validate_download_file ストアド・プロシージャを使用して、リモート・データベースで検証されます。

参照

「sp_hook_dbmlsync_validate_download_file」293 ページ

『Mobile Link 管理ガイド』> 「ファイルベースのダウンロード」

「-bc オプション」130 ページ

-bg オプション

機能

ダウンロード・ファイルを作成するとき、このオプションはまだ同期していないリモート・データベースで使用できるファイルを作成します。

構文

dbmlsync -bc "*filename***" -bg** …

説明

-bg オプションを使用すると、ダウンロード・ファイルによってリモート・データベースの世代番号が更新されます。

このオプションを使用すると、同期していないリモート・データベースに適用できるダウンロード・ファイルを構築できます。このオプションを使用しない場合は、同期を行ってからダウンロード・ファイルを適用する必要があります。

-bg オプションで構築したダウンロード・ファイルは、スナップショット・ダウンロードです。新しいリモートの最終ダウンロード・タイムスタンプは、デフォルトでは 1900 年 1 月 1 日になっており、これはダウンロード・ファイル内の最終ダウンロード・タイムスタンプより前となるため、タイムスタンプ・ベースのダウンロードは同期していないリモート・データベースと連携しません。タイムスタンプ・ベースでファイル・ベースのダウンロードが動作するには、ダウンロード・ファイル内の最終ダウンロード・タイムスタンプがリモートと同じか、それより前である必要があります。

すでに同期したリモート・データベースには、-bg オプションで構築したダウンロード・ファイルを適用しないでください。-bg オプションは、ダウンロード・ファイルの作成時にリモート・データベースの世代番号を統合データベースの値に更新します。すでに同期したリモートとは、リモート・データベースがダウンロードを適用する前にデータがアップロードされないことを意味し、このためデータが失われる可能性があります。

参照

「-bc オプション」130 ページ

『Mobile Link 管理ガイド』> 「ファイルベースのダウンロード」

『Mobile Link 管理ガイド』> 「Mobile Link の世代番号」

『Mobile Link 管理ガイド』> 「新しいリモートの同期」

-c オプション

機能

リモート・データベースの接続パラメータを指定します。

構文

dbmlsync -c "*connection-string***"** …

説明

接続文字列では、Adaptive Server Anywhere リモート・データベースへの接続に使用する、dbmlsync の接続パーミッションを指定します。通常、REMOTE DBA 権限を持つユーザ ID が使用されます。

keyword=value の形式で、複数のパラメータをセミコロンで区切って接続文字列を指定します。いずれかのパラメータ名にスペースが含まれる場合は、接続文字列を二重引用符で囲んでください。

-c を指定しないと、[DBMLSync 設定] ダイアログが表示されます。この接続ダイアログのフィールドで、残りのコマンド・ライン・オプションを指定できます。

Adaptive Server Anywhere データベースに接続するための接続パラメータの完全なリストについては、『ASA データベース管理ガイド』> 「接続パラメータ」を参照してください。

-d オプション

機能　リモート・データベースに対する競合ロックを削除します。

構文　**dbmlsync -d** …

説明　同期中は、LockTables 拡張オプションを OFF に設定しないかぎり、同期するパブリケーションに関するすべてのテーブルがロックされて他のプロセスによる変更が加えられません。別の接続でこれらのいずれかのテーブルにロックがあると、同期は失敗します。このオプションを指定すると、Adaptive Server Anywhere は競合ロックを保持するリモート・データベースへの他の接続をすべて強制的に削除します。

参照　『Mobile Link クライアント』> 「同期中の同時実行性」

-dc オプション

機能　再起動可能なダウンロードを指定します。

構文　**dbmlsync -dc** …

説明　デフォルトでは、ダウンロード中に Mobile Link で障害が発生すると、ダウンロード・データはリモート・データベースに適用されません。ただし、受信したダウンロードの一部がリモート・デバイスのテンポラリ・ファイルに保存されているため、次に dbmlsync を起動するときに -dc オプションを指定すると、短時間でダウンロードを完了でき

ます。-dc オプションを指定すると、dbmlsync はダウンロードを再起動し、前のダウンロードで受信しなかった部分をダウンロードします。残りのデータをダウンロードできる場合は、完全なダウンロードがリモート・データベースに適用されます。

-dc オプションを使用したときに、アップロードされる新しいデータがある場合は、再起動可能なダウンロードは失敗します。

また、Adaptive Server Anywhere リモート・データベースの再起動可能なダウンロードは、ContinueDownload 拡張オプションまたは sp_hook_dbmlsync_end フックを使用して指定することもできます。

参照

『Mobile Link 管理ガイド』> 「失敗したダウンロードの再開」

「ContinueDownload (cd) 拡張オプション」145 ページ

「sp_hook_dbmlsync_end」268 ページ

「DownloadReadSize (drs) 拡張オプション」150 ページ

-dl オプション

機能

ログ・ファイル内のメッセージを表示します。

構文

dbmlsync -dl …

説明

通常、ファイルに出力のログを取るときは、dbmlsync ウィンドウに書き込まれるよりも多くのメッセージがログ・ファイルに書き込まれます。このオプションを使用すると、dbmlsync は通常はファイルにのみ書き込まれる情報をウィンドウにも出力します。このオプションを使用すると、同期の速度に影響する場合があります。

-ds オプション

機能

ダウンロード専用同期を指定します。

構文

dbmlsync -ds …

説明

ダウンロード専用同期が発生する場合、dbmlsync はローの操作またはデータをアップロードしません。しかし、スキーマと進行オフセットについての情報はアップロードします。

さらに、dbmlsync は、ダウンロード専用同期中に、リモートでの変更が上書きされないようにします。これを実行するには、ログをスキャンし、アップロードされるのを待機している操作に関連するローを検出します。これらのローのいずれかがダウンロード・ストリームによって修正されると、ダウンロード・ストリームはロールバックされ、同期は失敗します。この理由で同期が失敗した場合は、問題を訂正するために完全な同期を行う必要があります。

ダウンロード専用同期で同期されたリモートがある場合、ダウンロード専用同期がスキャンするログの量を減らすために、定期的に完全な同期を行ってください。そうしないと、ダウンロード専用同期が完了するのに次第に時間がかかるようになります。

-ds を使用すると、ConflictRetries 設定が無視され、dbmlsync はダウンロード専用同期をリトライしません。ダウンロード専用同期が失敗した場合は、通常の同期が行われるまで、ダウンロード専用同期は継続してエラーとなります。

参照

『Mobile Link 管理ガイド』> 「アップロード専用の同期とダウンロード専用の同期」

「DownloadOnly (ds) 拡張オプション」148 ページ

「UploadOnly (uo) 拡張オプション」172 ページ または 「-uo オプション」193 ページ

dbmlsync 拡張オプション

機能

拡張オプションを指定します。

構文

dbmlsync -e *extended-option*=*value*; …

extended-option:
adr cd cr ctp dbs dir drs ds eh el ft hrt inc isc lt mem mn mp p pp sa sc sch scn st sv tor uo v vn vm vo vr vs vu

パラメータ

拡張オプションは、長形式または省略形で指定できます。詳細については、以下の各オプションを参照してください。

説明

コマンド・ラインで -e オプションを使用して指定したオプションは、同じコマンド・ラインで要求したすべての同期に適用されます。たとえば次のコマンド・ラインでは、拡張オプション sv=test は pub1 と pub2 の両方の同期に適用されます。

```
dbmlsync -e "sv=test" -n pub1 -n pub2
```

単一のアップロードに拡張オプションを指定するには、-eu オプションを使用します。

拡張オプションは、dbmlsync コマンド・ライン上で -e オプションまたは -eu オプションを使用して指定するか、データベースに格納できます。拡張オプションをデータベースに格納するには、Sybase Central を使用するか、sp_hook_dbmlsync_set_extended_options イベント・フックを使用するか、次のいずれかの文で OPTIONS 句を使用します。

- CREATE SYNCHRONIZATION SUBSCRIPTION
- ALTER SYNCHRONIZATION SUBSCRIPTION
- CREATE SYNCHRONIZATION USER
- ALTER SYNCHRONIZATION USER
- 同期ユーザを指定しない CREATE SYNCHRONIZATION SUBSCRIPTION (これによって、拡張オプションがパブリケーションに対応付けられる)

dbmlsync は、データベースに格納されるオプションとコマンド・ラインで指定されるオプションを結合します。競合するオプションが指定されている場合、dbmlsync は次のようにして競合を解決します。メソッドで指定されるオプションは、次のリストの順番で先に出てくるものが後で出てくるものより優先されます。

- sp_hook_dbmlsync_set_extended_options イベント・フックで指定されているオプション
- コマンド・ラインで -eu オプションを使用して指定されているオプション

- コマンド・ラインで -e オプションを使用して指定されているオプション
- (SQL 文または Sybase Central で) サブスクリプションに対して指定されているオプション
- (SQL 文または Sybase Central で) ユーザに対して指定されているオプション
- (SQL 文または Sybase Central で) パブリケーションに対して指定されているオプション

拡張オプションは、ログと SYSSYNC システム・テーブルで確認できます。

拡張オプションを使用して同期をチューニングする方法については、『Mobile Link クライアント』> 「同期のカスタマイズ」を参照してください。

拡張オプションの詳しい説明については、次を参照してください。

参照

「-eu オプション」180 ページ

『ASA SQL リファレンス・マニュアル』> 「SYSSYNC システム・テーブル」

「sp_hook_dbmlsync_set_extended_options」284 ページ

例

次の dbmlsync コマンド・ラインは、dbmlsync を起動するときの拡張オプションの設定方法を示します。

```
dbmlsync -e "adr=host=localhost;dir=c:¥db¥logs"...
```

次の SQL 文は、拡張オプションをデータベースに格納する方法を示します。

```
CREATE SYNCHRONIZATION SUBSCRIPTION TO mypub
    FOR mluser
    ADDRESS 'host=localhost'
    OPTION schedule='weekday@11:30am-12:30pm',
dir='c:¥db¥logs'
```

次の dbmlsync コマンド・ラインは、オプションとその構文をリストする使用画面を開きます。

```
dbmlsync -l
```

CommunicationAddress (adr) 拡張オプション

機能

Mobile Link サーバに接続するネットワーク・プロトコル・オプションを指定します。

構文

dbmlsync -e adr=*protocol-option*; ...

パラメータ

- **TCP/IP プロトコル・オプション** tcpip プロトコルを指定する場合は、オプションで次のプロトコル・オプションを指定できます。

TCP/IP プロトコル・オプション	詳細の参照先
client_port=*nnnnn*[-*mmmmm*]	「client_port」51 ページ
host=*hostname*	「host」52 ページ
liveness_timeout=*n*	「liveness_timeout」58 ページ
network_connect_timeout=*seconds*	「network_connect_timeout」59 ページ
network_leave_open={**0**|**1**}	「network_leave_open」59 ページ
network_name=*name*	「network_name」60 ページ
port=*portnumber*	「port」62 ページ
security=*cipher*(*keyword*=*value*;…)	「security」64 ページ

- **HTTP プロトコル** http プロトコルを指定する場合は、オプションで次のプロトコル・オプションを指定できます。

HTTP プロトコル・オプション	詳細の参照先
buffer_size=*number*	「buffer_size」44 ページ
client_port=*nnnnn*[-*mmmmm*]	「client_port」51 ページ

HTTP プロトコル・オプション	詳細の参照先
custom_header=*header*	「custom_header」51 ページ
host=*hostname*	「host」52 ページ
network_connect_timeout=*seconds*	「network_connect_timeout」59 ページ
network_leave_open={**0**|**1**}	「network_leave_open」59 ページ
network_name=*name*	「network_name」60 ページ
persistent={**0**|**1**}	「persistent」61 ページ
port=*portnumber*	「port」62 ページ
proxy_host=*proxy_hostname*	「proxy_host」63 ページ
proxy_port=*proxy_portnumber*	「proxy_port」63 ページ
set_cookie=*cookie_name=cookie_value*	「set_cookie」67 ページ
url_suffix=*suffix*	「url_suffix」70 ページ
version=*versionnumber*	「version」71 ページ

- **HTTPS または HTTPS_FIPS プロトコル**　HTTPS プロトコルは Certicom RSA セキュリティを使用します。HTTPS_FIPS プロトコルは、Certicom が提供している別の FIPS 140-2 準拠の RSA 暗号化ソフトウェアを使用します。

別途ライセンスを取得できるオプションが必要
トランスポート・レイヤ・セキュリティを使用するには、別途ライセンスを取得できる SQL Anywhere Studio セキュリティ・オプションを入手する必要があります。このセキュリティ・オプションは、輸出規制対象品目です。

このコンポーネントのご注文については、『SQL Anywhere Studio の紹介』> 「別途ライセンスが入手可能なコンポーネント」を参照してください。

HTTPS プロトコルを指定する場合は、オプションで次のプロトコル・オプションを指定できます。

HTTPS/HTTPS_FIPS プロトコル・オプション	詳細の参照先
buffer_size=*number*	「buffer_size」44 ページ
certificate_company=*company_name*	「certificate_company」45 ページ
certificate_name=*name*	「certificate_name」47 ページ
certificate_unit=*company_unit*	「certificate_unit」49 ページ
client_port=*nnnnn*[-*mmmmm*]	「client_port」51 ページ
custom_header=*header*	「custom_header」51 ページ
host=*hostname*	「host」52 ページ
network_connect_timeout=*seconds*	「network_connect_timeout」59 ページ
network_leave_open={**0**|**1**}	「network_leave_open」59 ページ
network_name=*name*	「network_name」60 ページ
persistent={**0**|**1**}	「persistent」61 ページ
port=*portnumber*	「port」62 ページ
proxy_host=*proxy_hostname*	「proxy_host」63 ページ
proxy_port=*proxy_portnumber*	「proxy_port」63 ページ
set_cookie=*cookie_name* =*cookie_value*	「set_cookie」67 ページ
trusted_certificates=*filename*	「trusted_certificates」68 ページ

HTTPS/HTTPS_FIPS プロトコル・オプション	詳細の参照先
url_suffix=*suffix*	「url_suffix」70 ページ
version=*versionnumber*	「version」71 ページ

- **ActiveSync プロトコル**　ActiveSync 同期の実行中は、デスクトップ・マシンに存在する ActiveSync 用 Mobile Link プロバイダとデータをやり取りするために ActiveSync が使用されます。ActiveSync プロトコル・オプションでは、ActiveSync 用 Mobile Link プロバイダと Mobile Link 同期サーバ間の通信を記述します。

  ActiveSync のアドレスの文字列は、次の形式を取ります。

  stream=*desktop-protocol*;[*desktop-protocol-options*]

  文中の各項目を次に説明します。

  - *desktop-protocol*　は、ActiveSync 用 Mobile Link プロバイダと Mobile Link 同期サーバとの間で使用するネットワーク・プロトコルです。これには、**http**、**https**、**https_fips**、または **tcpip** を指定できます。デフォルトは **tcpip** です。
  - *desktop-protocol-options* は、上記のリストで説明されている TCP/IP、HTTP、HTTPS、または HTTPS_FIPS のオプションです。

別途ライセンスを取得できるオプションが必要

トランスポート・レイヤ・セキュリティを使用するには、別途ライセンスを取得できる SQL Anywhere Studio セキュリティ・オプションを入手する必要があります。このセキュリティ・オプションは、輸出規制対象品目です。

このコンポーネントのご注文については、『SQL Anywhere Studio の紹介』> 「別途ライセンスが入手可能なコンポーネント」を参照してください。

詳細については、「ActiveSync プロバイダ・インストール・ユーティリティ」34 ページを参照してください。

説明

Mobile Link ユーザのすべてのサブスクリプションが、1 つの統合データベースに対してのみ同期されていることを確認する必要があります。複数の統合データベースに同期されていると、データの損失や予期しない動作が発生する場合があります。

リダイレクタを使用している場合は、『Mobile Link 管理ガイド』> 「Mobile Link クライアントとサーバのリダイレクタ設定」を参照してください。

このオプションには省略形と長形式があり、**adr** または **CommunicationAddress** を使用できます。

このオプションも、パブリケーション、サブスクリプション、またはユーザを作成、変更する SQL 文を使用して、データベースに格納できます。詳細については、次の項を参照してください。

- 『ASA SQL リファレンス・マニュアル』> 「CREATE SYNCHRONIZATION SUBSCRIPTION 文 [Mobile Link]」
- 『ASA SQL リファレンス・マニュアル』> 「CREATE SYNCHRONIZATION USER 文 [Mobile Link]」

TCP/IP、HTTP、HTTPS、または HTTPS_FIPS プロトコルを指定する場合は、CommunicationType 拡張オプションを使用してください。

詳細については、「CommunicationType (ctp) 拡張オプション」143 ページを参照してください。

参照

『Mobile Link 管理ガイド』> 「Mobile Link クライアントとサーバのリダイレクタ設定」

例

次の dbmlsync コマンド・ラインは、dbmlsync を使用するときのこのオプションの設定方法を示します。

```
dbmlsync -e "adr=host=localhost"
```

コマンド・ラインで複数のネットワーク・プロトコル・オプションを指定するには、それらを一重引用符で囲みます。次に例を示します。

```
dbmlsync -e "adr='host=somehost;port=5001'"
```

データベースに Address または CommunicationType を格納するには、拡張オプションを使用するか、ADDRESS 句または TYPE 句を使用できます。次に例を示します。

```
CREATE SYNCHRONIZATION SUBSCRIPTION
    TO sales_publication
    FOR ml_user1
    TYPE 'https'
    ADDRESS host='localhost;port=2439'
```

CommunicationType (ctp) 拡張オプション

機能

Mobile Link サーバへの接続に使用するネットワーク・プロトコルの種類を指定します。

構文

dbmlsync -e ctp=*sync-type*; ...

説明

sync-type は、**tcpip**、**http**、**https**、**https_fips**、または **ActiveSync** のうちの 1 つを指定できます。デフォルトは **tcpip** です。

Mobile Link ユーザのすべてのサブスクリプションが、1 つの統合データベースに対してのみ同期されていることを確認する必要があります。複数の統合データベースに同期されていると、データの損失や予期しない動作が発生する場合があります。

このオプションには省略形と長形式があり、**ctp** または **CommunicationType** を使用できます。

このオプションも、パブリケーション、サブスクリプション、またはユーザを作成、変更する SQL 文を使用して、データベースに格納できます。詳細については、次の項を参照してください。

- 『ASA SQL リファレンス・マニュアル』> 「CREATE SYNCHRONIZATION SUBSCRIPTION 文 [Mobile Link]」
- 『ASA SQL リファレンス・マニュアル』> 「CREATE SYNCHRONIZATION USER 文 [Mobile Link]」

参照

- 『Mobile Link 管理ガイド』> 「Mobile Link トランスポート・レイヤ・セキュリティ」
- 「CommunicationAddress (adr) 拡張オプション」138 ページ

例

次の dbmlsync コマンド・ラインは、dbmlsync を使用するときのこのオプションの設定方法を示します。

```
dbmlsync -e "ctp=https"
```

データベースに Address または CommunicationType を格納するには、拡張オプションを使用するか、ADDRESS 句または TYPE 句を使用できます。次に例を示します。

```
CREATE SYNCHRONIZATION SUBSCRIPTION
    TO sales_publication
    FOR ml_user1
    TYPE 'tcpip'
    ADDRESS host='localhost'
```

ConflictRetries (cr) 拡張オプション

機能

競合のためにダウンロードが失敗した場合のリトライの回数を指定します。

構文

dbmlsync -e cr=*number*; ...

説明

-1 は、リトライが無制限に続くよう指定します。デフォルトは **-1** です。

このオプションは、LockTables オプションが OFF (デフォルトは ON) の場合にだけ役に立ちます。

このオプションには省略形と長形式があり、**cr** または **ConflictRetries** を使用できます。

拡張オプションのデータベースへの格納もできます。dbmlsync 拡張オプションの詳細については、「dbmlsync 拡張オプション」135 ページを参照してください。

参照

『Mobile Link 管理ガイド』> 「競合の解決」

例

次の dbmlsync コマンド・ラインは、dbmlsync を使用するときのこのオプションの設定方法を示します。

```
dbmlsync -e "cr=5"
```

次の SQL 文は、このオプションのデータベースへの格納方法を示します。

```
CREATE SYNCHRONIZATION SUBSCRIPTION
    TO sales_publication
    FOR ml_user1
    OPTION cr='5';
```

ContinueDownload (cd) 拡張オプション

機能　再起動可能なダウンロードを指定します。

構文　**dbmlsync -e cd={ ON | OFF }; ...**

説明　デフォルトでは、ダウンロード中に Mobile Link で障害が発生すると、ダウンロード・データはリモート・データベースに適用されません。ただし、受信したダウンロードの一部がリモート・デバイスのテンポラリ・ファイルに保存されているため、次に dbmlsync を起動するときに -e cd オプションを指定すると、短時間でダウンロードを完了できます。-e cd オプションを指定すると、dbmlsync はダウンロードを再起動し、前のダウンロードで受信しなかった部分をダウンロードします。残りのデータをダウンロードできる場合は、完全なダウンロードがリモート・データベースに適用されます。

-dc オプションを使用したときに、アップロードされる新しいデータがある場合は、再起動可能なダウンロードは失敗します。

また、Adaptive Server Anywhere リモート・データベースの再起動可能なダウンロードは、-dc オプションまたは sp_hook_dbmlsync_end フックを使用して指定することもできます。

このオプションには省略形と長形式があり、**cd** または **ContinueDownload** を使用できます。

拡張オプションのデータベースへの格納もできます。dbmlsync 拡張オプションの詳細については、「dbmlsync 拡張オプション」135 ページを参照してください。

参照　『Mobile Link 管理ガイド』> 「失敗したダウンロードの再開」

「sp_hook_dbmlsync_set_extended_options」284 ページ

「-dc オプション」133 ページ

「sp_hook_dbmlsync_end」268 ページ

例

次の dbmlsync コマンド・ラインは、dbmlsync を使用するときのこのオプションの設定方法を示します。

```
dbmlsync -e "cd=on"
```

次の SQL 文は、このオプションのデータベースへの格納方法を示します。

```
CREATE SYNCHRONIZATION SUBSCRIPTION
    TO sales_publication
    FOR ml_user1
    OPTION cd='on';
```

DisablePolling (p) 拡張オプション

機能

自動のログスキャンのポーリングを無効にします。

構文

dbmlsync -e p= { **ON** | **OFF** }; ...

説明

アップロード・ストリームを構築するには、dbmlsync はトランザクション・ログをスキャンする必要があります。通常、これは同期直前に行います。しかし、同期がスケジュールされている場合、デフォルトでは dbmlsync はスケジュールされた同期の間の時間にログをスキャンします。また、sp_hook_dbmlsync_delay フックが使用されている場合、デフォルトでは dbmlsync は同期直前に生じる一時停止でログをスキャンします。同期が始まるときログはすでに一部がスキャンされているので、この動作はより効率的です。このデフォルトの動作は、ログスキャンのポーリングと呼ばれます。

ログスキャンのポーリングはデフォルトでは on になっていますが、同期がスケジュールされているとき、または sp_hook_dbmlsync_delay フックが使用されているときしか効果がありません。これが有効な場合、ポーリングは決まった間隔で実行されます。dbmlsync はログの最後までスキャンし、ポーリング期間の間待機した後、ログの新しいトランザクションをスキャンします。デフォルトではポーリング期間は 1 分ですが、dbmlsync -pp オプションまたは PollingPeriod 拡張オプションで変更できます。

デフォルトでは、ログスキャンのポーリングを無効にしません (**OFF**)。

このオプションは、**dbmlsync -p** と同じです。

このオプションには省略形と長形式があり、**p** または **DisablePolling** を使用できます。

拡張オプションのデータベースへの格納もできます。dbmlsync 拡張オプションの詳細については、「dbmlsync 拡張オプション」135 ページを参照してください。

参照

「PollingPeriod (pp) 拡張オプション」163 ページ

「-p オプション」185 ページ

「-pp オプション」187 ページ

例

次の dbmlsync コマンド・ラインは、dbmlsync を使用するときのこのオプションの設定方法を示します。

```
dbmlsync -e "p=on"
```

次の SQL 文は、このオプションのデータベースへの格納方法を示します。

```
CREATE SYNCHRONIZATION SUBSCRIPTION
    TO sales_publication
    FOR ml_user1
    OPTION p='on';
```

DownloadBufferSize (dbs) 拡張オプション

機能

ダウンロード・バッファのサイズを指定します。

構文

dbmlsync -e dbs= *number*[**K** | **M**]; ...

説明

バッファのサイズはバイト単位で指定します。キロバイトまたはメガバイトの単位を指定するには、それぞれサフィックス k、m を使用します。

このオプションを 0 に設定すると、dbmlsync はダウンロード・ストリームのバッファを行いません。設定が 0 より大きく 4 k より小さい場合、dbmlsync は 4 k のバッファ・サイズを使用し、警告を発します。デフォルトは、Windows CE では **32 K** で、他のすべてのオペレーティング・システムでは **1 M** です。

ダウンロード・バッファリングによりワーカ・スレッドはダウンロードをより早く送信できるので、ダウンロードの受信確認を省略するという利点があります。

このオプションには省略形と長形式があり、**dbs** または **DownloadBufferSize** を使用できます。

拡張オプションのデータベースへの格納もできます。dbmlsync 拡張オプションの詳細については、「dbmlsync 拡張オプション」135 ページを参照してください。

例

次の dbmlsync コマンド・ラインは、dbmlsync を使用するときのこのオプションの設定方法を示します。

```
dbmlsync -e "dbs=32k"
```

次の SQL 文は、このオプションのデータベースへの格納方法を示します。

```
CREATE SYNCHRONIZATION SUBSCRIPTION
    TO sales_publication
    FOR ml_user1
    OPTION dbs='32k';
```

DownloadOnly (ds) 拡張オプション

機能

同期がダウンロード専用であることを指定します。

構文

dbmlsync -e ds={ ON | OFF }; ...

説明

ダウンロード専用同期が発生する場合、dbmlsync はローの操作またはデータをアップロードしません。しかし、スキーマと進行オフセットについての情報はアップロードします。

さらに、dbmlsync は、ダウンロード専用同期中に、リモートでの変更が上書きされないようにします。これを実行するには、ログをスキャンし、アップロードされるのを待機している操作に関連するローを検出します。これらのローのいずれかがダウンロード・ストリームによって修正されると、ダウンロード・ストリームはロールバックされ、同期は失敗します。この理由で同期が失敗した場合は、問題を訂正するために完全な同期を行う必要があります。

ダウンロード専用同期で同期されたリモートがある場合、ダウンロード専用同期がスキャンするログの量を減らすために、定期的に完全な同期を行ってください。そうしないと、ダウンロード専用同期が完了するのに次第に時間がかかるようになります。

デフォルトは **OFF** です (アップロードとダウンロード両方の完全な同期)。

このオプションには省略形と長形式があり、**ds** または **DownloadOnly** を使用できます。

拡張オプションのデータベースへの格納もできます。dbmlsync 拡張オプションの詳細については、「dbmlsync 拡張オプション」135 ページを参照してください。

参照

「-ds オプション」134 ページ

『Mobile Link 管理ガイド』> 「アップロード専用の同期とダウンロード専用の同期」

例

次の dbmlsync コマンド・ラインは、dbmlsync を使用するときのこのオプションの設定方法を示します。

```
dbmlsync -e "ds=on"
```

次の SQL 文は、このオプションのデータベースへの格納方法を示します。

```
CREATE SYNCHRONIZATION SUBSCRIPTION
    TO sales_publication
    FOR ml_user1
    OPTION ds='ON';
```

DownloadReadSize (drs) 拡張オプション

機能

再起動可能なダウンロードには、通信エラーの後に再送信する必要があるデータの最大量を指定します。

構文

dbmlsync -e drs= *number*[**K** | **M**]; ...

説明

DownloadReadSize オプションが有用なのは、再起動可能なダウンロードを実行するときだけです。

ダウンロードの読み込みサイズは、バイト単位で指定します。キロバイトまたはメガバイトの単位を指定するには、それぞれサフィックス k、m を使用します。

dbmlsync は、チャンク内のダウンロード・ストリームを読み込みます。このチャンクのサイズを定義するのが DownloadReadSize です。通信エラーが発生すると、処理中のチャンク全体が失われます。エラーが発生した時点によって、失われるバイト数は 0 ～ DownloadReadSize -1 の間です。たとえば、DownloadReadSize が 100 バイトで、497 バイトを読み込んだ後でエラーが発生した場合は、読み込んだ最後の 97 バイトが失われます。このようにして失われたバイト数は、ダウンロードが再起動されたときに再送信されます。

通常は、DownloadReadSize の値を大きくすると、正常な同期でのパフォーマンスが向上しますが、エラーが発生したときに再送信されるデータが多くなります。

このオプションの一般的な用途は、通信の信頼性が低いときにデフォルトのサイズを減らすことです。

デフォルトは **32 K** です。

このオプションには省略形と長形式があり、**drs** または **DownloadReadSize** を使用できます。

拡張オプションのデータベースへの格納もできます。dbmlsync 拡張オプションの詳細については、「dbmlsync 拡張オプション」135 ページを参照してください。

参照

- 『Mobile Link 管理ガイド』> 「失敗したダウンロードの再開」
- 「ContinueDownload (cd) 拡張オプション」145 ページ

- 「sp_hook_dbmlsync_end」268 ページ
- 「-dc オプション」133 ページ

例

次の dbmlsync コマンド・ラインは、dbmlsync を使用するときのこのオプションの設定方法を示します。

```
dbmlsync -e "drs=100"
```

次の SQL 文は、このオプションのデータベースへの格納方法を示します。

```
CREATE SYNCHRONIZATION SUBSCRIPTION
    TO sales_publication
    FOR ml_user1
    OPTION drs='100';
```

ErrorLogSendLimit (el) 拡張オプション

機能

同期エラーが発生した場合、dbmlsync からサーバに送信するリモート・ログ・ファイルのサイズを指定します。

構文

dbmlsync -e el=*number*[**K** | **M**]; ...

説明

このオプションはバイト単位で指定します。キロバイトまたはメガバイトの単位を指定するには、それぞれサフィックス k、m を使用します。

このオプションは、同期中にエラーが発生した場合に、dbmlsync が Mobile Link 同期サーバに送信する出力ログのバイト数を指定します。dbmlsync 出力ログを送信しない場合は、このオプションを **0** に設定します。

ErrorLogSendLimit を十分に大きい値に設定すると、dbmlsync は現在のセッションの出力ログ・メッセージ全体を、Mobile Link 同期サーバに送信します。たとえば、出力ログ・メッセージが古い出力ログ・ファイルに追加された場合、dbmlsync は現在のセッション中に生成された新しいメッセージだけを送信します。新しいメッセージ全体の長さが ErrorLogSendLimit を超える場合、dbmlsync は新しく生成されたエラー・メッセージとログ・メッセージの最後の部分だけを、指定されたサイズの範囲内で送信します。

注意：出力ログのサイズは、使用している冗長性の設定により影響を受けます。冗長性を調節するには、dbmlsync -v オプションを使用するか、"verbose" で始まる dbmlsync 拡張オプションを使用します。詳細については、「-v オプション」194 ページ と次に示す -e 冗長性オプションを参照してください。

デフォルトは **32 K** です。

このオプションには省略形と長形式があり、**el** または **ErrorLogSendLimit** を使用できます。

拡張オプションのデータベースへの格納もできます。dbmlsync 拡張オプションの詳細については、「dbmlsync 拡張オプション」135 ページを参照してください。

例

次の dbmlsync コマンド・ラインは、dbmlsync を使用するときのこのオプションの設定方法を示します。

```
dbmlsync -e "el=32k"
```

次の SQL 文は、このオプションのデータベースへの格納方法を示します。

```
CREATE SYNCHRONIZATION SUBSCRIPTION
    TO sales_publication
    FOR ml_user1
    OPTION el='32k';
```

FireTriggers (ft) 拡張オプション

機能

ダウンロードが適用されたときにリモート・データベースでトリガが起動されるように指定します。

構文

dbmlsync -e ft= { **ON** | **OFF** }; ...

説明

デフォルトは **ON** です。

このオプションには省略形と長形式があり、**ft** または **FireTriggers** を使用できます。

拡張オプションのデータベースへの格納もできます。dbmlsync 拡張オプションの詳細については、「dbmlsync 拡張オプション」135 ページを参照してください。

例

次の dbmlsync コマンド・ラインは、dbmlsync を使用するときのこのオプションの設定方法を示します。

```
dbmlsync -e "ft=off"
```

次の SQL 文は、このオプションのデータベースへの格納方法を示します。

```
CREATE SYNCHRONIZATION SUBSCRIPTION
    TO sales_publication
    FOR ml_user1
    OPTION ft='off';
```

HoverRescanThreshold (hrt) 拡張オプション

機能

スケジュールを使用している場合、再スキャンの実行までに累積可能な廃棄メモリ量を制限します。

構文

dbmlsync -e hrt= *number*[**K** | **M**]; ...

説明

メモリをバイト単位で指定します。キロバイトまたはメガバイトの単位を指定するには、それぞれサフィックス k、m を使用します。デフォルトは **1 M** です。

スケジュール・オプションが指定されている場合、または複数の dbmlsync -n オプションが指定されている場合、dbmlsync はホバリング・モードに入ります。ホバリングは、dbmlsync が起動され、シャットダウンの前に複数の同期を行うよう要求されたときに、ログをスキャンする時間を減らす機能です。ホバリングは、同期されるすべてのサブスクリプションが同じ Mobile Link ユーザに関連するときだけ発生可能です。

ホバリング中、dbmlsync はシステムを使用してログから読み込まれた操作を追跡します。システムは最初にメモリで情報を管理しますが、次にディスクに書き込みます。ホバリングが続く場合、dbmlsync はフラグメント化されたメモリを破棄します。破棄されるメモリの量は、ホバリング中に処理された操作の数および関連するローのサイズに比

例します (BLOB は計算に入れません)。リモート・データベースに同期されているユーザのパブリケーションが 1 つしかない場合、メモリは破棄されません。

破棄されたメモリは、完全な再スキャンが行われた後にリカバリできます。メモリがリカバリされるときの制御方法は 2 つあります。それは、HoverRescanThreshold 拡張オプションと sp_hook_dbmlsync_log_rescan ストアド・プロシージャを使用する方法です。

このオプションには省略形と長形式があり、**hrt** または **HoverRescanThreshold** を使用できます。

拡張オプションのデータベースへの格納もできます。dbmlsync 拡張オプションの詳細については、「dbmlsync 拡張オプション」135 ページを参照してください。

参照

「sp_hook_dbmlsync_log_rescan」271 ページ

例

次の dbmlsync コマンド・ラインは、dbmlsync を使用するときのこのオプションの設定方法を示します。

```
dbmlsync -e "hrt=2m"
```

次の SQL 文は、このオプションのデータベースへの格納方法を示します。

```
CREATE SYNCHRONIZATION SUBSCRIPTION
    TO sales_publication
    FOR ml_user1
    OPTION hrt='2m';
```

IgnoreHookErrors (eh) 拡張オプション

機能

フック関数内で発生したエラーを無視するように指定します。

構文

dbmlsync -e eh= { **ON** | **OFF** }; ...

説明

デフォルトは **OFF** です。

このオプションには省略形と長形式があり、**eh** または **IgnoreHookErrors** を使用できます。

このオプションは、dbmlsync -eh オプションと同じです。

拡張オプションのデータベースへの格納もできます。dbmlsync 拡張オプションの詳細については、「dbmlsync 拡張オプション」135 ページを参照してください。

例

次の dbmlsync コマンド・ラインは、dbmlsync を使用するときのこのオプションの設定方法を示します。

```
dbmlsync -e "eh=off"
```

次の SQL 文は、このオプションのデータベースへの格納方法を示します。

```
CREATE SYNCHRONIZATION SUBSCRIPTION
    TO sales_publication
    FOR ml_user1
    OPTION eh='off';
```

IgnoreScheduling (isc) 拡張オプション

機能

スケジュール設定を無視するように指定します。

構文

dbmlsync -e isc= { **ON** | **OFF** }; ...

説明

ON に設定すると、dbmlsync は拡張オプションで指定されたスケジュール情報を無視して、すぐに同期を行います。デフォルトは **OFF** です。

このオプションは、dbmlsync -is オプションと同じです。

このオプションには省略形と長形式があり、**isc** または **IgnoreScheduling** を使用できます。

拡張オプションのデータベースへの格納もできます。dbmlsync 拡張オプションの詳細については、「dbmlsync 拡張オプション」135 ページを参照してください。

参照

『Mobile Link クライアント』> 「同期のスケジュール」

「Schedule (sch) 拡張オプション」164 ページ

例

次の dbmlsync コマンド・ラインは、dbmlsync を使用するときのこのオプションの設定方法を示します。

```
dbmlsync -e "isc=off"
```

次の SQL 文は、このオプションのデータベースへの格納方法を示します。

```
CREATE SYNCHRONIZATION SUBSCRIPTION
    TO sales_publication
    FOR ml_user1
    OPTION isc='off';
```

Increment (inc) 拡張オプション

機能

インクリメンタル・アップロードのサイズを制御します。

構文

dbmlsync -e inc= *number* [**K** | **M**]; ...

説明

このオプションは、インクリメンタル・スキャンの最小サイズをバイト単位で指定します。キロバイトまたはメガバイトの単位を指定するには、それぞれサフィックス k、m を使用します。

このオプションを指定すると、アップロードは 1 つ以上の部分に分けて Mobile Link に送信されます。このオプションは、サイトが完全なアップロードを完了するだけの長い時間接続を維持できない場合に役立ちます。オプションが設定されていない場合、アップロードは 1 つにまとめて送信されます。

このオプションの値は、大まかに各アップロード部分のサイズを指定します。オプションの値は、次のように各アップロード部分のサイズを制御します。dbmlsync は、データベース・トランザクション・ログをスキャンして、アップロード・ストリームを構築します。このオプションを指定すると、dbmlsync はオプションで設定されたバイト数をスキャンし、未処理トランザクションがない最初の時点、つまりすべてのトランザクションがコミットまたはロールバックされた次の時点

までスキャンを続けます。dbmlsync は、スキャンした部分をアップロード部分として送信し、中断したところからログのスキャンを再開します。

Increment 拡張オプションに -tu を使用することはできません。

このオプションには省略形と長形式があり、**inc** または **Increment** を使用できます。

拡張オプションのデータベースへの格納もできます。dbmlsync 拡張オプションの詳細については、「dbmlsync 拡張オプション」135 ページを参照してください。

例

次の dbmlsync コマンド・ラインは、dbmlsync を使用するときのこのオプションの設定方法を示します。

```
dbmlsync -e "inc=32000"
```

次の SQL 文は、このオプションのデータベースへの格納方法を示します。

```
CREATE SYNCHRONIZATION SUBSCRIPTION
    TO sales_publication
    FOR ml_user1
    OPTION inc='32k';
```

LockTables (lt) 拡張オプション

機能

アーティクル (同期対象パブリケーション内のテーブルまたはテーブルの一部) をロックしてから同期を実行するように指定します。

構文

dbmlsync -e lt= { **ON** | **OFF** | **SHARE** | **EXCLUSIVE** }; ...

説明

SHARE は、dbmlsync が共有モードのすべての同期テーブルをロックすることを意味します。EXCLUSIVE は、dbmlsync が排他モードのすべての同期テーブルをロックすることを意味します。Windows CE 以外のプラットフォームでは、ON は SHARE と同じです。Windows CE デバイスでは、ON は EXCLUSIVE と同じです。デフォルトは **ON** です。

同期中の修正を許可するには、OFF に設定します。

共有ロックと排他ロックの詳細については、『ASA SQL ユーザーズ・ガイド』> 「ロックの仕組み」 と 『ASA SQL リファレンス・マニュアル』> 「LOCK TABLE 文」を参照してください。

Mobile Link アプリケーションでのテーブルのロックの詳細については、『Mobile Link クライアント』> 「同期中の同時実行性」を参照してください。

同期テーブルが排他モードでロックされているとき (Windows CE デバイスではデフォルト)、他の接続はそのテーブルにアクセスできません。このため、個別の接続を必要とする dbmlsync ストアド・プロシージャは、同期テーブルのいずれかにアクセスする必要がある場合は実行できません。個別の接続を必要とするストアド・プロシージャは、次のとおりです。

- sp_hook_dbmlsync_download_com_error
- sp_hook_dbmlsync_download_fatal_sql_error
- sp_hook_dbmlsync_download_log_ri_violation

このオプションには省略形と長形式があり、**lt** または **LockTables** を使用できます。

拡張オプションのデータベースへの格納もできます。dbmlsync 拡張オプションの詳細については、「dbmlsync 拡張オプション」135 ページを参照してください。

例

次の dbmlsync コマンド・ラインは、dbmlsync を使用するときのこのオプションの設定方法を示します。

```
dbmlsync -e "lt=on"
```

次の SQL 文は、このオプションのデータベースへの格納方法を示します。

```
CREATE SYNCHRONIZATION SUBSCRIPTION
    TO sales_publication
    FOR ml_user1
    OPTION lt='on';
```

Memory (mem) 拡張オプション

機能

キャッシュ・サイズを指定します。

構文

dbmlsync -e mem= *number*[**K** | **M**]; ...

説明

アップロード・ストリームを構築するために使用されるキャッシュをバイト単位で指定します。キャッシュのサイズを大きくすると、メモリ内に保持できるデータのページが多くなるため、ディスクの読み込みと書き込みの回数が減少し、パフォーマンスが向上します。

キロバイトまたはメガバイトの単位を指定するには、それぞれサフィックス k、m を使用します。デフォルトは **1 M** です。

このオプションには省略形と長形式があり、**mem** または **Memory** を使用できます。

拡張オプションのデータベースへの格納もできます。dbmlsync 拡張オプションの詳細については、「dbmlsync 拡張オプション」135 ページを参照してください。

参照

「-urc オプション」193 ページ

『Mobile Link 管理ガイド』> 「パフォーマンスに関するヒント」

例

次の dbmlsync コマンド・ラインは、dbmlsync を使用するときのこのオプションの設定方法を示します。

```
dbmlsync -e "mem=2M"
```

次の SQL 文は、このオプションのデータベースへの格納方法を示します。

```
CREATE SYNCHRONIZATION SUBSCRIPTION
    TO sales_publication
    FOR ml_user1
    OPTION mem='2m';
```

MirrorLogDirectory (mld) 拡張オプション

機能

古いミラー・ログ・ファイルを削除できるようにその場所を指定します。

構文

dbmlsync -e mld= *filename* ...

説明

このオプションにより、次の 2 つの状況のいずれかが発生する場合に、dbmlsync は古いミラー・ログ・ファイルを削除できます。

- ミラー・トランザクション・ログとは別のディレクトリにオフライン・ミラー・ログが置かれる

  または

- リモート・データベース・エンジン以外のマシンで dbmlsync が実行される

通常の設定では、アクティブなミラー・ログと名前の変更されたミラー・トランザクション・ログは同じディレクトリに置かれ、dbmlsync はリモート・データベースと同じマシンで実行されるため、このオプションは不要であり、古いミラー・ログ・ファイルは自動的に削除されます。

このディレクトリ内のトランザクション・ログが影響を受けるのは、DELETE_OLD_LOGS データベース・オプションが ON または DELAY に設定される場合だけです。

このオプションには省略形と長形式があり、**mld** または **MirrorLogDirectory** を使用できます。

拡張オプションのデータベースへの格納もできます。dbmlsync 拡張オプションの詳細については、「dbmlsync 拡張オプション」135 ページを参照してください。

参照

『ASA データベース管理ガイド』> 「DELETE_OLD_LOGS オプション [レプリケーション]」

例

次の dbmlsync コマンド・ラインは、dbmlsync を使用するときのこのオプションの設定方法を示します。

```
dbmlsync -e "mld=c:¥tmp¥file"
```

次の SQL 文は、このオプションのデータベースへの格納方法を示します。

```
CREATE SYNCHRONIZATION SUBSCRIPTION
    TO sales_publication
    FOR ml_user1
    OPTION mld='c:¥tmp¥file';
```

MobiLinkPwd (mp) 拡張オプション

機能

Mobile Link パスワードを指定します。

構文

dbmlsync -e mp=*password*; ...

説明

接続に使用するパスワードを指定します。このパスワードは、サブスクリプションが同期されている Mobile Link ユーザの正しいパスワードにする必要があります。このユーザは、dbmlsync -u オプションで指定することもできます。デフォルト値は **NULL** です。

Mobile Link ユーザがすでにパスワードを持っている場合は、拡張オプション **-e mn** で変更できます。

このオプションには省略形と長形式があり、**mp** または **MobiLinkPwd** を使用できます。

拡張オプションのデータベースへの格納もできます。dbmlsync 拡張オプションの詳細については、「dbmlsync 拡張オプション」135 ページを参照してください。

参照

「NewMobiLinkPwd (mn) 拡張オプション」162 ページ

「-mn オプション」182 ページ

「-mp オプション」182 ページ

例

次の dbmlsync コマンド・ラインは、dbmlsync を使用するときのこのオプションの設定方法を示します。

```
dbmlsync -e "mp=SQL"
```

次の SQL 文は、このオプションのデータベースへの格納方法を示します。

```
CREATE SYNCHRONIZATION SUBSCRIPTION
    TO sales_publication
    FOR ml_user1
    OPTION mp='SQL';
```

NewMobiLinkPwd (mn) 拡張オプション

機能

新しいパスワードを指定します。

構文

dbmlsync -e mn=*new-password*; ...

説明

サブスクリプションが同期されている Mobile Link ユーザのパスワードを指定します。このオプションは、既存のパスワードを変更する場合に使用します。デフォルト値は **NULL** です。

このオプションには省略形と長形式があり、**mn** または **NewMobiLinkPwd** を使用できます。

拡張オプションのデータベースへの格納もできます。dbmlsync 拡張オプションの詳細については、「dbmlsync 拡張オプション」135 ページを参照してください。

参照

「MobiLinkPwd (mp) 拡張オプション」161 ページ

「-mn オプション」182 ページ

「-mp オプション」182 ページ

例

次の dbmlsync コマンド・ラインは、dbmlsync を使用するときのこのオプションの設定方法を示します。

```
dbmlsync -e "mp=oldpassword; mn=newpassword"
```

次の SQL 文は、このオプションのデータベースへの格納方法を示します。

```
CREATE SYNCHRONIZATION SUBSCRIPTION
    TO sales_publication
    FOR ml_user1
    OPTION mn='SQL';
```

OfflineDirectory (dir) 拡張オプション

機能　オフライン・トランザクションのログを含むパスを指定します。

構文　**dbmlsync -e dir=***path*; ...

説明　このオプションには省略形と長形式があり、**dir** または **OfflineDirectory** を使用できます。

拡張オプションのデータベースへの格納もできます。dbmlsync 拡張オプションの詳細については、「dbmlsync 拡張オプション」135 ページを参照してください。

例　次の dbmlsync コマンド・ラインは、dbmlsync を使用するときのこのオプションの設定方法を示します。

```
dbmlsync -e "dir=c:¥db¥logs"
```

次の SQL 文は、このオプションのデータベースへの格納方法を示します。

```
CREATE SYNCHRONIZATION SUBSCRIPTION
    TO sales_publication
    FOR ml_user1
    OPTION dir='c:¥db¥logs';
```

PollingPeriod (pp) 拡張オプション

機能　ログスキャンのポーリング期間を指定します。

構文　**dbmlsync -e pp=***number* [**S** | **M** | **H** | **D**]; ...

説明　このオプションは、ログスキャンの間隔を指定します。サフィックス s、m、h、d を使用して、それぞれ秒、分、時間、日を指定します。デフォルトは **1** 分です。サフィックスを指定しない場合、デフォルトの時間単位は分です。

ログスキャンのポーリングは、同期をスケジュールしているとき、または sp_hook_dbmlsync_delay フックを使用しているときだけ実行されます。

ログスキャンのポーリングの説明は、「DisablePolling (p) 拡張オプション」146 ページを参照してください。

このオプションは、**dbmlsync -pp** と同じです。

このオプションには省略形と長形式があり、**pp** または **PollingPeriod** を使用できます。

拡張オプションのデータベースへの格納もできます。dbmlsync 拡張オプションの詳細については、「dbmlsync 拡張オプション」135 ページを参照してください。

参照

「DisablePolling (p) 拡張オプション」146 ページ

「-pp オプション」187 ページ

「-p オプション」185 ページ

「sp_hook_dbmlsync_delay」243 ページ

例

次の dbmlsync コマンド・ラインは、dbmlsync を使用するときのこのオプションの設定方法を示します。

```
dbmlsync -e "pp=5"
```

次の SQL 文は、このオプションのデータベースへの格納方法を示します。

```
CREATE SYNCHRONIZATION SUBSCRIPTION
    TO sales_publication
    FOR ml_user1
    OPTION pp='5';
```

Schedule (sch) 拡張オプション

機能

同期のスケジュールを指定します。

構文

dbmlsync -e sch=*schedule*; ...

*schedule***=** { **EVERY**:*hhhh*:*mm* | *singleSchedule* | **INFINITE**,… }

hhhh : **00**… **9999**

mm : **00**… **59**

singleSchedule : *day @hh*:*mm*[**AM** | **PM**] [**-***hh*:*mm*[**AM** | **PM**]]

hh : **00**… **24**

mm : **00**… **59**

day :
EVERYDAY | **WEEKDAY** | **MON** | **TUE** | **WED** | **THU** | **FRI** | **SAT** | **SUN** | *dayOfMonth*

dayOfMonth : **0**… **31**

パラメータ

EVERY　EVERY キーワードを指定すると、同期はすぐに発生し、その後は指定した期間の経過後に無限に繰り返されます。指定した期間より同期処理に長時間かかる場合、同期はすぐに再開されます。

singleSchedule　1 つ以上の単一スケジュールを指定すると、同期は指定した日時にのみ発生します。

間隔は *@hh*:*mm*–*hh*:*mm* (AM または PM はオプション指定) と指定します。AM または PM を指定しない場合は、24 時間制とみなされます。24:00 は翌日の午前 00:00 とみなされます。間隔を指定すると、同期は間隔中の任意の時点から始まります。間隔を指定すると同期を行う時間帯に幅ができるため、同じスケジュールを持つ複数のリモート・データベースが、まったく同じ時刻に同期を行うことがなく、同期サーバ側では輻輳が生じなくなります。

間隔の終了時刻は常に開始時刻より後の時刻として解釈されます。間隔に真夜中が含まれている場合は、翌日に終了します。dbmlsync が間隔の途中で始まる場合、同期は終了時刻より前の任意の時点で発生します。

EVERYDAY　EVERYDAY とは週の 7 日間すべてのことです。

WEEKDAY　WEEKDAY とは月曜日から金曜日までのことです。

曜日は、Mon、Tue のようになります。Monday のように、完全な形式を使用することもできます。使用している言語が英語でない場合、接続文字列でクライアントが要求する言語ではない場合、およびサーバのウィンドウに表示される言語ではない場合は、完全な形式の曜日名を使用する必要があります。

dayOfMonth　月の長さに関係なく、月の最終日を指定するには、*dayOfMonth* を 0 に設定します。

INFINITE　INFINITE キーワードを使用すると、dbmlsync が実行され、ログが定期的にスキャンされますが、同期がスケジュールされている場合でも同期されません。同期が発生するのは、別のプログラムから開始されたときだけです。このオプションを dbmlsync -wc オプションと組み合わせて使用すると、dbmlsync をウェイクアップし、同期を行うことができます。

詳細については、「-wc オプション」195 ページを参照してください。

説明

直前の同期がスケジュール時刻にまだ完了していない場合は、その同期が完了した時点で、スケジュールされた同期が開始されます。

デフォルトは、スケジュール設定なしです。

このオプションには省略形と長形式があり、**sch** または **Schedule** を使用できます。

拡張オプションのデータベースへの格納もできます。dbmlsync 拡張オプションの詳細については、「dbmlsync 拡張オプション」135 ページを参照してください。

schedule オプションの構文は、同期 SQL 文と dbmlsync コマンド・ラインのどちらの場合も同じです。

IgnoreScheduling 拡張オプションと -is オプションは、dbmlsync がスケジュールを無視して即時の同期を行えるようにします。詳細については、「IgnoreScheduling (isc) 拡張オプション」155 ページを参照してください。

スケジュールの詳細については、『Mobile Link クライアント』> 「同期のスケジュール」を参照してください。

例

次の dbmlsync コマンド・ラインは、dbmlsync を使用するときのこのオプションの設定方法を示します。

```
dbmlsync -e "sch=WEEKDAY@8:00am,SUN@9:00pm"
```

次の SQL 文は、このオプションのデータベースへの格納方法を示します。

```
CREATE SYNCHRONIZATION SUBSCRIPTION
    TO sales_publication
    FOR ml_user1
    OPTION sch='WEEKDAY@8:00am,SUN@9:00pm';
```

ScriptVersion (sv) 拡張オプション

機能

スクリプト・バージョンを指定します。

構文

dbmlsync -e sv=*version-name*; ...

説明

スクリプト・バージョンは、同期中に統合データベースの Mobile Link によって実行されるスクリプトを決定します。デフォルトのスクリプト・バージョンは **default** です。

このオプションには省略形と長形式があり、**sv** または **ScriptVersion** を使用できます。

拡張オプションのデータベースへの格納もできます。dbmlsync 拡張オプションの詳細については、「dbmlsync 拡張オプション」135 ページを参照してください。

例

次の dbmlsync コマンド・ラインは、dbmlsync を使用するときのこのオプションの設定方法を示します。

```
dbmlsync -e "sv=SyaAd001"
```

次の SQL 文は、このオプションのデータベースへの格納方法を示します。

```
CREATE SYNCHRONIZATION SUBSCRIPTION
    TO sales_publication
    FOR ml_user1
    OPTION sv='SysAd001';
```

SendColumnNames (scn) 拡張オプション

機能

カラム名がアップロードで送信されるように指定します。

構文

dbmlsync -e scn= { **ON** | **OFF** }; ...

説明

このオプションを ON に設定すると、dbmlsync はリモート・データベースからサーバにカラム名を送信します。dbmlsrv9 -za または -ze オプションを使用して自動的にスクリプトを生成する場合に、このオプションが必要です。このオプションはアップロードのサイズを大きくするため、-za または -ze を使用していない場合はおすすめしません。

デフォルトは **OFF** です。

このオプションには省略形と長形式があり、**scn** または **SendColumnNames** を使用できます。

拡張オプションのデータベースへの格納もできます。dbmlsync 拡張オプションの詳細については、「dbmlsync 拡張オプション」135 ページを参照してください。

参照

『Mobile Link 管理ガイド』> 「-za オプション」

-ze オプション

例

次の dbmlsync コマンド・ラインは、dbmlsync を使用するときのこのオプションの設定方法を示します。

```
dbmlsync -e "scn=on"
```

次の SQL 文は、このオプションのデータベースへの格納方法を示します。

```
CREATE SYNCHRONIZATION SUBSCRIPTION
    TO sales_publication
    FOR ml_user1
    OPTION scn='on';
```

SendDownloadACK (sa) 拡張オプション

機能

ダウンロード受信確認がクライアントからサーバへ送信されるように指定します。

構文

dbmlsync -e sa= { **ON** | **OFF** }; ...

説明

確認をオフにすると (デフォルト)、統合データベースでの競合が減り、ダウンロード・トランザクションの時間が短縮されるためにスループットが向上します。ダウンロード・トランザクションの時間が短縮されるのは、リモート・クライアントがダウンロードを適用する間 Mobile Link がこれらのトランザクションを開いている必要がなくなり、できるかぎり早急にトランザクションがコミットまたはロールバックされるからです。クライアント側でダウンロード・バッファリングを有効にすると、ダウンロード肯定応答がなくなり最大限のパフォーマンスを実現できます。SendDownloadAck を OFF に設定することをおすすめします。

ダウンロード確認をオフにすることによるパフォーマンスの向上については、『Mobile Link 管理ガイド』> 「パフォーマンスに関するヒント」を参照してください。

注意：SendDownloadAck が ON に設定され、冗長モードである場合、受信確認行はクライアント・ログに書き込まれます。

デフォルトは **OFF** です。

このオプションには省略形と長形式があり、**sa** または **SendDownloadACK** を使用できます。

拡張オプションのデータベースへの格納もできます。dbmlsync 拡張オプションの詳細については、「dbmlsync 拡張オプション」135 ページを参照してください。

例

次の dbmlsync コマンド・ラインは、dbmlsync を使用するときのこのオプションの設定方法を示します。

```
dbmlsync -e "sa=off"
```

次の SQL 文は、このオプションのデータベースへの格納方法を示します。

```
CREATE SYNCHRONIZATION SUBSCRIPTION
    TO sales_publication
    FOR ml_user1
    OPTION sa='off';
```

SendTriggers (st) 拡張オプション

機能

アップロード時にトリガの動作が送信されるように指定します。

構文

dbmlsync -e st= { **ON** | **OFF** }; ...

説明

カスケード削除もトリガの動作と見なされます。

デフォルトは **OFF** です。

このオプションには省略形と長形式があり、**st** または **SendTriggers** を使用できます。

拡張オプションのデータベースへの格納もできます。dbmlsync 拡張オプションの詳細については、「dbmlsync 拡張オプション」135 ページを参照してください。

例

次の dbmlsync コマンド・ラインは、dbmlsync を使用するときのこのオプションの設定方法を示します。

```
dbmlsync -e "st=on"
```

次の SQL 文は、このオプションのデータベースへの格納方法を示します。

```
CREATE SYNCHRONIZATION SUBSCRIPTION
    TO sales_publication
    FOR ml_user1
    OPTION st='on';
```

TableOrder (tor) 拡張オプション

機能

アップロード・ストリームでのテーブルの順序を指定します。

構文

dbmlsync -e tor= *tables* ; ...

説明

このオプションを使用すると、アップロードされるテーブルの順序を指定できます。*tables* はカンマで区切ったリストで指定します。アップロードされるすべてのテーブルを指定する必要があります。同期に含まれていないテーブルをリストすると、そのテーブルは無視されます。

テーブルの順序を指定することにより、参照整合性を確実にします。たとえば、Table1 が Table2 を参照する場合、Table2 は Table1 の前にアップロードする必要があります。

指定したテーブルの順序では、テーブルが循環外部キー関係を持たないかぎり、順序が後に来るテーブルを参照する外部キーを持つテーブルがあってはいけません。デフォルトでは、dbmlsync はこの条件を満たすテーブルの順序を選択します。

このオプションを使用しなければならないケースはありません。このオプションは、何らかの理由で (通常は統合データベースでスクリプトを簡単にするため) 特定の順序でテーブルをアップロードするユーザのために提供されています。

このオプションには省略形と長形式があり、**tor** または **TableOrder** を使用できます。

拡張オプションのデータベースへの格納もできます。dbmlsync 拡張オプションの詳細については、「dbmlsync 拡張オプション」135 ページを参照してください。

参照

『Mobile Link 管理ガイド』> 「参照整合性と同期」

例

次の dbmlsync コマンド・ラインは、dbmlsync を使用するときのこのオプションの設定方法を示します。

```
dbmlsync -e "tor=admin,parent,child"
```

次の SQL 文は、このオプションのデータベースへの格納方法を示します。

```
CREATE SYNCHRONIZATION SUBSCRIPTION
    TO sales_publication
    FOR ml_user1
    OPTION tor='admin,parent,child';
```

UploadOnly (uo) 拡張オプション

機能

同期がアップロードだけを含むように指定します。

構文

dbmlsync -e uo={ ON | OFF } ; ...

説明

アップロード専用の同期中に、dbmlsync は通常の完全な同期と全く同じように、Mobile Link 同期サーバへのアップロードを準備し送信します。しかし、ダウンロード・ストリームを返信する代わりに、Mobile Link はアップロードのコミットが成功したかどうかを示す確認だけを送信します。

デフォルトは **OFF** です。

このオプションには省略形と長形式があり、**uo** または **UploadOnly** を使用できます。

拡張オプションのデータベースへの格納もできます。dbmlsync 拡張オプションの詳細については、「dbmlsync 拡張オプション」135 ページを参照してください。

参照

『Mobile Link 管理ガイド』> 「アップロード専用の同期とダウンロード専用の同期」

「DownloadOnly (ds) 拡張オプション」148 ページ

例

次の dbmlsync コマンド・ラインは、dbmlsync を使用するときのこのオプションの設定方法を示します。

```
dbmlsync -e "uo=on"
```

次の SQL 文は、このオプションのデータベースへの格納方法を示します。

```
CREATE SYNCHRONIZATION SUBSCRIPTION
    TO sales_publication
    FOR ml_user1
    OPTION uo='on';
```

Verbose (v) 拡張オプション

機能

完全冗長を指定します。

構文

dbmlsync -e v= { **ON** | **OFF** } ; ...

説明

このオプションが指定する高いレベルの冗長性は、パフォーマンスに影響する場合があるので、通常は開発段階にだけ使用してください。

このオプションは、**dbmlsync -v+** と同じです。-v オプションと拡張オプションの両方を指定して、競合が発生した場合は、-v オプションが拡張オプションよりも優先されます。競合が発生しない場合は、冗長ログ・オプションが追加され、指定したすべてのオプションが使用されます。拡張オプションでロギングの冗長性を設定しても、ロギングは直ちに有効にならないため、起動情報のログは取られません。最初の同期が行われる時点では、-v オプションを指定した場合と拡張オプションを指定した場合のロギングの動作は同じです。

詳細については、「-v オプション」194 ページを参照してください。

デフォルトは **OFF** です。

このオプションには省略形と長形式があり、**v** または **Verbose** を使用できます。

拡張オプションのデータベースへの格納もできます。dbmlsync 拡張オプションの詳細については、「dbmlsync 拡張オプション」135 ページを参照してください。

例

次の dbmlsync コマンド・ラインは、dbmlsync を使用するときのこのオプションの設定方法を示します。

```
dbmlsync -e "v=on"
```

次の SQL 文は、このオプションのデータベースへの格納方法を示します。

```
CREATE SYNCHRONIZATION SUBSCRIPTION
    TO sales_publication
    FOR ml_user1
    OPTION v='on';
```

VerboseHooks (vs) 拡張オプション

機能

フック・スクリプトに関係するメッセージのログを取るように指定します。

構文 **dbmlsync -e vs=** { **ON** | **OFF** } ; ...

説明 このオプションは、**dbmlsync -vs** と同じです。-v オプションと拡張オプションの両方を指定して、競合が発生した場合は、-v オプションが拡張オプションよりも優先されます。競合が発生しない場合は、冗長ログ・オプションが追加され、指定したすべてのオプションが使用されます。拡張オプションでロギングの冗長性を設定しても、ロギングは直ちに有効にならないため、起動情報のログは取られません。最初の同期が行われる時点では、-v オプションを指定した場合と拡張オプションを指定した場合のロギングの動作は同じです。

詳細については、「-v オプション」194 ページを参照してください。

デフォルトは **OFF** です。

このオプションには省略形と長形式があり、**vs** または **VerboseHooks** を使用できます。

拡張オプションのデータベースへの格納もできます。dbmlsync 拡張オプションの詳細については、「dbmlsync 拡張オプション」135 ページを参照してください。

参照 「dbmlsync クライアント・イベント・フック」225 ページ

例 次の dbmlsync コマンド・ラインは、dbmlsync を使用するときのこのオプションの設定方法を示します。

```
dbmlsync -e "vs=on"
```

次の SQL 文は、このオプションのデータベースへの格納方法を示します。

```
CREATE SYNCHRONIZATION SUBSCRIPTION
    TO sales_publication
    FOR ml_user1
    OPTION vs='on';
```

VerboseMin (vm) 拡張オプション

機能 少量の情報のログを取るように指定します。

構文 **dbmlsync -e vm=** { **ON** | **OFF** } ; ...

説明

このオプションは、**dbmlsync -v** と同じです。-v オプションと拡張オプションの両方を指定して、競合が発生した場合は、-v オプションが拡張オプションよりも優先されます。競合が発生しない場合は、冗長ログ・オプションが追加され、指定したすべてのオプションが使用されます。拡張オプションでロギングの冗長性を設定しても、ロギングは直ちに有効にならないため、起動情報のログは取られません。最初の同期が行われる時点では、-v オプションを指定した場合と拡張オプションを指定した場合のロギングの動作は同じです。

詳細については、「-v オプション」194 ページを参照してください。

デフォルトは **OFF** です。

このオプションには省略形と長形式があり、**vm** または **VerboseMin** を使用できます。

拡張オプションのデータベースへの格納もできます。dbmlsync 拡張オプションの詳細については、「dbmlsync 拡張オプション」135 ページを参照してください。

例

次の dbmlsync コマンド・ラインは、dbmlsync を使用するときのこのオプションの設定方法を示します。

```
dbmlsync -e "vm=on"
```

次の SQL 文は、このオプションのデータベースへの格納方法を示します。

```
CREATE SYNCHRONIZATION SUBSCRIPTION
    TO sales_publication
    FOR ml_user1
    OPTION vm='on';
```

VerboseOptions (vo) 拡張オプション

機能

指定したコマンド・ライン・オプション (拡張オプションを含む) に関する情報のログを取るよう指定します。

構文

dbmlsync -e vo= { **ON** | **OFF** } ; ...

説明

このオプションは、**dbmlsync -vo** と同じです。-v オプションと拡張オプションの両方を指定して、競合が発生した場合は、-v オプションが拡張オプションよりも優先されます。競合が発生しない場合は、冗長ログ・オプションが追加され、指定したすべてのオプションが使用されます。拡張オプションでロギングの冗長性を設定しても、ロギングは直ちに有効にならないため、起動情報のログは取られません。最初の同期が行われる時点では、-v オプションを指定した場合と拡張オプションを指定した場合のロギングの動作は同じです。

詳細については、「-v オプション」194 ページを参照してください。

デフォルトは **OFF** です。

このオプションには省略形と長形式があり、**vo** または **VerboseOptions** を使用できます。

拡張オプションのデータベースへの格納もできます。dbmlsync 拡張オプションの詳細については、「dbmlsync 拡張オプション」135 ページを参照してください。

例

次の dbmlsync コマンド・ラインは、dbmlsync を使用するときのこのオプションの設定方法を示します。

```
dbmlsync -e "vo=on"
```

次の SQL 文は、このオプションのデータベースへの格納方法を示します。

```
CREATE SYNCHRONIZATION SUBSCRIPTION
    TO sales_publication
    FOR ml_user1
    OPTION vo='on';
```

VerboseRowCounts (vn) 拡張オプション

機能

アップロードおよびダウンロードされるローの数のログを取るように指定します。

構文

dbmlsync -e vn= { **ON** | **OFF** } ; ...

説明

このオプションは、**dbmlsync -vn** と同じです。-v オプションと拡張オプションの両方を指定して、競合が発生した場合は、-v オプションが拡張オプションよりも優先されます。競合が発生しない場合は、冗長ログ・オプションが追加され、指定したすべてのオプションが使用されます。拡張オプションでロギングの冗長性を設定しても、ロギングは直ちに有効にならないため、起動情報のログは取られません。最初の同期が行われる時点では、-v オプションを指定した場合と拡張オプションを指定した場合のロギングの動作は同じです。

詳細については、「-v オプション」194 ページを参照してください。

デフォルトは **OFF** です。

このオプションには省略形と長形式があり、**vn** または **VerboseRowCounts** を使用できます。

拡張オプションのデータベースへの格納もできます。dbmlsync 拡張オプションの詳細については、「dbmlsync 拡張オプション」135 ページを参照してください。

例

次の dbmlsync コマンド・ラインは、dbmlsync を使用するときのこのオプションの設定方法を示します。

```
dbmlsync -e "vn=on"
```

次の SQL 文は、このオプションのデータベースへの格納方法を示します。

```
CREATE SYNCHRONIZATION SUBSCRIPTION
    TO sales_publication
    FOR ml_user1
    OPTION vn='on';
```

VerboseRowValues (vr) 拡張オプション

機能

アップロードおよびダウンロードされるローの値のログを取るように指定します。

構文

dbmlsync -e vr= { **ON** | **OFF** } ; ...

説明

このオプションは、**dbmlsync -vr** と同じです。-v オプションと拡張オプションの両方を指定して、競合が発生した場合は、-v オプションが拡張オプションよりも優先されます。競合が発生しない場合は、冗長ログ・オプションが追加され、指定したすべてのオプションが使用されます。拡張オプションでロギングの冗長性を設定しても、ロギングは直ちに有効にならないため、起動情報のログは取られません。最初の同期が行われる時点では、-v オプションを指定した場合と拡張オプションを指定した場合のロギングの動作は同じです。

詳細については、「-v オプション」194 ページを参照してください。

デフォルトは **OFF** です。

このオプションには省略形と長形式があり、**vr** または **VerboseRowValues** を使用できます。

拡張オプションのデータベースへの格納もできます。dbmlsync 拡張オプションの詳細については、「dbmlsync 拡張オプション」135 ページを参照してください。

例

次の dbmlsync コマンド・ラインは、dbmlsync を使用するときのこのオプションの設定方法を示します。

```
dbmlsync -e "vr=on"
```

次の SQL 文は、このオプションのデータベースへの格納方法を示します。

```
CREATE SYNCHRONIZATION SUBSCRIPTION
    TO sales_publication
    FOR ml_user1
    OPTION vr='on';
```

VerboseUpload (vu) 拡張オプション

機能

アップロード・ストリームに関する情報のログを取るように指定します。

構文

dbmlsync -e vu= { **ON** | **OFF** } ; ...

説明

このオプションは、**dbmlsync -vu** と同じです。-v オプションと拡張オプションの両方を指定して、競合が発生した場合は、-v オプションが拡張オプションよりも優先されます。競合が発生しない場合は、冗長ログ・オプションが追加され、指定したすべてのオプションが使用されます。拡張オプションでロギングの冗長性を設定しても、ロギングは直ちに有効にならないため、起動情報のログは取られません。最初の同期が行われる時点では、-v オプションを指定した場合と拡張オプションを指定した場合のロギングの動作は同じです。

詳細については、「-v オプション」194 ページを参照してください。

デフォルトは **OFF** です。

このオプションには省略形と長形式があり、**vu** または **VerboseUpload** を使用できます。

拡張オプションのデータベースへの格納もできます。dbmlsync 拡張オプションの詳細については、「dbmlsync 拡張オプション」135 ページを参照してください。

例

次の dbmlsync コマンド・ラインは、dbmlsync を使用するときのこのオプションの設定方法を示します。

```
dbmlsync -e "vu=on"
```

次の SQL 文は、このオプションのデータベースへの格納方法を示します。

```
CREATE SYNCHRONIZATION SUBSCRIPTION
    TO sales_publication
    FOR ml_user1
    OPTION vu='on';
```

-eh オプション

機能

フック関数で発生したエラーを無視します。

構文

dbmlsync -eh …

-ek オプション

機能

高度に暗号化されたデータベースの暗号化キーをコマンド・ラインで直接指定できます。

構文

dbmlsync -ek *key* …

説明

高度に暗号化されたデータベースを扱う場合、データベースやトランザクション・ログ (オフライン・トランザクションなど) を使用するには、常に暗号化キーを使用します。高度に暗号化されたデータベースの場合、-ek または -ep をどちらか指定します。両方同時には指定できません。高度に暗号化されたデータベースでは、キーを指定しないとコマンドが失敗します。

-ep オプション

機能

暗号化キーを入力するよう要求します。

構文

dbmlsync -ep …

説明

このオプションでは、暗号化キーを入力するためのダイアログ・ボックスが表示されます。クリア・テキストでは暗号化キーを見ることができないようにすることで、高いセキュリティが得られます。高度に暗号化されたデータベースの場合、-ek または -ep をどちらか指定します。両方同時には指定できません。高度に暗号化されたデータベースでは、キーを指定しないとコマンドが失敗します。

-eu オプション

機能

拡張アップロード・オプションを指定します。

構文

dbmlsync -n *publication-name* **-eu** *keyword=value*;…

説明

コマンド・ラインで -eu オプションを使用して指定した拡張オプションは、その直前の -n オプションで指定される同期のみに適用されます。たとえば、次のコマンド・ラインでは、拡張オプション sv=test は pub2 の同期のみに適用されます。

```
dbmlsync -n pub1 -n pub2 -eu sv=test
```

拡張オプションが複数設定された場合の処理方法については、「dbmlsync 拡張オプション」135 ページを参照してください。

拡張オプションの完全なリストは、「dbmlsync 拡張オプション」135 ページを参照してください。

-is オプション

機能　スケジュールの指示を無視して、直ちに同期を行います。

構文　**dbmlsync -is** …

説明　同期をスケジュールする拡張オプションを無視します。

スケジュールの詳細については、『Mobile Link クライアント』> 「同期のスケジュール」を参照してください。

-k オプション

機能　同期が終了したときに dbmlsync を停止します。

構文　**dbmlsync -k** …

説明　-k オプションを使用すると、同期が成功するか、-o オプションまたは -ot オプションを使用して出力ログ・ファイルが指定されている場合に、同期の完了後に dbmlsync を終了します。

参照　「-o オプション」183 ページ

「-ot オプション」184 ページ

-l オプション

機能　使用可能な拡張オプションをリストします。

構文 **dbmlsync -l** …

説明 dbmlsync コマンド・ラインと一緒に使用すると、使用可能な拡張オプションがすべて表示されます。

-mn オプション

機能 同期するユーザの新しいパスワードを指定します。

構文 **dbmlsync -mn** *password* …

説明 Mobile Link ユーザのパスワードを変更します。

詳細については、「Mobile Link ユーザの認証」9 ページを参照してください。

参照 「MobiLinkPwd (mp) 拡張オプション」161 ページ

「NewMobiLinkPwd (mn) 拡張オプション」162 ページ

「-mp オプション」182 ページ

-mp オプション

機能 同期するユーザのパスワードを指定します。

構文 **dbmlsync -mp** *password* …

説明 Mobile Link ユーザ認証用のパスワードを指定します。

詳細については、「Mobile Link ユーザの認証」9 ページを参照してください。

参照 「MobiLinkPwd (mp) 拡張オプション」161 ページ

「NewMobiLinkPwd (mn) 拡張オプション」162 ページ

「-mn オプション」182 ページ

-n オプション

機能　同期パブリケーションの名前を指定します。

構文　**dbmlsync -n** *pubname* …

説明　同期パブリケーションの名前です。-n オプションを複数指定すると、複数の同期パブリケーションを同期できます。

-n を使用して複数のパブリケーションを同期するには、次の 2 つの方法があります。

- `-n pub1,pub2,pub3` と指定して、1 つのアップロード・ストリームで pub1、pub2、pub3 をアップロードする。

  この方法では、各パブリケーションで拡張オプションを設定した場合、リスト内の最初のパブリケーションで設定したオプションのみが使用されます。2 番目以降のパブリケーションで設定した拡張オプションは無視されます。

- `-n pub1 -n pub2 -n pub3` と指定して、最初のアップロード・ストリームで pub1、2 番目で pub2、3 番目で pub3 をアップロードする。

  `-n pub1 -n pub2` を指定するなど、連続した同期が非常に速く行われると、サーバがまだ前の同期を処理しているときに、dbmlsync が同期の処理を開始する場合があります。この場合、2 番目の同期は、同時同期ができないことを示すエラーで失敗します。この状況が発生した場合は、sp_hook_dbmlsync_delay ストアド・プロシージャを定義して、各同期の前に遅延を作成できます。通常は数秒から 1 分が十分な遅延です。

  詳細については、「sp_hook_dbmlsync_delay」243 ページを参照してください。

-o オプション

機能　出力をログ・ファイルに送信します。

構文　**dbmlsync -o** *filename* …

説明

出力をログ・ファイルに追加します。デフォルトでは、画面に出力が表示されます。

参照

「-os オプション」184 ページ

「-ot オプション」184 ページ

-os オプション

機能

出力ログの最大サイズを指定します (このサイズに達するとログの名前が変更される)。

構文

dbmlsync -os *size* [**K** | **M** | **G**]...

説明

size には、出力メッセージのログを取るファイルの最大サイズを、バイト単位で指定します。単位をキロバイト、メガバイト、またはギガバイトで指定するには、それぞれサフィックス k、m、または g を使用してください。デフォルトでは、サイズは無制限となります。最小のサイズ制限は 10 KB です。

dbmlsync ユーティリティは、現在のファイル・サイズを確認してから、ファイルに出力メッセージのログを取ります。ログ・メッセージのファイル・サイズが指定したサイズを超えた場合、dbmlsync ユーティリティは出力ファイルの名前を *yymmddxx.dbr* に変更します。*yymmdd* は年月日を表し、*xx* は AA ～ ZZ のアルファベット順の英字です。

このオプションを使用すると、古いログ・ファイルを手動で削除してディスク領域を解放できます。

参照

「-o オプション」183 ページ

「-ot オプション」184 ページ

-ot オプション

機能

ログ・ファイルをトランケートし、このファイルに出力メッセージを追加します。

構文　**dbmlsync -ot** *logfile* …

説明　機能は -o オプションと同じですが、ログ・ファイルがトランケートされてからメッセージが書き込まれます。

参照　「-o オプション」183 ページ

「-os オプション」184 ページ

-p オプション

機能　ログスキャンのポーリングを無効にします。

構文　**dbmlsync -p** …

説明　アップロード・ストリームを構築するには、dbmlsync はトランザクション・ログをスキャンする必要があります。通常、これは同期直前に行います。しかし、同期がスケジュールされている場合、または sp_hook_dbmlsync_delay フックが使用されている場合、dbmlsync はデフォルトでは同期直前に発生する一時停止でログをスキャンします。同期が開始されるとログはすでに少なくとも一部がスキャンされるため、この動作はより効率的です。このデフォルトの動作は、ログスキャンのポーリングと呼ばれます。

ログスキャンのポーリングはデフォルトではオンになっていますが、スケジュール・オプションを使用して同期がスケジュールされているとき、または sp_hook_dbmlsync_delay フックが使用されているときしか効果がありません。ポーリングが有効な場合は、設定された間隔で実行されます。これはデフォルトでは 1 分ですが、dbmlsync -pp オプションで変更できます。

デフォルトでは、ログスキャンのポーリングを無効にしません (**OFF**)。

このオプションは、**dbmlsync -e p** と同じです。

参照　「DisablePolling (p) 拡張オプション」146 ページ

「PollingPeriod (pp) 拡張オプション」163 ページ

「-pp オプション」187 ページ

-pd オプション

機能

Windows CE 用の指定された dll をプリロードします。

構文

dbmlsync -pd *dllname*;...

説明

Windows CE で dbmlsync を実行するとき、-pd オプションを使用して、ロードする必要がある dll を指定します。そうしないと、正しい dll がロードされず、エラーが発生する場合があります。

次に、各通信プロトコル用にロードする必要がある dll を示します。

プロトコル	DLL
TCP/IP	dbmlsock9.dll
HTTP	dbmlhttp9.dll
HTTPS	dbmlhttps9.dll

複数の dll は、セミコロンで区切ったリストで指定します。次に例を示します。

```
-pd dbmlsock9.dll;dbmlhttp9.dll
```

-pi オプション

機能

Mobile Link 同期サーバを ping します。

構文

dbmlsync -pi -c *connection_string* **-e sv=***script_version*
[**-n** *pubname*] [**-u** *ml_username*]

説明

ping オプションを使用すると、接続情報が正しいかどうかをテストできます。-pi を使用すると、dbmlsync は同期を開始しません。

ping できるようにするには、Mobile Link 同期サーバのユニークなアドレスを指定してください。つまり、接続パラメータとスクリプト・バージョンの他に、パブリケーション名または Mobile Link ユーザ名 (あるいはその両方) を含めてください。パブリケーションまたはユーザはリモートの接続情報を保持しています。ユーザのサブスクリプションが複数のパブリケーションに対して作成されている場合、またはパブリケーションに複数のユーザのサブスクリプションがある場合は、パブリケーション名とユーザ名の両方を指定してください。たとえば、パブリケーションに対するサブスクリプションが 1 つだけの場合は、ユーザを指定せずにパブリケーションを指定できます。

Mobile Link 同期サーバは、ping 要求を受信すると、統合データベースに接続し、ユーザを認証し、ユーザ認証ステータスと値をクライアント (dbmlsync または Ultra Light) に送信します。

ping に成功した場合、Mobile Link サーバは情報メッセージを発行します。ping に失敗した場合は、エラー・メッセージを発行します。

Mobile Link サーバがコマンド・ライン・オプション -zu+ を指定して実行されていると、Mobile Link ユーザ名が ml_user システム・テーブルに見つからない場合は Mobile Link サーバがユーザを ml_user に追加します。

Mobile Link 同期サーバは次のスクリプトを実行できます (スクリプトが存在する場合)。

- begin_connection
- authenticate_user
- authenticate_user_hashed
- end_connection

クライアントは、サーバの ping 中は同期を実行できません。

-pp オプション

機能　ログスキャンの頻度を指定します。

構文 **dbmlsync -pp** *number* [**h** | **m** | **s**]…

説明 このオプションは、ログスキャンの間隔を指定します。サフィックス s、m、h、d を使用して、それぞれ秒、分、時間、日を指定します。デフォルトは **1** 分です。サフィックスを指定しない場合、デフォルトの時間単位は分です。

ログスキャンのポーリングは、同期をスケジュールしているとき、または sp_hook_dbmlsync_delay フックを使用しているときだけ実行されます。

ログスキャンのポーリングの説明は、「-p オプション」185 ページを参照してください。

参照 「PollingPeriod (pp) 拡張オプション」163 ページ

「DisablePolling (p) 拡張オプション」146 ページ

「-p オプション」185 ページ

-q オプション

機能 Mobile Link 同期クライアントを最小化ウィンドウで起動します。

構文 **dbmlsync -q** …

説明 これは、Windows オペレーティング・システム専用です。

-r オプション

機能 リモート・データベースと統合データベースのオフセットが一致しないとき、リモート・オフセットを使用するように指定します。

リモート・オフセットが統合オフセットよりも小さい場合に (リモート・データベースがバックアップからリストアされたときなど)、-rb オプションを使用できます。-r オプションは下位互換のために提供されており、-rb と同じです。統合オフセットよりもリモート・オフ

セットが大きい場合に使用する -ra オプションは、非常にまれな環境のためだけに提供されており、データ損失の原因となる可能性があります。

構文

dbmlsync { **-r | -ra | -rb** } …

説明

進行オフセットの詳細については、「進行オフセット」79 ページを参照してください。

-rb　リモート・データベースがバックアップからリストアされると、デフォルトの動作がデータ損失の原因となることがあります。この場合、リモート・データベースをリストアした後、初めて dbmlsync を実行するときに -rb を指定します。-rb を使用すると、リモート・データベースに記録されたオフセットが統合データベースから取得したオフセットよりも小さい場合、リモート・データベースに記録されているオフセットからアップロードが継続されます。-rb を使用し、リモートのオフセットが統合データベースからのオフセットより小さい場合、エラーが報告され、同期がアボートされます。

-rb オプションでは、すでにアップロードされたデータがアップロードされることがあります。これにより統合データベースで競合が起こる可能性があり、これは適切な競合解決スクリプトを使用して処理する必要があります。

-ra　-ra オプションは、非常にまれな場合にのみ使用してください。-ra を使用すると、リモートのオフセットが統合データベースから取得したオフセットよりも大きい場合、アップロードはリモート・データベースから取得したオフセットからリトライされます。-ra を使用し、リモートのオフセットが統合データベースからのオフセットより大きくない場合、エラーが報告され、同期がアボートされます。

-ra オプションの使用には注意してください。統合データベースをリストアした結果、オフセットが一致しなければ、2 つのオフセット間の差のうちリモート・データベース内で発生した変更が失われます。-ra オプションは、統合データベースがバックアップからリストアされ、リモート・データベースのトランザクション・ログがリモートのオフセットと同じポイントでトランケートされた場合に役立ちます。この場合、リモート・データベースからアップロードされたすべてのデータは、統合オフセットのポイントからリモート・オフセットのポイントまで失われます。

-sc オプション

機能

dbmlsync が各同期の前にスキーマ情報を再ロードするように指定します。

構文

dbmlsync -sc…

説明

バージョン 9.0 以前では、dbmlsync は各同期の前にデータベースからスキーマ情報を再ロードしていました。再ロードされた情報には、外部キー関係、パブリケーションの定義、データベースに格納された拡張オプション、データベースの設定に関する情報が含まれていました。速度の遅いハンドヘルド・デバイスでは、この情報をロードするのに通常 20 秒かかっていました。ほとんどの場合、この情報は同期と同期の間に変わることはありません。

バージョン 9.0 以降では、dbmlsync はデフォルトで起動時にのみスキーマ情報をロードします。各同期の前にこの情報をロードする場合は、-sc を指定します。

-tu オプション

機能

リモート・データベースの各トランザクションを、1 つの同期内で独立したトランザクションとしてアップロードするように指定します。

構文

dbmlsync -tu …

説明

-tu オプションを使用するときは、「**トランザクション・レベルのアップロード**」を作成します。こうすると、dbmlsync はリモート・データベースの各トランザクションを別個のトランザクションとしてアップロードします。リモート・データベースで同じローを 3 回変更し、そのたびに変更をコミットすると、この 3 つのトランザクションはいずれも次のアップロード・ストリームで適用されます。これらのトランザクションは、単一の接続の、単一の同期内でアップロードされますが、各トランザクションは正常にアップロードされた直後にコミットされます。

-tu オプションを使用すると、リモート・データベースのトランザクションの順序は、常に統合データベースで保持されます。ただし、トランザクションでの操作の順序は保持されないことがあります。これには、以下の 2 つの理由があります。

- Mobile Link は、常に外部キー関係に基づいて更新を適用します。たとえば、子と親のテーブルでデータが変更されると、Mobile Link はデータの挿入を親のテーブル、子のテーブルの順に行いますが、データの削除は子のテーブル、親のテーブルの順に行います。リモートの操作がこの順序に従っていない場合は、統合データベースでは操作の順序が異なることになります。
- トランザクション内の操作は結合されます。つまり、1 つのトランザクションで同じローを 3 回変更しても、最後の形式のローのみがアップロードされます。

使用法

-tu オプションを使用しないと、Mobile Link はリモート・データベースでのすべての変更を結合し、アップロード・ストリームの 1 つのトランザクションにします。つまり、同期と同期の間に同じローを 3 回変更しても、リモート・トランザクションの数に関係なく、最後の形式のローのみがアップロードされます。このデフォルトの動作は、効率がよく、多くの状況に最適です。

ただし、特定の状況では、統合データベースでリモート・トランザクションを保持したい場合があります。たとえば、リモート・データベースでトランザクションが発生したときに、そのトランザクションに基づいてアクションを行うトリガを統合データベースで定義したいことがあります。または、同期が発生した時点に関係なく、リモート・データベースのすべての変更が統合データベースに反映されるようにしたいこともあります。

さらに、アップロード・ストリームを小さいトランザクションに分割することには利点があります。多くの統合データベースは、小さなトランザクションを対象として最適化されているため、巨大なトランザクションは効率的でなく、多数の競合が発生することがあります。また、-tu オプションを使用すると、各トランザクションは正常にアップロードされたときに適用されるため、アップロード中に通信エラーが発生してもアップロード全体が失われることがありません。-tu オプションを使用している場合にアップロード・エラーが発生しても、正常にアップロードされたすべてのトランザクションが適用されます。

-tu オプションを使用しているときは、Mobile Link 同期サーバを -us オプションを付けて起動するとパフォーマンスが向上します。この -us オプションは、アップロードするデータがないテーブルのスクリプトを Mobile Link が呼び出さないようにします。

-tu オプションを指定すると、Mobile Link は SQL Remote とほぼ同じように動作します。主な相違点は、SQL Remote がすべての変更を発生順にリモート・データベースにレプリケートし、結合を行わないことです。-tu オプションを使用すると Mobile Link はすべてのトランザクションをレプリケートするため、このように動作させるには、すべての変更をコミットする必要があります。

Increment 拡張オプションに -tu を使用することはできません。

-u オプション

機能

Mobile Link ユーザ名を指定します。

構文

dbmlsync -u *ml_username* …

説明

dbmlsync コマンド・ラインでユーザを 1 人指定できます。この場合、*ml_username* は、処理するサブスクリプションに対応する CREATE SYNCHRONIZATION SUBSCRIPTION 文の FOR 句に使用されているユーザ名です。

このオプションを -n *publication* と併用して、dbmlsync が操作するサブスクリプションを識別します。各サブスクリプションは、*ml_username*, *publication* の組み合わせでユニークに識別されます。

コマンド・ラインで指定できるユーザ名は 1 つだけです。1 回の処理で同期するサブスクリプションはすべて、同じユーザと関連していなければなりません。-n オプションを使用してコマンド・ラインで指定した各パブリケーションにサブスクリプションが 1 つしかない場合は、-u オプションを省略できます。

-uo オプション

機能

同期にはアップロードのみが含まれ、ダウンロードは発生されないよう指定します。

構文

dbmlsync -uo…

説明

アップロード専用の同期中に、dbmlsync は通常の完全な同期と全く同じように、Mobile Link へのアップロードを準備し送信します。しかし、ダウンロード・ストリームを返信する代わりに、Mobile Link はアップロードのコミットが成功したかどうかを示す確認だけを送信します。

『Mobile Link 管理ガイド』> 「アップロード専用の同期とダウンロード専用の同期」

「DownloadOnly (ds) 拡張オプション」148 ページ

「UploadOnly (uo) 拡張オプション」172 ページ

-urc オプション

機能

同期でアップロードされるロー数の推定値を指定します。

構文

dbmlsync -urc *row-estimate* …

説明

パフォーマンスを改善するために、同期でアップロードされるロー数の推定値を指定できます。通常、推定値が大きいほど、アップロードが高速になりますが、多くのメモリを消費します。

指定する推定値に関係なく、同期は正しく実行されます。

参照

「Memory (mem) 拡張オプション」159 ページ

『Mobile Link 管理ガイド』> 「パフォーマンスに関するヒント」

-v オプション

機能

メッセージ・ログ・ファイルにログを取り、同期ウィンドウに表示する情報を指定できます。高度な冗長性を指定するとパフォーマンスに影響するので、通常は高度な冗長性を使用するのは開発段階だけにしてください。

構文

dbmlsync -v [*levels*] …

説明

-v オプションはメッセージ・ログ・ファイルと同期ウィンドウに影響します。dbmlsync コマンド・ラインで -o または -ot を指定すると、メッセージ・ログが記録されるだけです。

−v を単独で指定すると、少量の情報のログが取られます。

levels の値は次のとおりです。たとえば -vnrsu や -v+cp などのように、次のオプションを一度に 1 つまたは複数指定できます。

- **+** c と p 以外のすべてのログ・オプションをオンにする
- **c** ログ内の接続文字列を公開する
- **p** ログ内のパスワードを公開する
- **n** アップロードとダウンロードされたロー数のログを取る
- **o** 指定したコマンド・ライン・オプションと拡張オプションに関する情報のログを取る
- **r** アップロードとダウンロードされたローの値のログを取る
- **s** フック・スクリプトに関連するメッセージのログを取る
- **u** アップロード・ストリームに関する情報のログを取る

-v オプションと同様の機能を持つ拡張オプションがあります。-v オプションと拡張オプションの両方を指定して、競合が発生した場合は、-v オプションが拡張オプションよりも優先されます。競合が発生しない場合は、冗長ログ・オプションが追加され、指定したすべてのオプションが使用されます。拡張オプションでロギングの冗長性を設定しても、ロギングは直ちに有効にならないため、起動情報のログは取られません。最初の同期が行われる時点では、-v オプションを指定した場合と拡張オプションを指定した場合のロギングの動作は同じです。

参照

「Verbose (v) 拡張オプション」172 ページ

「VerboseHooks (vs) 拡張オプション」173 ページ

「VerboseMin (vm) 拡張オプション」174 ページ

「VerboseOptions (vo) 拡張オプション」175 ページ

「VerboseRowCounts (vn) 拡張オプション」176 ページ

「VerboseRowValues (vr) 拡張オプション」177 ページ

-wc オプション

機能

ウィンドウ・クラス名を指定します。

構文

dbmlsync -wc *class-name* …

説明

このオプションは、スケジュールが有効になっているときや、サーバによって開始される同期を使用しているときなどに、停止モードの dbmlsync をウェイクアップするのに使用可能なクラス名を指定します。

また、このウィンドウ・クラス名によって、ActiveSync 同期用のアプリケーションが識別されます。ActiveSync 同期に使用するアプリケーションの登録時に、クラス名を指定してください。

参照

- 『Mobile Link クライアント』> 「ActiveSync 用 Adaptive Server Anywhere クライアントの登録」
- 『Mobile Link クライアント』> 「ActiveSync 同期の使用」
- 「Schedule (sch) 拡張オプション」164 ページに示されている INFINITE キーワード
- 「同期のスケジュール」115 ページ

例

```
dbmlsync -wc dbmlsync_$message_end...
```

-x オプション

機能

出力メッセージがスキャンされた後、トランザクション・ログの名前を変更し、再起動します。

構文

dbmlsync -x [*size* [**K** | **M** | **G**]…

説明

オプションの *size* は、トランザクション・ログが指定されたサイズより大きい場合にのみ、名前が変更されることを意味します。単位をキロバイト、メガバイト、またはギガバイトで指定するには、それぞれサフィックス k、m、または g を使用してください。デフォルトのサイズは 0 です。

場合によっては、リモート・データベースのバックアップが実行されたり、データベース・サーバを停止するときにトランザクション・ログの名前を変更する代わりに、統合データベースにデータが同期されます。

リモート・データベース側でバックアップを定期的に実行しないと、トランザクション・ログが大きくなっていきます。トランザクション・ログのサイズを制御するには、-x オプションを使用する代わりに、Adaptive Server Anywhere のイベント・ハンドラを使用する方法があります。たとえば、次のイベント・ハンドラは、リモート・データベース側でトランザクション・ログのサイズが 5 MB を超えると、そのログの名前を変更します。このようなイベント・ハンドラを DELETE_OLD_LOGS データベース・オプションと併用すると、トランザクション・ログが占める領域を制御できます。

```
CREATE EVENT RenameLogLimit
 TYPE GrowLog
 WHERE event_condition( 'LogSize' ) > 5
 AT REMOTE
 HANDLER
 BEGIN
  If event_parameter( 'NumActive' ) <= 1 then
   BACKUP DATABASE DIRECTORY backupdir
   TRANSACTION LOG ONLY
   TRANSACTION LOG RENAME
  end if;
 END
```

この例では、NumActive によって、スタックされたハンドラの 1 つだけが実行されます。

参照

- 『ASA データベース管理ガイド』> 「スケジュールとイベントの使用によるタスクの自動化」
- 『ASA データベース管理ガイド』> 「DELETE_OLD_LOGS オプション [レプリケーション]」
- 『ASA SQL リファレンス・マニュアル』> 「CREATE EVENT 文」

第 7 章

Mobile Link SQL 文

この章の内容

この章では、SQL 文の詳細についてアルファベット順に説明します。

ALTER PUBLICATION 文

説明

この文を使用して、パブリケーションを変更します。Mobile Link では、パブリケーションが Adaptive Server Anywhere リモート・データベース内の同期データを識別します。SQL Remote では、統合データベース内とリモート・データベース内の両方のレプリケートされたデータがパブリケーションによって識別されます。

構文

ALTER PUBLICATION [*owner*.]*publication-name alterpub-clause*, …

alterpub-clause:
ADD TABLE *article-description*
| **MODIFY TABLE** *article-description*
| { **DELETE** | **DROP** } **TABLE** [*owner*.]*table-name*
| **RENAME** *publication-name*

owner, *publication-name*, *table-name* : *identifier*

article-description :
table-name [**(** *column-name*, … **)**]
[**WHERE** *search-condition*]
[**SUBSCRIBE BY** *expression*]

使用法

この文は、Mobile Link と SQL Remote にのみ適用されます。

ALTER PUBLICATION 文は、データベース内でパブリケーションを変更します。パブリケーションでは、「アーティクル」が 1 つのテーブルを表します。パブリケーションの変更とは、アーティクルの追加、修正、削除、またはパブリケーションの名前の変更を意味します。アーティクルを修正する場合は、そのアーティクル全体の仕様を入力してください。

Mobile Link パブリケーションのオプションは、ALTER SYNCHRONIZATION SUBSCRIPTION 文または CREATE SYNCHRONIZATION SUBSCRIPTION 文の ADD OPTION 句で設定します。

パーミッション

DBA 権限を持っているか、パブリケーションの所有者であることが必要です。文中で参照されるすべてのテーブルに対する排他アクセスが必要です。

関連する動作

オートコミット

参照

- 「CREATE PUBLICATION 文」207 ページ
- 「DROP PUBLICATION 文」216 ページ
- 「ALTER SYNCHRONIZATION SUBSCRIPTION 文 [Mobile Link]」202 ページ
- 「CREATE SYNCHRONIZATION SUBSCRIPTION 文 [Mobile Link]」210 ページ
- 『SQL Remote ユーザーズ・ガイド』> 「sp_add_article プロシージャ」
- 『SQL Remote ユーザーズ・ガイド』> 「sp_add_article_col プロシージャ」

標準と互換性

- **SQL/92**　ベンダ拡張。
- **SQL/99**　ベンダ拡張。

例

次の文は、customer テーブルを pub_contact パブリケーションに追加します。

```
ALTER PUBLICATION pub_contact
    ADD TABLE customer
```

ALTER SYNCHRONIZATION SUBSCRIPTION 文 [Mobile Link]

説明

この文を使用して、Adaptive Server Anywhere リモート・データベースで、Mobile Link ユーザによるパブリケーションへのサブスクリプションのプロパティを変更します。

構文

ALTER SYNCHRONIZATION SUBSCRIPTION
  TO *publication-name*
  [**FOR** *ml_username*, …]
  [**TYPE** *protocol*]
  [**ADDRESS** *protocol-options*]
  [**ADD OPTION** *option*=*value*, …]
  [**MODIFY OPTION** *option*=*value*, …]
  [**DELETE** { **ALL OPTION** | **OPTION** *option*, … }]

ml_username: *identifier*

protocol-type: **http** | **https** | **https_fips** |**tcpip** | **ActiveSync**

protocol-options: *string*

value: *string* | *integer*

パラメータ

TO 句　パブリケーション名を指定します。

FOR 句　1 つ以上の Mobile Link ユーザ名を指定します。

FOR 句を省略すると、パブリケーションに対するプロトコル・タイプ、プロトコル・オプション、拡張オプションが設定されます。

異なるロケーションで指定されるオプションを dbmlsync が処理する方法については、『Mobile Link クライアント』> 「拡張オプションと接続パラメータの優先順位」を参照してください。

TYPE 句　同期に使用するネットワーク・プロトコルを指定します。デフォルトのプロトコルは **tcpip** です。

通信プロトコルの詳細については、『Mobile Link クライアント』> 「CommunicationType (ctp) 拡張オプション」を参照してください。

ADDRESS 句　Mobile Link 同期サーバのロケーションを含むネットワーク・プロトコル・オプションを指定します。

プロトコル・オプションの完全なリストについては、『Mobile Link クライアント』> 「CommunicationAddress (adr) 拡張オプション」を参照してください。

ADD OPTION、MODIFY OPTION、DELETE OPTION、DELETE ALL OPTION 句　拡張オプションの追加、修正、削除、すべての削除ができます。それぞれの句に、オプションは 1 つしか指定できません。

各オプションの値に、"**=**"、"**,**"、"**;**" の記号は使用できません。

オプションの完全なリストについては、『Mobile Link クライアント』> 「dbmlsync 拡張オプション」を参照してください。

使用法

この文を使用して、Mobile Link リモート・データベースまたはリファレンス・データベース内で、同期サブスクリプションを変更します。

パーミッション

DBA 権限が必要です。パブリケーションで参照されるすべてのテーブルへの排他アクセスが必要です。

関連する動作

オートコミット

参照

- 「CREATE PUBLICATION 文」207 ページ
- 「CREATE SYNCHRONIZATION USER 文 [Mobile Link]」213 ページ

標準と互換性

- **SQL/92**　ベンダ拡張。
- **SQL/99**　ベンダ拡張。

例

sales パブリケーションについて、デフォルトのサブスクリプション値を含むデフォルトのサブスクリプションを作成します (FOR 句は省略します)。Mobile Link 同期サーバのアドレスを指定し、Certicom ルート証明書だけが信頼されるように指定します。

```
CREATE SYNCHRONIZATION SUBSCRIPTION
    TO sales_publication
    ADDRESS 'host=test.internal;port=2439;
       security=ecc_tls'
    OPTION memory='2m';
```

sales パブリケーションに対して、Mobile Link ユーザ ml_user1 のサブスクリプションを作成します。メモリ・オプションを、デフォルトのパブリケーションで指定される値ではなく 3 MB に設定します。

```
CREATE SYNCHRONIZATION SUBSCRIPTION
    TO sales_publication
    FOR 'ml_user1'
    OPTION memory='3m';
```

ALTER SYNCHRONIZATION USER 文 [Mobile Link]

説明

この文を使用して、Adaptive Server Anywhere リモート・データベースで Mobile Link ユーザのプロパティを変更します。

構文

ALTER SYNCHRONIZATION USER *ml_username*
[**TYPE** *protocol-type*]
[**ADDRESS** *protocol-options*]
[**ADD OPTION** *option*=*value*, …]
[**MODIFY OPTION** *option*=*value*, …]
[**DELETE** { **ALL OPTION** | **OPTION** *option* }]

ml_username: *identifier*

protocol-type: **http** | **https** | **https_fips** | **tcpip** | **ActiveSync**

protocol-options: *string*

value: *string* | *integer*

パラメータ

TYPE 句　同期に使用するネットワーク・プロトコルを指定します。

通信プロトコルの詳細については、『Mobile Link クライアント』> 「CommunicationType (ctp) 拡張オプション」を参照してください。

ADDRESS 句　Mobile Link 同期サーバのロケーションを含むネットワーク・プロトコル・オプションを指定します。

プロトコル・オプションの完全なリストについては、『Mobile Link クライアント』> 「CommunicationAddress (adr) 拡張オプション」を参照してください。

ADD OPTION、MODIFY OPTION、DELETE OPTION、DELETE ALL OPTION 句　拡張オプションの追加、修正、削除、すべての削除ができます。それぞれの句に、オプションは 1 つしか指定できません。

オプションの完全なリストについては、『Mobile Link クライアント』> 「dbmlsync 拡張オプション」を参照してください。

使用法

この文を使用して、Mobile Link リモート・データベース内で同期ユーザのプロパティを変更します。

パーミッション

DBA 権限が必要です。パブリケーションで参照されるすべてのテーブルへの排他アクセスが必要です。

関連する動作

オートコミット

参照

- 「ALTER SYNCHRONIZATION SUBSCRIPTION 文 [Mobile Link]」202 ページ
- 「CREATE SYNCHRONIZATION USER 文 [Mobile Link]」213 ページ
- 「CREATE SYNCHRONIZATION SUBSCRIPTION 文 [Mobile Link]」210 ページ

標準と互換性

- **SQL/92** ベンダ拡張。
- **SQL/99** ベンダ拡張。

CREATE PUBLICATION 文

説明

この文を使用して、パブリケーションを作成します。Mobile Link では、パブリケーションが Ultra Light または Adaptive Server Anywhere のリモート・データベース内の同期データを識別します。

構文

CREATE PUBLICATION [*owner*.]*publication-name*
　(TABLE *article-description*, … **)**

owner, *publication-name* : *identifier*

article-description :
　table-name [**(** *column-name*, … **)**]
　[**WHERE** *search-condition*]
　[**SUBSCRIBE BY** *expression*]

パラメータ

article-description　パブリケーションは複数のアーティクルで構成されています。各アーティクルは、テーブルまたはテーブルの一部です。アーティクルは、テーブルの縦の分割 (テーブル・カラムのサブセット)、横の分割 (テーブル・ローのサブセット)、または縦横の分割の場合があります。

WHERE 句　テーブル・ローのサブセットをアーティクルに含めるように定義します。パブリケーションのすべてのサブスクライバが同じサブセットを受信する場合に使用すると便利です。

SUBSCRIBE BY 句　アーティクルの定義には WHERE 句と SUBSCRIBE BY 句を併用できますが、SUBSCRIBE BY 句は SQL Remote でのみ使用されます。

使用法

この文は、Mobile Link と SQL Remote にのみ適用されます。

CREATE PUBLICATION 文はデータベースにパブリケーションを作成します。所有者名を指定すれば、別のユーザのパブリケーションも作成できます。

Mobile Link では、パブリケーションは Adaptive Server Anywhere リモート・データベースでは必須で、Ultra Light データベースではオプションです。このようなパブリケーションとそれに対するサブスクリプションによって、Mobile Link 同期サーバにアップロードされるデータが決定されます。パブリケーションとサブスクリプションを直

接作成して、リモート・データベースを構築できます。また、リモート・データベースのテンプレートとして機能する Adaptive Server Anywhere リファレンス・データベース内でパブリケーションとサブスクリプションを作成してから、Mobile Link 抽出ユーティリティを使用してリモート・データベースを構成する方法もあります。

Mobile Link パブリケーションのオプションは、ALTER SYNCHRONIZATION SUBSCRIPTION 文または CREATE SYNCHRONIZATION SUBSCRIPTION 文の ADD OPTION 句で設定します。

パーミッション

DBA 権限が必要です。文中で参照されるすべてのテーブルに対する排他アクセスが必要です。

関連する動作

オートコミット

参照

- 「ALTER PUBLICATION 文」200 ページ
- 「DROP PUBLICATION 文」216 ページ
- 「ALTER SYNCHRONIZATION SUBSCRIPTION 文 [Mobile Link]」202 ページ
- 「CREATE SYNCHRONIZATION SUBSCRIPTION 文 [Mobile Link]」210 ページ
- 『SQL Remote ユーザーズ・ガイド』> 「sp_create_publication プロシージャ」
- Adaptive Server Anywhere Mobile Link クライアント :『Mobile Link クライアント』> 「データのパブリッシュ」
- Ultra Light Mobile Link クライアント :『Mobile Link クライアント』> 「 同期させるデータの選択」

標準と互換性

- **SQL/92** ベンダ拡張。
- **SQL/99** ベンダ拡張。

例

次の文は、2 つのテーブルのすべてのカラムとローをパブリッシュします。

```
CREATE PUBLICATION pub_contact (
    TABLE contact,
    TABLE company
 )
```

次の文は、1 つのテーブルの一部のカラムのみをパブリッシュします。

```
CREATE PUBLICATION pub_customer (
    TABLE customer ( id, company_name, city )
 )
```

次の文は、customer テーブルの status カラムをテストする WHERE 句を組み込んで、アクティブな customer ローのみをパブリッシュします。

```
CREATE PUBLICATION pub_customer (
    TABLE customer ( id, company_name, city, state )
    WHERE status = 'active'
 )
```

CREATE SYNCHRONIZATION SUBSCRIPTION 文 [Mobile Link]

説明

この文を使用して、Adaptive Server Anywhere リモート・データベース内でパブリケーションに対する Mobile Link ユーザのサブスクリプションを作成します。

構文

CREATE SYNCHRONIZATION SUBSCRIPTION
  TO *publication-name*
  [**FOR** *ml_username*, …]
  [**TYPE** *protocol-type*]
  [**ADDRESS** *protocol-options*]
  [**OPTION** *option*=*value*, …]

ml_username: *identifier*

protocol-type: **http** | **https** | **https_fips** | **tcpip** | **ActiveSync**

protocol-options: *string*

value: *string* | *integer*

パラメータ

TO 句　パブリケーション名を指定します。

FOR 句　1 つ以上の Mobile Link ユーザ名を指定します。

ml_username は、リモート・データベースを識別する名前です。この名前は、ユニークである必要があります。

同期ユーザ名の詳細については、『Mobile Link クライアント』> 「Mobile Link ユーザの概要」を参照してください。

FOR 句を省略すると、パブリケーションに対するプロトコル・タイプ、プロトコル・オプション、拡張オプションが設定されます。

異なるロケーションで指定されるオプションを dbmlsync が処理する方法については、『Mobile Link クライアント』> 「拡張オプションと接続パラメータの優先順位」を参照してください。

TYPE 句　同期に使用するネットワーク・プロトコルを指定します。デフォルトのプロトコルは **tcpip** です。

ネットワーク・プロトコルの詳細については、『Mobile Link クライアント』> 「CommunicationType (ctp) 拡張オプション」を参照してください。

ADDRESS 句　Mobile Link 同期サーバのロケーションなどのネットワーク・プロトコル・オプションを指定します。複数のオプションは、セミコロンで区切ります。

プロトコル・オプションの完全なリストについては、『Mobile Link クライアント』> 「CommunicationAddress (adr) 拡張オプション」を参照してください。

OPTION 句　サブスクリプションについて拡張オプションを設定できます。FOR 句を指定しない場合、拡張オプションはパブリケーションのデフォルト設定として機能し、同期ユーザ用に設定された拡張オプションによって上書きされます。

異なるロケーションで指定されるオプションを dbmlsync が処理する方法については、『Mobile Link クライアント』> 「拡張オプションと接続パラメータの優先順位」を参照してください。

オプションの完全なリストについては、『Mobile Link クライアント』> 「dbmlsync 拡張オプション」を参照してください。

使用法

この文は、Mobile Link リモート・データベースまたはリファレンス・データベース内で、同期サブスクリプションを作成するために使用します。

パーミッション

DBA 権限が必要です。パブリケーションで参照されるすべてのテーブルへの排他アクセスが必要です。

関連する動作

オートコミット

参照

- 「CREATE PUBLICATION 文」207 ページ
- 「CREATE SYNCHRONIZATION USER 文 [Mobile Link]」213 ページ

標準と互換性

- **SQL/92**　ベンダ拡張。
- **SQL/99**　ベンダ拡張。

例

sales パブリケーションについて、デフォルトのサブスクリプション値を含むデフォルトのサブスクリプションを作成します (FOR 句は省略します)。Mobile Link 同期サーバのアドレスを指定し、Certicom ルート証明書だけが信頼されるように指定します。

```
CREATE SYNCHRONIZATION SUBSCRIPTION
    TO sales_publication
    ADDRESS 'host=test.internal;port=2439;
       security=ecc_tls'
    OPTION memory='2m';
```

sales パブリケーションに対して、Mobile Link ユーザ ml_user1 のサブスクリプションを作成します。メモリ・オプションを、デフォルトのパブリケーションで指定される値ではなく 3 MB に設定します。

```
CREATE SYNCHRONIZATION SUBSCRIPTION
    TO sales_publication
    FOR ml_user1
    OPTION memory='3m';
```

CREATE SYNCHRONIZATION USER 文 [Mobile Link]

説明

この文を使用して、Adaptive Server Anywhere リモート・データベース内で同期ユーザを作成します。

構文

CREATE SYNCHRONIZATION USER *ml_username*
[**TYPE** *protocol-type*]
[**ADDRESS** *protocol-options*]
[**OPTION** *option*=*value*, …]

ml_username: *identifier*

protocol-type: **tcpip** | **http** | **https** | **https_fips** | **ActiveSync**

protocol-options: *string*

value: *string* | *integer*

パラメータ

ml_username　リモート・データベースを識別する名前。この名前は、ユニークにする必要があります。

同期ユーザ名の詳細については、『Mobile Link クライアント』> 「Mobile Link ユーザの概要」を参照してください。

TYPE 句　同期に使用する通信プロトコルを指定します。オプションは、**tcpip**、**http**、**https**、**https_fips**、**ActiveSync** です。デフォルトのプロトコルは **tcpip** です。

通信プロトコルの詳細については、『Mobile Link クライアント』> 「CommunicationType (ctp) 拡張オプション」を参照してください。

ADDRESS 句　*protocol-options* を *keyword*=*value* の形式でセミコロンで区切って指定します。どのような設定を指定するかは、使用する通信プロトコル (TCPIP、HTTP、HTTPS、ActiveSync) に応じて異なります。

プロトコル・オプションの完全なリストについては、『Mobile Link クライアント』> 「CommunicationAddress (adr) 拡張オプション」を参照してください。

OPTION 句 OPTION 句では、*option=value* をカンマで区切ったリストで拡張オプションを設定できます。

各オプションの値に、等号とセミコロンは使用できません。データベース・サーバは、入力されたオプションを、妥当性を検査しないですべて受け入れます。そのため、オプションのスペルを間違えたり、無効な値を入力したりしても、dbmlsync コマンドを実行して同期を行うまでエラー・メッセージが表示されません。

同期ユーザ用に設定したオプションは、個々のサブスクリプション内で、または dbmlsync コマンド・ラインを使用して上書きできます。

拡張オプションの詳細については、『Mobile Link クライアント』> 「dbmlsync 拡張オプション」を参照してください。

説明

protocol-type、*protocol-options*、*options* は、複数の個所で設定できます。

異なるロケーションで指定されるオプションを dbmlsync が処理する方法については、『Mobile Link クライアント』> 「拡張オプションと接続パラメータの優先順位」を参照してください。

パーミッション

DBA 権限が必要です。

関連する動作

オートコミット

参照

- 「ALTER SYNCHRONIZATION USER 文 [Mobile Link]」205 ページ
- 「CREATE SYNCHRONIZATION SUBSCRIPTION 文 [Mobile Link]」210 ページ
- 「CREATE PUBLICATION 文」207 ページ
- 『Mobile Link 管理ガイド』> 「Mobile Link トランスポート・レイヤ・セキュリティ」

標準と互換性

- **SQL/92** ベンダ拡張。
- **SQL/99** ベンダ拡張。
- **Sybase** Open Client/Open Server でサポートされています。

例

次の例は、TCP/IP 経由でサーバ・マシン mlserver.mycompany.com と同期を行う、パスワード Sam を持つユーザ SSinger を作成します。ユーザ定義でのパスワードの使用には、セキュリティは適用*されません*。

```
CREATE SYNCHRONIZATION USER SSinger
 TYPE http
 ADDRESS 'host=mlserver.mycompany.com'
 OPTION MobiLinkPwd='Sam'
```

次の例は、factory014 という同期ユーザを作成し、dbmlsync がホバリングして 8 時間ごとの任意の時間に Certicom で暗号化された TCP/IP を通じて同期を行うようにします。任意の時間に行うことによって、複数の同時同期による Mobile Link サーバのパフォーマンスの低下が回避できます。

```
CREATE SYNCHRONIZATION USER factory014
 TYPE tcpip
 ADDRESS
'host=mycompany.manufacturing.mobilink1;security=certic
om_tls(certificate=mycompany_mobilink.crt;certificate_p
assword=thepassword)'
 OPTION Schedule='EVERY:08:00'
```

次の例は、HTTP との同期に使用される、sales5322 という同期ユーザを作成します。この例では、企業のファイアウォールの外側で Mobile Link 同期サーバを実行し、リダイレクタ (NSAPI Web サーバのリバース・プロキシ) を使用して同期要求をリダイレクトします。

```
CREATE SYNCHRONIZATION USER sales5322
 TYPE https
 ADDRESS
'host=www.mycompany.com;port=80;url_suffix=mlredirect/
ml/'
```

DROP PUBLICATION 文

説明

この文を使用して、パブリケーションを削除します。Mobile Link では、パブリケーションが Adaptive Server Anywhere リモート・データベース内の同期データを識別します。SQL Remote では、統合データベース内とリモート・データベース内の両方のレプリケートされたデータがパブリケーションによって識別されます。

構文

DROP PUBLICATION [*owner*.]*publication-name*

owner, *publication-name* : *identifier*

使用法

この文は、Mobile Link と SQL Remote にのみ適用されます。

パーミッション

DBA 権限が必要です。

関連する動作

オートコミット。パブリケーションに対するサブスクリプションがすべて削除されます。

参照

- 「ALTER PUBLICATION 文」200 ページ
- 「CREATE PUBLICATION 文」207 ページ
- 『SQL Remote ユーザーズ・ガイド』> 「sp_drop_publication プロシージャ」

標準と互換性

- **SQL/92** ベンダ拡張。
- **SQL/99** ベンダ拡張。

例

次の文は、pub_contact パブリケーションを削除します。

```
DROP PUBLICATION pub_contact
```

DROP SYNCHRONIZATION SUBSCRIPTION 文 [Mobile Link]

説明

この文を使用して、Mobile Link リモート・データベースまたは Mobile Link リファレンス・データベース内の同期サブスクリプションを削除します。また、指定されたパブリケーションの、デフォルト・サブスクリプション値が入っているデフォルト・サブスクリプションの削除にも使用できます。

構文

DROP SYNCHRONIZATION SUBSCRIPTION
　TO *publication-name*
　[**FOR** *ml_username*, …]

パラメータ

TO 句　パブリケーション名を指定します。

FOR 句　1 人以上の Mobile Link ユーザを指定します。

この句を省略すると、パブリケーションに対するデフォルトのサブスクリプションが削除されます。パブリケーションに対するサブスクリプションを作成していた Mobile Link ユーザは、デフォルト・パブリケーションの値をデフォルトとして継承します。

使用法

Mobile Link のリモートまたはリファレンス・データベース内の同期サブスクリプションを削除します。

パーミッション

DBA 権限が必要です。パブリケーションで参照されるすべてのテーブルへの排他アクセスが必要です。

関連する動作

オートコミット

標準と互換性

- **SQL/92**　ベンダ拡張。
- **SQL/99**　ベンダ拡張。

例

sales パブリケーションに対する、Mobile Link ユーザ ml_user1 のサブスクリプションを削除します。

```
DROP SYNCHRONIZATION SUBSCRIPTION
    TO sales_publication
    FOR "ml_user1"
```

(FOR 句を省略して) sales パブリケーションについて、デフォルトのサブスクリプション値を含むデフォルトのサブスクリプションを削除します。

```
DROP SYNCHRONIZATION SUBSCRIPTION
    TO sales_publication
```

DROP SYNCHRONIZATION USER 文 [Mobile Link]

説明

この文を使用して、Mobile Link リモート・データベースから同期ユーザを削除します。

構文

DROP SYNCHRONIZATION USER *ml_username, …*

ml_username: *identifier*

使用法

Mobile Link リモート・データベースから、1 人以上の同期ユーザを削除します。

パーミッション

DBA 権限が必要です。パブリケーションで参照されるすべてのテーブルへの排他アクセスが必要です。

関連する動作

ユーザに関連するすべてのサブスクリプションも削除されます。

標準と互換性

- **SQL/92**　ベンダ拡張。
- **SQL/99**　ベンダ拡張。

例

データベースから Mobile Link ユーザ ml_user1 を削除します。

```
DROP SYNCHRONIZATION USER ml_user1
```

START SYNCHRONIZATION DELETE 文 [Mobile Link]

説明

次の文を使用して Mobile Link 同期で行われた削除のロギングを再起動します。

構文

START SYNCHRONIZATION DELETE

使用法

通常、Adaptive Server Anywhere と Ultra Light は、同期の一部であるテーブルやカラム対して行われたすべての変更のログを自動的に取り、次の同期中にこれらの変更を統合データベースにアップロードします。STOP SYNCHRONIZATION DELETE 文を使用すると、削除処理の自動ロギングを一時的に中断できます。START SYNCHRONIZATION DELETE 文を使用すると、自動ロギングを再起動できます。

STOP SYNCHRONIZATION DELETE 文を実行すると、その接続で行われた削除処理は同期されません。この効果は、START SYNCHRONIZATION DELETE 文が実行されるまで継続します。STOP SYNCHRONIZATION DELETE を繰り返してもそれ以上効果はありません。

STOP SYNCHRONIZATION DELETE 文の実行回数に関わらず、START SYNCHRONIZATION DELETE 文を 1 回実行すれば、ロギングが再起動します。

アプリケーションでデータを同期しない場合は、START SYNCHRONIZATION DELETE を使用しないでください。

パーミッション

DBA 権限が必要です。

関連する動作

なし

参照

- 「STOP SYNCHRONIZATION DELETE 文 [Mobile Link]」222 ページ
- 『Ultra Light C/C++ ユーザーズ・ガイド』> 「StartSynchronizationDelete メソッド」

標準と互換性

- **SQL/92** ベンダ拡張。

- **SQL/99**　ベンダ拡張。
- **Sybase**　適用不可。

例

次の一連の SQL 文は、START SYNCHRONIZATION DELETE と STOP SYNCHRONIZATION DELETE の使用方法を示します。

```
-- Prevent deletes from being sent
 -- to the consolidated database
 STOP SYNCHRONIZATION DELETE;

 -- Remove all records older than 1 month
 -- from the remote database,
 -- NOT the consolidated database
 DELETE FROM PROPOSAL
 WHERE last_modified < months( CURRENT TIMESTAMP, -1 )

 -- Re-enable all deletes to be sent
 -- to the consolidated database
 -- DO NOT FORGET to start this
 START SYNCHRONIZATION DELETE;

 -- Commit the entire operation,
 -- otherwise rollback everything
 -- including the stopping of the deletes
 commit;
```

STOP SYNCHRONIZATION DELETE 文 [Mobile Link]

説明

次の文を使用して Mobile Link 同期で行われた削除のロギングを一時的に停止します。

構文

STOP SYNCHRONIZATION DELETE

使用法

通常、Adaptive Server Anywhere と Ultra Light リモート・データベースは、同期に組み込まれるテーブルやカラムに対して行われたすべての変更のログを自動的に取り、次の同期中にこれらの変更を統合データベースにアップロードします。この文を使用して、Adaptive Server Anywhere または Ultra Light リモート・データベースに対する変更のロギングを一時的に中断できます。

STOP SYNCHRONIZATION DELETE 文を実行すると、その接続でそれ以降行われた削除処理は同期されません。この効果は、START SYNCHRONIZATION DELETE 文が実行されるまで継続します。

STOP SYNCHRONIZATION DELETE を繰り返してもそれ以上効果はありません。STOP SYNCHRONIZATION DELETE 文の実行回数に関わらず、START SYNCHRONIZATION DELETE 文を 1 回実行すれば、ロギングが再起動します。

このコマンドは、リモート・データベースに対して訂正を行うには便利ですが、Mobile Link 同期を事実上無効にしてしまうので、使用の際には注意してください。

アプリケーションでデータを同期しない場合は、STOP SYNCHRONIZATION DELETE を使用しないでください。

パーミッション

DBA 権限が必要です。

関連する動作

なし

参照

- 『Ultra Light C/C++ ユーザーズ・ガイド』> 「StartSynchronizationDelete メソッド」
- 『Ultra Light C/C++ ユーザーズ・ガイド』> 「StopSynchronizationDelete メソッド」

◆ 「START SYNCHRONIZATION DELETE 文 [Mobile Link]」220 ページ

標準と互換性

- **SQL/92**　ベンダ拡張。
- **SQL/99**　ベンダ拡張。
- **Sybase**　適用不可。

例

例については、「START SYNCHRONIZATION DELETE 文 [Mobile Link]」220 ページ を参照してください。

第 8 章

dbmlsync クライアント・イベント・フック

この章の内容

クライアント同期処理のカスタマイズ

Adaptive Server Anywhere 同期クライアント dbmlsync には、一連のイベント・フックが用意されています。これを使用して、同期処理をカスタマイズできます。フックを実装した場合は、同期処理における特定の時点で呼び出されます。

イベント・フックを実装するには、特定の名前のストアド・プロシージャを作成します。ほとんどのイベント・フック・ストアド・プロシージャは、同期自体と同じ接続で実行されます。

イベント・フックを使用すると、同期イベントのログの記録、論理イベントまたは時刻に基づいた同期のスケジュール、同期動作のカスタマイズを行うことができます。

また、イベント・フックを使用して、パブリケーションで簡単に定義できないデータのサブセットを同期することもできます。たとえば、2 つのイベント・フック・プロシージャを記述して、テンポラリ・テーブル内のデータを同期することができます。この場合、一方のイベント・フック・プロシージャでは、同期の前にテンポラリ・テーブルから永久テーブルにデータをコピーし、他方では同期後にデータを逆にコピーします。

警告
同期処理の整合性は、組み込みトランザクションの順序に依存します。したがって、イベント・フック・プロシージャ内では、暗黙的または明示的なコミットもロールバックも実行しないでください。

同期イベント・フックの順序

次の疑似コードは、使用可能なイベントと、同期処理中に各イベントが呼び出されるポイントを示します。たとえば、sp_hook_dbmlsync_abort は最初に呼び出されるイベント・フックです。

各イベントによって特定のパラメータ値が利用可能になります。利用可能になったパラメータ値は、プロシージャを実装したときに使用できます。パラメータ値の中には、新しい値を返すように変更できるものがあります。それ以外のものは、読み込み専用です。これらのパラメータは、ストアド・プロシージャの引数ではありません。いずれの

イベント・フック・ストアド・プロシージャにも、引数は渡されません。代わりに、#hook_dict テーブル内のローを読み込んだり修正したりすることで、引数がやりとりされます。

たとえば、sp_hook_dbmlsync_begin にはパラメータが 1 つあります。このパラメータは、この同期呼び出しでアプリケーションが指定したユーザ名です。この値は、#hook_dict テーブルから取り出すことができます。

順序は Mobile Link 同期サーバでのイベントの順序と類似していますが、統合データベースとリモート・データベースに追加する論理の種類には、重複はほとんどありません。したがって、2 つのインタフェースは別のものとなります。

*_end フックが呼び出されるのは、対応する *_begin フックが呼び出されて正常に完了した場合です。*_begin フックが呼び出されたときに実装されていない場合は、正常に実行されたと見なされます。

アップロード・オプションの詳細については、「-tu オプション」190 ページと「Increment (inc) 拡張オプション」156 ページを参照してください。

```
sp_hook_dbmlsync_abort
 sp_hook_dbmlsync_set_extended_options
 loop until return codes direct otherwise (
     sp_hook_dbmlsync_abort
     sp_hook_dbmlsync_delay
 )
 sp_hook_dbmlsync_abort
 // start synchronization
 sp_hook_dbmlsync_begin
 // upload events
 for each upload segment
 // a normal synchronization has one upload segment
 // a transaction-level upload has one segment per
transaction
 // an incremental upload has one segment per upload
piece
  sp_hook_dbmlsync_logscan_begin
  sp_hook_dbmlsync_logscan_end
  sp_hook_dbmlsync_upload_begin
  sp_hook_dbmlsync_upload_end
 next upload event
```

```
// download events
 sp_hook_dbmlsync_validate_download_file (only called
    when -ba option is used)
 sp_hook_dbmlsync_download_begin
 for each table
     sp_hook_dbmlsync_download_table_begin
     sp_hook_dbmlsync_download_table_end
 next table
 sp_hook_dbmlsync_download_end
 sp_hook_dbmlsync_schema_upgrade
 // end synchronization
 sp_hook_dbmlsync_end
 sp_hook_dbmlsync_process_return_code
 sp_hook_dbmlsync_log_rescan
```

エラー処理

また、次のイベント・フック・プロシージャはエラー処理に使用できます。

```
sp_hook_dbmlsync_all_error
 sp_hook_dbmlsync_communication_error
 sp_hook_dbmlsync_connect_failed
 sp_hook_dbmlsync_download_com_error
 sp_hook_dbmlsync_download_SQL_error
 sp_hook_dbmlsync_download_fatal_SQL_error
 sp_hook_dbmlsync_download_ri_violation
 sp_hook_dbmlsync_download_log_ri_violation
 sp_hook_dbmlsync_misc_error
 sp_hook_dbmlsync_sql_error
```

実装された各プロシージャは、指定したタイプのエラーが発生するたびに自動的に実行されます。

イベント・フック・プロシージャの使用

この項では、イベント・フック・プロシージャの設計と使用におけるいくつかの注意事項について説明します。

注意

- イベント・フック・プロシージャでは、COMMIT 操作も ROLLBACK 操作も実行しないでください。プロシージャは同期と同じ接続で実行されるので、COMMIT または ROLLBACK を実行すると同期が妨害されます。

- 同じ名前で複数のフックを定義しないでください。(たとえば異なるユーザが) 同じ名前を持つ複数のフックを作成すると、どのフックが呼ばれるかは定義されません。
- フック・プロシージャは DBA 権限を持つユーザが作成してください。
- *_begin フックが正常に実行すると、その後にどのようなエラーが発生しても、対応する *_end フックが呼び出されます。*_end フックが定義されていても、*_begin フックが定義されていない場合は、通常 *_begin フックが呼び出される時点より前にエラーが発生しないかぎり、*_end フックが呼び出されます。

#hook_dict テーブル

フックが呼び出される直前に、dbmlsync は次の CREATE 文を使用してリモート・データベースに #hook_dict テーブルを作成します。テーブル名の前の # は、そのテーブルがテンポラリであることを意味します。

```
CREATE TABLE #hook_dict(
 name VARCHAR(128) NOT NULL UNIQUE,
 value VARCHAR(255) NOT NULL)
```

dbmlsync は #hook_dict テーブルを使用してフック関数に値を渡し、フック関数は #hook_dict テーブルを使用して dbmlsync に値を戻します。

たとえば、次の dbmlsync コマンド・ラインを実行します。

```
dbmlsync -c 'dsn=MyDsn' -n pub1,pub2 -u MyUser
```

sp_hook_dbmlsync_abort フックを呼び出す時点では、#hook_dict テーブルには次のようなローが含まれています。

名前	値
publication_0	pub1
publication_1	pub2
MobiLink user	MyUser

名前	値
Abort synchronization	false

アボート・フックを使用して、#hook_dict テーブルから値を取り出したり、その動作をカスタマイズしたりできます。たとえば、Mobile Link ユーザを取り出すには、次のように SELECT 文を使用します。

```
SELECT value
 FROM #hook_dict
 WHERE name = 'MobiLink user'
```

In/out パラメータは、dbmlsync の動作をフックで修正することによって更新できます。たとえば、次のような文を使用してテーブルの abort synchronization ローを更新することで、同期のアボートをフックから dbmlsync に命令することができます。

```
UPDATE #hook_dict
 SET value='true'
 WHERE name='abort synchronization'
```

各フックの記述には、#hook_dict テーブルのローがリストされます。

イベント・フック・プロシージャの所有者

イベント・フック接続は、ストアド・プロシージャを、所有者で識別せずに呼び出します。したがって、ストアド・プロシージャは次のいずれかによって所有されなければなりません。

- dbmlsync 接続で使用されるユーザ名 (通常は、REMOTE DBA 権限を持つユーザ)
- dbmlsync ユーザがメンバであるグループ ID

イベント・フック・プロシージャ用の接続

各イベント・フック・プロシージャは、同期自体と同じ接続で実行されます。ただし、次のプロシージャは例外です。

- sp_hook_dbmlsync_all_error

- sp_hook_dbmlsync_communication_error
- sp_hook_dbmlsync_download_com_error
- sp_hook_dbmlsync_download_fatal_sql_error
- sp_hook_dbmlsync_download_log_ri_violation
- sp_hook_dbmlsync_misc_error
- sp_hook_dbmlsync_sql_error

これらのプロシージャは、同期が失敗する前に呼び出されます。失敗すると、同期アクションがロールバックされます。別の接続で処理を行うと、これらのプロシージャを使用して失敗情報のログを取ることができ、このとき、ログ・アクションは同期アクションとともにはロールバックされません。

イベント引数

各フックは、パラメータ値を受け取ります。パラメータ値の中には、新しい値を返すように変更できるものがあります。それ以外のものは、読み込み専用です。

これらのパラメータは、#hook_dict テーブル内のローを読み込んだり修正したりすることでやりとりされます。定義は次のとおりです。

```
CREATE TABLE #hook_dict (
     name    VARCHAR( 128 ) NOT NULL UNIQUE,
     value    VARCHAR( 255 ) NOT NULL
  )
```

このテーブルの各ローには、1 つのパラメータの値があります。

dbmlsync は、#hook_dict テーブルを作成し、そのイベントのパラメータを追加してから、ストアド・プロシージャを呼び出します。プロシージャは、このテーブルから値を選択して読み込むことができます。

いくつかのパラメータは、フックから dbmlsync に値を渡すために使用できます。フックは、#hook_dict テーブルを更新して、値を渡します。

各イベントで指定されるパラメータ値のリストについては、「dbmlsync クライアント・イベント・フック」225 ページを参照してください。

例

次の例は、#hook_dict テーブル内の値を取り出し、設定する方法を示します。

次のサンプル sp_hook_dbmlsync_delay プロシージャは、#hook_dict テーブルを使用して引数を受け渡す様子を示します。このプロシージャでは、Mobile Link システムのスケジュールされたダウン時間 (18:00 ～ 19:00) 以外にのみ同期を行います。

```
CREATE PROCEDURE sp_hook_dbmlsync_delay()
 BEGIN
     DECLARE delay_val integer;
  SET delay_val=DATEDIFF(
    second, CURRENT TIME, '19:00');
  IF (delay_val>0 AND
      delay_val<3600)
  THEN
  UPDATE #hook_dict SET value=delay_val
    WHERE name='delay duration';
  END IF;
 END
```

次のプロシージャは、同期の開始時にリモート・データベース内で実行されます。現在の Mobile Link ユーザ名、つまり sp_hook_dbmlsync_begin イベントに使用可能なパラメータの 1 つを取り出して、コンソールに出力します。

```
CREATE PROCEDURE sp_hook_dbmlsync_begin()
 BEGIN
     DECLARE syncdef VARCHAR(150);
     SELECT '>>>syncdef = ' || value INTO syncdef
         FROM #hook_dict
         WHERE name ='MobiLink user name';
     MESSAGE syncdef TYPE INFO TO CONSOLE;
 END
```

イベント・フック・プロシージャ内でのエラーの無視

イベント・フック・プロシージャ内でエラーが発生した場合、デフォルトでは同期は停止します。dbmlsync ユーティリティで -eh オプションを指定すると、イベント・フック・プロシージャ内でエラーが発生しても無視するように指定できます。

詳細については、「IgnoreHookErrors (eh) 拡張オプション」154 ページを参照してください。

sp_hook_dbmlsync_abort

機能

このストアド・プロシージャは、同期処理をキャンセルする場合に使用します。

#hook_dict テーブルのロー

名前	値	説明
abort synchronization (in|out)	**True**|**False**	#hook_dict テーブルの abort synchronization ローを **True** に設定すると、dbmlsync はイベント終了後すぐに終了します。
publication_*n* (in)	パブリケーション名	パブリケーションが同期されています (*n* は整数)。アップロードされるパブリケーションごとに 1 つの publication_*n* エントリがあります。*n* の番号はゼロから始まります。
MobiLink user (in)	Mobile Link ユーザ名	同期対象となる Mobile Link ユーザ
exit code (in|out)	数値	abort synchronization を True に設定すると、この値を使用して、アボートされた同期のリターン・コードを設定できます。0 は同期が成功したことを示します。他の数は同期が失敗したことを示します。
script version (in|out)	スクリプト・バージョン名	同期に使用される Mobile Link スクリプト・バージョン

説明

この名前のプロシージャが存在する場合、そのプロシージャは dbmlsync の起動時に呼び出され、sp_hook_dbmlsync_delay フックにより各同期が遅延した後に再度呼び出されます。

abort synchronization の値を True に設定することによりフックがアボートを要求する場合、終了コードが sp_hook_dbmlsync_process_exit_code フックに渡されます。sp_hook_dbmlsync_process_exit_code フックが定義されていない場合、終了コードがプログラムの終了コードとして使用されます。

このプロシージャのアクションは、実行直後にコミットされます。

参照

- 「同期イベント・フックの順序」226 ページ
- 「sp_hook_dbmlsync_process_return_code」278 ページ

例

次のプロシージャは、毎日 19:00 ～ 20:00 にスケジュールされた保守作業時間中に、同期が行われないようにします。

```
create procedure sp_hook_dbmlsync_abort()
 begin
   declare down_time_start time;
   declare is_down_time varchar(128);
   set down_time_start='19:00';
   if abs( datediff( hour,down_time_start,now(*) ) ) < 1
   then


    set is_down_time='true';
   else
     set is_down_time='false';
   end if;
   UPDATE #hook_dict
   SET value = is_down_time
   WHERE name = 'abort synchronization'
 end
```

次の 2 つの理由のいずれかにより、同期をアボート可能なアボート・フックがあるとします。1 つ目の理由は、同期の正常終了を示すのに、dbmlsync に終了コード 0 を含ませるためです。また、2 つ目の理由は、エラー状態を示すのに、dbmlsync に 0 以外の終了コードを含ませるためです。sp_hook_dbmlsync_abort フックを次のように定義すれば、上記のことが可能です。

```
BEGIN
     IF [condition that defines the normal abort case]
THEN
         UPDATE #hook_dict SET value =  '0'
    WHERE name = 'exit code';
```

```
        UPDATE #hook_dict SET value =  'TRUE'
    WHERE name = 'abort synchronization';
     ELSEIF [condition that defines the error abort
case] THEN
        UPDATE #hook_dict SET value =  '1'
    WHERE name = 'exit code';
        UPDATE #hook_dict SET value =  'TRUE'
    WHERE name = 'abort synchronization';
     END IF;
 END;
```

sp_hook_dbmlsync_begin

機能

このストアド・プロシージャを使用して、同期処理の開始時にカスタム・アクションを追加します。

#hook_dict テーブルのロー

名前	値	説明
publication_*n* (in)	パブリケーション名	パブリケーションが同期されています (*n* は整数)。アップロードされるパブリケーションごとに 1 つの publication_*n* エントリがあります。*n* の番号はゼロから始まります。
MobiLink user (in)	Mobile Link ユーザ名	同期対象となる Mobile Link ユーザ
script version (in)	スクリプト・バージョン名	同期に使用される Mobile Link スクリプト・バージョン

説明

この名前のプロシージャが存在する場合、同期処理の開始時に呼び出されます。

このプロシージャのアクションは、実行直後にコミットされます。

参照

- 「同期イベント・フックの順序」226 ページ

例

次のテーブルを使用して、リモート・データベース上の同期イベントのログを取るとします。

```
CREATE TABLE SyncLog
 (
  "event_id"              integer NOT NULL DEFAULT
autoincrement ,
     "event_name"         varchar(128) NOT NULL ,
     "ml_user"               varchar(128) NULL ,
     "event_time"            timestamp NULL,
     "table_name"            varchar(128) NULL ,
     "upsert_count"          varchar(128) NULL ,
     "delete_count"          varchar(128) NULL ,
     "exit_code"             integer NULL ,
     "status_retval"         varchar(128) NULL ,
```

```
    "pubs"                      varchar(128) NULL ,
    "sync_descr "           varchar(128) NULL ,
     PRIMARY KEY ("event_id"),
)
```

次に、パブリケーションのリストをコンパイルする例を示します。同期処理の初めにパブリケーション・リストなどの同期情報のログを取ります。

```
CREATE PROCEDURE sp_hook_dbmlsync_begin ()
 BEGIN

     DECLARE pubs_list varchar(1024);
     DECLARE temp_str varchar(128);
     DECLARE qry varchar(128);



 -- insert publication list into pubs_list
     SELECT list(value) INTO pubs_list
      FROM #hook_dict
      WHERE name LIKE 'publication_%';



 -- log publication and synchronization information
     INSERT INTO
SyncLog(event_name,ml_user,pubs,event_time)
      SELECT
'dbmlsync_begin',#hook_dict.value,pubs_list,current
timestamp
      FROM #hook_dict
      WHERE name='MobiLink user';
 END
```

sp_hook_dbmlsync_connect_failed

機能

このストアド・プロシージャを使用して、Mobile Link 同期サーバに対する接続が失敗した場合に異なる通信タイプまたはアドレスを使用してリトライします。

#hook_dict テーブルのロー

名前	値	説明
publication_*n* (in)	パブリケーション名	パブリケーションが同期されています (*n* は整数)。アップロードされるパブリケーションごとに 1 つの publication_*n* エントリがあります。*n* の番号はゼロから始まります。
MobiLink user (in)	Mobile Link ユーザ名	同期対象となる Mobile Link ユーザ
script version (in)	スクリプト・バージョン名	同期に使用される Mobile Link スクリプト・バージョン
connection address (in|out)	接続アドレス	フックが呼び出されるとき、これは失敗した直前の通信の試行で使用されたアドレスです。この値を新しい接続アドレスに設定して通信を試行できます。retry を true に設定すると、次の通信の試行でこの値が使用されます。
connection type (in|out)	**tcpip** | **http** | **https** | **ActiveSync** | **ecc_tls** | **rsa_tls**	フックが呼び出されるとき、これは失敗した直前の通信の試行で使用された通信タイプです。この値は新しい接続タイプに設定して通信を試行できます。retry を true に設定すると、次の通信の試行でこの値が使用されます。
user data (in|out)	ユーザ定義のデータ	次の通信の試行が失敗した場合に使用されるステータス情報。たとえば、発生したリトライの回数を保存すると便利です。デフォルトは、空の文字列です。

名前	値	説明
allow remote ahead (in|out)	**true** | **false**	dbmlsync を -ra オプションで起動した場合だけ true です。このローは、現在の同期のために -ra オプションの読み込みまたは変更を行う場合だけ使用できます。詳細については、「-r オプション」188 ページを参照してください。
allow remote behind (in|out)	**true** | **false**	dbmlsync を -rb オプションを指定して起動した場合だけ true です。このローは、現在の同期のために -rb オプションの読み込みまたは変更を行う場合だけ使用できます。詳細については、「-r オプション」188 ページを参照してください。
retry (in|out)	**true** | **false**	失敗した接続の試行をリトライする場合にこの値を true に設定します。デフォルト値は FALSE です。

説明

この名前のプロシージャが存在する場合、Mobile Link 同期サーバへの接続の試行で dbmlsync が失敗したときにそのプロシージャが呼び出されます。

このフックが適用されるのは、データベースへの接続の試行ではなく、Mobile Link 同期サーバへの接続の試行に対してだけです。

進行オフセット不一致が発生した場合、dbmlsync は Mobile Link 同期サーバから切断してから再接続します。この種の再接続では、このフックが呼び出されず、最接続が失敗すると同期も失敗します。

このプロシージャのアクションは、実行直後にコミットされます。

例

この例は、sp_hook_dbmlsync_connect_failed フックを使用して最大 5 回まで接続をリトライします。

```
CREATE PROCEDURE sp_hook_dbmlsync_ml_connect_failed ()
 BEGIN
      DECLARE idx integer;
      DECLARE buf varchar( 128 );

      SELECT value
        INTO buf
        FROM #hook_dict
        WHERE name = 'user data';

      IF buf = '' THEN
        SELECT 1 INTO idx;
      ELSE
        SELECT CONVERT( integer, value ) + 1
          INTO idx
          FROM #hook_dict
          WHERE name = 'user data';
      END IF;

   IF idx <= 5 THEN
           UPDATE #hook_dict
           SET value = CONVERT( varchar(128), idx )
           WHERE name = 'user data';

           UPDATE #hook_dict
           SET value = 'TRUE'
           WHERE name = 'retry';
   END IF
 END
```

次に、接続情報が含まれるテーブルを使用する例を示します。接続の試行が失敗すると、フックはリストの次のサーバを試行します。

```
CREATE TABLE conn_list (
      label    integer primary key,
      addr  varchar( 128 ),
      type  varchar( 64 )
 );
 INSERT INTO conn_list
  VALUES ( 1, 'host=server1;port=91', 'tcpip' );
 INSERT INTO conn_list
  VALUES ( 2, 'host=server2;port=92', 'http' );
 INSERT INTO conn_list
  VALUES ( 3, 'host=server3;port=93', 'tcpip' );
 COMMIT;
```

```
CREATE PROCEDURE sp_hook_dbmlsync_ml_connect_failed ()
BEGIN
 DECLARE idx integer;
 DECLARE cnt integer;
 DECLARE buf varchar( 128 );

 SELECT value
  INTO buf
  FROM #hook_dict
  WHERE name = 'user data';

 IF buf = '' THEN
   SELECT 1 INTO idx;
 ELSE
   SELECT CONVERT( integer, value ) + 1
   INTO idx
      FROM #hook_dict
   WHERE name = 'user data';
    END IF;

 SELECT count( label ) INTO cnt FROM conn_list;

 IF idx <= cnt THEN
   UPDATE #hook_dict
        SET value = ( SELECT addr FROM conn_list WHERE
label = idx )
          WHERE name = 'connection address';


  UPDATE #hook_dict
    SET value = (SELECT type FROM conn_list WHERE
label=idx)
    WHERE name = 'connection type';

   UPDATE #hook_dict
    SET value = CONVERT( varchar(128), idx )
          WHERE name = 'user data';

         UPDATE #hook_dict
          SET value = 'TRUE'
          WHERE name = 'retry';
 END IF
END
```

sp_hook_dbmlsync_delay

機能

このストアド・プロシージャを使用して、同期が行われるタイミングを制御します。

#hook_dict テーブルのロー

名前	値	説明
delay duration (in|out)	秒数	プロシージャが **delay duration** の値を 0 に設定すると、dbmlsync の同期が進行します。**delay duration** が 0 以外の値の場合は、遅延フックが再び呼び出されるまでの秒数を示します。
maximum accumulated delay (in|out)	秒数	最大累積遅延は、各同期前の最大秒数を指定します。dbmlsync は、最後に実行された同期以降の遅延フックへのすべての呼び出しによって生じた合計遅延時間を追跡します。dbmlsync が実行を開始してから同期が行われないと、dbmlsync の起動時間から合計遅延時間が計算されます。合計遅延時間が Maximum Accumulated delay 値を上回ると、遅延フックを呼び出さずに同期が開始されます。
publication_*n* (in)	パブリケーション名	パブリケーションが同期されています (*n* は整数)。アップロードされるパブリケーションごとに 1 つの publication_*n* エントリがあります。*n* の番号はゼロから始まります。
MobiLink user (in)	Mobile Link ユーザ名	同期対象となる Mobile Link ユーザ
script version (in)	スクリプト・バージョン名	同期に使用される Mobile Link スクリプト・バージョン

説明

この名前のプロシージャが存在する場合、そのプロシージャは同期処理の始めの **sp_hook_dbmlsync_begin** の前に呼び出されます。

このプロシージャのアクションは、実行直後にコミットされます。

参照

- 「同期イベント・フックの順序」226 ページ

例

次のテーブルを使用して、リモート・データベース上の注文のログを取るとします。

```
CREATE TABLE OrdersTable(
   "id" integer primary key default autoincrement,
   "priority" varchar(128)
 )
```

次に、最大累積遅延 (1 時間) の間、同期を遅らせる例を示します。10 秒ごとにフックが呼び出され、OrdersTable 内に優先度の高いローがないかをチェックします。優先度の高いローが存在する場合、遅延期間はゼロに設定して同期処理を開始します。

```
CREATE PROCEDURE sp_hook_dbmlsync_delay()
 BEGIN
        -- Set the maximum delay between
synchronizations
     -- or before the first synchronization starts to 1
hour
     UPDATE #hook_dict SET value = '3600'  // 3600
seconds
      WHERE name = 'maximum accumulated delay';

     -- check if a high priority order exists in
OrdersTable
     IF EXISTS (SELECT * FROM OrdersTable where
priority='high') THEN
      -- start the synchronization to process the high
priority row
      UPDATE #hook_dict
       SET value = '0'
       WHERE name='delay duration';
     ELSE
      -- set the delay duration to call this procedure
again
      -- following a 10 second delay
      UPDATE #hook_dict
```

```
        SET value = '10'
        WHERE name='delay duration';
      END IF;
  END
```

sp_hook_dbmlsync_end フックでは、優先度の高いローを処理済みとしてマークすることができます。

```
-- change status of high priority row
  UPDATE OrdersTable
   SET priority = 'high-processed'
   WHERE priority = 'high';
```

sp_hook_dbmlsync_end の詳細については、「sp_hook_dbmlsync_end」268 ページ を参照してください。

sp_hook_dbmlsync_download_begin

機能

このストアド・プロシージャを使用して、同期処理のダウンロード処理開始時にカスタム・アクションを追加します。

#hook_dict テーブルのロー

名前	値	説明
publication_*n* (in)	パブリケーション名	パブリケーションが同期されています (*n* は整数)。アップロードされるパブリケーションごとに 1 つの publication_*n* エントリがあります。*n* の番号はゼロから始まります。
MobiLink user (in)	Mobile Link ユーザ名	同期対象となる Mobile Link ユーザ
script version (in)	スクリプト・バージョン名	同期に使用される Mobile Link スクリプト・バージョン

説明

この名前のプロシージャが存在する場合、同期処理のダウンロード処理開始時に呼び出されます。

ダウンロード・ストリームがコミットまたはロールバックされると、このプロシージャのアクションがコミットまたはロールバックされます。

参照

- 「同期イベント・フックの順序」226 ページ

例

次のテーブルを使用して、リモート・データベース上の同期イベントのログを取るとします。

```
CREATE TABLE SyncLog
 (
  "event_id"              integer NOT NULL DEFAULT
autoincrement ,
     "event_name"         varchar(128) NOT NULL ,
     "ml_user"               varchar(128) NULL ,
     "event_time"            timestamp NULL,
     "table_name"            varchar(128) NULL ,
     "upsert_count"          varchar(128) NULL ,
     "delete_count"          varchar(128) NULL ,
     "exit_code"             integer NULL ,
```

```
        "status_retval"         varchar(128) NULL ,
        "pubs"                     varchar(128) NULL ,
        "sync_descr "           varchar(128) NULL ,
         PRIMARY KEY ("event_id"),
    )
```

次に、パブリケーションのリストをコンパイルする例を示します。同期のダウンロード処理の初めにパブリケーション・リストなどの同期情報のログを取ります。

```
CREATE PROCEDURE sp_hook_dbmlsync_begin ()
 BEGIN

     DECLARE pubs_list varchar(1024);
     DECLARE temp_str varchar(128);
     DECLARE qry varchar(128);



 -- insert publication list into pubs_list
     SELECT list(value) INTO pubs_list
      FROM #hook_dict
      WHERE name LIKE 'publication_%';



 -- log publication and synchronization information
     INSERT INTO
SyncLog(event_name,ml_user,pubs,event_time)
      SELECT
'dbmlsync_download_begin',#hook_dict.value,
       pubs_list,current timestamp
      FROM #hook_dict
      WHERE name='MobiLink user';
 END
```

sp_hook_dbmlsync_download_com_error

機能

Mobile Link 同期サーバによって送信されたダウンロード・ストリームの読み込み中に通信エラーが発生した場合は、このストアド・プロシージャを使用してカスタム・アクションを追加します。

#hook_dict テーブルのロー

名前	値	説明
table name (in)	テーブル名	エラーの発生時に操作中だったテーブル。dbmlsync がテーブルを識別できない場合、値は空の文字列になります。
publication_*n* (in)	パブリケーション名	パブリケーションが同期されています (*n* は整数)。アップロードされるパブリケーションごとに 1 つの publication_*n* エントリがあります。*n* の番号はゼロから始まります。
MobiLink user (in)	Mobile Link ユーザ名	同期対象となる Mobile Link ユーザ
script version (in)	スクリプト・バージョン名	同期に使用される Mobile Link スクリプト・バージョン

説明

この名前のプロシージャが存在する場合、同期のダウンロード・フェーズ中に通信エラーが検出されると呼び出されます。その場合、ダウンロードは終了します。

このプロシージャは別個の接続で実行されるため、障害のログを取ることができます。それ以外の場合、ログのアクションは同期アクションとともにロールバックされます。dbmlsync が別個の接続を確立できないと、プロシージャは呼び出されません。

Windows CE デバイスのデフォルトでは、同期テーブルは排他モードでロックされます。このため、同期テーブルへのアクセスが必要な場合、このフックは正常に実行されません。また、このフックは、同期テーブルへのアクセスが必要で、ユーザが dbmlsync 拡張オプション

LockTables を EXCLUSIVE に設定している場合にも実行できません。詳細については、「LockTables (lt) 拡張オプション」157 ページを参照してください。

このプロシージャのアクションは、実行直後にコミットされます。

参照

- 「同期イベント・フックの順序」226 ページ

例

次のテーブルを使用して、通信エラーのログを取るとします。

```
CREATE TABLE SyncLogComErrorTable
 (
     " user_name "     varchar(255) NOT NULL ,
     " event_time "   timestamp NOT NULL ,
 )
```

次に、Mobile Link 同期サーバによって送信されたダウンロード・ストリームを読み込み中に通信エラーが発生した場合に、Mobile Link ユーザと現在のタイム・スタンプのログを取る例を示します。この情報は、リモート・データベースの SyncLogComErrorTable テーブルに格納されます。

```
CREATE PROCEDURE sp_hook_dbmlsync_download_com_error ()
 BEGIN
  INSERT INTO SyncLogComErrorTable (user_name,
event_time)
   SELECT #hook_dict.value, current timestamp
   FROM #hook_dict
   WHERE name = 'MobiLink user';
 END;
```

sp_hook_dbmlsync_download_end

機能

このストアド・プロシージャを使用して、同期処理のダウンロード処理終了時にカスタム・アクションを追加します。

#hook_dict テーブルのロー

名前	値	説明
publication_*n* (in)	パブリケーション名	パブリケーションが同期されています (*n* は整数)。アップロードされるパブリケーションごとに 1 つの publication_*n* エントリがあります。*n* の番号はゼロから始まります。
MobiLink user (in)	Mobile Link ユーザ名	同期対象となる Mobile Link ユーザ
script version (in)	スクリプト・バージョン名	同期に使用される Mobile Link スクリプト・バージョン

説明

この名前のプロシージャが存在する場合、同期処理のダウンロード処理終了時に呼び出されます。

ダウンロード・ストリームがコミットまたはロールバックされると、このプロシージャのアクションがコミットまたはロールバックされます。

参照

- 「同期イベント・フックの順序」226 ページ

例

次のテーブルを使用して、リモート・データベース上の同期イベントのログを取るとします。

```
CREATE TABLE SyncLog
 (
  "event_id"              integer NOT NULL DEFAULT
autoincrement ,
     "event_name"         varchar(128) NOT NULL ,
     "ml_user"               varchar(128) NULL ,
     "event_time"            timestamp NULL,
     "table_name"            varchar(128) NULL ,
     "upsert_count"          varchar(128) NULL ,
     "delete_count"          varchar(128) NULL ,
```

```
    "exit_code"             integer NULL ,
    "status_retval"         varchar(128) NULL ,
    "pubs"                     varchar(128) NULL ,
    "sync_descr "           varchar(128) NULL ,
     PRIMARY KEY ("event_id"),
)
```

次に、パブリケーションのリストをコンパイルする例を示します。同期のダウンロード処理の終りにパブリケーション・リストなどの同期情報のログを取ります。

```
CREATE PROCEDURE sp_hook_dbmlsync_begin ()
 BEGIN

     DECLARE pubs_list varchar(1024);
     DECLARE temp_str varchar(128);
     DECLARE qry varchar(128);



 -- insert publication list into pubs_list
     SELECT list(value) INTO pubs_list
      FROM #hook_dict
      WHERE name LIKE 'publication_%';



 -- log publication and synchronization information
     INSERT INTO
SyncLog(event_name,ml_user,pubs,event_time)
      SELECT
'dbmlsync_download_begin',#hook_dict.value,
       pubs_list,current timestamp
      FROM #hook_dict
      WHERE name='MobiLink user';
 END
```

sp_hook_dbmlsync_download_fatal_sql_error

機能

データベース・エラーによって同期ダウンロードのロールバックが発生すると動作します。

#hook_dict テーブルのロー

名前	値	説明
table name (in)	テーブル名	エラーの発生時に操作中だったテーブル。dbmlsync がテーブルを識別できない場合、値は空の文字列になります。
SQL error code (in)	SQL エラー・コード	操作が失敗した時にデータベースから返される SQL エラー・コードを識別します。
publication_*n* (in)	パブリケーション名	パブリケーションが同期されています (*n* は整数)。アップロードされるパブリケーションごとに 1 つの publication_*n* エントリがあります。*n* の番号はゼロから始まります。
MobiLink user (in)	Mobile Link ユーザ名	同期対象となる Mobile Link ユーザ
script version (in)	スクリプト・バージョン名	同期に使用される Mobile Link スクリプト・バージョン

説明

この名前のプロシージャが存在する場合、同期のダウンロード・フェーズ中に通信エラーが検出されると呼び出されます。これが発生するのは、無視できない SQL エラーが発生した場合、または sp_hook_dbmlsync_download_SQL_error フックがすでに呼び出されていて、エラーを無視しないように選択されている場合です。

このプロシージャは別個の接続で実行されるため、障害のログを取ることができます。それ以外の場合、ログのアクションは同期アクションとともにロールバックされます。dbmlsync が別個の接続を確立できないと、プロシージャは呼び出されません。

Windows CE デバイスのデフォルトでは、同期テーブルは排他モードでロックされます。このため、同期テーブルへのアクセスが必要な場合、このフックは正常に実行されません。また、このフックは、同期テーブルへのアクセスが必要で、ユーザが dbmlsync 拡張オプション LockTables を EXCLUSIVE に設定している場合にも実行できません。詳細については、「LockTables (lt) 拡張オプション」157 ページを参照してください。

このプロシージャのアクションは、実行直後にコミットされます。

参照

- 「同期イベント・フックの順序」226 ページ
- 「sp_hook_dbmlsync_download_sql_error」260 ページ

例

次のテーブルを使用して、SQL エラーのログを取るとします。

```
CREATE TABLE "DBA"."SyncLogComErrorTable"
 (
     " error_code "          varchar(255) NOT NULL ,
     " event_time "         timestamp NOT NULL ,
 )
```

次に、ダウンロード・ストリームの読み込み中に SQL エラーが発生した場合に、SQL エラー・コードと現在のタイム・スタンプのログを取る例を示します。この情報は、リモート・データベースの SyncLogSQLErrorTable に格納されます。

```
CREATE PROCEDURE
sp_hook_dbmlsync_download_fatal_sql_error ()
 BEGIN
  INSERT INTO SyncLogSQLErrorTable (error_code,
event_time)
   SELECT #hook_dict.value, current timestamp
   FROM #hook_dict
   WHERE name = 'SQL error code';
 END;
```

sp_hook_dbmlsync_download_log_ri_violation

機能

ダウンロード・プロセスの参照整合性違反のログを取ります。

#hook_dict テーブルのロー

名前	値	説明
Publication_*n* (in)	パブリケーション名	パブリケーションが同期されています (*n* は整数)。アップロードされるパブリケーションごとに 1 つの publication_*n* エントリがあります。*n* の番号はゼロから始まります。
MobiLink user (in)	Mobile Link ユーザ名	同期対象となる Mobile Link ユーザ
Foreign key table (in)	テーブル名	フック呼び出し対象の外部キー・カラムを含むテーブル
Primary key table (in)	テーブル名	フック呼び出し対象の外部キーが参照するテーブル
Role name (in)	ロール名	フック呼び出し対象の外部キーのロール名
Script version (in)	スクリプト・バージョン名	同期に使用される Mobile Link スクリプト・バージョン

説明

ダウンロード・ストリーム内のローがリモート・データベース上の外部キー関係に違反すると、ダウンロード参照整合性違反が発生します。このフックを使用すると、発生した参照整合性違反のログを取り、後でその原因を調べることができます。

ダウンロードが完了すると、コミットされる前に、dbmlsync は参照整合性違反があるかどうかをチェックします。参照整合性違反が見つかると、参照整合性違反を含む外部キーを識別して、sp_hook_dbmlsync_download_log_ri_violation を呼び出します (実装されている場合)。次に、sp_hook_dbmlsync_download_ri_conflict を呼び出します (実装されている場合)。それでも矛盾がある場合、dbmlsync は外部キー制約に違反するローを削除します。参照整合性違反を含む残りの外部キーに対して、このプロセスが繰り返されます。

このフックは、現在同期中のテーブルに関する参照整合性違反がある場合にのみ呼び出されます。同期中ではないテーブルに関する参照整合性違反がある場合、このフックは呼び出されず、同期が失敗します。

このフックは、dbmlsync がダウンロードに使用する接続とは別の接続で呼び出されます。このフックが使用する接続の独立性レベルは 0 のため、ダウンロード・ストリームから適用されていてまだコミットされていないローを判別できます。フックのアクションは完了直後にコミットされるので、ダウンロード・ストリームがコミットされるかロールバックされるかに関係なく、このフックによる変更は保存されます。

Windows CE デバイスのデフォルトでは、同期テーブルは排他モードでロックされます。このため、同期テーブルへのアクセスが必要な場合、このフックは正常に実行されません。また、このフックは、同期テーブルへのアクセスが必要で、ユーザが dbmlsync 拡張オプション LockTables を EXCLUSIVE に設定している場合にも実行できません。詳細については、「LockTables (lt) 拡張オプション」157 ページを参照してください。

参照整合性違反の問題を解決するために、このフックを使用しないでください。このフックはロギングにのみ使用してください。参照整合性違反を解決するには、sp_hook_dbmlsync_download_ri_violation を使用します。

参照

- 「sp_hook_dbmlsync_download_ri_violation」257 ページ
- 「同期イベント・フックの順序」226 ページ

例

次のテーブルを使用して、参照整合性違反のログを取るとします。

```
CREATE TABLE "DBA"." LogRIErrorTable "
 (
  "id"      int PRIMARY KEY NOT NULL,
  "name"       varchar(255) NOT NULL,
  "value"  timestamp NOT NULL
 )
```

次に、リモート・データベース上で参照整合性違反が検出されたときに、外部キー・テーブル名、プライマリ・キー・テーブル名などの同期情報のログを取る例を示します。この情報は、リモート・データベースの LogRIErrorTable に格納されます。

```
CREATE PROCEDURE
sp_hook_dbmlsync_download_log_ri_violation()
 BEGIN
  --
  -- get the next id for the LogRIErrorTable
  --
     DECLARE nextid int;

  SELECT max(id)
   INTO nextid
   FROM LogRIErrorTable;

  IF nextid is NULL THEN
   SET nextid = 0
  ELSE
   SET nextid = nextid +1
  END IF;

  INSERT INTO LogRIErrorTable(id,name, value )
   SELECT nextid,#hook_dict.name, #hook_dict.value
   FROM #hook_dict;
 END;
```

sp_hook_dbmlsync_download_ri_violation

機能

ダウンロード・プロセスの参照整合性違反を解決できます。

#hook_dict テーブルのロー

名前	値	説明
Publication_*n* (in)	パブリケーション名	パブリケーションが同期されています (*n* は整数)。アップロードされるパブリケーションごとに 1 つの publication_*n* エントリがあります。*n* の番号はゼロから始まります。
MobiLink user (in)	Mobile Link ユーザ名	同期対象となる Mobile Link ユーザ
Foreign key table (in)	テーブル名	フック呼び出し対象の外部キー・カラムを含むテーブル
Primary key table (in)	テーブル名	フック呼び出し対象の外部キーが参照するテーブル
Role name (in)	ロール名	フック呼び出し対象の外部キーのロール名
Script version (in)	スクリプト・バージョン名	同期に使用される Mobile Link スクリプト・バージョン

説明

ダウンロード・ストリーム内のローがリモート・データベース上の外部キー関係に違反すると、ダウンロード参照整合性違反が発生します。このフックを使用すると、dbmlsync が競合の原因であるローを削除する前に、参照整合性違反を解決できます。

ダウンロードが完了すると、コミットされる前に、dbmlsync は参照整合性違反があるかどうかをチェックします。参照整合性違反が見つかると、参照整合性違反を含む外部キーを識別して、sp_hook_dbmlsync_download_log_ri_violation を呼び出します (実装されている場合)。次に、sp_hook_dbmlsync_download_ri_conflict を呼び出します (実装されている場合)。それでも競合が解決されない場合は、dbmlsync がそのローを削除します。参照整合性違反を含む残りの外部キーに対して、このプロセスが繰り返されます。

このフックは、現在同期中のテーブルに関する参照整合性違反がある場合にのみ呼び出されます。同期中ではないテーブルに関する参照整合性違反がある場合、このフックは呼び出されず、同期が失敗します。

このフックは、dbmlsync がダウンロードに使用する接続と同じ接続で呼び出されます。データベースでデータの不整合が発生する可能性があるため、このフックに明示的または暗黙的なコミットが含まれないようにしてください。ダウンロード・ストリームがコミットまたはロールバックされると、このフックのアクションがコミットまたはロールバックされます。

他のフック・アクションとは異なり、このフック中に実行される操作は次の同期中にアップロードされません。

参照

- 「sp_hook_dbmlsync_download_log_ri_violation」254 ページ

例

この例は、次に示す Department テーブルと Employee テーブルを使用します。

```
CREATE TABLE Department(
  "department_id"    integer primary key
 );

 CREATE TABLE Employee(
   "employee_id"        integer primary key,
   "department_id"    integer,
   foreign key EMPLOYEE_FK1 (department_id) references
Department
 )
```

次の sp_hook_dbmlsync_download_ri_violation の定義は、Department テーブルと Employee テーブル間の参照整合性違反をクリーンアップします。この定義によって、外部キーの役割名が検証され、欠落している department_id の値が Department テーブルに挿入されます。

```
CREATE PROCEDURE
sp_hook_dbmlsync_download_ri_violation()
 BEGIN

 IF EXISTS (SELECT * FROM #hook_dict WHERE name = 'role
name'
  AND value = 'EMPLOYEE_FK1') THEN
```

```
  -- update the Department table with missing
department_id values
  INSERT INTO Department
   SELECT distinct department_id FROM Employee
   WHERE department_id NOT IN (SELECT department_id FROM
Department)

 END IF;
```

sp_hook_dbmlsync_download_sql_error

機能

Mobile Link 同期サーバによって送信されたダウンロード・ストリームの適用中に発生したデータベース・エラーを処理します。

#hook_dict テーブルのロー

名前	値	説明
Table name (in)	テーブル名	エラーの発生時に操作中だったテーブル。dbmlsync がテーブルを識別できない場合、値は空の文字列になります。
Continue (in|out)	**true** | **false**	エラーを無視して同期を継続するかどうかを示します。sp_hook_dbmlsync_download_fatal_sql_error フックを呼び出し、同期を停止するには、このパラメータを **False** に設定してください。このパラメータを **True** に設定すると、dbmlsync はエラーを無視し、同期を続行します。この結果、データが失われることがあります。
SQL error code (in)	SQL エラー・コード	操作が失敗した時にデータベースから返される SQL エラー・コードを識別します。
Publication_*n* (in)	パブリケーション名	パブリケーションが同期されています (*n* は整数)。アップロードされるパブリケーションごとに 1 つの publication_*n* エントリがあります。*n* の番号はゼロから始まります。
MobiLink user (in)	Mobile Link ユーザ名	同期対象となる Mobile Link ユーザ
Script version (in)	スクリプト・バージョン名	同期に使用される Mobile Link スクリプト・バージョン

説明

この名前のプロシージャが存在する場合、同期のダウンロード・フェーズ中にデータベース・エラーが検出されたときに呼び出されます。このプロシージャは、エラーを無視して同期を継続できる場合にのみ呼び出されます。致命的なエラーの場合は、sp_hook_dbmlsync_download_fatal_SQL_error プロシージャが呼び出されます。

警告

continue を True に設定すると、dbmlsync はデータベース・エラーを無視し、同期を続行します。失敗した操作をもう一度実行することはありません。このため、ダウンロードの一部またはすべてが失われることがあります。失われるデータの量は、発生したエラーの種類、発生したタイミング、リカバリするためにフックが使用される段階によって異なります。どのデータが失われるかを予測するのは非常に困難であるため、この機能を使用するときは最大限の注意を払ってください。ほとんどの場合、SQL エラーが発生した後は、処理を続行しないようにしてください。

ダウンロード・ストリームがコミットまたはロールバックされると、このプロシージャのアクションがコミットまたはロールバックされます。

参照

- 「同期イベント・フックの順序」226 ページ
- 「sp_hook_dbmlsync_download_fatal_sql_error」252 ページ

例

次のテーブルを使用して、SQL エラーのログを取るとします。

```
CREATE TABLE "DBA"." LogSQLErrorTable "
 (
  "id"      int PRIMARY KEY NOT NULL,
  "name"       varchar(255) NOT NULL,
  "value"  timestamp NOT NULL
 )
```

次に、同期のダウンロード・フェーズでデータベース・エラーが発生した場合に、SQL エラー・コード、テーブル名などの同期情報のログを取る例を示します。

```
CREATE PROCEDURE sp_hook_dbmlsync_download_sql_error()
 BEGIN
  --
  -- get the next LogSQLErrorTable id
  --
     DECLARE nextid int;

  SELECT max(id)
   INTO nextid
   FROM LogSQLErrorTable;

  IF nextid is NULL THEN
   SET nextid = 0
  ELSE
   SET nextid = nextid +1
  END IF;

  INSERT INTO LogSQLErrorTable(id,name, value )
   SELECT nextid,#hook_dict.name, #hook_dict.value
   FROM #hook_dict;
 END;
```

sp_hook_dbmlsync_download_table_begin

機能

このストアド・プロシージャを使用して、各テーブルがダウンロードされる直前にカスタム・アクションを追加します。

#hook_dict テーブルのロー

名前	値	説明
Table name (in)	テーブル名	操作が適用される予定のテーブル
Publication_*n* (in)	パブリケーション名	パブリケーションが同期されています (*n* は整数)。アップロードされるパブリケーションごとに 1 つの publication_*n* エントリがあります。*n* の番号はゼロから始まります。
MobiLink user (in)	Mobile Link ユーザ名	同期対象となる Mobile Link ユーザ
Script version (in)	スクリプト・バージョン名	同期に使用される Mobile Link スクリプト・バージョン

説明

この名前のプロシージャが存在する場合、ダウンロードされた操作がテーブルに適用される直前にテーブルごとに呼び出されます。ダウンロード・ストリームがコミットまたはロールバックされると、このプロシージャのアクションがコミットまたはロールバックされます。

参照

- 「同期イベント・フックの順序」226 ページ

例

次のテーブルを使用して、リモート・データベース上の同期イベントのログを取るとします。

```
CREATE TABLE SyncLog
 (
  "event_id"              integer NOT NULL DEFAULT
autoincrement ,
     "event_name"         varchar(128) NOT NULL ,
     "ml_user"               varchar(128) NULL ,
     "event_time"            timestamp NULL,
     "table_name"            varchar(128) NULL ,
     "upsert_count"          varchar(128) NULL ,
     "delete_count"          varchar(128) NULL ,
     "exit_code"             integer NULL ,
```

```
      "status_retval"         varchar(128) NULL ,
      "pubs"                     varchar(128) NULL ,
      "sync_descr "           varchar(128) NULL ,
       PRIMARY KEY ("event_id"),
 )
```

次に、テーブルがダウンロードされる直前に Mobile Link ユーザ、テーブル名、現在のタイムスタンプのログを取る例を示します。

```
CREATE PROCEDURE
sp_hook_dbmlsync_download_table_begin()
 BEGIN
      declare tbl varchar(255);

      -- load the table name from #hook_dict
   SELECT #hook_dict.value
    INTO tbl
    FROM #hook_dict
    WHERE #hook_dict.name = 'table name';

   INSERT INTO SyncLog (event_name, ml_user, table_name
     ,event_time)
    SELECT 'download_table_begin', #hook_dict.value,
tbl
     ,current timestamp
    FROM #hook_dict
    WHERE name = 'MobiLink user' ;
 END
```

sp_hook_dbmlsync_download_table_end

機能

このストアド・プロシージャを使用して、各テーブルがダウンロードされた直後にカスタム・アクションを追加します。

#hook_dict テーブルのロー

名前	値	説明
Table name (in)	テーブル名	操作が直前に適用されたテーブル
Delete count (in)	ローの数	ダウンロード・ストリームによってこのテーブルから削除されたローの数
Upsert count (in)	ローの数	ダウンロード・ストリームによってこのテーブルで更新または挿入されたローの数
Publication_*n* (in)	パブリケーション名	パブリケーションが同期されています (*n* は整数)。アップロードされるパブリケーションごとに 1 つの publication_*n* エントリがあります。*n* の番号はゼロから始まります。
MobiLink user (in)	Mobile Link ユーザ名	同期対象となる Mobile Link ユーザ
Script version (in)	スクリプト・バージョン名	同期に使用される Mobile Link スクリプト・バージョン

説明

この名前のプロシージャが存在する場合、ダウンロード・ストリームでの操作がすべてテーブルに適用された直後に呼び出されます。

ダウンロード・ストリームがコミットまたはロールバックされると、このプロシージャのアクションがコミットまたはロールバックされます。

参照

- 「同期イベント・フックの順序」226 ページ

例

次のテーブルを使用して、リモート・データベース上の同期イベントのログを取るとします。

```
CREATE TABLE SyncLog
 (
  "event_id"              integer NOT NULL DEFAULT
autoincrement ,
     "event_name"         varchar(128) NOT NULL ,
     "ml_user"               varchar(128) NULL ,
     "event_time"            timestamp NULL,
     "table_name"            varchar(128) NULL ,
     "upsert_count"          varchar(128) NULL ,
     "delete_count"          varchar(128) NULL ,
     "exit_code"             integer NULL ,
     "status_retval"         varchar(128) NULL ,
     "pubs"                     varchar(128) NULL ,
     "sync_descr "           varchar(128) NULL ,
      PRIMARY KEY ("event_id"),
 )
```

次に、テーブルがダウンロードされた直後に Mobile Link ユーザ、テーブル名、および挿入または更新されたローの数のログを取る例を示します。

```
CREATE PROCEDURE sp_hook_dbmlsync_download_table_end()
 BEGIN
     -- declare variables
     declare tbl varchar(255);
     declare upsertCnt varchar(255);
     declare deleteCnt varchar(255);

     -- load the table name from #hook_dict
     SELECT #hook_dict.value
      INTO tbl
      FROM #hook_dict
      WHERE #hook_dict.name = 'table name';

     -- load the upsert count from #hook_dict
     SELECT #hook_dict.value
      INTO upsertCnt
      FROM #hook_dict
      WHERE #hook_dict.name = 'upsert count';

     -- load the delete count from #hook_dict
     SELECT #hook_dict.value
      INTO deleteCnt
      FROM #hook_dict
      WHERE #hook_dict.name = 'delete count';
```

```
     INSERT INTO SyncLog (event_name, ml_user,
table_name,
       upsert_count, delete_count, event_time)
      SELECT 'download_table_end', #hook_dict.value,
tbl,
       upsertCnt, deleteCnt, current timestamp
      FROM #hook_dict
      WHERE name = 'MobiLink user' ;
 END
```

sp_hook_dbmlsync_end

機能

このストアド・プロシージャを使用して、同期が完了する直前にカスタム・アクションを追加します。

#hook_dict テーブルのロー

名前	値	説明
Restart (out)	**sync** | **download** | **none**	**sync** に設定すると、dbmlsync は完了した同期のリトライを行います。**True** も同じですが、廃止され、値 **sync** に置き換えられました。 **none** (デフォルト) に設定すると、dbmlsync はコマンド・ライン引数の指定に従って、停止するか、または再起動します。**False** も同じですが、廃止され、値 **none** に置き換えられました。 **download** に設定した場合、restartable download パラメータが **True** であると、dbmlsync は失敗したダウンロードを再起動します。
Exit code (in)	数値	0 (デフォルト) 以外の値が設定された場合は、同期エラーを表します。
Publication_*n* (in)	パブリケーション名	パブリケーションが同期されています (*n* は整数)。アップロードされるパブリケーションごとに 1 つの publication_*n* エントリがあります。*n* の番号はゼロから始まります。
MobiLink user (in)	Mobile Link ユーザ名	同期対象となる Mobile Link ユーザ

名前	値	説明
Upload status (in)	**committed** | **failed**	dbmlsync がアップロード・ストリームの受信確認を行おうとしたときに、Mobile Link 同期サーバから返されるステータスを指定します。 **committed**　Mobile Link 同期サーバがアップロード・ストリームを受信し、コミットしました。 **failed**　Mobile Link 同期サーバは、アップロード・ストリームをコミットしませんでした。
Script version (in)	スクリプト・バージョン名	同期に使用される Mobile Link スクリプト・バージョン
Restartable download (in)	**true** | **false**	**True** の場合は、現在の同期のダウンロードが失敗しており、再起動することができます。**False** の場合は、ダウンロードが正常に行われたか、または再起動することができません。
Restartable download size (in)	整数	restartable download パラメータが **True** である場合、このパラメータはダウンロードが失敗する前に受信したバイト数を示します。restartable download が **False** の場合、このパラメータの値は無効です。
error hook user state (in)	整数	この値にはエラーについての情報が含まれ、フック sp_hook_dbmlsync_all_error、sp_hook_dbmlsync_communication_error、sp_hook_dbmlsync_misc_error、または sp_hook_dbmlsync_sql_error から送信できます

説明

この名前のプロシージャが存在する場合、同期中に最後のイベントとして呼び出されます。

このプロシージャのアクションは、実行直後にコミットされます。

再起動パラメータを常に **sync** に設定するように sp_hook_dbmlsync_end フックが定義されており、ユーザが dbmlsync のコマンド・ラインで -n pub1, -n pub2, ... といった形式で複数のパブリケーションを指定している場合、dbmlsync は最初のパブリケーションを繰り返し同期し、2 番目のパブリケーションを同期しません。

参照

- 「クライアント同期処理のカスタマイズ」226 ページ
- 「同期イベント・フックの順序」226 ページ
- 『Mobile Link 管理ガイド』> 「失敗したダウンロードの再開」

例

次の例では、現在の同期のダウンロードが失敗して再起動が可能な場合、ダウンロードは手動で再起動されます。

```
CREATE PROCEDURE sp_hook_dbmlsync_end()
 BEGIN
   -- Restart the download if the download for the
current sync
   --  failed and can be restarted
   IF exists(SELECT * FROM #hook_dict
     WHERE name = 'restartable download' AND
value='true')
         THEN
    UPDATE #hook_dict SET value ='download' WHERE
name='restart';
   END IF;
 END
```

sp_hook_dbmlsync_log_rescan

機能

このストアド・プロシージャを使用して、再スキャンがいつ必要かプログラム的に決定できます。

#hook_dict テーブルのロー

名前	値	説明
Publication_*n* (in)	パブリケーション名	パブリケーションが同期されています (*n* は整数)。アップロードされるパブリケーションごとに 1 つの publication_*n* エントリがあります。*n* の番号はゼロから始まります。
MobiLink user (in)	Mobile Link ユーザ名	同期対象となる Mobile Link ユーザ
Discarded storage (in)	数値	最後の同期の後で破棄されるメモリのバイト数
Rescan (in|out)	**true** | **false**	フックによって True と設定されている場合、dbmlsync は次の同期の前に完全な再スキャンを行います。エントリ時には、この値は False に設定されます。
Script version (in)	スクリプト・バージョン名	同期に使用される Mobile Link スクリプト・バージョン

説明

コマンド・ラインで複数の -n オプションを指定すると、メモリ破棄の原因となる断片化が dbmlsync で起きる可能性があります。このフックにより、dbmlsync を使用してデータベース・トランザクション・ログの再スキャンを行いメモリを回復させるかどうかを決定できます。

再スキャンを強制する他の条件を満たさない場合、このフックは sp_hook_dbmlsync_end フックの直後に呼び出されます。

参照

- 「HoverRescanThreshold (hrt) 拡張オプション」153 ページ

例

次の例では、破棄された記憶領域が 1000 バイトを超える場合に #hook_dict テーブル内の再スキャン・フィールドを TRUE に設定します。

```
CREATE PROCEDURE sp_hook_dbmlsync_log_rescan ()
 BEGIN

  IF exists(SELECT * FROM #hook_dict
   WHERE name = 'Discarded storage' AND value>1000)
  THEN
   UPDATE #hook_dict SET value ='true' WHERE
name='Rescan';
  END IF;

 END
```

sp_hook_dbmlsync_logscan_begin

機能

このストアド・プロシージャを使用して、アップロード用にトランザクション・ログがスキャンされる直前にカスタム・アクションを追加します。

#hook_dict テーブルのロー

名前	値	説明
Starting log offset_*n* (in)	数値	スキャンの開始位置を示すログ・オフセット値。アップロードされるパブリケーションごとに 1 つの値があります。*n* の番号はゼロから始まります。
Log scan retry (in)	**true** | **false**	この同期でトランザクション・ログが初めてスキャンされる場合、この値は False、それ以外の場合は True。Mobile Link 同期サーバと dbmlsync でスキャン開始位置の情報が異なっている場合、ログは 2 回スキャンされます。
Publication_*n* (in)	パブリケーション名	パブリケーションが同期されています (*n* は整数)。アップロードされるパブリケーションごとに 1 つの publication_*n* エントリがあります。*n* の番号はゼロから始まります。
MobiLink user (in)	Mobile Link ユーザ名	同期対象となる Mobile Link ユーザ
Script version (in)	スクリプト・バージョン名	同期に使用される Mobile Link スクリプト・バージョン

説明

この名前のプロシージャが存在する場合、dbmlsync がトランザクション・ログをスキャンしてアップロード・ストリームをアセンブルする直前に呼び出されます。

このプロシージャのアクションは、実行直後にコミットされます。

参照

- 「同期イベント・フックの順序」226 ページ

例

次のテーブルを使用して、リモート・データベース上の同期イベントのログを取るとします。

```
CREATE TABLE SyncLog
 (
  "event_id"              integer NOT NULL DEFAULT
autoincrement ,
     "event_name"         varchar(128) NOT NULL ,
     "ml_user"               varchar(128) NULL ,
     "event_time"            timestamp NULL,
     "table_name"            varchar(128) NULL ,
     "upsert_count"          varchar(128) NULL ,
     "delete_count"          varchar(128) NULL ,
     "exit_code"             integer NULL ,
     "status_retval"         varchar(128) NULL ,
     "pubs"                     varchar(128) NULL ,
     "sync_descr "           varchar(128) NULL ,
      PRIMARY KEY ("event_id"),
 )
```

次に、アップロードのためにトランザクション・ログがスキャンされる直前に Mobile Link ユーザと現在のタイムスタンプのログを取る例を示します。

```
CREATE PROCEDURE sp_hook_dbmlsync_logscan_begin ()
 BEGIN
  -- log the synchronization event
  INSERT INTO SyncLog (event_nam
 e, ml_user,event_time)
   SELECT 'logscan_begin', #hook_dict.value, current
timestamp
   FROM #hook_dict
   WHERE name = 'MobiLink user' ;
 END
```

sp_hook_dbmlsync_logscan_end

機能

このストアド・プロシージャを使用して、アップロード用にトランザクション・ログがスキャンされた直後にカスタム・アクションを追加します。

#hook_dict テーブルのロー

名前	値	説明
Ending log offset (in)	数値	スキャンの終了位置を示すログ・オフセット値
Starting log offset_*n* (in)	数値	同期する各サブスクリプションの初期進行値。*n* 値は、Publication_*n* の値に対応します。たとえば、Starting log offset_1 は Publication_1 のオフセットです。
Log scan retry (in)	**true** | **false**	この同期でトランザクション・ログが初めてスキャンされる場合、この値は False、それ以外の場合は True。Mobile Link 同期サーバと dbmlsync でスキャン開始位置の情報が異なっている場合、ログは 2 回スキャンされます。
Publication_*n* (in)	パブリケーション名	パブリケーションが同期されています (*n* は整数)。アップロードされるパブリケーションごとに 1 つの publication_*n* エントリがあります。*n* の番号はゼロから始まります。
MobiLink user (in)	Mobile Link ユーザ名	同期対象となる Mobile Link ユーザ
Script version (in)	スクリプト・バージョン名	同期に使用される Mobile Link スクリプト・バージョン

説明

この名前のプロシージャが存在する場合、dbmlsync がアップロード・ストリームをアセンブルするためにトランザクション・ログをスキャンした直後に呼び出されます。

このプロシージャのアクションは、実行直後にコミットされます。

参照

- 「同期イベント・フックの順序」226 ページ

例

次のテーブルを使用して、リモート・データベース上の同期イベントのログを取るとします。

```
CREATE TABLE SyncLog
 (
  "event_id"              integer NOT NULL DEFAULT
autoincrement ,
     "event_name"         varchar(128) NOT NULL ,
     "ml_user"               varchar(128) NULL ,
     "event_time"            timestamp NULL,
     "table_name"            varchar(128) NULL ,
     "upsert_count"          varchar(128) NULL ,
     "delete_count"          varchar(128) NULL ,
     "exit_code"             integer NULL ,
     "status_retval"         varchar(128) NULL ,
     "pubs"                     varchar(128) NULL ,
     "sync_descr "           varchar(128) NULL ,
      PRIMARY KEY ("event_id"),
 )
```

次に、アップロードのためにトランザクション・ログがスキャンされた直後に Mobile Link ユーザと現在のタイムスタンプのログを取る例を示します。#hook_dict の log scan retry パラメータは、トランザクション・ログが複数回スキャンされるかどうかを示します。

```
CREATE PROCEDURE sp_hook_dbmlsync_logscan_end ()
 BEGIN

  declare scan_retry varchar(128);

  -- load the scan retry parameter from #hook_dict
  SELECT #hook_dict.value
   INTO scan_retry
   FROM #hook_dict
   WHERE #hook_dict.name = 'log scan retry';

  -- Determine if the log is being rescanned
  -- and log the synchronization event

  IF scan_retry='true' then
```

```
    INSERT INTO SyncLog (event_name,
ml_user,event_time,sync_descr)
     SELECT 'logscan_end', #hook_dict.value, current
timestamp,
      'Transaction log rescanned'
     FROM #hook_dict
     WHERE name = 'MobiLink user' ;
  ELSE
    INSERT INTO SyncLog (event_name,
ml_user,event_time,sync_descr)
     SELECT 'logscan_end', #hook_dict.value, current
timestamp,
      'Transaction log scanned normally'
     FROM #hook_dict
     WHERE name = 'MobiLink user' ;
  END IF;
 END
```

sp_hook_dbmlsync_process_return_code

機能

このストアド・プロシージャを使用して、リターン・コードを管理します。

#hook_dict テーブルのロー

名前	値	説明
Publication_*n* (in)	パブリケーション名	パブリケーションが同期されています (*n* は整数)。アップロードされるパブリケーションごとに 1 つの publication_*n* エントリがあります。*n* の番号はゼロから始まります。
MobiLink user (in)	Mobile Link ユーザ名	同期対象となる Mobile Link ユーザ
Fatal error (in)	**true** | **false**	dbmlsync を終了させる原因となるエラーのためにこのフックが呼び出されるときは True
Aborted synchronization (in)	**true** | **false**	sp_hook_dbmlsync_abort フックからのアボート要求のためにこのフックが呼び出されるときは True
Return code (in)	数値	直近の同期試行からのリターン・コード。0 は同期が成功したことを示します。他の値は同期が失敗したことを示します。この値は、そのフックを使用して同期をアボートするとき、sp_hook_dbmlsync_abort によって設定できます。
Last return code (in)	数値	最後にこのフックが呼び出されたときに #hook_dict テーブルの **new return code** ローに格納される値、またはこれがフックへの最初の呼び出しの場合は 0

名前	値	説明
New return code (in|out)	数値	そのプロセスに対して必要なリターン・コード。dbmlsync が終了するとき、dblmsync の **return code** はそのフックへの最後の呼び出しによってこのローに格納される値です。この値は -32768 から 32767 になります。
Script version (in)	スクリプト・バージョン名	同期に使用される Mobile Link スクリプト・バージョン

説明

コマンド・ラインで -n オプションを複数回指定する場合、スケジューリングを使用する場合、あるいは sp_hook_dbmlsync_end で restart パラメータを使用する場合、dbmlsync セッションは複数の同期を実行できます。これらの状況では、1 つ以上の同期が失敗すると、デフォルトのリターン・コードによってどれが失敗したのかが示されません。このフックを使用して、同期からのリターン・コードに基づいた dbmlsync プロセスのリターン・コードを定義できます。また、このフックを使用してリターン・コードのログを取ることもできます。

例

dbmlsync を実行して 5 つの同期を行い、リターン・コードで失敗した同期の数を示すとします。たとえば、リターン・コード 0 は失敗がないことを示し、1 は 1 つの同期が失敗したことを示します。これを実現するには、sp_hook_dbmlsync_process_return_code フックを次のように定義します。この場合、3 つの同期が失敗すると新しいリターン・コードは 3 になります。

```
CREATE PROCEDURE sp_hook_dbmlsync_process_return_code()
 BEGIN
     DECLARE    rc    integer;

     SELECT value INTO rc FROM #hook_dict WHERE name =
'return code';
     IF rc <> 0 THEN
        SELECT value INTO rc FROM #hook_dict WHERE name
```

```
= 'last return code';
        UPDATE #hook_dict SET value = rc + 1 WHERE name
= 'new return code';
     END IF;
 END;
```

参照

- 「同期イベント・フックの順序」226 ページ
- 「sp_hook_dbmlsync_abort」234 ページ

sp_hook_dbmlsync_schema_upgrade

機能

このストアド・プロシージャを使用して、スキーマを修正する SQL スクリプトを実行します。

#hook_dict テーブルのロー

名前	値	説明
Publication_*n* (in)	パブリケーション名	パブリケーションが同期されています (*n* は整数)。アップロードされるパブリケーションごとに 1 つの publication_*n* エントリがあります。*n* の番号はゼロから始まります。
MobiLink user (in)	Mobile Link ユーザ名	同期対象となる Mobile Link ユーザ
スクリプトのバージョン	スクリプト・バージョンの名前	同期しているスクリプト・バージョン
Drop hook (out)	**never** | **always** | **on success**	次のいずれかの値を取ります。 **never** - (デフォルト) データベースから sp_hook_dbmlsync_schema_upgrade フックを削除しません。 **always** - フックの実行を試行した後、データベースから sp_hook_dbmlsync_schema_upgrade フックを削除します。 **on success** - フックの実行に成功した場合、データベースから sp_hook_dbmlsync_schema_upgrade フックを削除します。dbmlsync の -eh オプションが使用されている場合、または dbmlsync の拡張オプション IgnoreHookErrors が True に設定されている場合、on success は always と同じです。

名前	値	説明
Script version (in)	スクリプト・バージョン名	同期に使用される Mobile Link スクリプト・バージョン

説明

このストアド・プロシージャは、配備されたリモート・データベースでスキーマの変更を行うためのものです。スキーマ更新のためにこのフックを使用すると、スキーマが更新される前にリモート・データベースのすべての変更が同期されます。これは、データベースが同期可能である状態を維持するために必要です。このフックが使用中の場合、dbmlsync の拡張オプション LockTables を off に設定しないでください (LockTables はデフォルトでは on です)。

同期中に Mobile Link によってアップロードが正常に適用され、受信確認されると、このフックは sp_hook_dbmlsync_download_end フックの後、sp_hook_dbmlsync_end フックの前に呼び出されます。このフックは、ダウンロード専用の同期中、またはファイル・ベースのダウンロードが作成または適用されるときには呼び出されません。

このフックで実行されるアクションは、フックが完了した直後にコミットされます。

参照

- 『Mobile Link 管理ガイド』> 「リモート・データベースでのスキーマの変更」

例

次の例では、sp_hook_dbmlsync_schema_upgrade プロシージャを使用して、リモート・データベース上の Dealer テーブルにカラムを追加します。アップグレードが成功すると、sp_hook_dbmlsync_schema_upgrade フックは削除されます。

```
CREATE PROCEDURE sp_hook_dbmlsync_schema_upgrade()
 BEGIN
  -- Upgrade the schema of the Dealer table
  -- here we add a column
  ALTER TABLE Dealer
   ADD dealer_description varchar(128);

  -- if the schema upgrade is successful, drop this hook
```

```
 UPDATE #hook_dict
  SET value = 'on success'
  WHERE name = 'drop hook';
END
```

sp_hook_dbmlsync_set_extended_options

機能

同期に適用される拡張オプションを指定して、次の同期の動作をプログラムによってカスタマイズするには、このストアド・プロシージャを使用します。

#hook_dict テーブルのロー

名前	値	説明
Publication_*n* (in)	パブリケーション名	パブリケーションが同期されています (*n* は整数)。アップロードされるパブリケーションごとに 1 つの publication_*n* エントリがあります。*n* の番号はゼロから始まります。
MobiLink user (in)	Mobile Link ユーザ名	同期対象となる Mobile Link ユーザ
Extended options (out)	*opt*=*val*;...	次の同期のために追加される拡張オプション

説明

この名前のプロシージャが存在する場合、これは sp_hook_dbmlsync_delay フックの直前に呼び出されます。

このフックで指定される拡張オプションは、パブリケーションと Mobile Link ユーザ・エントリによって識別される同期にのみ適用され、このフックが同じ同期を対象として次に呼び出されるまで適用されます。

このフックを使用して、スケジュール・オプションを指定することはできません。

このプロシージャのアクションは、実行直後にコミットされます。

参照

- 「同期イベント・フックの順序」226 ページ
- 「dbmlsync 拡張オプション」135 ページ
- 「拡張オプションと接続パラメータの優先順位」95 ページ

例

次の例では、sp_hook_dbmlsync_set_extended_options を使用して、SendColumnNames 拡張オプションを指定します。この拡張オプションは、pub1 が同期される場合だけ適用されます。

```
CREATE PROCEDURE sp_hook_dbmlsync_set_extended_options
()
 BEGIN
  IF exists(SELECT * FROM #hook_dict
   WHERE name LIKE 'publication_%' AND value='pub1')
  THEN
    -- specify the sendcolumnnames=on extended option
    UPDATE #hook_dict
              SET value = 'SendColumnNames=on'
     WHERE name = 'extended options';
  END IF;
 END
```

sp_hook_dbmlsync_upload_begin

機能

このストアド・プロシージャを使用して、アップロード転送の直前にカスタム・アクションを追加します。

#hook_dict テーブルのロー

名前	値	説明
Publication_*n* (in)	パブリケーション名	パブリケーションが同期されています (*n* は整数)。アップロードされるパブリケーションごとに 1 つの publication_*n* エントリがあります。*n* の番号はゼロから始まります。
MobiLink user (in)	Mobile Link ユーザ名	同期対象となる Mobile Link ユーザ
Script version (in)	スクリプト・バージョン名	同期に使用される Mobile Link スクリプト・バージョン

説明

この名前のプロシージャが存在する場合、dbmlsync がアップロード・ストリームを送信する直前に呼び出されます。

このプロシージャのアクションは、実行直後にコミットされます。

参照

- 「同期イベント・フックの順序」226 ページ

例

次のテーブルを使用して、リモート・データベース上の同期イベントのログを取るとします。

```
CREATE TABLE SyncLog
 (
  "event_id"              integer NOT NULL DEFAULT
autoincrement ,
     "event_name"         varchar(128) NOT NULL ,
     "ml_user"               varchar(128) NULL ,
     "event_time"            timestamp NULL,
     "table_name"            varchar(128) NULL ,
     "upsert_count"          varchar(128) NULL ,
     "delete_count"          varchar(128) NULL ,
     "exit_code"             integer NULL ,
     "status_retval"         varchar(128) NULL ,
```

```
    "pubs"                      varchar(128) NULL ,
    "sync_descr "           varchar(128) NULL ,
     PRIMARY KEY ("event_id"),
)
```

次に、アップロードの転送の直前に Mobile Link ユーザと現在のタイムスタンプのログを取る例を示します。

```
CREATE PROCEDURE sp_hook_dbmlsync_upload_begin ()
 BEGIN
  INSERT INTO SyncLog (event_name, ml_user,event_time)
   SELECT 'upload_begin', #hook_dict.value, current
timestamp
   FROM #hook_dict
   WHERE name = 'MobiLink user';
 END
```

sp_hook_dbmlsync_upload_end

機能

Mobile Link 同期サーバによってアップロード・ストリームが受信されたことを dbmlsync で確認した後に、このストアド・プロシージャを使用してカスタム・アクションを追加します。

#hook_dict テーブルのロー

名前	値	説明
Failure cause (in)	下の「説明」の値の範囲を参照	アップロード障害の原因。詳細については、下の「説明」を参照してください。

名前	値	説明
Upload status (in)	**retry** | **committed** | **failed** | **unknown**	dbmlsync がアップロード・ストリームの受信確認を行おうとしたときに、Mobile Link 同期サーバから返されるステータスを指定します。 **retry**　アップロード・ストリームの開始位置であるログ・オフセットの値が、Mobile Link 同期サーバと dbmlsync で異なっていました。Mobile Link 同期サーバは、アップロード・ストリームをコミットしませんでした。dbmlsync ユーティリティは、新しいログ・オフセットから始まる別のアップロード・ストリームを送信しようとします。 **committed**　Mobile Link 同期サーバがアップロード・ストリームを受信し、コミットしました。 **failed**　Mobile Link 同期サーバは、アップロード・ストリームをコミットしませんでした。 **unknown** Mobile Link 同期サーバが -tu オプションを使用して起動され、トランザクション・レベルのアップロードが発生しました。アップロードされる各トランザクションでは、sp_hook_dbmlsync_upload_begin フックと sp_hook_dbmlsync_upload_end フックが呼び出され、upload status の値は、最後のものを除き、常に **unknown** です。

名前	値	説明
Publication_*n* (in)	パブリケーション名	パブリケーションが同期されています (*n* は整数)。アップロードされるパブリケーションごとに 1 つの publication_*n* エントリがあります。*n* の番号はゼロから始まります。
MobiLink user (in)	Mobile Link ユーザ名	同期対象となる Mobile Link ユーザ
Script version (in)	スクリプト・バージョン名	同期に使用される Mobile Link スクリプト・バージョン
Authentication value (in)	値	この値は、サーバ上の authenticate_user スクリプト、authenticate_user_hashed スクリプト、または authenticate_parameters スクリプトによって生成されます。upload status が unknown であるとき、またはリモート・データベースと統合データベースに格納されているログ・オフセット間の競合が原因でアップロードが再送信された後に upload_end フックが呼び出されたとき、値は空の文字列になります。

説明

この名前のプロシージャが存在する場合、dbmlsync がアップロード・ストリームを送信し、Mobile Link 同期サーバから受信確認を受け取った直後に呼び出されます。

このプロシージャのアクションは、実行直後にコミットされます。

#hook_dict テーブルの failure cause ローに有効なパラメータ値の範囲は、次のとおりです。

- **UPLD_ERR_COMMUNICATIONS_FAILURE** 通信エラーが発生しました。

- **UPLD_ERR_LOG_OFFSET_MISMATCH**　リモート・データベースと統合データベースに格納されているログ・オフセット間に競合があるため、アップロードに失敗しました。
- **UPLD_ERR_GENERAL_FAILURE**　アップロードに失敗しました。原因は不明です。
- **UPLD_ERR_INVALID_USERID_OR_PASSWORD**　ユーザ ID またはパスワードが正しくありません。
- **UPLD_ERR_USERID_OR_PASSWORD_EXPIRED**　ユーザ ID またはパスワードの有効期限が切れています。
- **UPLD_ERR_USERID_ALREADY_IN_USE**　このユーザ ID はすでに使用されています。
- **UPLD_ERR_DOWNLOAD_NOT_AVAILABLE**　統合データベース側でアップロードがコミットされましたが、Mobile Link はダウンロード・ストリームを生成できなかったため、エラーが発生しました。
- **UPLD_ERR_PROTOCOL_MISMATCH**　*Dbmlsync* が Mobile Link 同期サーバから予期しないデータを受信しました。
- **UPLD_ERR_SQLCODE_n**　この *n* は整数です。統合データベースに SQL エラーが発生しました。指定された整数は、発生したエラーを表す SQLCODE です。

参照

- 「同期イベント・フックの順序」226 ページ

例

次のテーブルを使用して、リモート・データベース上の同期イベントのログを取るとします。

```
CREATE TABLE SyncLog
 (
  "event_id"              integer NOT NULL DEFAULT
autoincrement ,
     "event_name"         varchar(128) NOT NULL ,
     "ml_user"               varchar(128) NULL ,
     "event_time"            timestamp NULL,
     "table_name"            varchar(128) NULL ,
     "upsert_count"          varchar(128) NULL ,
     "delete_count"          varchar(128) NULL ,
```

```
    "exit_code"             integer NULL ,
    "status_retval"         varchar(128) NULL ,
    "pubs"                     varchar(128) NULL ,
    "sync_descr "           varchar(128) NULL ,
     PRIMARY KEY ("event_id"),
)
```

次に、Mobile Link 同期サーバがアップロード・ストリームを受信したかを dbmlsync が検証した後で Mobile Link ユーザと現在のタイムスタンプのログを取る例を示します。

```
CREATE PROCEDURE sp_hook_dbmlsync_upload_end ()
 BEGIN

  declare status_return_value varchar(255);

  -- store status_return_value
  SELECT #hook_dict.value
   INTO status_return_value
   FROM #hook_dict
   WHERE #hook_dict.name = 'upload status';

     INSERT INTO SyncLog (event_name, ml_user,
    status_retval, event_time)
   SELECT 'upload_end', #hook_dict.value,
    status_return_value, current timestamp
   FROM #hook_dict
   WHERE name = 'MobiLink user';
 END
```

sp_hook_dbmlsync_validate_download_file

機能

このフックを使用して、ダウンロード・ファイルがリモート・データベースに適用できるかどうかを決定するためのカスタム論理を実装します。

#hook_dict テーブルのロー

名前	値	説明
Publication_*n* (in)	パブリケーション名	パブリケーションが同期されています (*n* は整数)。アップロードされるパブリケーションごとに 1 つの publication_*n* エントリがあります。publication_*n* と generation number_*n* の *n* は一致します。*n* の番号はゼロから始まります。
MobiLink user (in)	Mobile Link ユーザ名	同期対象となる Mobile Link ユーザ
File last download time (in)		ダウンロード・ファイルの最後のダウンロード時間 (ダウンロード・ファイルには、最後のダウンロード時間とその前のダウンロード時間の間に変更されたすべてのローが含まれます)
File next last download time (in)		ダウンロード・ファイルの最後から 2 番目のダウンロード時間 (ダウンロード・ファイルには、最後のダウンロード時間とその前のダウンロード時間の間に変更されたすべてのローが含まれます)
File creation time (in)		ダウンロード・ファイルが作成された時間

名前	値	説明
File generation number_*n* (in)	数値	ダウンロード・ファイルからの世代番号。各 publication_*n* エントリに対して、1 つのファイル世代番号 number_*n* があります。publication_*n* と generation number_*n* の *n* は一致します。*n* の番号はゼロから始まります。
User data (in)	文字列	ダウンロード・ファイルが作成されたときに、dbmlsync -be オプションで指定した文字列
Apply file (in|out)	**True**|**False**	True (デフォルト) の場合、ダウンロード・ファイルは dbmlsync の他の検証チェックを通過したときだけ適用されます。False の場合、ダウンロード・ファイルはリモート・データベースに適用されません。
Check generation number (in|out)	**True**|**False**	true (デフォルト) の場合、dbmlsync は世代番号を検証します。ダウンロード・ファイル内の世代番号がリモート・データベース内の世代番号と一致しない場合、dbmlsync はダウンロード・ファイルを適用しません。false の場合、dbmlsync は世代番号をチェックしません。
Setting generation number (in)	**true** | **false**	ダウンロード・ファイルが作成されたときに -bg スイッチが使用された場合は True。-bg が使用された場合、リモート・データベースの世代番号はダウンロード・ファイルから更新され、通常の世代番号チェックは行われません。

説明

このストアド・プロシージャを使用して、ダウンロード・ファイルが適用できるか決定するためのカスタム・チェックを実装します。

ダウンロード・ファイルに含まれる世代番号またはタイムスタンプと、リモート・データベースに格納されている世代番号またはタイムスタンプを比較する場合、それらのデータは SYSSYNC テーブルと SYSPUBLICATION テーブルから問い合わせることができます。

-ba オプションが指定されていると、このフックが呼び出されます。このフックは、sp_hook_dbmlsync_upload_end フックの後、sp_hook_dbmlsync_download_begin フックの前に呼び出されます。

このフックのアクションは、フックが完了した直後にコミットされます。

参照

- 「-be オプション」131 ページ
- 「-bg オプション」131 ページ
- 『Mobile Link 管理ガイド』> 「ファイルベースのダウンロード」

例

次の例では、ユーザ文字列「sales manager data」を含まないダウンロードファイルを適用しません。

```
CREATE PROCEDURE
sp_hook_dbmlsync_validate_download_file ()
 BEGIN
   IF NOT exists(SELECT * FROM #hook_dict
    WHERE name = 'User data' AND value='sales manager
data')
   THEN
     UPDATE #hook_dict
                SET value = 'false' WHERE name = 'Apply
file';
   END IF;
 END
```

第 9 章

dbmlsync 統合コンポーネント

この章の内容

この章では、dbmlsync 統合コンポーネントの使用方法を説明します。dbmlsync 統合コンポーネントでは、Adaptive Server Anywhere クライアント・アプリケーションをカスタマイズできます。

概要

dbmlsync 統合コンポーネントは、同期をアプリケーションに追加するのに役立ちます。このコンポーネントには、Adaptive Server Anywhere クライアントの動作を調整するための一連のプロパティ、イベント、およびメソッドを提供します。

dbmlsync 統合コンポーネントは、2 つの形式で使用できます。いずれも、同じプロパティ、イベント、メソッドを公開しています。

- ビジュアル・コンポーネント。標準の dbmlsync ユーザ・インタフェースをアプリケーションに簡単に統合できます。
- 非ビジュアル・コンポーネント。ユーザ・インタフェースを使用せずに、または独自に作成したカスタム・ユーザ・インタフェースを使用して、コンポーネントの機能にアクセスできます。

dbmlsync 統合コンポーネントを使用すると、アプリケーションで同期を開始し、同期の進行状況についての情報を受け取ってから、同期イベントに基づいた特別な処理を実装できます。

dbmlsync 統合コンポーネントの代わりとして、Adaptive Server Anywhere に付属するデータベース・ツール・ライブラリの a_sync_db 構造体を使用できます。

a_sync_db の詳細については、『ASA プログラミング・ガイド』> 「a_sync_db 構造体」を参照してください。

Adaptive Server Anywhere クライアントの詳細については，「Mobile Link 同期クライアント」124 ページを参照してください。

サポートされるプラットフォーム

dbmlsync 統合コンポーネントは、ActiveX をサポートする Windows CE バージョンを含め、Mobile Link でサポートされているすべての Windows オペレーティング・システムで使用できます。

サポートされている開発環境には、Microsoft Visual Basic 6.0、eMbedded Visual Basic、Visual Studio .NET があります。

dbmlsync 統合コンポーネントの設定

dbmlsync 統合コンポーネントは、さまざまなプログラミング環境で使用できます。このコンポーネントの設定方法については、ご使用のプログラミング環境のマニュアルで参照してください。以下に、Windows フォーム・アプリケーションとコンソール・アプリケーションで統合コンポーネントを設定する手順を示します。

Windows フォーム・アプリケーション

❖ **Windows アプリケーションで dbmlsync 統合コンポーネントを設定するには、次の手順に従います。**

1. 参照をコンポーネントに追加します。
   - ビジュアル・コンポーネントの場合は、*dbmlsynccomg.dll* を使用します。これは、SQL Anywhere のインストール環境の *win32* サブディレクトリにあります。この DLL を開発環境のツールボックスに追加します。
   - 非ビジュアル・コンポーネントの場合は、*dbmlsynccom.dll* を使用します。これは、SQL Anywhere のインストール環境の *win32* サブディレクトリにあります。
2. コンポーネントのインスタンスを作成します。
   - ビジュアル・コンポーネントの場合は、ツールボックスを使用してフォーム上でインスタンスをドラッグ・アンド・ドロップします。
   - 非ビジュアル・コンポーネントの場合は、初期化コードでグローバル変数を宣言し、インスタンスを作成します。
     - たとえば、Visual Basic .NET を使用している場合、グローバル変数の宣言として次のように入力します。

```
Dim WithEvents dbmlsync1 As
DbmlsyncCOM.Dbmlsync
```

インスタンスを作成するには、次のように入力します。

```
dbmlsync1 = New DbmlsyncCOM.Dbmlsync
```

- たとえば、C# を使用している場合、グローバル変数の宣言として次のように入力します。

```
private DbmlsyncCOM.Dbmlsync
dbmlsync1;
```

インスタンスを作成するには、次のように入力します。

```
dbmlsync1 = new
DbmlsyncCOM.Dbmlsync();
```

注意： C# の場合、dbmlsync イベントのデリゲートを割り当てる (この作業の手順 3) 前にインスタンスを作成する必要があります。

3 同期イベントを処理する関数を追加します。

- たとえば、Visual Basic .NET を使用している場合は、関数のプロトタイプで Handles キーワードを使用します。

```
Private Sub dbmlsync1_ProgressMessage(ByVal msg
As String)
 Handles dbmlsync1.ProgressMessage
     lblProgressMessage.Text = msg
 End Sub
```

- C# を使用している場合は、イベント・ハンドラ関数を記述します。

```
private void dbmlsync1_ProgressMessage(string
msg)
 {
     lblProgressMessage.Text = msg;
 }
```

C# の初期化コード内では、デリゲートを使用して、dbmlsync イベントのイベント・ハンドラ関数を指定します。次の例では、dbmlsync ProgressMessage イベントを処理する dbmlsync1_ProgressMessage を割り当てています。

```
dbmlsync1.ProgressMessage +=
 new
DbmlsyncCOM._IDbmlsyncEvents_ProgressMessageEve
ntHandler(dbmlsync1_ProgressMessage);
```

イベントの詳細については、「dbmlsync 統合コンポーネントのイベント」312 ページを参照してください。

4 プロパティを設定します。

次の例は、UploadEventsEnabled プロパティを True に設定します。こうすると、UploadRow イベントが有効になります。

```
dbmlsync1.UploadEventsEnabled = True
```

UploadRow イベントの詳細については、「UploadRow イベント」325 ページを参照してください。

プロパティの詳細については、「dbmlsync 統合コンポーネントのプロパティ」306 ページを参照してください。

5 Run メソッドを使用して、1 つまたは複数の同期を開始します。次に例を示します。

```
dbmlsync1.Run("-c eng=rem1;uid=dba;pwd=sql")
```

Run メソッドの詳細については、「Run メソッド」304 ページを参照してください。

コンソール・アプリケーション

コンソール・アプリケーションが使用できるのは、非ビジュアル・コンポーネントだけです。

完全に修飾された dbmlsync 統合コンポーネントの名前は、DbmlsyncCOM.Dbmlsync です。

非ビジュアルな dbmlsync 統合コンポーネントを使用するコンソール・アプリケーションでは、Windows のメッセージ・キューをポンプする (キューからメッセージを取得する) 必要があります。たとえば、.NET アプリケーションでは、System.Windows.Application ネームスペースで以下のメソッドを使用する必要があります。

System.Windows.Application.Run() 現在のスレッドで、標準のアプリケーションのメッセージ・ループを起動します。

System.Windows.Application.Exit() すべてのメッセージ・ポンプに対して、終了してからすべてのアプリケーション・ウィンドウを閉じるように要求します。

❖ コンソール・アプリケーション (Visual Basic.NET) で dbmlsync 統合コンポーネントを設定するには、次の手順に従います。

1 参照を *System.Windows.Forms.dll* に追加します。

2 非ビジュアル・コンポーネント *dbmlsynccom.dll* に参照を追加します。この DLL は SQL Anywhere インストール環境の *win32* サブディレクトリにあります。

3 dbmlsync 統合コンポーネントのインスタンスを作成します。

- グローバル変数の宣言として、次のように入力します。

```
Dim WithEvents dbmlsync1 As
DbmlsyncCOM.Dbmlsync
```

- コンポーネントのインスタンスを作成するには、次のように入力します。

```
dbmlsync1 = New DbmlsyncCOM.Dbmlsync
```

4 プロパティを設定します。

次の例は、UploadEventsEnabled プロパティを True に設定します。こうすると、UploadRow イベントが有効になります。

```
dbmlsync1.UploadEventsEnabled = True
```

プロパティの詳細については、「dbmlsync 統合コンポーネントのプロパティ」306 ページを参照してください。

UploadRow イベントの詳細については、「UploadRow イベント」325 ページを参照してください。

5 Run メソッドを使用して、1 つまたは複数の同期を開始します。次に例を示します。

```
dbmlsync1.Run("-c eng=rem1;uid=dba;pwd=sql")
```

Run メソッドの詳細については、「Run メソッド」304 ページを参照してください。

6 System.Windows.Application Run メソッドを使用して、Windows メッセージ・キューのポンプを有効にします。

```
System.Windows.Forms.Application.Run()
```

7 同期イベントを処理する論理を追加します。

イベントの詳細については、「dbmlsync 統合コンポーネントのイベント」312 ページを参照してください。

8 System.Windows.Application Exit メソッドを使用して、すべてのメッセージ・ポンプに対して終了するよう要求します。

```
System.Windows.Application.Exit()
```

dbmlsync 統合コンポーネントのメソッド

次に、DbmlsyncCOM.Dbmlsync クラスによって実装されるメソッドを示します。

Run メソッド

機能

dbmlsync コマンド・ライン・オプションを使用して、1 つまたは複数の同期を開始します。

構文

Run(ByVal *cmdLine* As String **)**
DbmlsyncCOM.Dbmlsync のメンバ

パラメータ

cmdLine dbmlsync オプションを指定する文字列です。

説明

オプションのリストは、「dbmlsync オプション」129 ページを参照してください。

cmdLine パラメータには、dbmlsync コマンド・ライン・ユーティリティとの同期を実行する場合に使用するのと同じオプションを含める必要があります。たとえば、次のコマンド・ラインを使用して dbmlsync を実行するとします。

```
dbmlsync -c uid=dba;pwd=sql
```

これは、次の Run メソッドの呼び出しと同じです。

```
dbmlsync1.Run "-c uid=dba;pwd=sql"
```

例

次の例は、remote1 というリモート・データベースの同期を開始します。

```
dbmlsync1.Run "-c eng=remote1;uid=dba;pwd=sql"
```

Stop メソッド

機能

アクティブな同期に終了するように要求します。

構文

Stop( )

DbmlsyncCOM.Dbmlsync のメンバ

説明

Stop メソッドは、アクティブな同期を終了する要求を発行します。このメソッドはすぐに返されます。

ビジュアルな dbmlsync 統合コンポーネントに組み込まれた停止ボタンは、このメソッドを自動的に呼び出します。

例

次の例は、dbmlsync 統合コンポーネントのインスタンスである dbmlsync1 によって実行されている同期を停止します。

```
dbmlsync1.Stop
```

dbmlsync 統合コンポーネントのプロパティ

dbmlsync 統合コンポーネントのプロパティを使用すると、コンポーネントの動作をカスタマイズしたり、実行中の同期の状態を調べたりできます。

Path プロパティ

機能

dbmlsync.exe のロケーションを指定します。

構文

Public Property **Path( )** As String
DbmlsyncCOM.Dbmlsync のメンバ

説明

Windows の PATH 環境変数で指定されているディレクトリに *dbmlsync.exe* が置かれている場合は、このプロパティを設定する必要はありません。

例

次の例は、dbmlsync 統合コンポーネントのインスタンスのパスを設定します。

```
dbmlsync1.Path = "c:¥program files¥sybase¥sql anywhere
9¥win32"
```

UploadEventsEnabled プロパティ

機能

UploadRow イベントを有効にします。

構文

Public Property **UploadEventsEnabled()** As Boolean
DbmlsyncCOM.Dbmlsync のメンバ

説明

UploadRow イベントを処理する場合は、このプロパティを True に設定してください。デフォルトは False で、UploadRow イベントは無効になっています。

UploadRow イベントの詳細については、「UploadRow イベント」325 ページを参照してください。

例

次の例は、UploadEventsEnabled を True に設定します。

```
dbmlsync1.UploadEventsEnabled = True
```

DownloadEventsEnabled プロパティ

機能

DownloadRow イベントを有効にします。

構文

Public Property **DownloadEventsEnabled( )** As Boolean
DbmlsyncCOM.Dbmlsync のメンバ

説明

DownloadRow イベントを処理する場合は、このプロパティを True に設定してください。デフォルトは False で、DownloadRow イベントは無効になっています。

DownloadRow イベントの詳細については、「DownloadRow イベント」316 ページを参照してください。

例

次の例は、DownloadEventsEnabled を True に設定します。

```
dbmlsync1.DownloadEventsEnabled = True
```

ErrorMessageEnabled プロパティ

機能

MsgError 型のメッセージに対して Message イベントが呼び出されないようにします。

構文

Public Property **ErrorMessageEnabled( )** As Boolean
DbmlsyncCOM.Dbmlsync のメンバ

説明

エラー情報を Message イベントで処理しない場合は、パフォーマンスを向上させるため、このプロパティを False に設定してください。デフォルトは True で、Message イベントをトリガする MsgError 型のメッセージが有効になっています。

Message イベントの詳細については、「Message イベント」321 ページを参照してください。

例

次の例は、ErrorMessageEnabled を False に設定します。

```
dbmlsync1.ErrorMessageEnabled = False
```

WarningMessageEnabled プロパティ

機能

MsgWarning 型のメッセージに対して Message イベントが呼び出されないようにします。

構文

Public Property **WarningMessageEnabled( )** As Boolean
DbmlsyncCOM.Dbmlsync のメンバ

説明

警告情報を Message イベントで処理しない場合は、パフォーマンスを向上させるため、このプロパティを False に設定してください。デフォルトは True で、Message イベントをトリガする MsgWarning 型のメッセージが有効になっています。

Message イベントの詳細については、「Message イベント」321 ページを参照してください。

例

次の例は、WarningMessageEnabled を False に設定します。

```
dbmlsync1.WarningMessageEnabled = False
```

InfoMessageEnabled プロパティ

機能

MsgInfo 型のメッセージに対して Message イベントが呼び出されないようにします。

構文

Public Property **InfoMessageEnabled( )** As Boolean
DbmlsyncCOM.Dbmlsync のメンバ

説明

通常の進行状況情報を Message イベントで処理しない場合は、パフォーマンスを向上させるため、このプロパティを False に設定してください。デフォルトは True で、Message イベントをトリガする MsgInfo 型のメッセージが有効になっています。

Message イベントの詳細については、「Message イベント」321 ページを参照してください。

例

次の例は、InfoMessageEnabled を False に設定します。

```
dbmlsync1.InfoMessageEnabled = False
```

DetailedInfoMessageEnabled プロパティ

機能

MsgDetailedInfo 型のメッセージに対して Message イベントが呼び出されないようにします。

構文

Public Property **DetailedInfoMessageEnabled( )** As Boolean
DbmlsyncCOM.Dbmlsync のメンバ

説明

詳細な進行状況情報を Message イベントで処理しない場合は、パフォーマンスを向上させるため、このプロパティを False に設定してください。デフォルトは True で、Message イベントをトリガする MsgDetailedInfo 型のメッセージが有効になっています。

Message イベントの詳細については、「Message イベント」321 ページを参照してください。

例

次の例は、DetailedInfoMessageEnabled を False に設定します。

```
dbmlsync1.DetailedInfoMessageEnabled = False
```

UseVB6Types プロパティ

機能

Visual Basic 6 を使用している場合は、このプロパティを True に設定して、UploadRow イベントと DownloadRow イベントから返されるロー・データの処理を簡素化します。

構文

Public Property **DetailedInfoMessageEnabled( )** As Boolean
DbmlsyncCOM.Dbmlsync のメンバ

説明

Visual Basic 6 では、符号なしの 32 ビット値とすべての 64 ビット値がサポートされていません。これらの型のデータは、IRowTransferData オブジェクトの ColumnValue プロパティから返されることがあります。UseVB6Types を True に設定すると、これらの型のデータは Visual Basic 6 でサポートされる別の型に変換され、処理が簡素化されます。Uint32 の値は double 型に変換され、64 ビット値は文字列に変換されます。

IRowTransferData インタフェースの詳細については、「IRowTransferData インタフェース」328 ページを参照してください。

UploadRow イベントの詳細については、「UploadRow イベント」325 ページを参照してください。

DownloadRow イベントの詳細については、「DownloadRow イベント」316 ページを参照してください。

例

次の例は、Visual Basic 6.0 で使用される dbmlsync 統合コンポーネントのインスタンスのデータ型の強制変換を有効にします。

```
dbmlsync1.UseVB6Types = True
```

ExitCode プロパティ

機能

最新の Run メソッドの呼び出しによって開始された同期からの終了コードを返します。

構文

Public Property **ExitCode( )** As Integer

DbmlsyncCOM.Dbmlsync のメンバ

説明

ExitCode プロパティは、最後に呼び出された Run メソッドによって開始された同期の終了コードを返します。0 は同期が成功したことを示します。他の値は同期が失敗したことを示します。

注意：
DoneExecution イベントがトリガされる前にこのプロパティの値を取得すると、終了コードが無効な値になることがあります。

例

次の例は、DoneExecution イベントがトリガされた時点での最新の同期からの終了コードを表示します。

```
Private Sub dbmlsync1_DoneExecution() Handles
dbmlsync1.DoneExecution
         MsgBox(dbmlsync1.ExitCode)
     End Sub
```

EventChannelSize プロパティ

機能　メソッドの呼び出しの処理に使用される内部バッファのサイズを指定します。

構文　Public Property **EventChannelSize( )** As Integer
DbmlsyncCOM.Dbmlsync のメンバ

説明　ほとんどの場合、このプロパティを変更する必要はありません。

DispatchChannelSize プロパティ

機能　イベント情報の処理に使用される内部バッファのサイズを指定します。

構文　Public Property **DispatchChannelSize( )** As Integer
DbmlsyncCOM.Dbmlsync のメンバ

説明　ほとんどの場合、このプロパティを変更する必要はありません。

dbmlsync 統合コンポーネントのイベント

イベントは、クライアント・アプリケーションが同期の進行状況の情報を受け取り、それに基づいてアクションを行うためのメカニズムを提供します。

BeginDownload イベント

BeginDownload イベントは、同期のダウンロード処理開始時にトリガされます。

構文

Public Event **BeginDownload( )**
DbmlsyncCOM.Dbmlsync のメンバ

説明

このイベントを使用して、同期のダウンロード処理開始時にカスタム・アクションを追加します。

例

次に示す Visual Basic .NET の例は、BeginDownload イベントがトリガされたときにメッセージを出力します。

```
Private Sub dbmlsync1_BeginDownload()
 Handles dbmlsync1.BeginDownload

        MsgBox("Beginning Download")

 End Sub
```

BeginLogScan イベント

BeginLogScan イベントは、アップロード・ストリームをアセンブルするために dbmlsync がトランザクション・ログをスキャンする直前にトリガされます。

構文

Public Event **BeginLogScan(** ByVal *rescanLog* As Boolean **)**
DbmlsyncCOM.Dbmlsync のメンバ

パラメータ

rescanLog　この同期でトランザクション・ログが初めてスキャンされる場合、この値は False、それ以外の場合は True。Mobile Link 同期サーバと dbmlsync でスキャン開始位置の情報が異なっている場合、ログは 2 回スキャンされます。

説明

このイベントを使用して、アップロード用にトランザクション・ログがスキャンされる直前にカスタム・アクションを追加します。

例

次に示す Visual Basic .NET の例は、BeginLogScan イベントがトリガされたときにメッセージを出力します。

```
Private Sub dbmlsync1_BeginLogScan(
  ByVal rescanLog As Boolean
 )
 Handles dbmlsync1.BeginLogScan

        MsgBox("Begin Log Scan")

 End Sub
```

BeginSynchronization イベント

BeginSynchronization イベントは、各同期の開始時にトリガされます。

構文

Public Event **BeginSynchronization(_**
ByVal *userName* As String, _
ByVal *pubNames* As String _
)
DbmlsyncCOM.Dbmlsync のメンバ

パラメータ

userName　同期対象となる Mobile Link ユーザ。

pubNames　同期されるパブリケーション。パブリケーションが複数の場合は、カンマで区切ったリストで指定します。

説明

このイベントを使用して、同期の開始時にカスタム・アクションを追加します。

例

次に示す Visual Basic .NET の例は、BeginSynchronization イベントがトリガされたときにメッセージを出力します。メッセージには、ユーザ名とパブリケーション名が出力されます。

```
Private Sub dbmlsync1_BeginSynchronization(
  ByVal userName As String,
  ByVal pubNames As String
 )
 Handles dbmlsync1.BeginSynchronization

        MsgBox("Beginning synchronization for: " +
userName _
      + " publication: " + pubNames)

 End Sub
```

BeginUpload イベント

BeginUpload イベントは、アップロード転送の直前にトリガされます。

構文

Public Event **BeginUpload( )**
DbmlsyncCOM.Dbmlsync のメンバ

説明

このイベントを使用して、Mobile Link 同期サーバへのアップロード転送の直前にカスタム・アクションを追加します。

例

次に示す Visual Basic .NET の例は、BeginUpload イベントがトリガされたときにメッセージを出力します。

```
Private Sub dbmlsync1_BeginUpload()
 Handles dbmlsync1.BeginUpload

         MsgBox("Begin Upload")

 End Sub
```

ConnectMobilink イベント

ConnectMobilink イベントは、コンポーネントが Mobile Link 同期サーバに接続する直前にトリガされます。

構文

Public Event **ConnectMobilink( )**
DbmlsyncCOM.Dbmlsync のメンバ

説明

このイベントを使用して、リモート・データベースが Mobile Link 同期サーバに接続する直前にカスタム・アクションを追加します。この段階では、dbmlsync はアップロード・ストリームの生成を完了しています。

ConnectMobiLink イベントは、BeginSynchronization イベントの後で発生します。

例

次に示す Visual Basic .NET の例は、ConnectMobilink イベントがトリガされたときにメッセージを出力します。

```
Private Sub dbmlsync1_ConnectMobilink()
 Handles dbmlsync1.ConnectMobilink

         MsgBox("Connecting to the MobiLink
synchronization server")

 End Sub
```

DisconnectMobilink イベント

DisconnectMobilink イベントは、コンポーネントが Mobile Link 同期サーバから切断した直後にトリガされます。

構文

Public Event **DisconnectMobilink( )**
DbmlsyncCOM.Dbmlsync のメンバ

説明

このイベントを使用して、リモート・データベースが Mobile Link 同期サーバから切断した直後にカスタム・アクションを追加します。

例

次に示す Visual Basic .NET の例は、DisconnectMobilink イベントがトリガされたときにメッセージを出力します。

```
Private Sub dbmlsync1_DisconnectMobilink()
 Handles dbmlsync1.DisconnectMobilink

         MsgBox("Disconnected from the MobiLink
synchronization server")

 End Sub
```

DoneExecution イベント

DoneExecution イベントは、Run メソッドの呼び出しによって開始された同期がすべて完了したときにトリガされます。

構文

Public Event **DoneExecution( )**
DbmlsyncCOM.Dbmlsync のメンバ

説明

このイベントを使用して、Run メソッドの呼び出しによって開始された同期がすべて完了したときにカスタム・アクションを追加します。

例

ExitCode プロパティを使用する次の Visual Basic .NET 例は、最後に呼び出された Run メソッドによって開始された同期からの終了コードを出力します。

```
Private Sub dbmlsync1_DoneExecution()
 Handles dbmlsync1.DoneExecution

        MsgBox(dbmlsync1.ExitCode)

 End Sub
```

DownloadRow イベント

DownloadRow イベントは、Mobile Link 同期サーバからローがダウンロードされるときにトリガされます。

構文

Public Event **DownloadRow(**
ByVal *rowData* As DbmlsyncCOM.IRowTransferData
)
DbmlsyncCOM.Dbmlsync のメンバ

パラメータ

rowData ダウンロードされるローの詳細な情報が含まれている IRowTransferData オブジェクト。

IRowTransferData インタフェースの詳細については、「IRowTransferData インタフェース」328 ページを参照してください。

説明

このイベントを使用して、Mobile Link 同期サーバからダウンロードされるローを調べます。

DownloadRow イベントを有効にするには、DownloadEventsEnabled プロパティを使用します。

詳細については、「DownloadEventsEnabled プロパティ」307 ページを参照してください。

ローのダウンロード・イベントで削除操作が発生したときは、プライマリ・キー・カラムの値しか使用できません。

例

次に示す Visual Basic .NET の例は、DownloadRow イベントでローのすべてのカラムを反復処理します。この処理で、カラム値が null であるかどうかが判別され、カラム名と値が出力されます。

```
Private Sub dbmlsync1_DownloadRow(
  ByVal rowData As DbmlsyncCOM.IRowTransferData
 )
 Handles dbmlsync1.DownloadRow

 Dim liX As Integer
 For liX = 0 To rowData.ColumnCount - 1
     If VarType(rowData.ColumnValue(liX)) <>
VariantType.Null Then
         ' output the non-null column value
         MsgBox("Column " + CStr(liX) + ": " +
rowData.ColumnName(liX) + _
         ", " + CStr(rowData.ColumnValue(liX)))
     Else
         ' output 'NULL' for the column value
         MsgBox("Column " + CStr(liX) + ": " +
rowData.ColumnName(liX) + _
         ", " + "NULL")
     End If
 Next liX

 End Sub
```

EndDownload イベント

EndDownload イベントは、同期処理のダウンロード処理終了時にトリガされます。

構文

Public Event **EndDownload(**
long *upsertRows*,
long *deleteRows*
)
DbmlsyncCOM.Dbmlsync のメンバ

パラメータ

upsertRows ダウンロード処理で更新または挿入されたローの数を示します。

deleteRows ダウンロード処理で削除されたローの数を示します。

説明

このイベントを使用して、同期のダウンロード処理終了時にカスタム・アクションを追加します。

例

次に示す Visual Basic .NET の例は、EndDownload イベントがトリガされたときに、挿入、更新、削除されたローの数とメッセージを出力します。

```
Private Sub dbmlsync1_EndDownload(
  ByVal upsertRows As Integer,
  ByVal deleteRows As Integer
 )
 Handles dbmlsync1.EndDownload

     MsgBox("Download complete." + _
      CStr(upsertRows) + "Rows updated or inserted" + _
      CStr(deleteRows) + "Rows deleted")

 End Sub
```

EndLogScan イベント

EndLogScan イベントは、アップロード用にトランザクション・ログがスキャンされた直後にトリガされます。

構文

Public Event **EndLogScan( )**
DbmlsyncCOM.Dbmlsync のメンバ

説明

このイベントを使用して、アップロード用にトランザクション・ログがスキャンされた直後にカスタム・アクションを追加します。

例

次に示す Visual Basic .NET の例は、EndLogScan イベントがトリガされたときにメッセージを出力します。

```
Private Sub dbmlsync1_EndLogScan()
 Handles dbmlsync1.EndLogScan

        MsgBox("Scan of transaction log complete...")

 End Sub
```

EndSynchronization イベント

EndSynchronization イベントは、同期の完了時にトリガされます。

構文

Public Event **EndSynchronization(**
ByVal *exitCode* As Integer,
ByRef *restart* As Boolean
)
DbmlsyncCOM.Dbmlsync のメンバ

パラメータ

exitCode　0 (ゼロ) 以外の値に設定されている場合は、同期エラーが発生したことを示します。

restart　このイベントが呼び出されるときは、この値は False に設定されます。イベントによって値が True に変更された場合、dbmlsync は同期を再起動します。

説明

このイベントを使用して、同期の完了時にカスタム・アクションを追加します。

例

次に示す Visual Basic .NET の例は、EndSynchronization イベントを使用して、最大で 5 つの失敗した同期を再起動します。同期の再起動にすべて失敗すると、"All restart attempts failed" というメッセージと、終了コードが出力されます。同期が成功すると、"Synchronization succeeded" というメッセージと、終了コードが出力されます。

```
' Global variable for the number of restarts
 Dim numberOfRestarts As Integer

 Private Sub dbmlsync1_EndSynchronization(
  ByVal ExitCode As Integer,
  ByRef restart As Boolean
```

```
 )
 Handles dbmlsync1.EndSynchronization

     If numberOfRestarts < 5 Then
         MsgBox("Restart Number: " +
CStr(numberOfRestarts + 1))
         If ExitCode <> 0 Then
             ' restart the failed synchronization
             restart = True
             numberOfRestarts = numberOfRestarts + 1
         Else
             ' the last synchronization succeeded
             MsgBox("Synchronization succeeded. " + _
               "Exit code: " + CStr(ExitCode))
         End If
     Else
         MsgBox("All restart attempts failed. " + _
              "Exit code: " + CStr(ExitCode))
     End If

 End Sub
```

EndUpload イベント

EndUpload イベントは、Mobile Link 同期サーバへのアップロード・ストリーム転送の直後にトリガされます。

構文

Public Event **EndUpload( )**

DbmlsyncCOM.Dbmlsync のメンバ

説明

このイベントを使用して、dbmlsync から Mobile Link 同期サーバへのアップロード・ストリーム転送の直後にカスタム・アクションを追加します。

例

次に示す Visual Basic .NET の例は、EndUpload イベントがトリガされたときにメッセージを出力します。

```
Private Sub dbmlsync1_EndUpload()
 Handles dbmlsync1.EndUpload

     MsgBox("End Upload")

 End Sub
```

Message イベント

Message イベントは、dbmlsync が情報のログを取るときにトリガされます。

構文

Public Event **Message(_**
ByVal *msgClass* As DbmlsyncCOM.MessageClass, _
ByVal *msgID* As Integer, ByVal *msg* As String_
)
DbmlsyncCOM.Dbmlsync のメンバ

パラメータ

msgClass　メッセージの重大度を示します。次のいずれかの値を取ります。

- **MsgInfo** 同期の進行状況情報が含まれるメッセージです。
- **MsgDetailedInfo** MsgInfo と似ていますが、より詳細な情報が含まれています。
- **MsgWarning** 問題が発生する可能性があることを示すメッセージですが、正常な同期の妨げにはならない問題です。
- **MsgError** 正常な同期の妨げになる問題を示すメッセージです。

msgID　メッセージのユニークな識別子です。msgID がゼロの場合、メッセージにはユニークな識別子はありません。

msg　メッセージのテキストです。

説明

このイベントを使用して、dbmlsync がログを取った情報を受け取ります。

例

次に示す Visual Basic .NET の例は、dbmlsync がログを取ったメッセージをリストボックス・コントロールに追加します。

```
Private Sub dbmlsync1_Message(
  ByVal msgClass As DbmlsyncCOM.MessageClass,
  ByVal msgId As Integer, ByVal msg As String
 )
 Handles dbmlsync1.Message

     Select Case msgClass
```

```
            Case DbmlsyncCOM.MessageClass.MsgError
                lstMessages.Items.Add("Error: " + msg)
            Case DbmlsyncCOM.MessageClass.MsgWarning
                lstMessages.Items.Add("Warning: " + msg)
            Case DbmlsyncCOM.MessageClass.MsgInfo
                lstMessages.Items.Add("Info: " + msg)
        End Select

    End Sub
```

ProgressIndex イベント

ProgressIndex イベントは、dbmlsync が進行状況バーを更新したときにトリガされます。

構文

Public Event **ProgressIndex(_**
ByVal *index* As Integer, _
ByVal *max* As Integer _
)
DbmlsyncCOM.Dbmlsync のメンバ

パラメータ

インデックス　同期の進行状況を表す整数。

max　進行状況の最大値。この値がゼロである場合、最大値はこのイベントが最後に呼び出されてから変更されていません。

説明

このイベントを使用して、進行状況バーのような、進行状況を示すインジケータを更新します。

例

次に示す Visual Basic .NET の例は、Index の値に基づいて、進行状況バー・コントロールを更新します。インデックスの最大値は、同期の開始時に設定されます。

```
Private Sub dbmlsync1_ProgressIndex(
  ByVal index As Integer,
  ByVal max As Integer
 )
 Handles dbmlsync1.ProgressIndex

     If max <> 0 Then
```

```
            ProgressBar1.Maximum = max
        End If
        ProgressBar1.Value = index

    End Sub
```

ProgressMessage イベント

ProgressMessage イベントは、同期の進行状況情報が変更されたときにトリガされます。

構文

Public Event **ProgressMessage(** ByVal *msg* As String **)**
DbmlsyncCOM.Dbmlsync のメンバ

パラメータ

msg　新しい進行状況の文字列。

説明

このイベントを使用して、通常は dbmlsync の進行状況バー上に表示される文字列を受け取ります。

例

次に示す Visual Basic .NET の例は、ProgressMessage イベントがトリガされたときに進行状況のラベルの値を設定します。

```
Private Sub dbmlsync1_ProgressMessage(
  ByVal msg As String
 )
 Handles dbmlsync1.ProgressMessage

     lblProgressMessage.Text = msg

 End Sub
```

SetTitle イベント

SetTitle イベントは、ステータス情報が変更されたときにトリガされます。dbmlsync ユーティリティでは、この情報はタイトル・バーに表示されます。

構文

Public Event **SetTitle(** ByVal *title* **)** As String
)
DbmlsyncCOM.Dbmlsync のメンバ

パラメータ

title dbmlsync ウィンドウのタイトル・バー内のタイトル。

説明

このイベントを使用して、dbmlsync ウィンドウの値が変更されたときに、このウィンドウに通常表示されるタイトルを受け取ります。

例

次に示す Visual Basic .NET の例は、SetTitle イベントがトリガされたときに Windows フォームのタイトルを設定します。

```
Private Sub dbmlsync1_SetTitle(
  ByVal title As String
 )
 Handles dbmlsync1.SetTitle

        Me.Text = title

 End Sub
```

UploadAck イベント

UploadAck イベントは、コンポーネントがアップロード・ストリームの確認を Mobile Link 同期サーバから受信した後にトリガされます。

構文

Public Event **UploadAck(_**
ByVal *status* As DbmlsyncCOM.UploadAckStatus _
)
DbmlsyncCOM.Dbmlsync のメンバ

パラメータ

status アップロード・ストリームが処理された後に、Mobile Link によってリモートに返されたステータスを示します。以下のいずれかの値になります。

- **StatCommitted** Mobile Link 同期サーバがアップロード・ストリームを受信し、コミットしたことを示します。
- **StatRetry** アップロード・ストリームの開始位置であるログ・オフセットの値が、Mobile Link 同期サーバと dbmlsync で異なっていたことを示します。Mobile Link 同期サーバは、アップロード・ストリームをコミットしませんでした。コンポーネントは、Mobile Link 同期サーバのログ・オフセットから開始して別のアップロード・ストリームを送信します。

- **StatFailed** Mobile Link 同期サーバがアップロード・ストリームをコミットしなかったことを示します。

説明

このイベントを使用して、dbmlsync がアップロード・ストリームの確認を Mobile Link 同期サーバから受信した後にカスタム・アクションを追加します。

例

次に示す Visual Basic .NET の例は、UploadAck イベントがトリガされたときに、アップロードが失敗していた場合にメッセージを出力します。

```
Private Sub dbmlsync1_UploadAck(ByVal status As
DbmlsyncCOM.UploadAckStatus) Handles
dbmlsync1.UploadAck

     If status = DbmlsyncCOM.UploadAckStatus.StatFailed
Then
          MsgBox("Upload Failed")
     End If

 End Sub
```

UploadRow イベント

UploadRow イベントは、Mobile Link 同期サーバにローがアップロードされるときにトリガされます。

構文

Public Event **UploadRow(**
ByVal *rowData* As DbmlsyncCOM.IRowTransferData
)
DbmlsyncCOM.Dbmlsync のメンバ

パラメータ

rowData　アップロードされるローの詳細な情報が含まれている IRowTransferData オブジェクト。

IRowTransferData インタフェースの詳細については、「IRowTransferData インタフェース」328 ページを参照してください。

説明

このイベントを使用して、Mobile Link 同期サーバにアップロードされるローを調べます。

UploadRow イベントを有効にするには、UploadEventsEnabled プロパティを使用します。

詳細については、「UploadEventsEnabled プロパティ」306 ページを参照してください。

例

次に示す Visual Basic .NET の例は、UploadRow イベントでローのすべてのカラムを反復処理します。この処理で、カラム値が null であるかどうかが判別され、カラム名と値が出力されます。

```
Private Sub dbmlsync1_UploadRow(
  ByVal rowData As DbmlsyncCOM.IRowTransferData
 )
 Handles dbmlsync1.UploadRow

 Dim liX As Integer
 For liX = 0 To rowData.ColumnCount - 1
     If VarType(rowData.ColumnValue(liX)) <>
VariantType.Null Then
         ' output the non-null column value
         MsgBox("Column " + CStr(liX) + ": " +
rowData.ColumnName(liX) + _
         ", " + CStr(rowData.ColumnValue(liX)))
     Else
         ' output 'NULL' for the column value
         MsgBox("Column " + CStr(liX) + ": " +
rowData.ColumnName(liX) + _
         ", " + "NULL")
     End If
 Next liX

 End Sub
```

WaitingForUploadAck イベント

WaitingForUploadAck イベントは、コンポーネントが Mobile Link 同期サーバからのアップロード確認の待機を開始したときにトリガされます。

構文

Public Event **WaitingForUploadAck( )**

DbmlsyncCOM.Dbmlsync のメンバ

説明

このイベントを使用して、コンポーネントが Mobile Link 同期サーバからのアップロード確認を待機しているときにカスタム・アクションを追加します。

例

次に示す Visual Basic .NET の例は、WaitingForUploadAck イベントがトリガされたときにメッセージを出力します。

```
Private Sub dbmlsync1_WaitingForUploadAck()
 Handles dbmlsync1.WaitingForUploadAck

     MsgBox("Waiting for Upload Acknowledgement")

 End Sub
```

IRowTransferData インタフェース

Public Interface **IRowTransferData**
DbmlsyncCOM のメンバ

UploadRow イベントと DownloadRow イベントは、パラメータとして DbmlsyncCOM.IRowTransferData オブジェクトを受け入れ、アップロードされたローとダウンロードされたローを調べます。このインタフェースは、テーブル名、ローの操作、カラム名などのローの詳細な情報を定義します。

RowOperation プロパティ

機能

ローで実行される操作を指定します。

構文

Public Property **RowOperation( )** As DbmlsyncCOM.RowEventOp
DbmlsyncCOM.IRowTransferData のメンバ

説明

このプロパティの値は、次のいずれかです。

OpInsert　ローが挿入されました。

OpUpdate　ローが更新されました。

OpDelete　ローが削除されました。

OpTruncate　テーブルがトランケートされました (テーブルのすべてのローが削除されました)。RowOperation プロパティがこの値を保持しているときは、ColumnName プロパティと ColumnValue プロパティは無効な情報を返します。

注意 : DownloadRow イベントの場合、アップサート (更新または挿入) 操作に値 OpInsert が指定されます。

TableName プロパティ

機能

アップロード操作またはダウンロード操作が発生するテーブルの名前です。

構文

Public Property **TableName( )** As String
DbmlsyncCOM.IRowTransferData のメンバ

説明

TableName プロパティは、アップロード操作またはダウンロード操作が発生するテーブルの名前を指定します。次の例は、UploadRow イベントでの TableName プロパティの使い方を示しています。

UploadRow イベントの詳細については、「UploadRow イベント」325 ページを参照してください。

例

Visual Basic .NET の例を示します。

```
Private Sub dbmlsync1_UploadRow(
  ByVal rowData As DbmlsyncCOM.IRowTransferData
 )
 Handles dbmlsync1.UploadRow

     MsgBox ("Table name:" + rowData.TableName)

 End Sub
```

ColumnName プロパティ

機能

アップロード操作またはダウンロード操作が発生するローのカラム名を取得します。

構文

Public Property **ColumnName(**ByVal *index* As String**)** As Object
DbmlsyncCOM.IRowTransferData のメンバ

パラメータ

インデックス　取得するカラム名を指定するゼロベースの整数。インデックス値の範囲は、ゼロから、ColumnCount プロパティの値未満までです。

ColumnCount プロパティの詳細については、「ColumnCount プロパティ」332 ページを参照してください。

説明

同じインデックスで ColumnValue プロパティを使用して、対応するカラム値を取得できます。

例

次に示す Visual Basic .NET の例は、UploadRow イベントでローのすべてのカラムを反復処理します。この処理で、カラム値が null であるかどうかが判別され、カラム名と値が出力されます。

UploadRow イベントの詳細については、「UploadRow イベント」325 ページを参照してください。

```
Private Sub dbmlsync1_UploadRow(
  ByVal rowData As DbmlsyncCOM.IRowTransferData
 )
 Handles dbmlsync1.UploadRow

 Dim liX As Integer
 For liX = 0 To rowData.ColumnCount - 1
     If VarType(rowData.ColumnValue(liX)) <>
VariantType.Null Then
         ' output the non-null column value
         MsgBox("Column " + CStr(liX) + ": " +
rowData.ColumnName(liX) + _
         ", " + CStr(rowData.ColumnValue(liX)))
     Else
         ' output 'NULL' for the column value
         MsgBox("Column " + CStr(liX) + ": " +
rowData.ColumnName(liX) + _
         ", " + "NULL")
     End If
 Next liX

 End Sub
```

ColumnValue プロパティ

機能

アップロード操作またはダウンロード操作が発生するカラムの値を取得します。

構文

Public Property **ColumnValue(** ByVal *index* As Integer **)** As Object
DbmlsyncCOM.IRowTransferData のメンバ

パラメータ

インデックス　取得するカラム値を指定するゼロベースの整数。インデックス値の範囲は、ゼロから、ColumnCount プロパティの値未満までです。

ColumnCount プロパティの詳細については、「ColumnCount プロパティ」332 ページを参照してください。

説明

更新操作が発生した場合、このプロパティで指定されるカラム値は更新が適用された後の値です。

同じインデックスで ColumnName プロパティを使用して、対応するカラム名を取得できます。

blob のカラム値を、このプロパティを通じて使用することはできません。blob のカラムが検出された場合、ColumnValue は文字列 "(blob)" です。

例

次に示す Visual Basic .NET の例は、UploadRow イベントでローのすべてのカラムを反復処理します。この処理で、カラム値が null であるかどうかが判別され、カラム名と値が出力されます。

UploadRow イベントの詳細については、「UploadRow イベント」325 ページを参照してください。

```
Private Sub dbmlsync1_UploadRow(
  ByVal rowData As DbmlsyncCOM.IRowTransferData
 )
 Handles dbmlsync1.UploadRow

 Dim liX As Integer
 For liX = 0 To rowData.ColumnCount - 1
     If VarType(rowData.ColumnValue(liX)) <>
VariantType.Null Then
         ' output the non-null column value
         MsgBox("Column " + CStr(liX) + ": " +
rowData.ColumnName(liX) + _
         ", " + CStr(rowData.ColumnValue(liX)))
     Else
         ' output 'NULL' for the column value
         MsgBox("Column " + CStr(liX) + ": " +
rowData.ColumnName(liX) + _
         ", " + "NULL")
     End If
 Next liX

 End Sub
```

ColumnCount プロパティ

機能

アップロード操作またはダウンロード操作が発生するローに含まれるカラム数です。

構文

Public Property **ColumnCount( )** As Integer
DbmlsyncCOM.IRowTransferData のメンバ

説明

ColumnCount プロパティは、アップロード操作またはダウンロード操作が発生するローのカラム数を指定します。次の例は、UploadRow イベントでの ColumnCount プロパティの使い方を示しています。

UploadRow イベントの詳細については、「UploadRow イベント」325 ページを参照してください。

例

Visual Basic .NET の例を示します。

```
Private Sub dbmlsync1_UploadRow(
  ByVal rowData As DbmlsyncCOM.IRowTransferData
 )
 Handles dbmlsync1.UploadRow

     MsgBox "Number of Columns:" +
CStr(rowData.ColumnCount)

 End Sub
```

第 10 章

dbmlsync の DBTools インタフェース

この章の内容

この章では、dbmlsync の DBTools インタフェースを設定して使用する方法について説明します。このインタフェースを使用することで、Adaptive Server Anywhere クライアント・アプリケーションをカスタマイズできます。

概要

データベース・ツール (DBTools) は、同期を含むデータベース管理をアプリケーションに統合するために使用できるライブラリです。データベース管理ユーティリティはすべて、DBTools によって構築されます。

DBTools ライブラリの詳細については、『ASA プログラミング・ガイド』> 「データベース・ツール・インタフェース」を参照してください。

dbmlsync 用の DBTools インタフェースを使用することで、Mobile Link 同期クライアント・アプリケーションに同期機能を統合できます。たとえば、このインタフェースを使用し、カスタム・ユーザ・インタフェースに dbmlsync の出力メッセージを表示できます。

dbmlsync 用の DBTools インタフェースを使用する代わりに、dbmlsync 統合コンポーネントを使用することもできます。

dbmlsync 統合コンポーネントの詳細については、「この章の内容」297 ページを参照してください。

dbmlsync 用の DBTools インタフェースは、Mobile Link 同期クライアントを設定および実行する次の要素から構成されます。

- **a_sync_db 構造体** この構造体は、dbmlsync コマンド・ライン・オプションに対応する設定を保持します。これらの設定によって同期をカスタマイズできます。この構造体には、同期と進行状況情報を受け取るコールバック関数へのポインタも含まれます。

  a_sync_db の詳細については、『ASA プログラミング・ガイド』> 「a_sync_db 構造体」を参照してください。

- **a_syncpub 構造体** この構造体は、パブリケーション情報を保持します。同期用パブリケーションのリンク・リストを指定できます。

  a_syncpub の詳細については、『ASA プログラミング・ガイド』> 「a_syncpub 構造体」を参照してください。

- **DBSynchronizeLog 関数** この関数は同期処理を開始します。この関数のパラメータは、a_sync_db インスタンスへのポインタだけです。

  DBSynchronizeLog の詳細については、『ASA プログラミング・ガイド』> 「DBSynchronizeLog 関数」を参照してください。

dbmlsync の DBTools インタフェースの設定

この項では、dbmlsync の DBTools インタフェースの基本的な使用手順を示します。

DBTools ライブラリの詳細については、『ASA プログラミング・ガイド』> 「データベース・ツール・インタフェースの概要」を参照してください。

ご使用の開発環境でのインポート・ライブラリの使用についての詳細は、『ASA プログラミング・ガイド』> 「データベース・ツール・インタフェースの使い方」を参照してください。

❖ DBTtools インタフェースを使用して dbmlsync の設定と起動を行うには、次の手順に従います。

1 DBTools ヘッダ・ファイルをインクルードします。

DBTools ヘッダ・ファイル *dbtools.h* は、DBTools ライブラリのエントリ・ポイントをリストし、必要なデータ型を定義します。

```
#include "dbtools.h"
```

2 DBTools インタフェースを起動します。

- a_dbtools_info 構造体の宣言と初期化を行います。

```
a_dbtools_info   info;
 short ret;
 ...
 // clear a_dbtools_info fields
 memset( &info, 0, sizeof( info ) );
 info.errorrtn = dbsyncErrorCallBack;
```

dbsyncErrorCallBack はエラー・メッセージを処理する関数であり、この作業の手順 4 で定義します。

- DBToolsInit 関数を使用して DBTools を起動します。

```
ret = DBToolsInit( &info );
 if( ret != 0 ) {
  printf("dbtools initialization failure ¥n");
 }
```

DBTools の初期化の詳細については、次の項を参照してください。

- 『ASA プログラミング・ガイド』> 「データベース・ツール・インタフェースの使い方」
- 『ASA プログラミング・ガイド』> 「a_dbtools_info 構造体」
- 『ASA プログラミング・ガイド』> 「DBToolsInit 関数」

3 a_sync_db インスタンスを初期化します。

- a_sync_db インスタンスを宣言します。たとえば、次のように dbsync_info というインスタンスを宣言します。

```
a_sync_db dbsync_info;
```

- a_sync_db 構造体のフィールドをクリアします。

```
memset( &dbsync_info, 0, sizeof( dbsync_info ) );
```

- 必須の a_sync_db のフィールドを設定します。

```
dbsync_info.version = DB_TOOLS_VERSION_NUMBER;
 dbsync_info.output_to_mobile_link = 1;
 dbsync_info.default_window_title
   = "dbmlsync dbtools sample";
```

- データベース接続文字列を設定します。

```
dbsync_info.connectparms = "uid=dba;pwd=sql";
```

データベース接続パラメータの詳細については、「-c オプション」132 ページを参照してください。

- 同期をカスタマイズするための他の a_sync_db のフィールドを設定します。

ほとんどのフィールドは、dbmlsync コマンド・ライン・オプションに対応しています。この対応の詳細については、*dbtools.h* を参照してください。

次の例では、冗長オペレーションが指定されています。

```
dbsync_info.verbose_upload = 1;
 dbsync_info.verbose_option_info = 1;
 dbsync_info.verbose_row_data = 1;
 dbsync_info.verbose_row_cnts = 1;
```

- 他の a_sync_db のフィールドを初期化します。

a_sync_db のフィールドの詳細については、『ASA プログラミング・ガイド』> 「a_sync_db 構造体」を参照してください。

4 同期中にフィードバックを受け取るコールバック関数を作成し、これらの関数を適切な a_sync_db のフィールドに割り当てます。

次の関数は、標準出力ストリームを使用して dbmlsync のエラー、ログ、および進行状況情報を表示します。

DBTools のコールバック関数の詳細については、『ASA プログラミング・ガイド』> 「コールバック関数の使い方」を参照してください。

- たとえば、生成されたエラー・メッセージを処理する dbsyncErrorCallBack という関数を作成します。

```
extern short _callback dbsyncErrorCallBack(
char *str )
 {
     if( str != NULL ) {
         printf( "Error Msg    %s¥n", str );
     }
     return 0;
 }
```

- たとえば、生成された警告メッセージを処理する dbsyncWarningCallBack という関数を作成します。

```
extern short _callback dbsyncWarningCallBack(
char *str )
 {
     if( str != NULL ) {
         printf( "Warning Msg  %s¥n", str );
     }
     return 0;
 }
```

- たとえば、冗長情報メッセージを受け取る dbsyncLogCallBack という関数を作成します。この情報メッセージは、ウィンドウに表示する代わりにファイルに記録できます。

```
extern short _callback dbsyncLogCallBack( char
*str )
 {
     if( str != NULL ) {
         printf( "Log Msg      %s¥n", str );
     }
     return 0;
 }
```

- たとえば、同期中に生成された情報メッセージを受け取る dbsyncMsgCallBack という関数を作成します。

```
 extern short _callback dbsyncMsgCallBack( char
*str )
 {
     if( str != NULL ) {
         printf( "Display Msg  %s¥n", str );
     }
     return 0;
 }
```

- たとえば、進行状況テキストを受け取る dbsyncProgressMessageCallBack という関数を作成します。dbmlsync ユーティリティでは、このテキストは進行状況バーの真上に表示されます。

```
 extern short _callback
dbsyncProgressMessageCallBack(
  char *str )
 {
     if( str != NULL ) {
         printf( "ProgressText %s¥n", str );
     }
     return 0;
 }
```

- たとえば、進行状況インジケータまたは進行状況バーを更新するために情報を受け取る dbsyncProgressIndexCallBack という関数を作成します。この関数は、次の 2 つのパラメータを受け取ります。

  - **index** 同期の現在の進行状況を表す整数。
  - **max** 進行状況の最大値。この値がゼロである場合、最大値はこのイベントが最後に呼び出されてから変更されていません。

```
extern short _callback
dbsyncProgressIndexCallBack
 (a_sql_uint32 index, a_sql_uint32 max )
 {
     printf( "ProgressIndex    Index %d Max:
%d¥n",
             index, max );
     return 0;
 }
```

次に、このコールバックへの呼び出しの一般的な順序を示します。

```
// example calling sequence
 // dbsyncProgressIndexCallBack( 0, 100 );
 //    dbsyncProgressIndexCallBack( 25, 0 );
 //    dbsyncProgressIndexCallBack( 50, 0 );
 //    dbsyncProgressIndexCallBack( 75, 0 );
 //    dbsyncProgressIndexCallBack( 100, 0 );
```

この順序では、0% 完了、25% 完了、50% 完了、75% 完了、100% 完了に設定された進行状況バーが表示されます。

- たとえば、ステータス情報を受け取る dbsyncWindowTitleCallBack という関数を作成します。dbmlsync ユーティリティでは、この情報はタイトル・バーに表示されます。

```
extern short _callback
dbsyncWindowTitleCallBack(
  char *title )
 {
     printf( "Window Title     %s¥n", title );
     return 0;
 }
```

dbsyncMsgQueueCallBack 関数は、遅延またはスリープが必要な場合に呼び出します。この関数は、*dllapi.h* に定義されている次の値のいずれかを返す必要があります。

- **MSGQ_SLEEP_THROUGH** 要求したミリ秒の間ルーチンがスリープしたことを示します。ほとんどの場合はこの値が返されるようにします。
- **MSGQ_SHUTDOWN_REQUESTED** できるだけ早く同期を終了したいことを示します。
- **MSGQ_SYNC_REQUESTED** 要求したミリ秒に達しないうちにルーチンがスリープ状態を終了したこと、および、同期が現在進行中でない場合はただちに次の同期をとり始めることを示します。

```
extern short _callback dbsyncMsgQueueCallBack(
             a_sql_uint32
sleep_period_in_milliseconds )
 {

  printf( "Sleep %d ms¥n",
```

```
sleep_period_in_milliseconds );
  Sleep( sleep_period_in_milliseconds );
  return MSGQ_SLEEP_THROUGH;
 }
```

- 適切な a_sync_db 同期構造体のフィールドにコールバック関数のポインタを割り当てます。

```
// set call back functions
 dbsync_info.errorrtn    = dbsyncErrorCallBack;
 dbsync_info.warningrtn  =
dbsyncWarningCallBack;
 dbsync_info.logrtn      = dbsyncLogCallBack;
 dbsync_info.msgrtn      = dbsyncMsgCallBack;
 dbsync_info.msgqueuertn =
dbsyncMsgQueueCallBack;
 dbsync_info.progress_index_rtn
      = dbsyncProgressIndexCallBack;
 dbsync_info.progress_msg_rtn
      = dbsyncProgressMessageCallBack;
 dbsync_info.set_window_title_rtn
      = dbsyncWindowTitleCallBack;
```

5 どのパブリケーションを同期するかを指定する a_syncpub 構造体のリンク・リストを作成します。

リンク・リスト内の各ノードは、dbmlsync コマンド・ライン上の -n オプションの 1 つのインスタンスに対応します。

- a_syncpub インスタンスを宣言します。たとえば、これを publication_info とします。

```
a_syncpub publication_info;
```

- a_syncpub フィールドを初期化し、同期するパブリケーションを指定します。

  たとえば、単一の同期セッションで template_p1 パブリケーションと template_p2 パブリケーションをまとめて識別するには、次のように指定します。

```
publication_info.next = NULL; // linked list
terminates
 publication_info.pub_name =
"template_p1,template_p2";
 publication_info.ext_opt  =
"sv=template_ver1";
 publication_info.alloced_by_dbsync = 0;
```

これは、dbmlsync コマンド・ラインで `-n template_p1,template_p2` と指定するのと同じです。

ext_opt フィールドを使用して指定された関連スクリプト・バージョンは、dbmlsync -eu オプションと同じ機能を提供します。

詳細については、「-eu オプション」180 ページを参照してください。

- a_sync_db インスタンスの upload_defs フィールドにパブリケーション構造体を割り当てます。

```
dbsync_info.upload_defs   = &publication_info;
```

a_syncpub 構造体のリンク・リストを作成できます。リンク・リスト内の各 a_syncpub インスタンスは、dbmlsync -n オプションの使用に対応します。

dbmlsync -n オプションの詳細については、「-n オプション」183 ページを参照してください。

a_syncpub 構造体の詳細については、『ASA プログラミング・ガイド』> 「a_syncpub 構造体」を参照してください。

6 DBSynchronizeLog 関数を使用して dbmlsync を実行します。

次に示すコード・リスト内の sync_ret_val は、『ASA プログラミング・ガイド』> 「ソフトウェア・コンポーネントのリターン・コード」にリストされている戻り値です。

```
printf("Running dbmlsync using dbtools
interface...¥n");
 sync_ret_val = DBSynchronizeLog(&dbsync_info);
 printf("¥n Done... synchronization return value is:
%I ¥n", sync_ret_val);
```

手順 6 は、同じパラメータ値または異なるパラメータ値を指定して複数回繰り返すことができます。

7 DBTools インタフェースをシャットダウンします。

DBToolsFini 関数は、DBTools リソースを開放します。

```
DBToolsFini( &info );
```

DBToolsFini 関数の詳細については、『ASA プログラミング・ガイド』> 「DBToolsFini 関数」を参照してください。

第 3 部　Mobile Link 用の Ultra Light クライアント

第 3 部では、Mobile Link 同期のために Ultra Light クライアントを設定し実行する方法について説明します。

第 11 章

Ultra Light クライアント

この章の内容

この章では、Ultra Light データベースを Mobile Link クライアントとして使用する方法について説明します。この章では、同期を最大限に活用するための Ultra Light データベースの設計について説明します。また、ここではネットワーク・プロトコルについて説明するほか、Palm OS デバイスと Windows CE デバイスを同期に対応するように設定する方法に関する資料が用意されています。

Ultra Light の詳細については、『『Ultra Light データベース・ユーザーズ・ガイド』> 「Ultra Light データベース・ユーザーズ・ガイド」』を参照してください。

Adaptive Server Anywhere データベースを Mobile Link クライアントとして使用する方法については、「Adaptive Server Anywhere クライアント」75 ページを参照してください。

概要

Ultra Light の開発者は、Ultra Light データベースのデータを中央の統合データベースと同期できます。統合データベースには、パーソナル・アプリケーション用のデスクトップ・データベースも、共有のエンタープライズ・データ用のマルチユーザ・データベースも指定できます。

Ultra Light アプリケーションへの同期の追加

通常、モバイル・データベースと組み込みデータベースには、統合データベース内のデータすべてを含むことはできません。ただし、実際にはローカルで必要なデータは、モバイル化したい特定のアプリケーションで使用されるデータです。Ultra Light には、このようなデータベースの一部を使用し、これを統合データベースと同期させる機能があります。

各 Ultra Light データベースのテーブルには、統合データベースのローとカラムのサブセットを含めることができます。たとえば、統合データベースの顧客テーブルには、カラム数が 100 以上、ロー数が 100,000 以上含まれることがあります。しかし、Ultra Light データベースでは、カラムが 4、ローが 1000 だけで十分なこともあります。Mobile Link では、各リモート・データベースにダウンロードするサブセットを正確に定義できます。

❖ Ultra Light アプリケーションに同期を追加するには、次の手順に従います。

1. 同期ストリームを準備します。

   ネットワーク・プロトコルを選択し、そのプロトコルに必要なプロトコル・オプションを設定します。

   詳細については、「ネットワーク・プロトコルの選択」361 ページを参照してください。

2. 同期関数を呼び出します。

   同期関数は、使用している開発モデルと、アプリケーションが使用しているネットワーク・プロトコルによって決まります。

   詳細については、「同期関数の呼び出し」363 ページを参照してください。

別途ライセンスを取得できるオプションが必要
トランスポート・レイヤ・セキュリティを使用するには、別途ライセンスを取得できる SQL Anywhere Studio セキュリティ・オプションを入手する必要があります。このセキュリティ・オプションは、輸出規制対象品目です。

このコンポーネントのご注文については、『SQL Anywhere Studio の紹介』> 「別途ライセンスが入手可能なコンポーネント」を参照してください。

同期させるデータの選択

デフォルトでは、同期をデータベースに追加する場合、すべてのデータが同期されます。パブリケーションやさまざまな方法によって、どのデータを同期させるかをカスタマイズできます。

別々に同期するためのデータ・セットの設計

静的インタフェースの場合、Ultra Light データベースのスキーマは、アプリケーション内に組み込まれたクエリで定義されます。リファレンス・データベースにパブリケーションを追加すると、別々に同期するデータ・セットを定義できます。パブリケーションを使用して同期対象の変更を定義しない場合は、すべての変更が同期します。

SQL Anywhere では、さまざまな用途にパブリケーションを使用します。パブリケーションはアーティクルのセットで構成されます。通常は、各アーティクルは 1 つのテーブルの全体を構成します。または、テーブル内にデータのサブセットを定義できます。

Ultra Light アプリケーション用に定義されたアーティクルでは、WHERE 句を指定してローのサブセットを使用できますが、カラムのサブセットや SUBSCRIBE BY 句は使用できません。HotSync 同期を制御する Ultra Light パブリケーションのアーティクルでは、WHERE 句は使用できません。

❖ Ultra Light データベースのデータ・サブセットを同期させるには、次の手順に従います。

1. 同期させるデータを表すパブリケーションを作成します。

   詳細については、「パブリケーションの作成」352 ページを参照してください。

2. 静的インタフェースの場合、-v コマンド・ライン・オプションでパブリケーションを指定して Ultra Light ジェネレータを実行します。

   詳細については、『Ultra Light データベース・ユーザーズ・ガイド』> 「Ultra Light ジェネレータ」を参照してください。

3 同期関数を呼び出して、パブリケーションを指定します。

パブリケーションを指定しなければ、データベースに加えた変更はすべて同期されます。パブリケーションを 1 つまたは複数指定すると、表示されたパブリケーションのうち、その数に該当する変更しか同期されません。

詳細については、次の項目を参照してください。

- **MobileVB**『Ultra Light for MobileVB ユーザーズ・ガイド』> 「ULSyncParms クラス」を参照してください。
- **ActiveX**『Ultra Light ActiveX ユーザーズ・ガイド』> 「ULSyncParms クラス」を参照してください。
- **Native Ultra Light for Java**『API リファレンス』の「**ianywhere.native_ultralite.SyncParms**」を参照してください。
- **Embedded SQL と静的型 C++ API**『Ultra Light C/C++ ユーザーズ・ガイド』> 「publication 同期パラメータ」を参照してください。
- **静的型 Java API**『Ultra Light 静的型 Java ユーザーズ・ガイド』> 「publication 同期パラメータ」を参照してください。
- **Ultra Light for M-Business Anywhere**『UltraLite for M-Business Anywhere User's Guide』> 「Class SyncParms」を参照してください。

パブリケーションの作成

コンポーネント

パブリケーションは、スキーマ・ペインタ、ULXML ユーティリティを使用するか、リファレンス・データベースから Ultra Light データベースに追加できます。

❖ **Ultra Light データベースからデータをパブリッシュするには、次の手順に従います (スキーマ・ペインタの場合)。**

1 Ultra Light データベースに接続します。

2 左ウィンドウ枠で、[同期] フォルダを開きます。

3 [パブリケーションの追加] をダブルクリックします。

4 パブリケーションを含めるテーブルのセットを指定します。

5 [OK] をクリックして変更を保存します。

リファレンス・データベース

Ultra Light 同期では、パブリケーション内の各アーティクルに、完全なテーブルか WHERE 句のいずれかを入れることができます。

❖ **Ultra Light のリファレンス・データベースからデータをパブリッシュするには、次の手順に従います (Sybase Central の場合)。**

1 DBA 権限のあるユーザとしてデータベースに接続します。

2 [パブリケーション] フォルダを開き、右ペイン上で右クリックし、[新規] - [パブリケーション] をクリックします。

3 新しく作成するパブリケーションの名前を入力します。[次へ] をクリックします。

4 [テーブル] タブで、[使用可能なテーブル] のリストからテーブルを 1 つ選択します。[追加] をクリックします。選択したテーブルが、右側の [選択したテーブル] リストに表示されます。

5 必要であれば、さらにテーブルを追加します。テーブルの順序は重要ではありません。

6 必要な場合は、[WHERE] タブをクリックし、パブリケーションに含めるローを指定します。カラムのサブセットは指定できません。HotSync 同期を使用している場合は、WHERE 句を指定しないでください。

7 [終了] をクリックします。

❖ Ultra Light のリファレンス・データベースからデータをパブリッシュするには、次の手順に従います (SQL の場合)。

1 DBA 権限のあるユーザとしてデータベースに接続します。

2 CREATE PUBLICATION 文を実行して、新しく作成するパブリケーション名とパブリッシュするテーブルを指定します。

詳細については、『ASA SQL リファレンス・マニュアル』> 「CREATE PUBLICATION 文」を参照してください。

優先度の高い変更の同期

パブリケーションは Ultra Light データベースの特定の部分を同期できます。パブリケーションをアップロード専用同期またはダウンロード専用同期と組み合わせると、優先度の高い変更を効率よく同期することが可能です。アップロード専用同期とダウンロード専用同期はどちらも双方向の同期より時間がかかりません。

詳細については、「同期させるデータの選択」351 ページと、次の項を参照してください。

- **MobileVB**『Ultra Light for MobileVB ユーザーズ・ガイド』> 「ULSyncParms クラス」を参照してください。
- **ActiveX**『Ultra Light ActiveX ユーザーズ・ガイド』> 「ULSyncParms クラス」を参照してください。
- **Native Ultra Light for Java**『API リファレンス』の「**ianywhere.native_ultralite.SyncParms**」を参照してください。
- **Embedded SQL と静的型 C++ API**『Ultra Light C/C++ ユーザーズ・ガイド』> 「upload_only 同期パラメータ」を参照してください。
- **静的型 Java API**『Ultra Light 静的型 Java ユーザーズ・ガイド』> 「upload_only 同期パラメータ」を参照してください。

- **Ultra Light for M-Business Anywhere**『UltraLite for M-Business Anywhere User's Guide』> 「Class SyncParms」を参照してください。

Ultra Light データベースへの非同期テーブルの組み込み

デフォルトでは、Ultra Light データベース内のすべてのテーブルは、統合データベースと同期されます。しかし、同期から除外されるテーブルを Ultra Light データベースに組み込むことができます。ただし、リファレンス・データベースの作成時にこれらのテーブルを明示的に識別してください。

名前の末尾が nosync で終わるテーブルは同期はしません。これらのテーブルは、統合データベースに関連しない永続的なデータ用として使用します。同期からは除外されていますが、テーブルの使用方法自体は Ultra Light データベース内の他のテーブルとまったく同じです。

またはパブリケーションを使用して同じ効果を上げることができます。詳細については、「別々に同期するためのデータ・セットの設計」351 ページを参照してください。

同期を制御するクライアント固有のデータの使用

Ultra Light のアプリケーションには、同期を制御するクライアント固有のデータを必要とするものがあります。しかしこういったデータは統合データベースでは不要です。たとえば、Ultra Light のアプリケーションを使用して複数のチャネルやトピックの中から関心のあるものを示し、この情報を使って適切なローをダウンロードする場合を考えてみます。

名前の末尾が allsync で終わるテーブルを Ultra Light データベース内に作成すると、そのテーブルのすべてのローは、最後の同期以降に変更が行われたかどうかに関わらず、それぞれの同期のたびに同期されます。

allsync テーブルには、ユーザ固有またはクライアント固有のデータを保管できます。同期するときに、統合データベースのテンポラリ・テーブルにこのテーブルのデータをアップロードすると、他のスクリプトで同期を制御するのにそのデータを使用できます。そのため、統合データベースにそのデータを保持する必要はありません。

Ultra Light データベースへの読み込み専用テーブルの組み込み

アプリケーションによっては、ローカル・レベルで更新されていないテーブルが Ultra Light データベースに組み込まれています。価格リストや会社方針などがこの例に当たります。テーブルを効率良く同期するには、これらのテーブルをパブリケーションに組み込み、ダウンロード専用同期を使用してそのパブリケーションを同期します。ダウンロード専用同期では、同期の前に Ultra Light データベースに対するすべての変更をコミットする必要はありません。

ダウンロード専用同期を使用する場合は、ローカル・レベルでデータ変更が行われていないことを確認してください。ローカル・レベルでデータが変更されていると、SQLE_DOWNLOAD_CONFLICT エラーによって同期は失敗します。

双方向同期とは異なり、ダウンロード専用同期では Ultra Light データベースに対してあらかじめすべての変更をコミットする必要はありません。コミットしなかった変更は同期に含まれず、アップロードされません。したがって、不完全なトランザクションでも問題は起こりません。

ダウンロード専用同期については、以下を参照してください。

- **MobileVB**『Ultra Light for MobileVB ユーザーズ・ガイド』> 「ULSyncParms クラス」を参照してください。
- **ActiveX**『Ultra Light ActiveX ユーザーズ・ガイド』> 「ULSyncParms クラス」を参照してください。
- **Native Ultra Light for Java**『API リファレンス』の「**ianywhere.native_ultralite.SyncParms**」を参照してください。
- **Embedded SQL と静的型 C++ API**『Ultra Light C/C++ ユーザーズ・ガイド』> 「download_only 同期パラメータ」を参照してください。

- **静的型 Java API**『Ultra Light 静的型 Java ユーザーズ・ガイド』> 「download_only 同期パラメータ」を参照してください。
- **Ultra Light for M-Business Anywhere**『UltraLite for M-Business Anywhere User's Guide』> 「Class SyncParms」を参照してください。

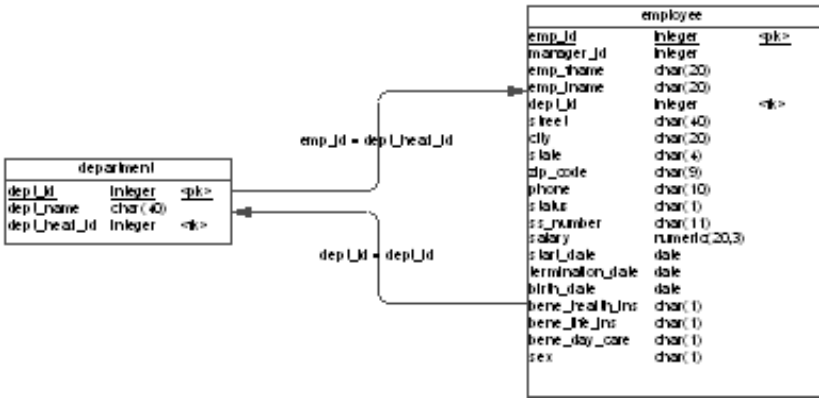
外部キー循環

この項では、Ultra Light 同期での特定の制限事項について説明します。この制限事項は、外部キーによってテーブルがリンクし合ったことによる循環が形成されたために生じます。

Ultra Light リモート・データベースからの Mobile Link 同期では、1 回のトランザクションですべての変更を統合データベースにコミットする必要があります。複数のテーブルに対するこの 1 回のトランザクションを容易にするために、各テーブルに対する挿入、更新、削除を、プライマリ・テーブルへの操作が先に実行され、関連する外部テーブルへの操作が後に実行される順序に並べてください。これによって、外部テーブルへの挿入が外部キーの参照整合性制約を満たしていることが保証されます (削除などの他の操作でも同様です)。

Ultra Light アナライザは、リファレンス・データベースのスキーマに基づいて、プライマリ・テーブルが外部テーブルよりも前にアップロードされるように、リモート・データベース内のすべてのテーブルを自動的に順番に並べます。スキーマに外部キー・サイクルが含まれていないかぎり、この順序付けは常に行われます。

次の図に、2 つのテーブル間の単純な外部キー・サイクルを示します。



Ultra Light アナライザが外部キー・サイクルを検出した場合、アナライザがエラーなく正常に完了するためには、サイクルを停止する必要があります。同期トランザクションが正常に適用されるようにするには、リファレンス・データベースと統合データベースの両方で外部キー・サイクルを停止する必要があります。

Adaptive Server Anywhere 統合データベースとリファレンス・データベースでは、いずれかの外部キーを **check on commit** (コミット時にチェック) に設定することで、外部キーの参照整合性を、操作の開始時ではなくコミット・フェーズ中にチェックするようにできます。他のデータベース・ベンダも同様の方法を用意している場合がありますが、用意されていない場合には、外部キー・サイクルをなくすようにスキーマを再設計してください。

例

```
create table c (
     id integer not null primary key,
     c_pk integer not null
 );
 create table p (
     pk integer not null primary key,
     c_id integer not null,
     foreign key p_to_c (c_id) references c(id)
 );
 alter table c
 add foreign key c_to_p (c_pk)
 references p(pk)
 check on commit;
```

Ultra Light ネットワーク・プロトコル

各 Ultra Light データベースは、ネットワーク・プロトコルによって Mobile Link 同期サーバと同期します。このネットワーク・プロトコルは、Ultra Light アプリケーションで指定されます。使用可能なネットワーク・プロトコルには、TCP/IP ベース・ネットワークの TCP/IP、HTTP と HTTPS があります。また、Palm Computing Platform での HotSync 同期と、Windows CE での ActiveSync 同期がサポートされています。

詳細については、「ネットワーク・プロトコルの選択」361 ページを参照してください。

サポートされているネットワーク・プロトコル

Ultra Light データベースは、TCP/IP、HTTP、HTTPS などの一連のネットワーク・プロトコルを使用して Mobile Link 同期サーバと同期することができます。ActiveSync 同期は、一部の開発モデル下の Windows CE アプリケーションで実行できます。HotSync は、Palm OS アプリケーションで実行できます。

各ネットワーク・プロトコルには、対応するプロトコル・オプションがあります。これらのオプションは、Mobile Link 同期サーバの場所やネットワーク固有の制御パラメータなど、プロトコルに必要な値を設定します。

Ultra Light クライアントの同期には、次のネットワーク・プロトコルがサポートされます。

コンポーネント	TCP/IP	HTTP	HTTPS	ActiveSync (Windows CE のみ)	HotSync (Palm OS のみ)
Ultra Light ActiveX	√	√	√		
Ultra Light for MobileVB	√	√	√[1]		√
Native Ultra Light for Java	√	√	√	√	

コンポーネント	TCP/IP	HTTP	HTTPS	ActiveSync (Windows CE のみ)	HotSync (Palm OS のみ)
Ultra Light.NET	√	√	√	√	
Ultra Light C++	√	√	√	√	√
Embedded SQL	√	√	√	√	√
静的型 C++ API	√	√	√	√	√
静的型 Java API	√[2]	√	√		

1. 別途注文してください。
2. セキュリティ機能のある同期を使用するには、別のプロトコルを使用してください。

別途ライセンスを取得できるオプションが必要
トランスポート・レイヤ・セキュリティを使用するには、別途ライセンスを取得できる SQL Anywhere Studio セキュリティ・オプションを入手する必要があります。このセキュリティ・オプションは、輸出規制対象品目です。

このコンポーネントのご注文については、『SQL Anywhere Studio の紹介』> 「別途ライセンスが入手可能なコンポーネント」を参照してください。

ネットワーク・プロトコルの選択

各ネットワーク・プロトコルには、動作を制御する一連のオプションがあります。ネットワーク・プロトコルを選択する場合には、これらの同期プロトコル・オプションを設定する必要があります。

ネットワーク・プロトコルとその関連同期プロトコル・オプションを選択する方法は、使用している Ultra Light 開発モデルによって異なります。

- Ultra Light for MobileVB と Ultra Light ActiveX の場合、ネットワーク・プロトコルは同期パラメータの 1 つとして、ULSyncParms オブジェクトの Stream プロパティで設定されます。プロトコル・オプションは、StreamParms プロパティでキーワードと値ペアのセットとして用意されています。

  詳細については、『Ultra Light for MobileVB ユーザーズ・ガイド』> 「ULSyncParms クラス」と 『Ultra Light ActiveX ユーザーズ・ガイド』> 「ULSyncParms クラス」を参照してください。

- Native Ultra Light for Java アプリケーションの場合、ネットワーク・プロトコルは **SyncParms** オブジェクトの **setStream** メソッドで設定されます。

  詳細については、『API リファレンス』の「**ianywhere.native_ultralite.SyncParms**」を参照してください。

- Embedded SQL と静的型 C++ API アプリケーションの場合、ネットワーク・プロトコルは、ul_synch_info 構造体のストリーム・メンバに設定されます。プロトコル・オプションは、ul_synch_info 構造体の stream_parms メンバに文字列として指定されます。次のコードは、TCP/IP 同期の一例です。

  ```
  ul_synch_info info;
   ...
   info.stream = ULSocketStream();
   info.stream_parms = UL_TEXT( "host=myserver" );
  ```

  詳細については、次の項目を参照してください。

  - 『Ultra Light C/C++ ユーザーズ・ガイド』> 「stream 同期パラメータ」
  - 『Ultra Light C/C++ ユーザーズ・ガイド』> 「stream_parms 同期パラメータ」

- 静的 Java アプリケーションの場合、プロトコル・オプションは **setStreamParms** メソッドを使用して指定します。次の例は、このメソッドを呼び出す方法を示します。

```
UlSynchOptions synch_options = new
UlSynchOptions();
 synch_opts.setStream( new UlSocketStream() );
 synch_opts.setStreamParms(
"host=myserver;port=2439" );
```

詳細については、『Ultra Light 静的型 Java ユーザーズ・ガイド』> 「stream 同期パラメータ」 と 『Ultra Light 静的型 Java ユーザーズ・ガイド』> 「stream_parms 同期パラメータ」 を参照してください。

同期関数の呼び出し

データを Mobile Link 同期サーバと同期させるために、Ultra Light アプリケーションは同期関数を呼び出します。特定の関数は、使用している開発モデルと、選択したネットワーク・プロトコルによって決まります。

次の種類のネットワーク・プロトコルを区別すると役に立ちます。

- **外部で開始された同期ストリーム** ActiveSync および HotSync 同期は、外部アプリケーションによって開始されます。

  HotSync 同期の呼び出しについては、以下を参照してください。

  - **MobileVB**『Ultra Light for MobileVB ユーザーズ・ガイド』> 「データの同期」を参照してください。
  - **Embedded SQL と静的型 C++ API**『Ultra Light C/C++ ユーザーズ・ガイド』> 「Palm アプリケーションに HotSync 同期を追加する」を参照してください。

- **直接開始されたネットワーク・プロトコル** TCP/IP、HTTP、HTTPS ネットワーク・プロトコルは、Ultra Light から直接開始されます。

  これらのプロトコルの同期関数の呼び出しについては、以下を参照してください。

- **Ultra Light for MobileVB**『Ultra Light for MobileVB ユーザーズ・ガイド』> 「Synchronize メソッド」を参照してください。
- **Ultra Light ActiveX**『Ultra Light ActiveX ユーザーズ・ガイド』> 「Synchronize メソッド」を参照してください。
- **Native Ultra Light for Java**『API リファレンス』の「**ianywhere.native_ultralite.Connection.synchronize**」を参照してください。
- **Ultra Light.NET** Ultra Light.NET の『API リファレンス』の「Synchronize」を参照してください。
- **Embedded SQL**『Ultra Light C/C++ ユーザーズ・ガイド』>「ULSynchronize 関数」を参照してください。
- **静的型 C++ API**『Ultra Light C/C++ ユーザーズ・ガイド』> 「Synchronize メソッド」を参照してください。
- **静的型 Java API**『Ultra Light 静的型 Java ユーザーズ・ガイド』> 「synchronize メソッド」を参照してください。

プライマリ・キーの一意性の管理

リファレンス・データベースにあるカラムのデフォルト値は、GLOBAL AUTOINCREMENT 型として宣言できます。このデフォルト設定は、ユニークな値を管理するカラムのすべてに適用できますが、特にプライマリ・キーの場合に役立ちます。この機能は、Mobile Link 同期などによって複数のデータベース間でデータがレプリケートされる設定で、ユニークな値を生成するタスクを容易にします。

このデフォルトのグローバル・オートインクリメントを指定すると、そのカラムの値のドメインが分割されます。各分割には同じ数の値が含まれます。たとえば、データベース内の整数カラムの分割サイズを 1000 に設定した場合、1 つの分割が 1001 から 2000 まで拡大します。また、2 つ目の分割は 2001 から 3000 まで拡大し、以降、同じように拡大していきます。

リファレンス・データベース内でグローバル・オートインクリメントとしてカラムを宣言する方法については、「デフォルトのグローバル・オートインクリメント・カラムの宣言」366 ページを参照してください。

Ultra Light データベースでグローバル・オートインクリメント・カラムを使用するには、まずデータベースの各コピーにユニークなグローバル・データベース ID 番号を割り当てます。Ultra Light は、データベースの ID 番号により一意化された各データベースにデフォルト値を設定します。たとえば、上の例のデータベースに ID 番号 1 を割り当てた場合、このデータベースのデフォルト値は 1001 ～ 2000 の範囲から選択されます。このデータベースの別のコピーで、ID 番号 2 が割り当てられたデータベースからは、2001 ～ 3000 の範囲にある同一カラムのデフォルト値が指定されます。

グローバル・データベース ID 番号を割り当てる方法については、「グローバル・データベース識別子の設定」367 ページを参照してください。

Mobile Link リモート・データベースでグローバル・オートインクリメントの値を使用する方法については、『Mobile Link 管理ガイド』>「グローバル・オートインクリメントを使用したユニークなプライマリ・キーの管理」を参照してください。

デフォルトのグローバル・オートインクリメント・カラムの宣言

Adaptive Server Anywhere リファレンス・データベースで、デフォルトのカラム値を宣言します。Ultra Light アプリケーションを構築するとき、このデフォルトのカラム値が Ultra Light データベースに継承されます。Sybase Central でカラムのプロパティを選択するか、TABLE 文または ALTER TABLE 文に DEFAULT GLOBAL AUTOINCREMENT 句を指定することで、リファレンス・データベースにデフォルト値を設定できます。

オプションで、AUTOINCREMENT キーワードの直後にカッコで分割サイズを指定できます。この分割サイズには任意の正の整数を設定できますが、通常、分割サイズは、サイズの値がすべての分割で不足しないように選択されています。

カラムの型が INT または UNSIGNED INT である場合、デフォルトの分割サイズは 2^{16} = 65536 です。他の型のカラムの場合、デフォルトの分割サイズは 2^{32} = 4294967296 です。これらのデフォルト値は適切でないことがあるため、分割サイズを明示的に指定するのが最も賢明です。

たとえば、次の文では 2 つのカラム (顧客 ID 番号を保持する整数カラム、顧客名を保持する文字列カラム) を持つ簡単なリファレンス・テーブルが作成されます。

```
CREATE TABLE customer (
     id   INT          DEFAULT GLOBAL AUTOINCREMENT
(5000),
     name VARCHAR(128) NOT NULL,
     PRIMARY KEY (id)
 )
```

上の例では、選択された分割サイズは 5000 です。

Ultra Light アプリケーションと Adaptive Server Anywhere データベースでは、一部のデータ型のデフォルト分割サイズが異なります。Ultra Light アプリケーションと同じようにリファレンス・データベースを動作させるのであれば、分割サイズを明示的に宣言します。

GLOBAL AUTOINCREMENT の詳細については、『ASA SQL リファレンス・マニュアル』> 「CREATE TABLE 文」を参照してください。

グローバル・データベース識別子の設定

アプリケーションを配備するときには、各データベースに対して必ず異なる ID 番号を割り当てる必要があります。ID 番号の作成と配布はさまざまな方法で実行できます。テーブルに値を設定し、ユーザ名など、ユニークなプロパティに基づいて、各データベースに適切なローをダウンロードするのも 1 つの方法です。

この ID 番号の設定方法は、使用するプログラミング・インタフェースによって異なります。

- **Ultra Light for MobileVB**『Ultra Light for MobileVB ユーザーズ・ガイド』> 「プロパティ」を参照してください。
- **Ultra Light ActiveX**『Ultra Light ActiveX ユーザーズ・ガイド』> 「プロパティ」を参照してください。
- **Native Ultra Light for Java**『API リファレンス』の「**ianywhere.native_ultralite.Connection.databaseID**」を参照してください。
- **Ultra Light.NET** Ultra Light.NET の『API リファレンス』の「Connection」を参照してください。
- **Embedded SQL**『Ultra Light C/C++ ユーザーズ・ガイド』> 「ULSetDatabaseID 関数」を参照してください。
- **静的型 C++ API**『Ultra Light C/C++ ユーザーズ・ガイド』> 「SetDatabaseID メソッド」を参照してください。
- **静的型 Java API**『Ultra Light 静的型 Java ユーザーズ・ガイド』> 「setDatabaseID メソッド」を参照してください。
- **Ultra Light for M-Business Anywhere**『UltraLite for M-Business Anywhere User's Guide』> 「Method setDatabaseID」を参照してください。

デフォルト値の選択方法

配備された各 Ultra Light アプリケーションのグローバル・データベース識別子には、ユニークな正の整数を設定してから、デフォルト値を割り当てます。各データベースは、これらの ID 番号によってユニークに識別されます。

詳細は、「グローバル・データベース識別子の設定」367 ページを参照してください。

特定のデータベースのデフォルト値の範囲は、$pn + 1$ ～ $p(n + 1)$ です。ここで、p は分割サイズを、n はグローバル・データベース ID 番号を示します。たとえば、分割サイズに 1000 を、グローバル・データベース ID 番号に 3 を設定すると、範囲は 3001 ～ 4000 になります。

Ultra Light アプリケーションでは、次の規則を適用してデフォルト値を選択します。

- カラムに現在の分割の値が含まれていない場合、最初のデフォルト値は $pn + 1$ である。
- カラムに現在の分割の値が含まれていても、そのすべてが $p(n + 1)$ 未満であれば、この範囲内でこれまで使用した最大値より 1 大きい値が次のデフォルト値になる。
- デフォルトのカラム値は、現在の分割以外のカラムの値の影響を受けない。つまり、$pn + 1$ より小さいか $p(n + 1)$ より大きい数には影響されない。Mobile Link 同期を介して別のデータベースからレプリケートされた場合に、このような値が存在する可能性があります。

警告

Mobile Link 同期を介してダウンロードされたカラム値は、デフォルト値のカウンタを更新しません。したがって、ある Mobile Link クライアントによって別のクライアントの分割に値が挿入された場合は、エラーが発生する可能性があります。この問題を回避するため、Ultra Light アプリケーションのコピーでは、そのコピー自体の分割にだけ値が挿入されるようにします。

グローバル・データベース ID 番号がデフォルト値の 2147483647 に設定されると、カラムには NULL 値が挿入されます。NULL 値が許可されていない場合にローを挿入しようとすると、エラーが発生します。たとえば、テーブルのプライマリ・キーにカラムが含まれている場合に、この状況が発生します。

グローバル・データベース ID 番号には負の値は設定できないので、正の値が常に選択されます。ID 番号の最大値を制限するのは、カラムのデータ型と分割サイズだけです。

デフォルトの Null 値は、分割で値が不足したときにも生成されます。この場合には、データベースに新しいグローバル・データベース ID 番号を割り当てて、別の分割からデフォルト値を選択できるようにしてください。カラムで null が許可されていない場合に NULL 値を挿入しようとすると、エラーが発生します。

特定の分割で値が不足する場合は、そのデータベースに新しいデータベース ID を割り当てることができます。方法が適切なものであれば、新しいデータベース ID 番号を割り当てることができます。未使用のデータベース ID 値のプールを管理する方法も、その 1 つです。このプールは、プライマリ・キー・プールと同じ方法で管理されます。

デフォルト値の範囲が不足するかどうかを判別する方法については、「使用できるデフォルト値の数の検出」370 ページを参照してください。

明示的なプライマリ・キー・プールを使用してプライマリ・キーの一意性を維持する方法については、『Mobile Link 管理ガイド』> 「ユニークなプライマリ・キーの管理」を参照してください。

最後に割り当てられた値の割り出し

最後の挿入操作中に選択された値は、取り出すことができます。これらの値はプライマリ・キーで頻繁に使用されるので、生成された値がわかると、最初のローのプライマリ・キーを参照するローをより簡単に挿入できます。

Embedded SQL から、次の文を使用すると、最後に割り当てられたグローバル・オートインクリメントのデフォルト値を取得できます。

```
select @@identity
```

C++ API からは、**ULConnection** オブジェクトに **GetLastIdentity()** メソッドを使用するとこの値を取得できます。

戻り値は、符号なし 64 ビット整数であるデータベース・データ型 UNSIGNED BIGINT です。この文では最後に割り当てられたデフォルト値がわかるだけなので、間違った結果を取らないために INSERT 文を実行した直後にこの値を取り出してください。

ときには、1 つの INSERT 文にグローバル・オートインクリメント型のカラムが複数含まれていることがあります。この場合、戻り値は生成されたデフォルト値のいずれか 1 つですが、そのうちのどの値であるかを判別する信頼できる方法はありません。このため、このような状況を回避するようなデータベースの設計と INSERT 文の記述を行ってください。

使用できるデフォルト値の数の検出

デフォルト値は、グローバル・データベース ID 番号によって識別される分割で、最大値に達するまでその分割から選択されます。最大値に達した場合、またはまもなくその状態になる場合は、データベースに新しい ID 番号を割り当ててください。

プログラミング・インタフェースを使うと、使用済みの番号の割合を取得できます。戻り値は、これまでに使用された値のパーセンテージを表す 0 ～ 100 の範囲の short 型です。たとえば、値 99 は、未使用の値がほとんど残っていないので、データベースに新しい ID 番号を割り当てる必要があることを示します。

この ID 番号の設定方法は、使用するプログラミング・インタフェースによって異なります。

- **Ultra Light for MobileVB**『Ultra Light for MobileVB ユーザーズ・ガイド』> 「プロパティ」を参照してください。
- **Ultra Light ActiveX**『Ultra Light ActiveX ユーザーズ・ガイド』> 「プロパティ」を参照してください。
- **Native Ultra Light for Java**『API リファレンス』の「**ianywhere.native_ultralite.Connection.databaseID**」を参照してください。

- **Ultra Light.NET** Ultra Light.NET の『API リファレンス』の「Connection」を参照してください。
- **Embedded SQL**『Ultra Light C/C++ ユーザーズ・ガイド』>「ULGlobalAutoincUsage 関数」を参照してください。
- **静的型 C++ API**『Ultra Light C/C++ ユーザーズ・ガイド』>「GlobalAutoincUsage メソッド」を参照してください。
- **静的型 Java API**『Ultra Light 静的型 Java ユーザーズ・ガイド』>「globalAutoincUsage メソッド」を参照してください。
- **Ultra Light for M-Business Anywhere**『UltraLite for M-Business Anywhere User's Guide』>「Method getGlobalAutoIncrementUsage」を参照してください。

Palm Computing Platform での Ultra Light データベースの同期

ここでは、Palm Computing Platform に固有の同期に関する詳細を説明します。

Palm Computing Platform における Ultra Light の詳細については、『Ultra Light C/C++ ユーザーズ・ガイド』> 「Palm Computing Platform 用 Ultra Light アプリケーションの開発」を参照してください。

同期方法の選択

Palm Computing Platform での同期は、HotSync を使用して行われるか、TCP/IP または HTTP を使用する標準のネットワーク・プロトコルを介して行われます。同期方法には、それぞれに利点と欠点があります。

- **複数のアプリケーション**　Palm デバイスに複数の Ultra Light アプリケーションがインストールされている場合は、HotSync を呼び出したときに、すべての Ultra Light アプリケーションが同期します。複数のアプリケーションを TCP/IP または HTTP 接続で同期する場合は、各アプリケーションを順に起動し、同期します。
- **汎用シリアル・バスのサポート**　HotSync 同期では、USB を自動サポートします。
- **パブリケーション**　HotSync を使用した同期には、WHERE 句を入れることはできません。

  詳細については、「別々に同期するためのデータ・セットの設計」351 ページを参照してください。

HotSync 同期の概要

Palm デバイス上の Ultra Light アプリケーションは、他のプラットフォーム上の Ultra Light アプリケーションと同じように、TCP/IP または HTTP プロトコルによって同期できます。また、Palm 固有の HotSync プロトコルを使って同期することもできますが、他のプロトコルとは動作方法が異なります。この項では、HotSync による同期のアーキテクチャについて説明します。

HotSync 同期では、次の順序でイベントが発生します。

1. Ultra Light アプリケーションの終了時に、終了時の状態が保存されます。この状態情報は、Ultra Light データベースとは別の Palm データベースに保存されます。

   詳細については、『Ultra Light C/C++ ユーザーズ・ガイド』>「Ultra Light Palm アプリケーションの状態の保存」を参照してください。

2. Palm デバイスの同期時に、HotSync によって Mobile Link コンジットが呼び出され、Mobile Link 同期サーバと同期します。Mobile Link コンジットが Ultra Light データベースからページを読み込み、Mobile Link 同期サーバにアップロードを送信します。

3. Mobile Link 同期サーバが、更新を統合データベースに統合し、ダウンロード・ストリームをコンジットに送信します。

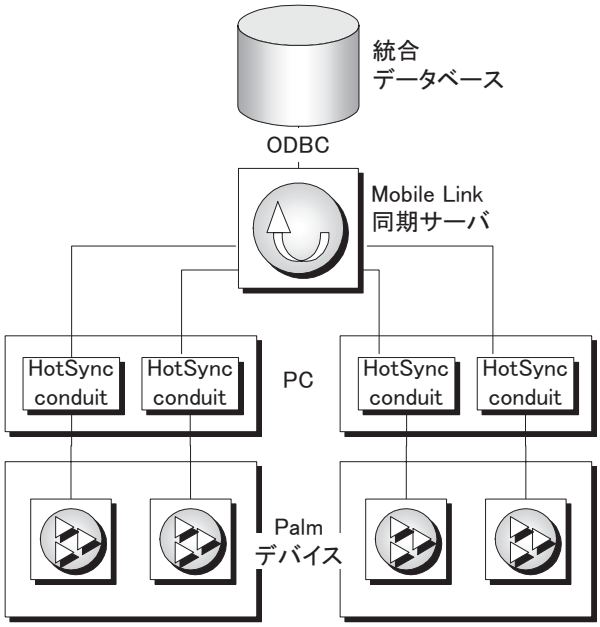
4. コンジットが、Palm デバイス上の Ultra Light データベースにダウンロード・ストリームを統合します。

5. アプリケーションの起動時に、直前に保存された Ultra Light アプリケーションの状態がロードされます。

   詳細については、『Ultra Light C/C++ ユーザーズ・ガイド』>「Ultra Light Palm アプリケーションの状態のリストア」を参照してください。

HotSync のアーキテクチャ

次の図に HotSync のアーキテクチャを示します。アプリケーションごとに、別々の HotSync コンジットが必要です。1 台の PC に複数の HotSync コンジットを配置できます。



Mobile Link HotSync コンジットの設定方法については、「Mobile Link HotSync コンジットの設定」379 ページを参照してください。

HotSync 設定の概要

HotSync 同期では、HotSync Manager が Mobile Link HotSync コンジット (*dbhsync9.dll*) を起動します。HotSync コンジットは、デバイスから読み込みを行ってからアップロード・ストリームを Mobile Link 同期サーバに送信します。次に、HotSync コンジットはダウンロード・ストリームを Mobile Link 同期サーバから受信し、ダウンロード内容をデバイスに書き込みます。

Mobile Link HotSync コンジットは、TCP/IP、HTTP、HTTPS プロトコルのいずれかを使って Mobile Link 同期サーバと同期します。

ほとんどの場合、Mobile Link HotSync コンジットだけがユーザのデスクトップ・マシンに配備されます。

HotSync のアーキテクチャについては、「HotSync 同期の概要」373 ページを参照してください。

❖ Mobile Link HotSync コンジットをインストールおよび設定するには、次の手順に従います。

1. Mobile Link コンジットのファイルをユーザのマシンに配置します。

   作業手順については、「コンジットのファイル」375 ページを参照してください。

2. Mobile Link コンジットを HotSync Manager に登録します。これにより、HotSync Manager は Mobile Link コンジットを使用できるようになります。

   作業手順については、「HotSync Manager への Mobile Link HotSync コンジットの登録」376 ページを参照してください。

3. Ultra Light の **ul_synch_info** 構造体に **stream_parms** パラメータを指定しなかった場合は、HotSync Manager からこれらのパラメータを入力します。

   作業手順については、「Mobile Link HotSync コンジットの設定」379 ページを参照してください。

   Ultra Light 同期の呼び出しで **stream_parms** パラメータを指定する方法については、『Ultra Light C/C++ ユーザーズ・ガイド』> 「Palm アプリケーションに HotSync 同期を追加する」を参照してください。

4. 暗号化されたデータベースを使用している場合は、[conduit 設定] ダイアログに暗号化キーを入力します。このキーを入力しない場合は、同期のたびにキーを入力する必要があります。

   作業手順については、「Mobile Link HotSync コンジットの設定」379 ページを参照してください。

コンジットのファイル

HotSync コンジットは以下のファイルで構成されます。

- **dbhsync9.dll**　HotSync Manager が呼び出す DLL。

- **dblgen9.dll**　言語リソース・ライブラリ。英語以外の言語では、ファイル名の en という文字が、言語を示す 2 文字の省略形に置換され、*dblgde9.dll* や *dblgja9.dll* になります。

- **ストリーム用の DLL**　コンジットと Mobile Link 同期サーバとの通信には DLL が必要です。各ネットワーク・プロトコルごとに個別の DLL が提供されています。

  - TCP/IP の場合は、*dbmlsock9.dll* を使用します。
  - HTTP の場合は、*dbmlsock9.dll* と *dbmlhttp9.dll* を使用します。
  - HTTPS の場合は、*dbmlhttps9.dll* を使用します。
  - この通信で暗号化を使用する場合は、暗号化 DLL である *dbmltls9.dll* も追加する必要があります。

これらのファイルはシステム・パスの同じディレクトリになければなりません。SQL Anywhere Studio をインストールしている場合、これらのファイルはインストール・ディレクトリのオペレーティング・システム用のサブディレクトリにインストールされています。このサブディレクトリは、すでにシステム・パスに含まれています。ただし、これらのファイルを使用するために SQL Anywhere Studio をインストールする必要はありません。

HotSync Manager への Mobile Link HotSync コンジットの登録

Ultra Light には *dbcond9* というコマンド・ラインの「**コンジット・インストール・ユーティリティ**」があります。このユーティリティは、HotSync Manager が Mobile Link コンジットを使用できるようにするレジストリ・エントリ・セットを作成します。このユーティリティには以下のファイルが必要です。

- *dbcond9.exe*
- *condmgr.dll*

❖ コンジット・インストール・ユーティリティを配備するには、次の手順に従います。

1. 配備ディレクトリの最上位レベルを選択します。

たとえば、*c:¥deploy* という名前のディレクトリを選択します。

2　配備ディレクトリのフル・パスを値とするレジストリ・エントリを追加します。

レジストリ・エントリは、次のように指定します。

```
HKEY_CURRENT_USER¥Software¥Sybase¥Adaptive Server
Anywhere¥version-string¥Location
```

ここで *version-string* は、SQL Anywhere Studio のバージョン (**9.0** など) を示す数値です。このエントリが *HKEY_CURRENT_USER* で見つからない場合、ソフトウェアは *HKEY_LOCAL_MACHINE* を調べます。

3　*dbcond9.exe* ファイルは、配備ディレクトリの *win32* サブディレクトリに追加してください。

4　*condmgr.dll* ファイルを追加します。

condmgr.dll ファイルは、配備ディレクトリの *win32¥condmgr* サブディレクトリに追加してください。

SQL Anywhere Studio をインストールすると、必要なレジストリ・エントリが作成され、適切なロケーションへファイルが配置されます。

❖ Mobile Link HotSync コンジットを HotSync Manager に登録するには、次の手順に従います。

1　HotSync コンジットのファイルとコンジット・インストール・ユーティリティのファイルがあることを確認します。

2　HotSync コンジット・インストール・ユーティリティを起動します。コマンド・ラインで、Palm アプリケーションの作成者 ID と、HotSync がコンジットを識別するための名前を指定します。

たとえば次のコマンドは、作成者 ID が **Syb2** であるアプリケーションの **CustDB** というコンジットをインストールします。次の例は CustDB サンプル・アプリケーションの設定です。

```
dbcond9 "Syb2" -n CustDB
```

コンジット・インストール・ユーティリティの詳細については、『Ultra Light データベース・ユーザーズ・ガイド』>「HotSync コンジット・インストーラ」を参照してください。

注意

HotSync 同期のセカンダリ・ロケーションは、使用中の Palm Computing Platform のバージョンによって異なります。このセカンダリ・ロケーションは、*HKEY_CURRENT_USER¥Software¥U.S. Robotics* または *HKEY_CURRENT_USER¥Software¥Palm Computing* の下に存在することがあります。

Mobile Link HotSync コンジットが正しくインストールされたことをチェックする

次に示すのは、コンジットのインストールと動作を確認するための手順です。

❖ HotSync コンジットが正しくインストールされたことをチェックするには、次の方法に従います。

1 次のようにして、コンジットがインストールされていることを確認します。

- PC のシステム・トレイで、[HotSync Manager] を右クリックします。
- ポップアップ・メニューで [動作設定] を選択します。

  ユーザ別のコンジット・リストが表示されます。自分のコンジットがリストに表示されていることを確認します。

2 システム環境変数 UL_DEBUG_CONDUIT に、任意の値を設定します。

3 HotSync Manager を停止し、再起動します。

4 Mobile Link コンジットが正しくインストールされていれば、2 つのダイアログ・ボックスが表示されます。ダイアログが表示されない場合は、コンジットが正しくインストールされていません。

5 環境変数の設定を解除します。

6 HotSync Manager を停止し、再起動します。

Mobile Link を起動してから HotSync を使用してください。
HotSync を使用するには、Mobile Link 同期サーバを起動し、Mobile Link HotSync コンジットからの接続を受け取れる状態にしてください。Mobile Link 同期サーバは同じコンピュータに存在しなくてもかまいませんが、ネットワークを通じてアクセスできる必要があります。

Mobile Link HotSync コンジットの設定

Mobile Link HotSync コンジットは、Ultra Light アプリケーションと統合データベースを同期するために Mobile Link 同期サーバと通信する必要があります。コンジットが Mobile Link 同期サーバを検知するために必要な情報は、ULSetSynchInfo 関数に渡す Ultra Light **ul_synch_info** 構造体の **stream_parms** メンバに指定できます。**stream_parms** の値を指定しないか、null 値を指定した場合は、必要なパラメータを HotSync Manager から入力できます。

また、高度に暗号化された Ultra Light データベースを使用している場合は、同期のたびに入力する必要がないように暗号化キーを保存できます。

Palm Desktop ソフトウェアをインストールしている場合は、Adaptive Server Anywhere をインストールすると、**CustDB** サンプル・アプリケーションのレジストリ・エントリが作成されます。これらのエントリをもとに独自のアプリケーションを作成できます。

stream_parms の詳細については、『Ultra Light C/C++ ユーザーズ・ガイド』> 「Palm アプリケーションに HotSync 同期を追加する」を参照してください。

❖ Mobile Link HotSync コンジットを同期に対応するように設定するには、次の手順に従います。

1. システム・トレイ内の HotSync Manager アイコンを右クリックし、ポップアップ・メニューから [動作設定] を選択します。
2. コンジット名リストから Mobile Link HotSync コンジットを選択し、[変更] をクリックします。
3. [同期パラメータ] テキスト・ボックスにネットワーク・プロトコル・オプションのセットを入力します。これらのオプションは、"stream" エントリがネットワーク・プロトコルのタイプ (TCP/IP、HTTP、HTTPS) の指定に使用されていることを除き、**stream_parms** パラメータのオプションと同じです。次に例を示します。

   ```
   stream=tcpip;host=localhost
   ```

   詳細については、「HotSync プロトコル・オプション」435 ページを参照してください。
4. データベースが高度に暗号化されている場合は、[暗号化キー] テキスト・ボックスに暗号化キーを入力できます。キーを入力しない場合は、同期のたびに暗号化キーの入力を求めるプロンプトが表示されます。
5. [OK] をクリックします。エントリが完成します。HotSync コンジットが使用可能になりました。

レジストリのロケーション

プロトコル・オプションと暗号化キーは、*HKEY_CURRENT_USER¥Software¥Sybase¥Adaptive Server Anywhere¥9.0¥Conduit¥Creator-ID* 内のレジストリに格納されます。*Creator-ID* はアプリケーションによって異なります。

HotSync 同期のセカンダリ・ロケーションは、使用中の Palm Computing Platform のバージョンによって異なります。レジストリは、*HKEY_CURRENT_USER¥Software¥U.S. Robotics* フォルダまたは *HKEY_CURRENT_USER¥Software¥Palm Computing* フォルダに作成されます。

HotSync ログ・ファイル

HotSync は、各同期がいつ実行され、インストールされている各コンジットが正しく動作したかどうかを記録します。HotSync ログ・ファイルは、Pilot または Palm のディレクトリの *User¥HotSync.log* です。

HotSync ログ・ファイル内の詳細なデバッグ情報を取得するには、UL_DEBUG_CONDUIT_LOG 環境変数を設定します。この変数は、HotSync コンジットに問題がある場合に開発時に使用すると便利です。デフォルトでは、この環境変数は設定されていません。

UL_DEBUG_CONDUIT_LOG = 1　この環境変数を 1 に設定すると、同期パラメータ、レジストリ・ロケーション、ライブラリのロードの試行回数などの基本的な情報が HotSync ログ・ファイルに書き込まれます。

UL_DEBUG_CONDUIT_LOG = 2　2 に設定すると、詳細な情報が HotSync ログ・ファイルに書き込まれます。

新しい設定を有効にするには、HotSync を再起動する必要があります。

環境変数を設定する方法については『ASA データベース管理ガイド』> 「環境変数の設定」を参照してください。

Mobile Link HotSync コンジットの配備

HotSync 同期を使用するアプリケーションについては、各エンド・ユーザがデスクトップ・マシンに Mobile Link HotSync コンジットをインストールしてください。このインストールでは、次の作業を行う必要があります。

- **コンジットのファイルの配備**　コンジット用のファイルをエンド・ユーザのシステム・パスのロケーションにインストールします。

  コンジットのファイルの一覧は、「コンジットのファイル」375 ページを参照してください。

- **コンジットのインストール**　コンジット・インストール・ユーティリティをエンド・ユーザに配備してそのユーティリティを実行する指示を与えられます。または、HotSync Manager を使用してコンジットをインストールできます。

  作業手順については、「HotSync Manager への Mobile Link HotSync コンジットの登録」376 ページを参照してください。

- **コンジットの設定**　Ultra Light の **ul_synch_info** 構造体に **stream_parms** パラメータを指定しなかった場合は、HotSync Manager からこれらのパラメータを入力します。また、暗号化されたデータベースを使用している場合は、暗号化キーも入力できます。

  作業手順については、「Mobile Link HotSync コンジットの設定」379 ページを参照してください。

TCP/IP、HTTP、HTTPS 同期の設定

この項では、TCP/IP または HTTP 同期を使用する Ultra Light Palm アプリケーションに対して同期を設定する方法について説明します。

HTTP 通信または TCP/IP 通信の同期アーキテクチャについては、『Mobile Link 管理ガイド』> 「同期システムの要素」を参照してください。

Palm Computing Platform の TCP/IP による同期を設定する

Palm デバイスで TCP/IP のネットワークを使う方法は 2 つあります。いずれの方法でも、リモート・アクセス・サービス (RAS) に接続する必要があります。2 つの方法の相違点は、RAS への接続方法です。

- **モデムを使って ISP にダイヤル接続する**　インターネット・サービス・プロバイダ (ISP) がリモート・アクセス・サービス (RAS) へのアクセスを提供していることが必要です。接続のコンポーネントは次のとおりです。

```
Application
 <--> Palm Net Library
 <--> Palm modem
 <--> NT RAS
 <--> TCP/IP network
```

- **シリアル・ポートを使って Windows NT マシンに接続する**　接続のコンポーネントは次のとおりです。

```
Application
 <--> Palm Net Library
 <--> serial cable
 <--> NT RAS
 <--> TCP/IP network
```

TCP/IP を使用する場合、Mobile Link 同期サーバは、TCP/IP でアクセス可能なネットワーク上にあれば、どのマシンでもかまいません。

同期を実行する前に以下の条件を確認します。

1. デバイスがクレードル内にあることを確認します。
2. RAS を実行する Windows NT マシンにシリアル・ポートを使って接続する場合、HotSync Manager とシリアル・ポートを使う他のアプリケーションが停止していることを確認します。Windows NT では、一度に 1 つのアプリケーションしかシリアル・ポートを使用できません。
3. Mobile Link 同期サーバが起動していることを確認します。デフォルトでは、Mobile Link 同期サーバは、ポート 2439 で TCP/IP 通信を受信します。
4. Palm デバイスではネットワークの設定が完了しており、ネットワークに接続できる状態にあることを確認します。モデムを使って ISP にダイヤルアップ接続する場合は、モデムの設定も必要です。

Palm Computing Platform の HTTP または HTTPS による同期を設定する

HTTP または HTTPS 同期を使用するには、先に RAS TCP/IP 同期を設定します。RAS の設定については、「Palm Computing Platform の TCP/IP による同期を設定する」382 ページを参照してください。

HTTP または HTTPS を使用する場合、Mobile Link 同期サーバは、プロトコルでアクセス可能なネットワーク上にあれば、どのマシンでもかまいません。

❖ **HTTP または HTTPS を使用して同期するには、次の手順に従います。**

1 Palm デバイスをクレードルに入れます。

2 RAS を実行するデスクトップ・マシンにシリアル・ポートを使って接続する場合、HotSync Manager とシリアル・ポートを使用する他のアプリケーションを停止してください。
Windows では、一度に 1 つのアプリケーションしかシリアル・ポートを使用できません。

3 Mobile Link 同期サーバを起動します。

4 Palm デバイスでネットワークの設定が行われ、ネットワークに接続できる状態であることを確認します。モデムを使って ISP にダイヤルアップ接続する場合は、モデムの設定も必要です。

詳細については、「Palm Computing Platform の TCP/IP による同期を設定する」382 ページを参照してください。

リモート・アクセス・サービスの設定

TCP/IP、HTTP、または HTTPS で Palm アプリケーションを同期するには、リモート・アクセス・サービスがデスクトップ・マシンにインストールされている必要があります。リモート・アクセス・サービス・ソフトウェアは、Ultra Light または Mobile Link に含まれていません。リモート・アクセス・サービス・ソフトウェアの設定は簡単ではないため、この項では設定作業に役立つ手順をいくつか説明します。

同期にモデムを使用する場合の RAS の設定

この方法を使う場合は、ダイヤルインするときにリモート・アクセス・サービスにアクセスできることが必要です。

❖ Palm デバイスでモデムによる RAS TCP/IP を設定するには、次の手順に従います。

1 Palm デバイスをモデム・モジュールに接続してモデムを取り付けます。

2 [環境設定] パネルを開き、画面の右上にあるドロップダウン・リストから [ネットワーク] を選択します。

3 Windows RAS サービスを選択します。

4 ダイヤルイン・ユーザ名とパスワードを設定します。

5 電話番号をリモート・アクセス・サービスにアクセスできる番号に設定します。この番号は加入している ISP に問い合わせてください。

6 [詳細] をクリックします。

7 接続タイプを設定します (通常は PPP)。

8 ネットワーク管理者の指示に従って DNS と IP アドレスを設定します。

9 [スクリプト] を押し、ISP から指示されたスクリプトを入力します。このスクリプトの例を次に示します。

```
Wait For: Username:
 Delay: 1
 Send UserID:
 Send CR:
 Wait For: Password:
 Delay: 1
 Send Password:
 Send CR:
 Wait For: >
 Delay: 1
 Send: ppp
 Send CR:
 End:
```

[OK] を何度か押して [環境設定] の [ネットワーク] に戻ります。

これで、TCP/IP 接続をテストする準備ができました。

シリアル・ポート接続の場合の RAS の設定

この手順には、Windows NT と Palm Computing デバイスの両方での操作が含まれます。

❖ Windows NT でシリアル・ポートによる RAS TCP/IP を設定するには、次の手順に従います。

1 [コントロール パネル] から [モデム] を開きます。クレードルが接続されている COM ポートの [**2 台の PC 間のダイヤルアップ・ネットワーキング・シリアル・ケーブル**] にモデムが定義されていることを確認します。

2 モデムの速度を使用しているボー・レートに設定します。デフォルトは 19200 です。

3 TCP/IP プロトコルがインストールされていることを確認します。[スタート] － [設定] － [コントロール パネル] を選択し、[ネットワーク] アイコンをダブルクリックします。[プロトコル] タブをクリックします。TCP/IP エントリがない場合は、[追加] を選択してインストールしてください。

4 [IP 転送を行う] を選択します (TCP/IP プロパティの [ルーティング] タブ)。

5 [サービス] タブでリモート・アクセス・サービスがインストールされていることを確認します。リモート・アクセス・サービスのエントリがない場合は、[追加] を選択してインストールしてください。

   リモート・アクセス・サービスのプロパティで、ポート・リストにクレードルの COM ポートがない場合は、そのポートの [**2 台の PC 間のダイヤルアップ・ネットワーキング・シリアル・ケーブル**] を追加します。

6 このエントリを呼び出しの受信ができるように設定します。RAS ネットワークのプロパティで暗号化設定を [**クリア テキストを含む任意の認証を許可する**] に設定します。[RAS ネットワークのプロパティ] で TCP/IP クライアントのみを許可します。

7 TCP/IP を設定します。クライアントがネットワーク全体にアクセスできるようにします。TCP/IP アドレスの割り当てはネットワークによって異なります。詳細はネットワーク管理者に問い合わせてください。

8 ダイヤルイン・アクセスのユーザを追加します。[スタート] － [プログラム] － [管理ツール] － [ユーザー マネージャ] を選択します。[**ユーザーは次回ログオン時にパスワード変更が必要**] をオフにします。[ダイヤルイン] ボタンを選択し、[コールバックしない] を選択してユーザにダイヤルイン・パーミッションを与えます。

9 RAS COM ポートと HotSync Manager が使用するポートが同じである場合は、HotSync Manager またはその COM ポートを使用する他のアプリケーションを停止してください。

10 リモート アクセス管理を起動します。[スタート] － [プログラム] － [管理ツール] － [リモート アクセス管理] を選択します。

11 RAS サービスを起動します。[サーバー] － [リモート アクセス サービスの開始] を選択します。このサービスはローカル・マシンで開始してください。

HotSync Manager またはこのシリアル・ポートを使用する他のアプリケーションと RAS サービスは同時に動作できません。一方を実行するためには他方を停止してください。Windows NT では、2 つのアプリケーションが同じシリアル・ポートにアクセスすることはできないからです。HotSync Manager を再起動するためには、RAS サービスを停止 (リモート アクセス管理で [サーバー] － [リモート アクセス サービスの停止] を選択) する必要があります。または、別々のシリアル・ポートを使用することもできます。

RAS サービスが動作していれば、シリアル・ポートで接続要求を受信できます。

❖ Palm デバイスでシリアル・ポートによる RAS TCP/IP を設定するには、次の手順に従います。

1 [環境設定] パネルを開き、画面の右上にあるドロップダウン・リストから [ネットワーク] を選択します。

2 Windows RAS サービスを選択します。

3 ダイヤルイン・ユーザ名とパスワードを設定します。

4 シリアル・ポートを使用するように Palm を設定します。

- Palm OS 3.3 以上では、[ダイレクト シリアル] を選択します。
- それ以前の Palm OS バージョンでは、電話番号を **00** (ゼロが 2 つ) に設定します。これは Palm にモデムではなくシリアル・ポートを使用するように指示する特殊な電話番号です。

5 [詳細] をクリックします。

6 接続タイプを設定します (通常は PPP)。

7 ネットワーク管理者の指示に従って DNS と IP アドレスを設定します。

8 [スクリプト] を押し、次のスクリプトを入力します。

```
Send: CLIENT
 Send CR:
 Delay: 1
 Send: CLIENT
 END
```

[OK] を何度か押して [環境設定] の [ネットワーク] に戻ります。

これで、TCP/IP 接続をテストする準備ができました。

テストとトラブルシューティング

❖ 接続をテストするには、次の手順に従います。

1. **モデムを使用**　Palm デバイスをモデムに接続し、ISP から指示された手順で ISP のネットワークに接続します。接続できたら、Palm デバイスの [環境設定] － [ネットワーク] で [接続] ボタンを押します。

2. **シリアル・ポートを使用**　Windows NT マシンで RAS が動作していることを確認します。Palm デバイスをクレードルに入れ、クレードルを Windows NT マシンの正しい COM ポートに接続します。Palm デバイスの [環境設定] － [ネットワーク] で [接続] ボタンを押します。

TCP/IP では、2 つのレベルのサービスがあります。最下位レベルでは、次の形式の IP 番号を使って別の TCP/IP ホストに接続できます。

```
NNN.NNN.NNN.NNN
```

次のレベルでは、DNS サーバが正しく設定されている場合、ホスト名を使用して別のホストに接続できます。

```
some_host_machine.any_company.com
```

通常、番号よりは名前の方が覚えやすいので、DNS サービスを使用した方が便利です。最下位レベルの TCP/IP サービスと IP 番号があれば、TCP/IP を使って Ultra Light アプリケーションを同期できます。

Palm デバイスで TCP/IP 接続のトラブルシューティングとして実行できる作業がいくつかあります。

- 接続フェーズ中に Palm デバイスのスクロール・ダウン・ボタンを押すと、接続の進行状況が表示されます。
- [環境設定] の [ネットワーク] パネルから接続ログにアクセスできます。[オプション] － [ログの表示] を選択します。ネットワーク接続に関する情報が表示されます。このログは対話型のユーティリティで、接続情報の制御と表示に使用できます。ヘルプを表示するには、[?] を入力します。

- Palm から TCP/IP 接続をテストするためのツールがいくつか用意されています。そのほとんどは次の Web サイトで入手できます。

  http://www.roadcoders.com
  http://www.palmcentral.com

Windows NT で実行できるトラブルシューティング作業もあります。

- [リモート アクセス管理] で実行中のサーバをダブルクリックします。
- 該当するポートを選択し、[ポートの状態] を選択します。[ポートの状態] ダイアログに回線の状態 (接続完了または呼び出し待ち状態) が表示され、送受信されているバイト数を確認できます。

Windows CE での Ultra Light データベースの同期

Windows CE 上の Ultra Light アプリケーションは、標準のネットワーク・プロトコル (TCP/IP、HTTP、または HTTPS) を使用して同期できます。また、Embedded SQL、静的 C++ API、または Native Ultra Light for Java を使用して構築された Ultra Light アプリケーションも ActiveSync を使用できます。

ここでは、ActiveSync 同期について説明します。

ActiveSync 用 Mobile Link プロバイダのインストール

ActiveSync で使用するアプリケーションを登録するには、まずインストール・ユーティリティ *dbasinst.exe* を使用して、ActiveSync 用 Mobile Link プロバイダをインストールします。

ActiveSync 用 Mobile Link プロバイダには、デスクトップ・コンポーネントとデバイス・コンポーネントが含まれています。デスクトップ・マシンを通じて同期を行う各デバイスに対して、このプロバイダを設定してください。

ActiveSync 用 Mobile Link プロバイダをインストールしたら、各アプリケーションを個別に登録してください。作業手順については、「ActiveSync で使用するアプリケーションの登録」393 ページを参照してください。

❖ ActiveSync 用 Mobile Link プロバイダをインストールするには、次の手順に従います。

1. 使用中のマシンに ActiveSync ソフトウェアがインストールしてあり、Windows CE デバイスが接続されていることを確認します。
2. 次のコマンドを入力して、Mobile Link プロバイダをインストールします。

```
dbasinst -k desk-path -v dev-path
```

desk-path はプロバイダのデスクトップ・コンポーネント (*dbasdesk.dll*) のロケーション、*dev-path* はデバイス・コンポーネント (*dbasdev.dll*) のロケーションです。

使用中のマシンに SQL Anywhere がインストールしてある場合は、*dbasdesk.dll* は SQL Anywhere ディレクトリの *win32* サブディレクトリにあります。また、*dbasdev.dll* は *CE* サブディレクトリのプラットフォーム固有のディレクトリにあります。これらのディレクトリは、デフォルトの検索ロケーションです。-k オプションと -v オプションはどちらも省略できます。

詳細については、「ActiveSync プロバイダ・インストール・ユーティリティ」34 ページを参照してください。

3 マシンを再起動します。

マシンを再起動すると、ActiveSync で新しいプロバイダが認識されます。

4 Mobile Link プロバイダを有効にします。

- [ActiveSync] ウィンドウで [オプション] をクリックします。
- リストにある [Mobile Link] 項目を有効にして [OK] をクリックし、Mobile Link プロバイダをアクティブにします。
- 登録されたアプリケーションのリストを表示するには、もう一度 [オプション] をクリックし、Mobile Link プロバイダを選択して [設定] をクリックします。

  アプリケーションの登録についての詳細は、「ActiveSync で使用するアプリケーションの登録」393 ページを参照してください。

ActiveSync で使用するアプリケーションの登録

ActiveSync で使用するアプリケーションを登録するには、ActiveSync プロバイダのインストール・ユーティリティを使用する方法と、ActiveSync ソフトウェア自体を使用する方法があります。この項では、ActiveSync ソフトウェアを使用する方法について説明します。

もう 1 つの方法については、「ActiveSync プロバイダ・インストール・ユーティリティ」34 ページを参照してください。

❖ ActiveSync で使用するアプリケーションを登録するには、次の手順に従います。

1. ActiveSync 用 Mobile Link プロバイダがインストールされていることを確認します。

   詳細は、「ActiveSync 用 Mobile Link プロバイダのインストール」391 ページを参照してください。

2. デスクトップ・マシンで、ActiveSync ソフトウェアを起動します。

3. [ActiveSync] ウィンドウで [オプション] を選択します。

4. 情報タイプのリストから、[Mobile Link] を選択し、[設定] をクリックします。

5. [Mobile Link 同期] ダイアログで [新規] をクリックします。[プロパティ] ダイアログが表示されます。

6. アプリケーションについて次の情報を入力します。

   - **[アプリケーション名]**　ActiveSync ユーザ・インタフェースに表示されるアプリケーションを識別する名前。
   - **[クラス名]**　アプリケーションの登録済みクラス名。

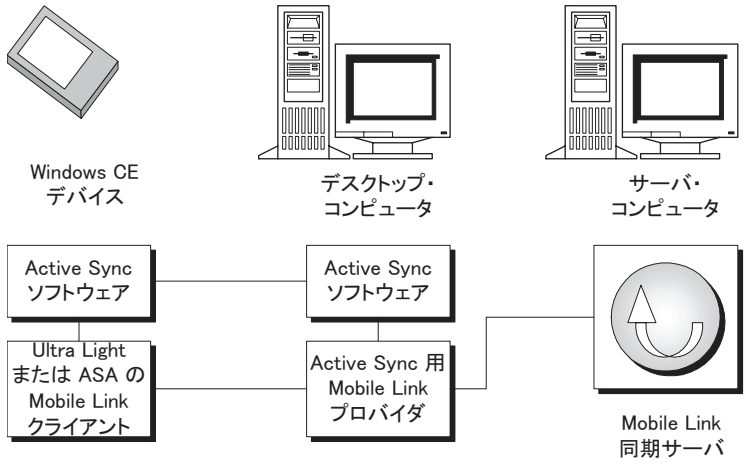
     『Ultra Light C/C++ ユーザーズ・ガイド』> 「アプリケーションに対するクラス名の割り当て」を参照してください。

- **[パス]** アプリケーションのデバイス上のロケーション。
- **[引数]** ActiveSync でアプリケーションの起動時に使用するコマンド・ライン引数。

7 [OK] をクリックしてアプリケーションを登録します。

ActiveSync を使用するアプリケーションの配備

ActiveSync 同期を使用するアプリケーションは、デバイスにコピーするだけでなく、ActiveSync に登録してください。また、ActiveSync 用 Mobile Link プロバイダをそれぞれのデスクトップ・マシンにインストールしてください。次の図は、ActiveSync のアーキテクチャを示しています。



❖ **ActiveSync アプリケーションを配備するには、次の手順に従います。**

1 ActiveSync 用 Mobile Link プロバイダをエンド・ユーザの各コンピュータにインストールします。

SQL Anywhere では、ActiveSync プロバイダ・インストール・ユーティリティが提供されています。これは、*dbasinst.exe* コマンド・ライン・ユーティリティです。

dbasinst.exe コマンド・ライン・ユーティリティについては、「ActiveSync 用 Mobile Link プロバイダのインストール」391 ページと 「ActiveSync プロバイダ・インストール・ユーティリティ」34 ページを参照してください。

2 ActiveSync で使用するアプリケーションを登録します。

これを登録するには、ActiveSync を使用するか、ActiveSync プロバイダ・インストール・ユーティリティ *dbasinst.exe* を使用します。

アプリケーションの登録については、「ActiveSync で使用するアプリケーションの登録」393 ページを参照してください。

3 アプリケーションをデバイスにコピーします。

アプリケーションが単一の実行プログラムであり、ランタイム・ライブラリに静的にリンクしている場合には、ActiveSync プロバイダ・インストール・ユーティリティ *dbasinst.exe* を使用します。

アプリケーションに複数のファイルが含まれている場合 (たとえば、静的ランタイム・ライブラリを使用するのではなく、Ultra Light ランタイム DLL を使用した場合) には、別の方法でそれらのファイルをデバイスにコピーしてください。

第 12 章

Ultra Light 同期パラメータ

この章の内容

この章では、Mobile Link を使用して Ultra Light アプリケーションを同期するために設定できるすべてのパラメータについて詳しく説明します。

この章で説明するパラメータが適用されるのは、Ultra Light リモート・データベースだけです。Adaptive Server Anywhere リモート・データベースを同期させる場合は、「Adaptive Server Anywhere クライアントの同期パラメータ」123 ページを参照してください。

同期パラメータ

同期パラメータは、Ultra Light データベースと Mobile Link 同期サーバ間の同期を制御します。パラメータの設定方法は、使用している Ultra Light インタフェースによって異なります。この項では、パラメータの影響について説明し、これらの設定方法が説明された他の項へのリンクを示します。

Stream Type および Version パラメータが必要です。これらが設定されていない場合、同期関数は例外 (SQLCode.SQLE_SYNC_INFO_INVALID またはこれと同じもの) をスローします。

一部の同期パラメータは競合することがあります。Download Only、Ping、Resume Partial Download、または Upload Only は一度に 1 つのみ指定できます。これらのパラメータが複数 true に設定されると、同期関数は例外 (SQLCode.SQLE_SYNC_INFO_INVALID またはこれと同じもの) をスローします。

Authentication Parameters 同期パラメータ

機能

authentication_parameters スクリプトのパラメータを指定します。

使用法

このパラメータを使用して、authentication_parameters スクリプトに必要な値を指定します。このような値には、ユーザ名やパスワードなどがあります。

このパラメータを使用する場合、パラメータの数も指定してください。「Number of Authentication Parameters パラメータ」407 ページを参照してください。

指定可能な値

文字列の配列。文字列の値として Null は使用できませんが、空の文字列は指定できます。

参照

「Number of Authentication Parameters パラメータ」407 ページ

『Mobile Link 管理ガイド』> 「authenticate_parameters 接続イベント」

インタフェース

- **MobileVB**『Ultra Light for MobileVB ユーザーズ・ガイド』> 「ULSyncParms クラス」
- **ActiveX**『Ultra Light ActiveX ユーザーズ・ガイド』> 「ULSyncParms クラス」
- **Ultra Light.NET**『Ultra Light.NET ユーザーズ・ガイド』> 「AuthenticationParms プロパティ」(iAnywhere.Data.UltraLite ネームスペース) と 『Ultra Light.NET ユーザーズ・ガイド』> 「AuthenticationParms プロパティ」(iAnywhere.UltraLite ネームスペース) を参照してください。
- **Native Ultra Light for Java**『API リファレンス』の「**ianywhere.native_ultralite.SyncParms**」を参照してください。
- **C++ コンポーネント、Embedded SQL、および静的型 C++ API**『Ultra Light C/C++ ユーザーズ・ガイド』> 「auth_parms パラメータ」を参照してください。
- **静的型 Java API**『Ultra Light 静的型 Java ユーザーズ・ガイド』> 「auth_parms パラメータ」を参照してください。

Authentication Status 同期パラメータ

機能

Mobile Link のユーザ認証のステータスをレポートします。Mobile Link 同期サーバが、この情報をクライアントに提供します。

使用法

カスタム認証スキームを実装している場合、authenticate_user または authenticate_user_hashed 同期スクリプトは、このパラメータの有効値の 1 つを返します。

このパラメータは Mobile Link 同期サーバによって設定されるため、読み込み専用です。

指定可能な値

使用できる値は、インタフェース固有の列挙に格納されています。

統合データベースでカスタム **authenticate_user** 同期スクリプトが異なる値を返す場合は、『Mobile Link 管理ガイド』> 「authenticate_user 接続イベント」で指定されているルールに従って値が解釈されます。

参照

「Mobile Link ユーザの認証」9 ページ

インタフェース

- **MobileVB**『Ultra Light for MobileVB ユーザーズ・ガイド』> 「ULSyncResult クラス」
- **ActiveX**『Ultra Light ActiveX ユーザーズ・ガイド』> 「ULSyncResult クラス」
- **Ultra Light.NET**『Ultra Light.NET ユーザーズ・ガイド』> 「AuthStatus プロパティ」(iAnywhere.Data.UltraLite ネームスペース) と 『Ultra Light.NET ユーザーズ・ガイド』> 「AuthStatus プロパティ」(iAnywhere.UltraLite ネームスペース) を参照してください。
- **Native Ultra Light for Java**『API リファレンス』の「**ianywhere.native_ultralite.SyncResult**」を参照してください。
- **C++ コンポーネント、Embedded SQL、および静的型 C++ API**『Ultra Light C/C++ ユーザーズ・ガイド』> 「auth_status パラメータ」を参照してください。
- **静的型 Java API**『Ultra Light 静的型 Java ユーザーズ・ガイド』> 「auth_status パラメータ」を参照してください。
- **Ultra Light for M-Business Anywhere**『UltraLite for M-Business Anywhere User's Guide』> 「Class SyncResult」を参照してください。

Authentication Value 同期パラメータ

機能

カスタム Mobile Link のユーザ認証スクリプトの結果をレポートします。Mobile Link 同期サーバが、この情報をクライアントに提供します。

デフォルト

デフォルトの Mobile Link ユーザ認証メカニズムによって設定される値については、「Authentication Status 同期パラメータ」399 ページで説明します。

使用法

このパラメータは Mobile Link 同期サーバによって設定されるため、読み込み専用です。

参照

『Mobile Link 管理ガイド』> 「authenticate_user 接続イベント」

『Mobile Link 管理ガイド』> 「authenticate_user_hashed 接続イベント」

「Authentication Status 同期パラメータ」399 ページ

インタフェース

- **MobileVB**『Ultra Light for MobileVB ユーザーズ・ガイド』> 「ULSyncResult クラス」
- **ActiveX**『Ultra Light ActiveX ユーザーズ・ガイド』> 「ULSyncResult クラス」
- **Ultra Light.NET**『Ultra Light.NET ユーザーズ・ガイド』> 「AuthValue プロパティ」(iAnywhere.Data.UltraLite ネームスペース) と 『Ultra Light.NET ユーザーズ・ガイド』> 「AuthValue プロパティ」(iAnywhere.UltraLite ネームスペース) を参照してください。
- **Native Ultra Light for Java**『API リファレンス』の「**ianywhere.native_ultralite.SyncParms**」を参照してください。
- **C++ コンポーネント、Embedded SQL、および静的型 C++ API**『Ultra Light C/C++ ユーザーズ・ガイド』> 「auth_value 同期パラメータ」を参照してください。
- **静的型 Java API**『Ultra Light 静的型 Java ユーザーズ・ガイド』> 「auth_value 同期パラメータ」を参照してください。
- **Ultra Light for M-Business Anywhere**『UltraLite for M-Business Anywhere User's Guide』> 「Class SyncResult」を参照してください。

Checkpoint Store 同期パラメータ

機能

同期中にデータベースのチェックポイントを追加して、同期プロセス中のデータベースの拡張を制限します。

チェックポイント操作は、アプリケーションと Palm コンジットの I/O 操作を増やすため、同期が低速になります。このオプションは、多くの更新を行う大規模なダウンロードに最適です。低速なフラッシュ・メモリを使用するデバイスでは、パフォーマンスの低下が望ましくないことがあります。

デフォルト

デフォルトでは、必要なチェックポイントのみが実行されます。

インタフェース

- **MobileVB**『Ultra Light for MobileVB ユーザーズ・ガイド』> 「ULSyncParms クラス」
- **ActiveX**『Ultra Light ActiveX ユーザーズ・ガイド』> 「ULSyncParms クラス」
- **Ultra Light.NET**『Ultra Light.NET ユーザーズ・ガイド』> 「CheckpointStore プロパティ」(iAnywhere.Data.UltraLite ネームスペース) と 『Ultra Light.NET ユーザーズ・ガイド』> 「CheckpointStore プロパティ」(iAnywhere.UltraLite ネームスペース) を参照してください。
- **Native Ultra Light for Java**『API リファレンス』の「**ianywhere.native_ultralite.SyncParms**」を参照してください。
- **C++ コンポーネント、Embedded SQL、および静的型 C++ API**『Ultra Light C/C++ ユーザーズ・ガイド』> 「checkpoint_store 同期パラメータ」を参照してください。
- **静的型 Java API** 使用できません。
- **Ultra Light for M-Business Anywhere**『UltraLite for M-Business Anywhere User's Guide』> 「Class SyncParms」を参照してください。

Disable Concurrency 同期パラメータ

機能

この同期中は他のスレッドからデータベースへのアクセスを禁止します。

デフォルト

このパラメータはブール値であり、デフォルトは false です (同時データベース・アクセスを許可)。ダウンロード時のデータ・アクセスは読み込み、書き込みともに可能です。それ以外は読み込み専用です。

参照

『Ultra Light データベース・ユーザーズ・ガイド』> 「Ultra Light での同時実行性について」

インタフェース

- **MobileVB** 使用できません。
- **ActiveX** 使用できません。
- **Ultra Light.NET**『Ultra Light.NET ユーザーズ・ガイド』> 「DisableConcurrency プロパティ」(iAnywhere.Data.UltraLite ネームスペース) と 『Ultra Light.NET ユーザーズ・ガイド』> 「DisableConcurrency プロパティ」(iAnywhere.UltraLite ネームスペース) を参照してください。
- **Native Ultra Light for Java**『API リファレンス』の「**ianywhere.native_ultralite.SyncParms**」を参照してください。
- **C++ コンポーネント、Embedded SQL、および静的型 C++ API**『Ultra Light C/C++ ユーザーズ・ガイド』> 「disable_concurrency 同期パラメータ」を参照してください。
- **静的型 Java API** 使用できません。
- **Ultra Light for M-Business Anywhere**『UltraLite for M-Business Anywhere User's Guide』> 「Class SyncParms」を参照してください。

Download Only 同期パラメータ

機能

この同期中は、Ultra Light データベースから変更をアップロードしません。

デフォルト

このパラメータはブール値であり、デフォルトは false です。

参照

「Ultra Light データベースへの読み込み専用テーブルの組み込み」356 ページ

「Upload Only 同期パラメータ」428 ページ

インタフェース

- **MobileVB**『Ultra Light for MobileVB ユーザーズ・ガイド』>「ULSyncParms クラス」
- **ActiveX**『Ultra Light ActiveX ユーザーズ・ガイド』>「ULSyncParms クラス」
- **Ultra Light.NET**『Ultra Light.NET ユーザーズ・ガイド』>「DownloadOnly プロパティ」(iAnywhere.Data.UltraLite ネームスペース) と 『Ultra Light.NET ユーザーズ・ガイド』>「DownloadOnly プロパティ」(iAnywhere.UltraLite ネームスペース) を参照してください。
- **Native Ultra Light for Java**『API リファレンス』の「**ianywhere.native_ultralite.SyncParms**」を参照してください。
- **C++ コンポーネント、Embedded SQL、および静的型 C++ API**『Ultra Light C/C++ ユーザーズ・ガイド』>「download_only 同期パラメータ」を参照してください。
- **静的型 Java API**『Ultra Light 静的型 Java ユーザーズ・ガイド』> 「download_only 同期パラメータ」を参照してください。
- **Ultra Light for M-Business Anywhere**『UltraLite for M-Business Anywhere User's Guide』> 「Class SyncParms」を参照してください。

Ignored Rows 同期パラメータ

機能

同期中にスクリプトがないために Mobile Link 同期サーバによってローが 1 つでも無視されると、このブール・パラメータが **true** に設定されます。

このパラメータは読み込み専用です。

インタフェース

- **MobileVB**『Ultra Light for MobileVB ユーザーズ・ガイド』>「ULSyncResult クラス」

- **ActiveX**『Ultra Light ActiveX ユーザーズ・ガイド』> 「ULSyncResult クラス」
- **Ultra Light.NET**『Ultra Light.NET ユーザーズ・ガイド』> 「IgnoredRows プロパティ」(iAnywhere.Data.UltraLite ネームスペース) と 『Ultra Light.NET ユーザーズ・ガイド』> 「IgnoredRows プロパティ」(iAnywhere.UltraLite ネームスペース) を参照してください。
- **Native Ultra Light for Java**『API リファレンス』の「**ianywhere.native_ultralite.SyncResult**」を参照してください。
- **C++ コンポーネント、Embedded SQL、および静的型 C++ API**『Ultra Light C/C++ ユーザーズ・ガイド』> 「ignored_rows 同期パラメータ」を参照してください。
- **静的型 Java API**『Ultra Light 静的型 Java ユーザーズ・ガイド』> 「ignored_rows 同期パラメータ」を参照してください。
- **Ultra Light for M-Business Anywhere**『UltraLite for M-Business Anywhere User's Guide』> 「Class SyncResult」を参照してください。

Keep Partial Download 同期パラメータ

機能

同期時の通信エラーが原因でダウンロードが失敗すると、このパラメータは、変更をロールバックしないでダウンロードを保持するかどうかを制御します。

デフォルト

デフォルト設定は false です。この場合、ダウンロードが失敗したあと、変更はすべてロールバックされます。

インタフェース

- **MobileVB**『Ultra Light for MobileVB ユーザーズ・ガイド』> 「ULSyncParms クラス」
- **ActiveX**『Ultra Light ActiveX ユーザーズ・ガイド』> 「ULSyncParms クラス」

- **Ultra Light.NET**『Ultra Light.NET ユーザーズ・ガイド』> 「KeepPartialDownload プロパティ」(iAnywhere.Data.UltraLite ネームスペース) と 『Ultra Light.NET ユーザーズ・ガイド』> 「KeepPartialDownload プロパティ」(iAnywhere.UltraLite ネームスペース) を参照してください。
- **Native Ultra Light for Java**『API リファレンス』の「**ianywhere.native_ultralite.SyncParms**」を参照してください。
- **C++ コンポーネント、Embedded SQL、および静的型 C++ API**『Ultra Light C/C++ ユーザーズ・ガイド』> 「keep_partial_download 同期パラメータ」を参照してください。
- **静的型 Java API** 使用できません。
- **Ultra Light for M-Business Anywhere**『UltraLite for M-Business Anywhere User's Guide』> 「Class SyncParms」を参照してください。

参照

- 『Mobile Link 管理ガイド』> 「失敗したダウンロードの再開」
- 「Resume Partial Download 同期パラメータ」414 ページ

New Password 同期パラメータ

機能

ユーザ名に対する新しい Mobile Link パスワードを設定します。

デフォルト

これはオプションの文字列パラメータです。

参照

「Mobile Link ユーザの認証」9 ページ

インタフェース

- **MobileVB**『Ultra Light for MobileVB ユーザーズ・ガイド』> 「ULSyncParms クラス」
- **ActiveX**『Ultra Light ActiveX ユーザーズ・ガイド』> 「ULSyncParms クラス」

- **Ultra Light.NET**『Ultra Light.NET ユーザーズ・ガイド』> 「NewPassword プロパティ」(iAnywhere.Data.UltraLite ネームスペース) と 『Ultra Light.NET ユーザーズ・ガイド』> 「NewPassword プロパティ」(iAnywhere.UltraLite ネームスペース) を参照してください。
- **Native Ultra Light for Java**『API リファレンス』の「**ianywhere.native_ultralite.SyncParms**」を参照してください。
- **C++ コンポーネント、Embedded SQL、および静的型 C++ API**『Ultra Light C/C++ ユーザーズ・ガイド』> 「new_password 同期パラメータ」を参照してください。
- **静的型 Java API**『Ultra Light 静的型 Java ユーザーズ・ガイド』> 「new_password 同期パラメータ」を参照してください。
- **Ultra Light for M-Business Anywhere**『UltraLite for M-Business Anywhere User's Guide』> 「Class SyncParms」を参照してください。

Number of Authentication Parameters パラメータ

機能

authentication_parameters スクリプトに渡される認証パラメータの数を指定します。

デフォルト

パラメータは指定されていません。

参照

「Authentication Parameters 同期パラメータ」398 ページ

『Mobile Link 管理ガイド』> 「authenticate_parameters 接続イベント」

インタフェース

- **MobileVB**『Ultra Light for MobileVB ユーザーズ・ガイド』> 「ULSyncParms クラス」
- **ActiveX**『Ultra Light ActiveX ユーザーズ・ガイド』> 「ULSyncParms クラス」

- **Ultra Light.NET**『Ultra Light.NET ユーザーズ・ガイド』> 「AuthenticationParms プロパティ」(iAnywhere.Data.UltraLite ネームスペース) または 『Ultra Light.NET ユーザーズ・ガイド』> 「AuthenticationParms プロパティ」(iAnywhere.UltraLite ネームスペース) を参照してください。

- **Native Ultra Light for Java**『API リファレンス』の「**ianywhere.native_ultralite.SyncParms**」を参照してください。

- **C++ コンポーネント、Embedded SQL、および静的型 C++ API**『Ultra Light C/C++ ユーザーズ・ガイド』> 「num_auth_parms パラメータ」を参照してください。

- **静的型 Java API**『Ultra Light 静的型 Java ユーザーズ・ガイド』> 「num_auth_parms パラメータ」を参照してください。

- **Ultra Light for M-Business Anywhere**『UltraLite for M-Business Anywhere User's Guide』> 「Class SyncParms」を参照してください。

Observer 同期パラメータ

機能

同期をモニタするコールバック関数またはイベント・ハンドラへのポインタです。

参照

「User Data 同期パラメータ」429 ページ

インタフェース

- **MobileVB** イベントを持つ接続オブジェクトを宣言します。『Ultra Light for MobileVB ユーザーズ・ガイド』> 「ULConnection クラス」を参照してください。

- **ActiveX** DatabaseManager オブジェクトを開くときに CreateObjectWithEvents を使用します。『Ultra Light ActiveX ユーザーズ・ガイド』> 「ULDatabaseManager クラス」を参照してください。

- **Ultra Light.NET**『Ultra Light.NET ユーザーズ・ガイド』> 「メンバ ULSyncProgressListener」(iAnywhere.Data.UltraLite ネームスペース) と 『Ultra Light.NET ユーザーズ・ガイド』> 「SyncProgressListener インタフェース」(iAnywhere.UltraLite ネームスペース) を参照してください。
- **Native Ultra Light for Java**『API リファレンス』の「**ianywhere.native_ultralite.SyncProgressListener**」を参照してください。
- **C++ コンポーネント、Embedded SQL、および静的型 C++ API**『Ultra Light C/C++ ユーザーズ・ガイド』> 「observer 同期パラメータ」を参照してください。
- **静的型 Java API**『Ultra Light 静的型 Java ユーザーズ・ガイド』> 「observer 同期パラメータ」を参照してください。

Partial Download Retained 同期パラメータ

機能

同期時の通信エラーが原因でダウンロードが失敗すると、このパラメータは、変更をロールバックしないで、ダウンロードされたこの変更が適用されたかどうかを制御します。

デフォルト

このパラメータは Ultra Light によって設定されます。部分的ダウンロードが保持されていない場合、この値は false です。

部分的ダウンロードが保持されるのは、Keep Partial Download が true に設定されている場合のみです。

インタフェース

- **MobileVB**『Ultra Light for MobileVB ユーザーズ・ガイド』> 「ULSyncResult クラス」
- **ActiveX**『Ultra Light ActiveX ユーザーズ・ガイド』> 「ULSyncResult クラス」
- **Ultra Light.NET**『Ultra Light.NET ユーザーズ・ガイド』> 「PartialDownloadRetained プロパティ」(iAnywhere.Data.UltraLite ネームスペース) と『Ultra Light.NET

ユーザーズ・ガイド』> 「PartialDownloadRetained プロパティ」(iAnywhere.UltraLite ネームスペース) を参照してください。

- **Native Ultra Light for Java**『API リファレンス』の「**ianywhere.native_ultralite.SyncResult**」を参照してください。
- **C++ コンポーネント、Embedded SQL、および静的型 C++ API**『Ultra Light C/C++ ユーザーズ・ガイド』> 「partial_download_retained 同期パラメータ」を参照してください。
- **静的型 Java API** 使用できません。
- **Ultra Light for M-Business Anywhere**『UltraLite for M-Business Anywhere User's Guide』> 「Class SyncResult」を参照してください。

参照

- 『Mobile Link 管理ガイド』> 「失敗したダウンロードの再開」
- 「Keep Partial Download 同期パラメータ」405 ページ
- 「Resume Partial Download 同期パラメータ」414 ページ

Password 同期パラメータ

機能

ユーザ名に対する Mobile Link パスワードを指定する文字列です。このユーザ名とパスワードは他のデータベース・ユーザ ID やパスワードとは別のもので、アプリケーションを Mobile Link 同期サーバに対して識別し、認証するために使用されます。

デフォルト

これはオプションの文字列パラメータです。

参照

「Mobile Link ユーザの認証」9 ページ

インタフェース

- **MobileVB**『Ultra Light for MobileVB ユーザーズ・ガイド』> 「ULSyncParms クラス」
- **ActiveX**『Ultra Light ActiveX ユーザーズ・ガイド』> 「ULSyncParms クラス」

- **Ultra Light.NET**『Ultra Light.NET ユーザーズ・ガイド』> 「Password プロパティ」(iAnywhere.Data.UltraLite ネームスペース) と 『Ultra Light.NET ユーザーズ・ガイド』> 「Password プロパティ」(iAnywhere.UltraLite ネームスペース) を参照してください。
- **Native Ultra Light for Java**『API リファレンス』の「**ianywhere.native_ultralite.SyncParms**」を参照してください。
- **C++ コンポーネント、Embedded SQL、および静的型 C++ API**『Ultra Light C/C++ ユーザーズ・ガイド』> 「password 同期パラメータ」を参照してください。
- **静的型 Java API**『Ultra Light 静的型 Java ユーザーズ・ガイド』> 「password 同期パラメータ」を参照してください。
- **Ultra Light for M-Business Anywhere**『UltraLite for M-Business Anywhere User's Guide』> 「Class SyncParms」を参照してください。

Ping 同期パラメータ

機能

Ultra Light クライアントと Mobile Link 同期サーバ間の通信を確認します。このパラメータが true に設定されている場合は、同期は行われません。

Mobile Link 同期サーバは、ping 要求を受信すると、統合データベースに接続し、ユーザを認証し、ユーザ認証ステータスと値をクライアントに送信します。

ping に成功した場合、Mobile Link サーバは情報メッセージを発行します。ping に失敗した場合は、エラー・メッセージを発行します。

Mobile Link サーバがコマンド・ライン・オプション -zu+ を指定して実行されていると、Mobile Link ユーザ名が ml_user システム・テーブルに見つからない場合は Mobile Link サーバがユーザを ml_user に追加します。

Mobile Link 同期サーバに次のスクリプトが存在する場合、ping 要求に対してこれらのスクリプトを実行できます。

- begin_connection
- authenticate_user
- authenticate_user_hashed
- end_connection

デフォルト

これはオプションのブール・パラメータです。

参照

「-pi オプション」186 ページ

インタフェース

- **MobileVB**『Ultra Light for MobileVB ユーザーズ・ガイド』> 「ULSyncParms クラス」
- **ActiveX**『Ultra Light ActiveX ユーザーズ・ガイド』> 「ULSyncParms クラス」
- **Ultra Light.NET**『Ultra Light.NET ユーザーズ・ガイド』> 「PingOnly プロパティ」(iAnywhere.Data.UltraLite ネームスペース) と 『Ultra Light.NET ユーザーズ・ガイド』> 「PingOnly プロパティ」(iAnywhere.UltraLite ネームスペース) を参照してください。
- **Native Ultra Light for Java**『API リファレンス』の「**ianywhere.native_ultralite.SyncParms**」を参照してください。
- **C++ コンポーネント、Embedded SQL、および静的型 C++ API**『Ultra Light C/C++ ユーザーズ・ガイド』> 「ping 同期パラメータ」を参照してください。
- **静的型 Java API**『Ultra Light 静的型 Java ユーザーズ・ガイド』> 「ping 同期パラメータ」を参照してください。
- **Ultra Light for M-Business Anywhere**『UltraLite for M-Business Anywhere User's Guide』> 「Class SyncParms」を参照してください。

publication 同期パラメータ

機能

同期させるパブリケーションを指定します。

デフォルト

パブリケーションを指定しない場合、すべてのデータが同期されます。

使用法

同期時には、publication パラメータを「**パブリケーション・マスク**」、つまり、パブリケーション定数を OR で結合したリストに設定します。

参照

『Ultra Light データベース・ユーザーズ・ガイド』> 「Ultra Light ジェネレータ」

「別々に同期するためのデータ・セットの設計」351 ページ

インタフェース

- **MobileVB**『Ultra Light for MobileVB ユーザーズ・ガイド』> 「ULPublicationSchema クラス」
- **ActiveX**『Ultra Light ActiveX ユーザーズ・ガイド』> 「ULPublicationSchema クラス」
- **Ultra Light.NET**『Ultra Light.NET ユーザーズ・ガイド』> 「ULPublicationSchema クラス」(iAnywhere.Data.UltraLite ネームスペース) と 『Ultra Light.NET ユーザーズ・ガイド』> 「PublicationSchema クラス」(iAnywhere.UltraLite ネームスペース) を参照してください。
- **Native Ultra Light for Java**『API リファレンス』の「**ianywhere.native_ultralite.PublicationSchema**」を参照してください。
- **C++ コンポーネント、Embedded SQL、および静的型 C++ API**『Ultra Light C/C++ ユーザーズ・ガイド』> 「publication 同期パラメータ」を参照してください。
- **静的型 Java API**『Ultra Light 静的型 Java ユーザーズ・ガイド』> 「publication 同期パラメータ」を参照してください。

- **Ultra Light for M-Business Anywhere**『UltraLite for M-Business Anywhere User's Guide』> 「Class PublicationSchema」を参照してください。

Resume Partial Download 同期パラメータ

機能

失敗したダウンロードを再開します。同期によって変更はアップロードされず、失敗したダウンロードでダウンロードされるはずだった変更のみがダウンロードされます。

デフォルト

False

参照

- 『Mobile Link 管理ガイド』> 「失敗したダウンロードの再開」
- 「Keep Partial Download 同期パラメータ」405 ページ
- 「Partial Download Retained 同期パラメータ」409 ページ

インタフェース

- **MobileVB**『Ultra Light for MobileVB ユーザーズ・ガイド』> 「ULSyncParms クラス」
- **ActiveX**『Ultra Light ActiveX ユーザーズ・ガイド』> 「ULSyncParms クラス」
- **Ultra Light.NET**『Ultra Light.NET ユーザーズ・ガイド』> 「ResumePartialDownload プロパティ」(iAnywhere.Data.UltraLite ネームスペース) と 『Ultra Light.NET ユーザーズ・ガイド』> 「ResumePartialDownload プロパティ」(iAnywhere.UltraLite ネームスペース) を参照してください。
- **Native Ultra Light for Java**『API リファレンス』の「**ianywhere.native_ultralite.SyncParms**」を参照してください。
- **C++ コンポーネント、Embedded SQL、および静的型 C++ API**『Ultra Light C/C++ ユーザーズ・ガイド』> 「resume_partial_download 同期パラメータ」を参照してください。
- **静的型 Java API** 使用できません。

- **Ultra Light for M-Business Anywhere**『UltraLite for M-Business Anywhere User's Guide』> 「Class SyncParms」を参照してください。

Security 同期パラメータ

機能

Mobile Link 同期サーバとのメッセージ交換に Certicom 暗号化テクノロジを使用するように、Ultra Light クライアントを設定します。

別途ライセンスを取得できるオプションが必要
トランスポート・レイヤ・セキュリティを使用するには、別途ライセンスを取得できる SQL Anywhere Studio セキュリティ・オプションを入手する必要があります。このセキュリティ・オプションは、輸出規制対象品目です。

このコンポーネントのご注文については、『SQL Anywhere Studio の紹介』> 「別途ライセンスが入手可能なコンポーネント」を参照してください。

このパラメータは、静的型 Java API では使用されません。静的型 Java API からセキュリティ機能のある同期を使用するには、別個のストリームを選択します。詳細については、「UlSecureRSASocketStream 同期パラメータ」443 ページと 「UlSecureSocketStream 同期パラメータ」444 ページを参照してください。

デフォルト

このパラメータは、デフォルトでトランスポート・レイヤ・セキュリティなしに対応する null に設定されます。

使用法

同期ストリームに加えてセキュリティ・ストリームを指定します。指定可能な値は次のとおりです。

- **ULSecureCerticomTLSStream()**　Certicom によって提供される楕円曲線暗号化トランスポート・レイヤ・セキュリティ。このストリームを使用する場合は、アプリケーションを *ulecc.llib* にリンクするか、あるいはアプリケーションとともに *ulecc9.dll* (Unicode アプリケーションの場合は *ulecc9w.dll*) を提供する必要があります。

- **ULSecureRSATLSStream()** Certicom によって提供される RSA トランスポート・レイヤ・セキュリティ。このストリームを使用する場合は、アプリケーションを *ulrsa.lib* にリンクするか、あるいはアプリケーションとともに *ulrsa9.dll* (Unicode アプリケーションの場合は *ulrsa9w.dll*) を提供する必要があります。

例

次の C++ コードは、TCP/IP による RSA トランスポート・レイヤ・セキュリティを使用するために同期を設定します。

```
//C++
 ul_synch_info info;
 ...
 info.stream = ULSocketStream();
 info.security = ULSecureRSATLSStream();
```

参照

- 『Mobile Link 管理ガイド』> 「Mobile Link トランスポート・レイヤ・セキュリティ」
- 「Security Parameters 同期パラメータ」416 ページ
- 「certificate_company」45 ページ
- 「certificate_name」47 ページ
- 「certificate_unit」49 ページ
- 「trusted_certificates」68 ページ「certificate_company」45 ページ

インタフェース

- **C++ コンポーネント、Embedded SQL、および静的型 C++ API**『Ultra Light C/C++ ユーザーズ・ガイド』> 「security 同期パラメータ」を参照してください。
- **静的型 Java API** 個別の同期ストリームを使用します。「UlSecureRSASocketStream 同期パラメータ」443 ページと「UlSecureSocketStream 同期パラメータ」444 ページを参照してください。

Security Parameters 同期パラメータ

機能

トランスポート・レイヤ・セキュリティを使用する場合に必要なオプションを設定します。このパラメータは、**security** パラメータとともに使用してください。

詳細については、「Security 同期パラメータ」415 ページを参照してください。

このパラメータは静的 Java アプリケーションには適用されません。Ultra Light 静的 Java アプリケーションからセキュリティ機能のある同期を使用するには、別個のストリームを選択します。詳細については、「UlSecureRSASocketStream 同期パラメータ」443 ページと「UlSecureSocketStream 同期パラメータ」444 ページを参照してください。

デフォルト

デフォルト設定は、空の文字列です。

使用法

ULSecureCerticomTLSStream() と ULSecureRSATLSStream() セキュリティ・パラメータは、次のオプションで構成される文字列を使用します。オプションのパラメータは、セミコロンで区切られた文字列で指定します。

- **certificate_company**　証明書に記されている組織フィールドがこの値と一致する場合にだけ、Ultra Light アプリケーションはサーバ証明書を受け入れます。デフォルトでは、このフィールドはチェックされていません。
- **certificate_unit**　証明書に記されている部署フィールドがこの値と一致する場合にだけ、Ultra Light アプリケーションはサーバ証明書を受け入れます。デフォルトでは、このフィールドはチェックされていません。
- **certificate_name**　証明書に記されている共通名フィールドがこの値と一致する場合にだけ、Ultra Light アプリケーションはサーバ証明書を受け入れます。デフォルトでは、このフィールドはチェックされていません。
- **trusted_certificates**　この値を指定すると、Ultra Light アプリケーションはデータベース・スキーマ自体からではなく永続的な記憶領域から証明書を取り出します。このオプションは、Palm OS では使用できません。

次に例を示します。

```
ul_synch_info info;
 ...
 info.stream = ULSocketStream();
 info.security = ULSecureCerticomTLSStream();
 info.security_parms =
      UL_TEXT( "certificate_company=Sybase" )
      UL_TEXT( ";" )
      UL_TEXT( "certificate_unit=Sales" );
```

Palm OS で安全な同期を使用する場合、trusted_certificates は使用できません。アプリケーション自体に証明書を埋め込むには、Ultra Light ジェネレータで -r コマンド・ライン・オプションを使用する必要があります。詳細については、『Ultra Light データベース・ユーザーズ・ガイド』> 「Ultra Light ジェネレータ」を参照してください。

参照

- 「Security 同期パラメータ」415 ページ
- 「certificate_company」45 ページ
- 「certificate_name」47 ページ
- 「certificate_unit」49 ページ
- 「trusted_certificates」68 ページ「certificate_company」45 ページ

インタフェース

- **C++ コンポーネント、Embedded SQL、および静的型 C++ API**『Ultra Light C/C++ ユーザーズ・ガイド』> 「security_parms 同期パラメータ」を参照してください。
- **静的型 Java API** 個別の同期ストリームを使用します。

Send Column Names 同期パラメータ

機能

true に設定されている場合、Ultra Light は各カラム名を Mobile Link 同期サーバに送信します。

このパラメータは、通常は同期スクリプトを自動的に生成するために Mobile Link 同期サーバで -za または -ze スイッチとともに使用されます。

このパラメータは Ultra Light 静的 Java アプリケーションには使用できません。

デフォルト

False。

参照　『Mobile Link 管理ガイド』> 「-za オプション」

インタフェース

- **MobileVB**『Ultra Light for MobileVB ユーザーズ・ガイド』> 「ULSyncParms クラス」
- **ActiveX**『Ultra Light ActiveX ユーザーズ・ガイド』> 「ULSyncParms クラス」
- **Ultra Light.NET**『Ultra Light.NET ユーザーズ・ガイド』> 「SendColumnNames プロパティ」(iAnywhere.Data.UltraLite ネームスペース) と 『Ultra Light.NET ユーザーズ・ガイド』> 「SendColumnNames プロパティ」(iAnywhere.UltraLite ネームスペース) を参照してください。
- **Native Ultra Light for Java**『API リファレンス』の「**ianywhere.native_ultralite.SyncParms**」を参照してください。
- **C++ コンポーネント、Embedded SQL、および静的型 C++ API**『Ultra Light C/C++ ユーザーズ・ガイド』> 「send_column_names 同期パラメータ」を参照してください。
- **静的型 Java API** この機能は静的型 Java API には使用できません。
- **Ultra Light for M-Business Anywhere**『UltraLite for M-Business Anywhere User's Guide』> 「Class SyncParms」を参照してください。

Send Download Acknowledgement 同期パラメータ

機能

このブール・パラメータを true に設定すると、Mobile Link 同期サーバに対して、クライアントからダウンロード確認が提供されることを通知します。

クライアントがダウンロード確認を送信する場合、Mobile Link 同期サーバのワーカ・スレッドは、クライアントがダウンロードを適用するまで待機します。クライアントがダウンロード確認を送信しない場合、Mobile Link 同期サーバは、次の同期のため、まもなく解放されます。

デフォルト False

インタフェース

- **MobileVB**『Ultra Light for MobileVB ユーザーズ・ガイド』> 「ULSyncParms クラス」
- **ActiveX**『Ultra Light ActiveX ユーザーズ・ガイド』> 「ULSyncParms クラス」
- **Ultra Light.NET**『Ultra Light.NET ユーザーズ・ガイド』> 「SendDownloadAck プロパティ」(iAnywhere.Data.UltraLite ネームスペース) と 『Ultra Light.NET ユーザーズ・ガイド』> 「SendDownloadAck プロパティ」(iAnywhere.UltraLite ネームスペース) を参照してください。
- **Native Ultra Light for Java**『API リファレンス』の「**ianywhere.native_ultralite.SyncParms**」を参照してください。
- **C++ コンポーネント、Embedded SQL、および静的型 C++ API**『Ultra Light C/C++ ユーザーズ・ガイド』> 「send_download_ack 同期パラメータ」を参照してください。
- **静的型 Java API** このパラメータは静的 Java アプリケーションには使用できません。
- **Ultra Light for M-Business Anywhere**『UltraLite for M-Business Anywhere User's Guide』> 「Class SyncParms」を参照してください。

Stream Error 同期パラメータ

機能 通信エラー・レポート情報を保持する構造体を設定します。

適用対象 このパラメータは、C/C++ インタフェースにのみ適用されます。

Ultra Light C++ コンポーネント以外の Ultra Light コンポーネントは、Sync Result パラメータの一部として通信エラー情報を受け取ります。「Sync Result 同期パラメータ」426 ページを参照してください。

この機能は Ultra Light 静的 Java アプリケーションには使用できません。

デフォルト

このパラメータは Mobile Link 同期サーバによって設定されるため、読み込み専用です。このパラメータが設定されるのは、同期時に通信エラーが発生する場合のみです。

インタフェース

- **MobileVB**『Ultra Light for MobileVB ユーザーズ・ガイド』>「ULSyncResult クラス」
- **ActiveX**『Ultra Light ActiveX ユーザーズ・ガイド』>「ULSyncResult クラス」
- **Ultra Light.NET** ianywhere.UltraLite.SyncResult を参照してください。
- **Native Ultra Light for Java**『API リファレンス』の「**ianywhere.native_ultralite.SyncParms**」を参照してください。
- **C++ コンポーネント、Embedded SQL、および静的型 C++ API**『Ultra Light C/C++ ユーザーズ・ガイド』>「stream_error 同期パラメータ」を参照してください。
- **静的型 Java API** この機能は静的 Java には使用できません。
- **Ultra Light for M-Business Anywhere**『UltraLite for M-Business Anywhere User's Guide』>「Class SyncResult」を参照してください。

Stream Type 同期パラメータ

機能

同期に使用する Mobile Link ネットワーク・プロトコルを設定します。

ほとんどのネットワーク・プロトコルでは、Mobile Link 同期サーバ・アドレスや他の動作を識別するプロトコル・オプションが必要です。これらのオプションは、**stream_parms** パラメータで指定します。

詳細については、「Stream Parameters 同期パラメータ」425 ページを参照してください。

デフォルト

このパラメータは必須です。デフォルト値はありません。

使用法

ネットワーク・プロトコルがオプションを必要とする場合は、Stream Parameters を使用してそのオプションを渡すか、あるいは Stream Parameters パラメータを null に設定します。

stream 関数には以下の種類がありますが、すべてのターゲット・プラットフォームですべての関数を使用できるわけではありません。

ネットワーク・プロトコル	説明
ActiveSync	ActiveSync 同期 (Windows CE のみ)。 プロトコルの一覧は、「ActiveSync プロトコル・オプション」433 ページを参照してください。
HTTP	HTTP を介して同期。 HTTP プロトコルは基本となるトランスポートとして TCP/IP を使用します。Ultra Light アプリケーションは Web ブラウザとして機能し、Mobile Link 同期サーバは Web サーバとして機能します。Ultra Light アプリケーションは、サーバへのデータ送信のために POST 要求を送り、サーバからのデータの読み込みのために GET 要求を送ります。 プロトコルの一覧は、「HTTP プロトコル・オプション」439 ページを参照してください。

ネットワーク・プロトコル	説明
HTTPS と HTTPS_FIPS	HTTPS プロトコルを介して同期。 HTTPS プロトコルは、基本のプロトコルとして SSL または TLS を使用します。これは、インターネット・プロトコル (HTTP と TCP/IP) 上で動作します。 HTTPS プロトコルでは、Certicom が提供するテクノロジを使用する必要があります。Certicom テクノロジを使用するには、別途ライセンスを取得できる SQL Anywhere Studio セキュリティ・オプションを入手する必要があります。このセキュリティ・オプションは、輸出規制対象品目です。詳細については、『SQL Anywhere Studio の紹介』> 「別途ライセンスが入手可能なコンポーネント」を参照してください。 プロトコルの一覧は、「HTTPS プロトコル・オプション」441 ページを参照してください。
TCP/IP	TCP/IP を介して同期。 プロトコルの一覧は、「TCP/IP プロトコル・オプション」438 ページを参照してください。
UlSecureSocketStream()	楕円曲線暗号化を使用したトランスポート・レイヤ・セキュリティでの TCP/IP または HTTP 同期。このストリームは静的 Java アプリケーションでのみ使用できます。 プロトコルの一覧は、「UlSecureSocketStream 同期パラメータ」444 ページを参照してください。

ネットワーク・プロトコル	説明
UlSecureRSASocketStream()	RSA 暗号化を使用したトランスポート・レイヤ・セキュリティでの TCP/IP または HTTP 同期。このストリームは静的 Java アプリケーションでのみ使用できます。 プロトコルの一覧は、「UlSecureSocketStream 同期パラメータ」444 ページを参照してください。

インタフェース

- **MobileVB**『Ultra Light for MobileVB ユーザーズ・ガイド』> 「ULSyncParms クラス」
- **ActiveX**『Ultra Light ActiveX ユーザーズ・ガイド』> 「ULSyncParms クラス」
- **Ultra Light.NET**『Ultra Light.NET ユーザーズ・ガイド』> 「Stream プロパティ」(iAnywhere.Data.UltraLite ネームスペース) と 『Ultra Light.NET ユーザーズ・ガイド』> 「Stream プロパティ」(iAnywhere.UltraLite ネームスペース) を参照してください。
- **Native Ultra Light for Java**『API リファレンス』の「**ianywhere.native_ultralite.SyncParms**」を参照してください。
- **C++ コンポーネント、Embedded SQL、および静的型 C++ API**『Ultra Light C/C++ ユーザーズ・ガイド』> 「stream 同期パラメータ」を参照してください。
- **静的型 Java API**『Ultra Light 静的型 Java ユーザーズ・ガイド』> 「stream 同期パラメータ」を参照してください。
- **Ultra Light for M-Business Anywhere**『UltraLite for M-Business Anywhere User's Guide』> 「Class SyncParms」を参照してください。

Stream Parameters 同期パラメータ

機能

ネットワーク・プロトコルを設定するオプションを設定します。

ネットワーク・プロトコル・オプションのリストで、セミコロンで区切られます。各オプションは *keyword*=*value* の形式で、許可されるキーワード・セットはネットワーク・プロトコルによって異なります。

各プロトコルで使用できるオプションのリストについては、次の各項を参照してください。

- 「ActiveSync プロトコル・オプション」433 ページ
- 「HotSync プロトコル・オプション」435 ページ
- 「HTTP プロトコル・オプション」439 ページ
- 「HTTPS プロトコル・オプション」441 ページ
- 「TCP/IP プロトコル・オプション」438 ページ
- 「UlSecureRSASocketStream 同期パラメータ」443 ページ
- 「UlSecureSocketStream 同期パラメータ」444 ページ

デフォルト

これはオプションの文字列パラメータです。デフォルトは null です。

参照

「Ultra Light 同期クライアントのネットワーク・プロトコルのオプション」433 ページ

インタフェース

- **MobileVB**『Ultra Light for MobileVB ユーザーズ・ガイド』>「ULSyncParms クラス」
- **ActiveX**『Ultra Light ActiveX ユーザーズ・ガイド』>「ULSyncParms クラス」
- **Ultra Light.NET**『Ultra Light.NET ユーザーズ・ガイド』>「StreamParms プロパティ」(iAnywhere.Data.UltraLite ネームスペース) と 『Ultra Light.NET ユーザーズ・ガイド』>「StreamParms プロパティ」(iAnywhere.UltraLite ネームスペース) を参照してください。

- **Native Ultra Light for Java**『API リファレンス』の「**ianywhere.native_ultralite.SyncParms**」を参照してください。
- **C++ コンポーネント、Embedded SQL、および静的型 C++ API**『Ultra Light C/C++ ユーザーズ・ガイド』> 「stream_parms 同期パラメータ」を参照してください。
- **静的型 Java API**『Ultra Light 静的型 Java ユーザーズ・ガイド』> 「stream_parms 同期パラメータ」を参照してください。
- **Ultra Light for M-Business Anywhere**『UltraLite for M-Business Anywhere User's Guide』> 「Class SyncParms」を参照してください。

Sync Result 同期パラメータ

機能

同期のステータスをレポートします。

適用対象

このパラメータは、Ultra Light コンポーネントにのみ適用されます。

C/C++ インタフェースは、この情報を個々のパラメータとして受け取ります。

通信エラー情報は Ultra Light 静的 Java アプリケーションには使用できません。

デフォルト

このパラメータは Mobile Link 同期サーバによって設定されるため、読み込み専用です。このパラメータが設定されるのは、同期時に通信エラーが発生する場合のみです。

説明

Sync Result は、さまざまな情報が個々のフィールドに格納された複合パラメータです。

- **Authentication Status** 認証の成功または失敗をレポートします。「Authentication Status 同期パラメータ」399 ページを参照してください。
- **Ignored Rows** 無視されるローの数をレポートします。「Ignored Rows 同期パラメータ」404 ページを参照してください。

- **Stream Error 情報** Stream Error 情報には、Stream Error Code、Stream Error Context、Stream Error ID、Stream Error System が含まれています。
- **Upload OK** アップロード・フェーズの成功または失敗をレポートします。「Upload OK 同期パラメータ」427 ページを参照してください。

インタフェース

- **MobileVB**『Ultra Light for MobileVB ユーザーズ・ガイド』>「ULSyncResult クラス」
- **ActiveX**『Ultra Light ActiveX ユーザーズ・ガイド』>「ULSyncResult クラス」
- **Ultra Light.NET**『Ultra Light.NET ユーザーズ・ガイド』>「ULSyncResult クラス」(iAnywhere.Data.UltraLite ネームスペース) と 『Ultra Light.NET ユーザーズ・ガイド』> 「SyncResult クラス」(iAnywhere.UltraLite ネームスペース) を参照してください。
- **Native Ultra Light for Java**『API リファレンス』の「**ianywhere.native_ultralite.SyncParms**」を参照してください。
- **C++ コンポーネント、Embedded SQL、静的型 C++ API**『Ultra Light C/C++ ユーザーズ・ガイド』> 「stream_error 同期パラメータ」を参照してください。
- **静的型 Java API** この機能は静的 Java には使用できません。
- **Ultra Light for M-Business Anywhere**『UltraLite for M-Business Anywhere User's Guide』> 「Class SyncResult」を参照してください。

Upload OK 同期パラメータ

機能

Mobile Link 同期サーバにアップロードされたデータのステータスをレポートします。

使用法

このパラメータは Mobile Link 同期サーバによって設定されるため、読み込み専用です。

同期後、このパラメータには、アップロードが成功した場合は **true**、それ以外の場合は **false** が入ります。同期エラーがあったかどうかについてこのパラメータを確認することによって、エラーが発生する前にデータが正常にアップロードされたかどうかを確認できます。

インタフェース

- **MobileVB**『Ultra Light for MobileVB ユーザーズ・ガイド』> 「ULSyncResult クラス」
- **ActiveX**『Ultra Light ActiveX ユーザーズ・ガイド』> 「ULSyncResult クラス」
- **Ultra Light.NET**『Ultra Light.NET ユーザーズ・ガイド』> 「UploadOK プロパティ」(iAnywhere.Data.UltraLite ネームスペース) と 『Ultra Light.NET ユーザーズ・ガイド』> 「UploadOK プロパティ」(iAnywhere.UltraLite ネームスペース) を参照してください。
- **Native Ultra Light for Java**『API リファレンス』の「**ianywhere.native_ultralite.SyncParms**」を参照してください。
- **C++ コンポーネント、Embedded SQL、および静的型 C++ API**『Ultra Light C/C++ ユーザーズ・ガイド』> 「upload_ok 同期パラメータ」を参照してください。
- **静的型 Java API**『Ultra Light 静的型 Java ユーザーズ・ガイド』> 「upload_ok 同期パラメータ」を参照してください。
- **Ultra Light for M-Business Anywhere**『UltraLite for M-Business Anywhere User's Guide』> 「Class SyncResult」を参照してください。

Upload Only 同期パラメータ

機能

現在の同期中にダウンロードは発生しないことを示します。これにより、特に低速の通信リンクでは、通信時間を節約できます。true に設定すると、クライアントは Mobile Link 同期サーバからのアップロード確認を待ってから、同期セッションを正常終了します。

デフォルト

このパラメータはオプションのブール値であり、デフォルトは false です。

参照

「優先度の高い変更の同期」354 ページ

「Download Only 同期パラメータ」403 ページ

インタフェース

- **MobileVB**『Ultra Light for MobileVB ユーザーズ・ガイド』>「ULSyncParms クラス」
- **ActiveX**『Ultra Light ActiveX ユーザーズ・ガイド』>「ULSyncParms クラス」
- **Ultra Light.NET**『Ultra Light.NET ユーザーズ・ガイド』>「UploadOnly プロパティ」(iAnywhere.Data.UltraLite ネームスペース) と 『Ultra Light.NET ユーザーズ・ガイド』>「UploadOnly プロパティ」(iAnywhere.UltraLite ネームスペース) を参照してください。
- **Native Ultra Light for Java**『API リファレンス』の「**ianywhere.native_ultralite.SyncParms**」を参照してください。
- **C++ コンポーネント、Embedded SQL、および静的型 C++ API**『Ultra Light C/C++ ユーザーズ・ガイド』>「upload_only 同期パラメータ」を参照してください。
- **静的型 Java API**『Ultra Light 静的型 Java ユーザーズ・ガイド』>「upload_only 同期パラメータ」を参照してください。
- **Ultra Light for M-Business Anywhere**『UltraLite for M-Business Anywhere User's Guide』>「Class SyncParms」を参照してください。

User Data 同期パラメータ

機能

アプリケーション固有の情報を同期 observer で使用できるようにします。

使用法

同期 observer コールバック関数またはイベント・ハンドラの実装時に、User Data パラメータを使用して情報を指定することによって、アプリケーション固有の情報を使用可能にできます。

Ultra Light.NET などのコンポーネントは、ユーザ・データの処理に別個のパラメータを必要しないので、User Data パラメータはありません。

参照

「Observer 同期パラメータ」408 ページ

インタフェース

- **MobileVB**『Ultra Light for MobileVB ユーザーズ・ガイド』> 「ULConnection クラス」
- **ActiveX**『Ultra Light ActiveX ユーザーズ・ガイド』> 「ULDatabaseManager クラス」
- **Native Ultra Light for Java**『API リファレンス』の「**ianywhere.native_ultralite.Connection**」を参照してください。
- **C++ コンポーネント、Embedded SQL、および静的型 C++ API**『Ultra Light C/C++ ユーザーズ・ガイド』> 「user_data 同期パラメータ」を参照してください。
- **静的型 Java API**『Ultra Light 静的型 Java ユーザーズ・ガイド』> 「user_data 同期パラメータ」を参照してください。

User Name 同期パラメータ

機能

ユーザ名を指定する文字列であり、Mobile Link 同期サーバが Mobile Link クライアントをユニークに識別するために使用します。Mobile Link では、この値を使用して、ダウンロードする内容の決定、同期ステータスの記録、同期中の割り込みからの復帰を行います。

デフォルト

パラメータにはデフォルト値がないので、明示的に設定してください。

使用法

ユーザ認証がオフの状態で Mobile Link 同期サーバが実行されていないかぎり、ユーザ名が必要です。詳細については、『Mobile Link 管理ガイド』> 「-zu オプション」を参照してください。

参照

「Mobile Link ユーザの認証」9 ページ

「Mobile Link ユーザ」7 ページ

インタフェース

- **MobileVB**『Ultra Light for MobileVB ユーザーズ・ガイド』> 「ULSyncParms クラス」
- **ActiveX**『Ultra Light ActiveX ユーザーズ・ガイド』> 「ULSyncParms クラス」
- **Ultra Light.NET**『Ultra Light.NET ユーザーズ・ガイド』> 「UserName プロパティ」(iAnywhere.Data.UltraLite ネームスペース) と 『Ultra Light.NET ユーザーズ・ガイド』> 「UserName プロパティ」(iAnywhere.UltraLite ネームスペース) を参照してください。
- **Native Ultra Light for Java**『API リファレンス』の「**ianywhere.native_ultralite.SyncParms**」を参照してください。
- **C++ コンポーネント、Embedded SQL、および静的型 C++ API**『Ultra Light C/C++ ユーザーズ・ガイド』> 「user_name 同期パラメータ」を参照してください。
- **静的型 Java API**『Ultra Light 静的型 Java ユーザーズ・ガイド』> 「user_name 同期パラメータ」を参照してください。
- **Ultra Light for M-Business Anywhere**『UltraLite for M-Business Anywhere User's Guide』> 「Class SyncParms」を参照してください。

Version 同期パラメータ

機能

統合データベースの同期スクリプトは、それぞれバージョン文字列で区別されます。たとえば、異なる文字列バージョンによって特定される 2 種類の **download_cursor** スクリプトがあります。Ultra Light アプリケーションは、バージョン文字列により、同期スクリプトのセットから 1 種類を選択できます。

デフォルト

これは必須の文字列パラメータです。

参照

『Mobile Link 管理ガイド』> 「スクリプト・バージョン」

インタフェース

- **MobileVB**『Ultra Light for MobileVB ユーザーズ・ガイド』> 「ULSyncParms クラス」

- **ActiveX**『Ultra Light ActiveX ユーザーズ・ガイド』> 「ULSyncParms クラス」
- **Ultra Light.NET**『Ultra Light.NET ユーザーズ・ガイド』> 「Version プロパティ」(iAnywhere.Data.UltraLite ネームスペース) と 『Ultra Light.NET ユーザーズ・ガイド』> 「Version プロパティ」(iAnywhere.UltraLite ネームスペース) を参照してください。
- **Native Ultra Light for Java**『API リファレンス』の「**ianywhere.native_ultralite.SyncParms**」を参照してください。
- **C++ コンポーネント、Embedded SQL、および静的型 C++ API**『Ultra Light C/C++ ユーザーズ・ガイド』> 「version 同期パラメータ」を参照してください。
- **静的型 Java API**『Ultra Light 静的型 Java ユーザーズ・ガイド』> 「version 同期パラメータ」を参照してください。
- **Ultra Light for M-Business Anywhere**『UltraLite for M-Business Anywhere User's Guide』> 「Class SyncParms」を参照してください。

Ultra Light 同期クライアントのネットワーク・プロトコルのオプション

この項では、各ネットワーク・プロトコルで使用できるオプションを示します。ネットワーク・プロトコルのオプションは、アドレス情報 (ホストとポート) やプロトコル固有の情報などを提供することによって、クライアントが Mobile Link 同期サーバを特定して正しく通信できるようにします。

ActiveSync プロトコル・オプション

ActiveSync 同期ストリームには、Windows CE で動作する Native Ultra Light for Java、Embedded SQL、静的型 C++ API アプリケーションからしかアクセスできません。

ActiveSync 同期を選択する手順は、次のとおりです。

- Native Ultra Light for Java の場合は、StreamType.ACTIVE_SYNC を syncParms.setStream メソッドの引数として指定します。次に例を示します。

  ```
  _conn.syncParms.setStream( StreamType.ACTIVE_SYNC );
  ```

  詳細については、『Native UltraLite for Java API Reference』の「**ianywhere.native_ultralite.StreamType**」と「**ianywhere.native_ultralite.SyncParms**」を参照してください。

- Embedded SQL と静的型 C++ API の場合は、ULActiveSyncStream() をネットワーク・プロトコルとして指定します。次に例を示します。

  ```
  ul_synch_info info;
   ...
   info.stream = ULActiveSyncStream();
  ```

  詳細については、『Ultra Light C/C++ ユーザーズ・ガイド』>「ULActiveSyncStream 関数」を参照してください。

プロトコル・オプションの意味

プロトコル・オプションは、デスクトップ・マシン上で動作する Mobile Link ActiveSync プロバイダから Mobile Link 同期サーバへの接続を制御します。

プロトコル・オプションは次の形式をとります。

```
stream=stream_name;provider_stream_parameters
```

stream_name は Mobile Link 同期サーバと通信するときにコンジットが使用するプロトコルを示します。次のいずれかを指定できます。

- **tcpip**
- **http**
- **https**

provider_stream_parameters は、ActiveSync プロバイダが使用するプロトコル・オプションのセットであり、使用するプロトコルのプロトコル・オプションと同じ形式です。指定されたプロトコルに対して、*provider_stream_parameters* は、プロトコルのプロトコル・オプションと同じデフォルトを使用します。*stream_name* のデフォルト値は **tcpip** です。

たとえば、次の静的 C++ コードでは HTTP プロトコルを使用します。

```
ULInitSynchInfo( &info );
 info.stream = ULActiveSyncStream();
 info.stream_parms = "stream=http";
 ULSynchronize( &sqlca, &info );
```

provider_stream_parameters の詳細については、「TCP/IP プロトコル・オプション」438 ページ、「HTTP プロトコル・オプション」439 ページ、および 「HTTPS プロトコル・オプション」441 ページを参照してください。

ActiveSync 同期への暗号化の追加

ストリームに Certicom 暗号化を追加するには、デスクトップ・マシン上のファイルにルート証明書を入れてください。これは、暗号化情報が **security** 同期パラメータに埋め込まれる他の Ultra Light アプリケーションの場合とは異なります。

プロトコル・オプションは、Adaptive Server Anywhere Mobile Link クライアントの場合と同じように指定する必要があります。このフォーマットは、次のとおりです。

security=*cipher*{ *keyword*=*value*;… }

cipher には certicom_tls を指定し、キーワードは次のリストから指定します。

- **certificate_company**　証明書の組織フィールド。
- **certificate_unit**　証明書の組織単位フィールド。
- **certificate_name**　証明書の共通名フィールド。
- **trusted_certificates**　信用された証明書の場所。

たとえば、静的 C++ アプリケーションは次のような行を使用することがあります。

```
info.stream_parms =
"stream=tcpip;security=ecc_tls(trusted_certificates=tru
sted.crt)";
```

詳細については、次の項を参照してください。

- 「certificate_company」45 ページ
- 「certificate_name」47 ページ
- 「certificate_unit」49 ページ
- 「trusted_certificates」68 ページ

HotSync プロトコル・オプション

HotSync 同期ストリームには、Palm Computing Platform で動作する Ultra Light for MobileVB アプリケーション、Embedded SQL アプリケーション、静的型 C++ API アプリケーションからしかアクセスできません。HTTP や TCP/IP 同期とは異なり、HotSync 同期は、Ultra Light アプリケーション内の同期機能ではなく HotSync Manager によって外部的に開始されます。

HotSync 同期を選択する手順は、次のとおりです。

- Ultra Light for MobileVB の場合は、ULSyncParms.Stream として ULStreamType 列挙から ulPalmConduit を選択します。

  詳細については、『Ultra Light for MobileVB ユーザーズ・ガイド』> 「ULSyncParms クラス」を参照してください。

- Embedded SQL や静的型 C++ API の場合、アプリケーションの ULSetSynchInfo メソッドに ul_synch_info 構造体を指定します。ストリーム・パラメータは無視されます。UL_NULL も指定できます。

  詳細については、『Ultra Light C/C++ ユーザーズ・ガイド』> 「ULSetSynchInfo 関数」を参照してください。

プロトコル・オプションの意味

HotSync 同期の場合、プロトコル・オプションではデバイスから HotSync Manager または HotSync Server への接続は制御*されません*。代わりに、HotSync マネージャまたはサーバ側で動作している Mobile Link コンジットから Mobile Link 同期サーバへの接続を指定します。

引数の形式は次のとおりです。

`stream=stream_name;`*conduit_stream_parameters*

stream_name は Mobile Link 同期サーバと通信するときにコンジットが使用するプロトコルを示します。次のいずれかを指定できます。

- **tcpip**
- **http**
- **https**

conduit_stream_parameters は、コンジットが使用するプロトコル・オプションのセットで、使用するプロトコルの **stream_parms** 引数と同じ形式です。指定されたストリームに対して、*conduit_stream_parameters* は、プロトコルの **stream_parms** 引数と同じデフォルトを使用します。*stream_name* のデフォルト値は tcpip です。

たとえば、次の Embedded SQL コードでは HTTP 同期ストリームを使用します。

```
ULInitSynchInfo( &info );
 info.stream_parms = "stream=http";
```

conduit_stream_parameters の詳細については、「TCP/IP プロトコル・オプション」438 ページ、「HTTP プロトコル・オプション」439 ページ、および 「HTTPS プロトコル・オプション」441 ページを参照してください。

Null 値とデフォルト設定

HotSync 同期を使用して、プロトコル・オプションを指定しない場合、コンジットはレジストリでプロトコル名とオプションを探します。有効なネットワーク・プロトコルが見つからないと、デフォルトのプロトコルとプロトコル・オプションが使用されます。このデフォルト・ストリーム・パラメータの設定は、次のとおりです。

```
stream=tcpip;host=localhost
```

HotSync 同期への暗号化の追加

ストリームに Certicom 暗号化を追加するには、デスクトップ・マシン上のファイルにルート証明書を入れてください。これは、暗号化情報が **security** 同期パラメータに埋め込まれる他の Ultra Light アプリケーションの場合とは異なります。

プロトコル・オプションは、Adaptive Server Anywhere Mobile Link クライアントの場合と同じように指定する必要があります。このフォーマットは、次のとおりです。

security=*cipher*{ *keyword*=*value*;… }

cipher には certicom_tls を指定し、キーワードは次のリストから指定します。

- **certificate_company**　証明書の組織フィールド
- **certificate_unit**　証明書の組織単位フィールド
- **certificate_name**　証明書の共通名フィールド
- **trusted_certificates**　信用された証明書の場所

たとえば、静的型 C++ アプリケーションでは次のとおりです。

```
info.stream_parms =
"stream=tcpip;security=ecc_tls(trusted_certificates=tru
sted.crt)";
```

詳細については、次の項を参照してください。

- 「certificate_company」45 ページ
- 「certificate_name」47 ページ
- 「certificate_unit」49 ページ
- 「trusted_certificates」68 ページ

TCP/IP プロトコル・オプション

TCP/IP 同期ストリームには、すべての Ultra Light インタフェースからアクセスできます。

TCP/IP 同期ストリームの選択

同期ストリームとして TCP/IP を選択するには、次のようにします。

- Ultra Light for MobileVB と Ultra Light ActiveX の場合は、ULSyncParms.Stream として ULStreamType 列挙から ulTCPIP を選択します。

  詳細については、『Ultra Light for MobileVB ユーザーズ・ガイド』> 「ULSyncParms クラス」 と 『Ultra Light ActiveX ユーザーズ・ガイド』> 「ULSyncParms クラス」を参照してください。

- Native Ultra Light for Java の場合は、StreamType.TCPIP を SyncParms.setStream() の引数として指定します。

  詳細については、『Native UltraLite for Java API Reference』の「**ianywhere.native_ultralite.StreamType**」と「**ianywhere.native_ultralite.SyncParms**」を参照してください。

- Embedded SQL または静的型 C++ API の場合は、ULSocketStream() をストリーム同期パラメータの引数として指定します。

  詳細については、『Ultra Light C/C++ ユーザーズ・ガイド』> 「ULSocketStream 関数」を参照してください。

- 静的型 Java API では、UlSynchOptions.setStream() の引数として UlSocketStream を指定します。次に例を示します。

```
UlSynchOptions opts = new UlSynchOptions;
 opts.setStream(new UlSocketStream() );
```

詳細については、『Ultra Light 静的型 Java ユーザーズ・ガイド』> 「stream 同期パラメータ」を参照してください。

プロトコル・オプション

TCP/IP プロトコルを使用する場合、次のプロトコル・オプションから選択できます。

パラメータ	詳細の参照先
client_port= *nnnnn*[**-** *mmmmm*]	「client_port」51 ページ
host=*hostname*	「host」52 ページ
liveness_timeout=*n*	「liveness_timeout」58 ページ
network_connect_timeout=*seconds*	「network_connect_timeout」59 ページ
network_leave_open={**0**|**1**}	「network_leave_open」59 ページ
network_name=*name*	「network_name」60 ページ
port=*portnumber*	「port」62 ページ

HTTP プロトコル・オプション

HTTP 同期ストリームには、すべての Ultra Light コンポーネントからアクセスできます。

HTTP 同期ストリームの選択

HTTPS を同期ストリームとして選択するには、次のようにします。

- Ultra Light for MobileVB と Ultra Light ActiveX の場合は、ULSyncParms.Stream として ULStreamType 列挙から ulHTTP を選択します。

  詳細については、『Ultra Light for MobileVB ユーザーズ・ガイド』> 「ULSyncParms クラス」を参照してください。

- Native Ultra Light for Java の場合は、StreamType.HTTP を SyncParms.setStream() の引数として指定します。

  詳細については、『Native UltraLite for Java API Reference』の「**ianywhere.native_ultralite.StreamType**」と「**ianywhere.native_ultralite.SyncParms**」を参照してください。

- Embedded SQL または静的型 C++ API の場合は、ULHTTPStream() をストリーム同期パラメータの引数として指定します。

  詳細については、『Ultra Light C/C++ ユーザーズ・ガイド』>「ULHTTPStream 関数」を参照してください。

プロトコル・オプション

HTTP ストリームを使用する場合、次のプロトコル・オプションから選択できます。

パラメータ	詳細の参照先
buffer_size=*nnnn*	「buffer_size」44 ページ
client_port= *nnnnn*[**-***mmmmm*]	「client_port」51 ページ
custom_header=*header*	「custom_header」51 ページ
host=*hostname*	「host」52 ページ
network_connect_timeout=*seconds*	「network_connect_timeout」59 ページ
network_leave_open={**0**|**1**}	「network_leave_open」59 ページ
network_name=*name*	「network_name」60 ページ
persistent={**0**|**1**}	「persistent」61 ページ
port=*portnumber*	「port」62 ページ
proxy_host=*proxy_hostname*	「proxy_host」63 ページ

パラメータ	詳細の参照先
proxy_port= *proxy_portnumber*	「proxy_port」63 ページ
set_cookie=*cookie_name***=** *cookie_value*	「set_cookie」67 ページ
url_suffix=*suffix*	「url_suffix」70 ページ
version=*versionnumber*	「version」71 ページ

参照

『Mobile Link 管理ガイド』> 「Mobile Link クライアントとサーバのリダイレクタ設定」

HTTPS プロトコル・オプション

HTTPS 同期ストリームには、すべての Ultra Light コンポーネントからアクセスできます。

HTTPS 同期ストリームの選択

HTTPS を同期ストリームとして選択するには、次のようにします。

- Ultra Light for MobileVB と Ultra Light for eMbedded Visual Basic の場合は、ULSyncParms.Stream として ULStreamType 列挙から ulHTTPS を選択します。

  詳細については、『Ultra Light for MobileVB ユーザーズ・ガイド』> 「ULSyncParms クラス」を参照してください。

- Native Ultra Light for Java の場合は、StreamType.HTTPS を SyncParms.setStream() の引数として指定します。

  詳細については、『Native UltraLite for Java API Reference』の「**ianywhere.native_ultralite.StreamType**」と「**ianywhere.native_ultralite.SyncParms**」を参照してください。

- Embedded SQL または静的型 C++ API の場合は、ULHTTPSStream() をストリーム同期パラメータの引数として指定します。

詳細については、『Ultra Light C/C++ ユーザーズ・ガイド』> 「ULHTTPSStream 関数」を参照してください。

別途ライセンスを取得できるオプションが必要
トランスポート・レイヤ・セキュリティを使用するには、別途ライセンスを取得できる SQL Anywhere Studio セキュリティ・オプションを入手する必要があります。このセキュリティ・オプションは、輸出規制対象品目です。

このコンポーネントのご注文については、『SQL Anywhere Studio の紹介』> 「別途ライセンスが入手可能なコンポーネント」を参照してください。

プロトコル・オプション

HTTPS ストリームを使用する場合、次のプロトコル・オプションから選択できます。

パラメータ	詳細の参照先
buffer_size=*nnnn*	「buffer_size」44 ページ
certificate_company=*company_name*	「certificate_company」45 ページ
certificate_name=*name*	「certificate_name」47 ページ
certificate_unit=*company_unit*	「certificate_unit」49 ページ
client_port= *nnnnn*[**-***mmmmm*]	「client_port」51 ページ
custom_header=*header*	「custom_header」51 ページ
host=*hostname*	「host」52 ページ
network_connect_timeout=*seconds*	「network_connect_timeout」59 ページ
network_leave_open={**0**|**1**}	「network_leave_open」59 ページ
network_name=*name*	「network_name」60 ページ

パラメータ	詳細の参照先
persistent={**0**|**1**}	「persistent」61 ページ
port=*portnumber*	「port」62 ページ
proxy_host=*proxy_hostname*	「proxy_host」63 ページ
proxy_port=*proxy_portnumber*	「proxy_port」63 ページ
set_cookie=*cookie_name***=***cookie_value*	「set_cookie」67 ページ
trusted_certificates=*filename*	「trusted_certificates」68 ページ
url_suffix=*suffix*	「url_suffix」70 ページ
version=*versionnumber*	「version」71 ページ

参照

- 『Mobile Link 管理ガイド』> 「トランスポート・レイヤ・セキュリティを使用する Ultra Light クライアントの設定」
- 『Mobile Link 管理ガイド』> 「Mobile Link クライアントとサーバのリダイレクタ設定」

UlSecureRSASocketStream 同期パラメータ

Certicom RSA 暗号化を使用したトランスポート・レイヤ・セキュリティに静的型 Java アプリケーションからアクセスするには、UlSecureRSASocketStream オブジェクトを使用し、別個のストリームとしてアクセスします。これは、同期構造体に別個のパラメータを指定する他の Ultra Light アプリケーションとは異なる動作です。

別途ライセンスを取得できるオプションが必要

トランスポート・レイヤ・セキュリティを使用するには、別途ライセンスを取得できる SQL Anywhere Studio セキュリティ・オプションを入手する必要があります。このセキュリティ・オプションは、輸出規制対象品目です。

このコンポーネントのご注文については、『SQL Anywhere Studio の紹介』> 「別途ライセンスが入手可能なコンポーネント」を参照してください。

プロトコル・オプション

UlSecureSocketStream のパラメータは、セミコロンで区切られた文字列形式で指定します。UlSecureRSASocketStream を使用する場合、次のプロトコル・オプションから選択できます。

パラメータ	詳細の参照先
certificate_company=*company_name*	「certificate_company」45 ページ
certificate_name=*name*	「certificate_name」47 ページ
certificate_unit=*company_unit*	「certificate_unit」49 ページ
client_port= *nnnnn*[**-***mmmmm*]	「client_port」51 ページ
host=*hostname*	「host」52 ページ
liveness_timeout=*n*	「liveness_timeout」58 ページ
port=*portnumber*	「port」62 ページ

UlSecureSocketStream 同期パラメータ

Certicom 楕円曲線暗号化を使用したトランスポート・レイヤ・セキュリティに静的型 Java アプリケーションからアクセスするには、UlSecureSocketStream オブジェクトを使用し、別個のストリームとしてアクセスします。

別途ライセンスを取得できるオプションが必要
トランスポート・レイヤ・セキュリティを使用するには、別途ライセンスを取得できる SQL Anywhere Studio セキュリティ・オプションを入手する必要があります。このセキュリティ・オプションは、輸出規制対象品目です。

このコンポーネントのご注文については、『SQL Anywhere Studio の紹介』> 「別途ライセンスが入手可能なコンポーネント」を参照してください。

詳細については、『Ultra Light 静的型 Java ユーザーズ・ガイド』> 「stream 同期パラメータ」と『Ultra Light 静的型 Java ユーザーズ・ガイド』> 「トランスポート・レイヤ・セキュリティ」を参照してください。

プロトコル・オプション

UlSecureSocketStream のパラメータは、セミコロンで区切られた文字列形式で指定します。UlSecureRSASocketStream を使用する場合、次のプロトコル・オプションから選択できます。

パラメータ	詳細の参照先
certificate_company=*company_name*	「certificate_company」45 ページ
certificate_name=*name*	「certificate_name」47 ページ
certificate_unit=*company_unit*	「certificate_unit」49 ページ
client_port= *nnnnn*[**-***mmmmm*]	「client_port」51 ページ
host=*hostname*	「host」52 ページ
liveness_timeout=*n*	「liveness_timeout」58 ページ
port=*portnumber*	「port」62 ページ

索引

記号

#hook_dict テーブル
dbmlsync 229
説明 231
.NET
Mobile Link のユーザ認証 27
@@identity グローバル変数
Ultra Light での使用 369
@data オプション
Mobile Link [dbmlsync] 129

A

a_dbtools_info 構造体
初期化 336
a_sync_db 構造体
概要 334
初期化 337
a_syncpub 構造体
概要 334
ActiveSync
ASA クライアントへの Mobile Link プロバイダのインストール 110
ASA クライアントへのアプリケーションの登録 112
dbmlsync のクラス名 195
Mobile Link ActiveSync プロバイダ [dbasinst] 34
Mobile Link ASA クライアント 108
Mobile Link ASA クライアント用の CREATE SYNCHRONIZATION USER 文 109
Mobile Link Ultra Light アプリケーションの配備 394
Mobile Link プロバイダのインストール 34
Ultra Light クライアントへの Mobile Link プロバイダのインストール 391
Ultra Light クライアントへのアプリケーションの登録 393
Ultra Light での設定 433
Ultra Light のトランスポート・レイヤ・セキュリティ 434
通信オプション 141
ActiveSync オプション
Mobile Link 141
ActiveSync で使用するアプリケーションの登録
Mobile Link Ultra Light クライアント 393
ActiveSync 同期の使用
Mobile Link ASA クライアント 108
ActiveSync プロトコル・オプション
Ultra Light Mobile Link クライアント 433
ActiveSync プロバイダ・インストール・ユーティリティ [dbasinst]
構文 34
ActiveSync 用 Adaptive Server Anywhere クライアントの登録 112
ActiveSync 用 Mobile Link プロバイダのインストール
ASA クライアント 110
Ultra Light クライアント 391
ActiveSync 用の Adaptive Server Anywhere リモート・データベースの設定 109
ActiveSync を使用するアプリケーションの配備
Ultra Light クライアント 394
ActiveX
Mobile Link dbmlsync 統合コンポーネント 297
Adaptive Server Anywhere
Mobile Link クライアントとしての使用 4
Adaptive Server Anywhere クライアント

ActiveSync の登録 112
dbmlsync 124
Mobile Link の説明 75
概要 4
Adaptive Server Anywhere クライアントの同期パラメータ
リスト 123
Adaptive Server Anywhere バージョン 7 Mobile Link クライアント 117
Adaptive Server Anywhere リモート・データベース
Mobile Link の説明 75
ADDRESS 句
CREATE SYNCHRONIZATION USER (Mobile Link) 213
adr dbmlsync 拡張オプション
説明 138
allsync テーブル
Ultra Light データベース 355
ALTER PUBLICATION 文
SQL 構文 (Mobile Link) 200
ALTER SYNCHRONIZATION UBSCRIPTION 文
SQL 構文 (Mobile Link) 202
ALTER SYNCHRONIZATION USER 文
SQL 構文 (Mobile Link) 205
-ap オプション
Mobile Link [dbmlsync] 129
authenticate_user
事前に定義されたスクリプトの使用 28
説明 26
-a オプション
Mobile Link [dbmlsync] 129

B

-ba オプション
Mobile Link [dbmlsync] 130
-bc オプション
Mobile Link [dbmlsync] 130
BeginDownload イベント
dbmlsync 統合コンポーネント 312
BeginLogScan イベント
dbmlsync 統合コンポーネント 312
BeginSynchronization イベント
dbmlsync 統合コンポーネント 313
BeginUpload イベント
dbmlsync 統合コンポーネント 314
-be オプション
Mobile Link [dbmlsync] 131
-bg オプション
Mobile Link [dbmlsync] 131
buffer_size ストリーム・パラメータ
HTTPS による Ultra Light 同期 441
HTTP による Ultra Light 同期 439
buffer_size プロトコル・オプション
Mobile Link クライアント 44

C

cd dbmlsync 拡張オプション
説明 145
Certicom
Ultra Light のトランスポート・レイヤ・セキュリティ 415, 416
certificate_company プロトコル・オプション
Mobile Link クライアント 45
certificate_name プロトコル・オプション
Mobile Link クライアント 47
certificate_unit プロトコル・オプション
Mobile Link クライアント 49
checkpoint_store 同期パラメータ
Ultra Light の同期パラメータ 401
Checkpoint Store
Ultra Light 同期パラメータ 401
client_port プロトコル・オプション
Mobile Link クライアント 51
ColumnCount プロパティ
dbmlsync 統合コンポーネント 332
ColumnName
dbmlsync 統合コンポーネント 329
ColumnValue プロパティ
dbmlsync 統合コンポーネント 330
commit_state カラム

説明　7
COMMIT 文
イベント・フック・プロシージャ　228
CommunicationAddress dbmlsync 拡張オプション
説明　138
CommunicationType dbmlsync 拡張オプション
説明　143
ConflictRetries dbmlsync 拡張オプション
説明　144
ConflictRetries 同期オプション
ASA クライアント　105
ConnectMobilink イベント
dbmlsync 統合コンポーネント　314
ContinueDownload dbmlsync 拡張オプション
説明　145
cr dbmlsync 拡張オプション
説明　144
CREATE PUBLICATION 文
SQL 構文 (Mobile Link)　207
CREATE SYNCHRONIZATION SUBSCRIPTION 文
Mobile Link ASA クライアント用の ActiveSync　109
SQL 構文 (Mobile Link)　210
CREATE SYNCHRONIZATION USER 文
Mobile Link ASA クライアント用の ActiveSync　109
SQL 構文 (Mobile Link)　213
ctp dbmlsync 拡張オプション
説明　143
custom_header プロトコル・オプション
Mobile Link クライアント　51
-c オプション
Mobile Link [dbmlsync]　132

D

dbasdesk.dll
インストール　34
dbasdev.dll
インストール　34
dbasinst ユーティリティ
ASA クライアントへの ActiveSync 用 Mobile Link プロバイダのインストール　110
Ultra Light への ActiveSync 用 Mobile Link プロバイダのインストール　391
オプション　34
構文　34
dbcond9 ユーティリティ
Ultra Light における配備　376
Ultra Light の HotSync コンジット　376
dbhsync9.dll
Ultra Light の HotSync コンジット　375
dblgen9.dll
Ultra Light における HotSync コンジットの配備　375
dbmlhttp9.dll
Ultra Light アプリケーションの配備　375
dbmlhttps9.dll
Ultra Light アプリケーションの配備　375
dbmlsock9.dll
Ultra Light アプリケーションの配備　375
dbmlsynccom.dll
dbmlsync 統合コンポーネント　298
dbmlsynccomg.dll
dbmlsync 統合コンポーネント　298
dbmlsync オプション
アルファベット順のリスト　129
dbmlsync 拡張オプション　135
dbmlsync 統合コンポーネント
IRowTransfer インタフェース　328
イベント　312
サポートされるプラットフォーム　298
設定　299
説明　297
プロパティ　306
メソッド　304
dbmlsync の DBTools インタフェース
説明　333
dbmlsync の DBTools インタフェースの設定
説明　336
dbmlsync のデータベース・ツール・インタフェース

説明　333
dbmlsync ユーティリティ
#hook_dict テーブル　229
DBTools インタフェース　333
Mobile Link ASA クライアント用の ActiveSync　108
Mobile Link 同期のカスタマイズ　226
sp_hook_dbmlsync_abort フック　234
sp_hook_dbmlsync_begin　237
sp_hook_dbmlsync_connect_failed　239
sp_hook_dbmlsync_delay　243
sp_hook_dbmlsync_download_begin　246
sp_hook_dbmlsync_download_com_error　248
sp_hook_dbmlsync_download_end　250
sp_hook_dbmlsync_download_fatal_sql_error　252
sp_hook_dbmlsync_download_log_ri_violation　254
sp_hook_dbmlsync_download_ri_violation　257
sp_hook_dbmlsync_download_sql_error　260
sp_hook_dbmlsync_download_table_begin　263
sp_hook_dbmlsync_download_table_end　265
sp_hook_dbmlsync_end　268
sp_hook_dbmlsync_log_rescan　271
sp_hook_dbmlsync_logscan_begin　273
sp_hook_dbmlsync_logscan_end　275
sp_hook_dbmlsync_process_return_code　278
sp_hook_dbmlsync_schema_upgrade　281
sp_hook_dbmlsync_set_extended_options　284
sp_hook_dbmlsync_upload_begin　286
sp_hook_dbmlsync_upload_end　288
sp_hook_dbmlsync_validate_download_file　293
アプリケーションからの同期の開始　106
イベント・フック　225
エラー処理イベント・フック　228
オプション　124
拡張オプション　135
構文　124
使用　102
進行オフセット　79
説明　123
ダイヤルアップ　138
統合コンポーネント　297
同時実行性　105
トランザクション・ログ　104
バージョン 7 クライアントの使用　117
パーミッション　102
パスワード　24
パスワードの変更　25
複数のユーザ　102
dbmltls9.dll
Ultra Light アプリケーションの配備　375
dbmluser ユーティリティ
使用　24
dbs dbmlsync 拡張オプション
説明　147
dbser9.dll
Ultra Light アプリケーションの配備　375
DBSynchronizeLog 関数
概要　335
dbtools.h
ASA クライアントの同期　106
DBToolsFini 関数
使用　344
DBToolsInit 関数
dbtools の起動　336
DBTools インタフェース　vii
dbmlsync　333
dbmlsync の設定　336
-dc オプション
Mobile Link [dbmlsync]　133
DetailedInfoMessageEnabled プロパティ
dbmlsync 統合コンポーネント　309
dir dbmlsync 拡張オプション
説明　163
Disable Concurrency
Ultra Light 同期パラメータ　402
DisablePolling dbmlsync 拡張オプション
説明　146
DisconnectMobilink イベント
dbmlsync 統合コンポーネント　315
DispatchChannelSize プロパティ
dbmlsync 統合コンポーネント　311
dllapi.h
dbmlsync の DBTools インタフェース　341
-dl オプション

Mobile Link [dbmlsync] 134
DoneExecution イベント
dbmlsync 統合コンポーネント 316
download_only 同期パラメータ
Ultra Light 403
DownloadBufferSize dbmlsync 拡張オプション
説明 147
DownloadEventsEnabled プロパティ
dbmlsync 統合コンポーネント 307
Download Only
Ultra Light 同期パラメータ 403
DownloadOnly dbmlsync 拡張オプション
説明 148
DownloadReadSize dbmlsync 拡張オプション
説明 150
DownloadRow イベント
dbmlsync 統合コンポーネント 316
DROP PUBLICATION 文
SQL 構文 (Mobile Link) 216
説明 90
DROP SYNCHRONIZATION SUBSCRIPTION 文
SQL 構文 (Mobile Link) 217
説明 100
DROP SYNCHRONIZATION USER 文
SQL 構文 (Mobile Link) 219
drs dbmlsync 拡張オプション
説明 150
ds dbmlsync 拡張オプション
説明 148
-ds オプション
Mobile Link [dbmlsync] 134
-d オプション
Mobile Link [dbasinst] 34
Mobile Link [dbmlsync] 133

E

-e adr
dbmlsync 拡張オプション 138
ECC
Mobile Link ASA クライアントの暗号を解く鍵 64
ecc_tls
Mobile Link ASA クライアントの暗号を解く鍵 64
-e cd
dbmlsync 拡張オプション 145
-e CommunicationAddress
dbmlsync 拡張オプション 138
-e CommunicationType
dbmlsync 拡張オプション 143
-e ConflictRetries
dbmlsync 拡張オプション 144
-e ContinueDownload
dbmlsync 拡張オプション 145
-e cr
dbmlsync 拡張オプション 144
-e ctp
dbmlsync 拡張オプション 143
-e dbs
dbmlsync 拡張オプション 147
-e dir
dbmlsync 拡張オプション 163
-e DisablePolling
dbmlsync 拡張オプション 146
-e DownloadBufferSize
dbmlsync 拡張オプション 147
-e DownloadOnly
dbmlsync 拡張オプション 148
-e DownloadReadSize
dbmlsync 拡張オプション 150
-e drs
dbmlsync 拡張オプション 150
-e ds
dbmlsync 拡張オプション 148
-e eh
dbmlsync 拡張オプション 154
-e el
dbmlsync 拡張オプション 151
-e ErrorLogSendLimit
dbmlsync 拡張オプション 151
-e FireTriggers
dbmlsync 拡張オプション 152

-e ft
dbmlsync 拡張オプション 152
eh dbmlsync 拡張オプション
説明 154
-e HoverRescanThreshold
dbmlsync 拡張オプション 153
-e hrt
dbmlsync 拡張オプション 153
-eh オプション
Mobile Link [dbmlsync] 179
-e IgnoreHookErrors
dbmlsync 拡張オプション 154
-e IgnoreScheduling
dbmlsync 拡張オプション 155
-e inc
dbmlsync 拡張オプション 156
-e Increment
dbmlsync 拡張オプション 156
-e isc
dbmlsync 拡張オプション 155
-ek オプション
Mobile Link [dbmlsync] 180
el dbmlsync 拡張オプション
説明 151
-e LockTables
dbmlsync 拡張オプション 157
-e lt
dbmlsync 拡張オプション 157
-e mem
dbmlsync 拡張オプション 159
-e Memory
dbmlsync 拡張オプション 159
-e MirrorLogDirectory
dbmlsync 拡張オプション 159
-e mld
dbmlsync 拡張オプション 159
-e mn
dbmlsync 拡張オプション 162
-e MobiLinkPwd
dbmlsync 拡張オプション 161
-e mp
dbmlsync 拡張オプション 161
EndDownload イベント
dbmlsync 統合コンポーネント 317
EndLogScan イベント
dbmlsync 統合コンポーネント 318
EndSynchronization イベント
dbmlsync 統合コンポーネント 319
EndUpload イベント
dbmlsync 統合コンポーネント 320
-e NewMobiLinkPwd
dbmlsync 拡張オプション 162
-e OfflineDirectory
dbmlsync 拡張オプション 163
-e p
dbmlsync 拡張オプション 146
-e PollingPeriod
dbmlsync 拡張オプション 163
-e pp
dbmlsync 拡張オプション 163
-ep オプション
Mobile Link [dbmlsync] 180
ErrorLogSendLimit dbmlsync 拡張オプション
説明 151
ErrorMessageEnabled プロパティ
dbmlsync 統合コンポーネント 307
-e sa
dbmlsync 拡張オプション 169
-e sch
dbmlsync 拡張オプション 164
-e Schedule
dbmlsync 拡張オプション 164
-e scn
dbmlsync 拡張オプション 168
-e ScriptVersion
dbmlsync 拡張オプション 167
-e SendColumnNames
dbmlsync 拡張オプション 168
-e SendDownloadACK
dbmlsync 拡張オプション 169
-e SendTriggers
dbmlsync 拡張オプション 170
-e st
dbmlsync 拡張オプション 170

-e sv
dbmlsync 拡張オプション 167
-e TableOrder
dbmlsync 拡張オプション 170
-e tor
dbmlsync 拡張オプション 170
-e uo
dbmlsync 拡張オプション 172
-e UploadOnly
dbmlsync 拡張オプション 172
-eu オプション
Mobile Link [dbmlsync] 180
-e v
dbmlsync 拡張オプション 172
EventChannelSize プロパティ
dbmlsync 統合コンポーネント 311
-e Verbose
dbmlsync 拡張オプション 172
-e VerboseHooks
dbmlsync 拡張オプション 173
-e VerboseMin
dbmlsync 拡張オプション 174
-e VerboseOptions
dbmlsync 拡張オプション 175
-e VerboseRowCounts
dbmlsync 拡張オプション 176
-e VerboseRowValues
dbmlsync 拡張オプション 177
-e VerboseUpload
dbmlsync 拡張オプション 178
-e vm
dbmlsync 拡張オプション 174
-e vn
dbmlsync 拡張オプション 176
-e vo
dbmlsync 拡張オプション 175
-e vr
dbmlsync 拡張オプション 177
-e vs
dbmlsync 拡張オプション 173
-e vu
dbmlsync 拡張オプション 178
ExitCode プロパティ
dbmlsync 統合コンポーネント 310
-e オプション
Mobile Link [dbmlsync] 135

F

FIPS
Mobile Link ASA クライアント・ストリームの暗号化 64
ActiveSync 同期 141
dbmlsync の HTTPS 同期オプション 140
FireTriggers dbmlsync 拡張オプション
説明 152
ft dbmlsync 拡張オプション
説明 152

G

GetLastIdentity メソッド
Ultra Light 同期 369
getNewPassword メソッド
Ultra Light 406
getPassword メソッド
Ultra Light での同期 410
getScriptVersion メソッド
Ultra Light 431
getStream メソッド
Ultra Light 421
getUploadOK メソッド
Ultra Light 427
getUploadOnly メソッド
Ultra Light 428
getUserName メソッド
Ultra Light 430
GLOBAL_DATABASE_ID オプション
Ultra Light での設定 367

H

host プロトコル・オプション

Mobile Link クライアント 52
HotSync Manager への Mobile Link HotSync コンジットの登録
Ultra Light クライアント 376
HotSync コンジット
Ultra Light における設定 379
Ultra Light におけるテスト 378
HotSync 設定の概要
Ultra Light クライアント 374
HotSync 同期
Ultra Light クライアント 373
Ultra Light 内のアーキテクチャ 373
Ultra Light の Palm Computing Platform 374
Ultra Light のトランスポート・レイヤ・セキュリティ 437
設定 435
HotSync 同期の概要
Ultra Light クライアント 373
HoverRescanThreshold dbmlsync 拡張オプション
説明 153
hrt dbmlsync 拡張オプション
説明 153
HTTP
HTTP を使用した Mobile Link ASA クライアント 138
HTTP を使用した Mobile Link Ultra Light クライアント 439
Ultra Light での同期 439
http_password プロトコル・オプション
Mobile Link クライアント 53
http_proxy_password プロトコル・オプション
Mobile Link クライアント 54
http_proxy_userid プロトコル・オプション
Mobile Link クライアント 56
http_userid プロトコル・オプション
Mobile Link クライアント 57
HTTPS
HTTPS を使用した Mobile Link ASA クライアント 140
HTTPS を使用した Mobile Link Ultra Light クライアント 441
HTTPS_FIPS
HTTPS_FIPS を使用した Mobile Link ASA クライアント 140
HTTPS_FIPS 同期
dbmlsync のオプション 140
HTTPS 同期
dbmlsync のオプション 140
Palm Computing Platform と Ultra Light 383
HTTP ストリーム・パラメータ
Ultra Light 同期 439, 441
HTTP 同期
dbmlsync のオプション 138
Palm Computing Platform と Ultra Light 383

I

ignored_rows 同期パラメータ
Ultra Light での同期 404
Ignored Rows
Ultra Light 同期パラメータ 404
IgnoreHookErrors dbmlsync 拡張オプション
説明 154
IgnoreScheduling dbmlsync 拡張オプション
説明 155
IMAP サーバ
Mobile Link 認証 28
inc dbmlsync 拡張オプション
説明 156
Increment dbmlsync 拡張オプション
説明 156
InfoMessageEnabled プロパティ
dbmlsync 統合コンポーネント 308
IRowTransferData
dbmlsync 統合コンポーネント 328
isc dbmlsync 拡張オプション
説明 155
-is オプション
Mobile Link [dbmlsync] 181

J

Java
Mobile Link のユーザ認証 27

K

Keep Partial Download
Ultra Light 同期パラメータ 405
-k オプション
Mobile Link [dbasinst] 34
Mobile Link [dbmlsync] 181

L

LDAP サーバ
Mobile Link 認証 28
liveness_timeout プロトコル・オプション
Mobile Link クライアント 58
LockTables dbmlsync 拡張オプション
説明 157
LockTables 同期オプション
ASA クライアント 105
lt dbmlsync 拡張オプション
説明 157
-l オプション
Mobile Link [dbasinst] 34
Mobile Link [dbmlsync] 181

M

mem dbmlsync 拡張オプション
説明 159
Memory dbmlsync 拡張オプション
説明 159
Message イベント
dbmlsync 統合コンポーネント 321
MirrorLogDirectory dbmlsync 拡張オプション
説明 159
ml_user
説明 10
古いバージョン上への ASA クライアントのインストール 79
ml_username
説明 10
mld dbmlsync 拡張オプション
説明 159
mlxtract ユーティリティ
構文 38
mn dbmlsync 拡張オプション
説明 162
-mn オプション
Mobile Link [dbmlsync] 182
Mobile Link
ALTER PUBLICATION 文 200
ALTER SYNCHRONIZATION SUBSCRIPTION 文 202
ALTER SYNCHRONIZATION USER 文 205
ASA クライアント 75
ASA クライアントのスケジュール 115
CREATE PUBLICATION 文 207
CREATE SYNCHRONIZATION SUBSCRIPTION 文 210
CREATE SYNCHRONIZATION USER 文 213
dbmlsync イベント・フック 225
dbmlsync オプション 123
DROP PUBLICATION 文 216
DROP SYNCHRONIZATION SUBSCRIPTION 文 217
START SYNCHRONIZATION DELETE 文 220
STOP SYNCHRONIZATION DELETE 文 222
Ultra Light クライアント 347
クライアントの接続パラメータ 44
参照整合性違反のロギング 254
バージョン 7 で廃止された機能 117
フック 225
ユーザ 9
ユーティリティ 33
Mobile Link ActiveSync プロバイダ・インストール・ユーティリティ [dbasinst]
構文 34
Mobile Link HotSync コンジットが正しくイン

ストールされたことをチェックする
Ultra Light クライアント 378
Mobile Link HotSync コンジットの設定
Ultra Light クライアント 379
Mobile Link HotSync コンジットの配備
Ultra Light クライアント 381
Mobile Link クライアント・データベース抽出ユーティリティ [mlxtract]
構文 38
Mobile Link クライアントの紹介 3
Mobile Link サブスクリプションの削除
ASA クライアント 100
Mobile Link サブスクリプションの変更
ASA クライアント 99
Mobile Link 同期
ASA クライアント 75
ASA クライアントのスケジュール 115
Ultra Light クライアント 347
Mobile Link 同期クライアント
dbmlsync オプション 124
説明 123
Mobile Link 同期サーバ
Ultra Light の HotSync 374
Mobile Link 同期サブスクリプション
ASA クライアント 98
Mobile Link 同期ユーザのサブスクリプションの作成
ASA クライアント 98
Mobile Link のセキュリティ
新しいユーザ 22
カスタム・ユーザ認証 26
パスワード 20
パスワードの変更 25
ユーザ認証アーキテクチャ 16
ユーザ認証パスワード 24
ユーザ認証メカニズムの選択 14
ユーザの認証 9
Mobile Link のパフォーマンス
アップロードするロー数の推定 193
Mobile Link ユーザ
ALTER SYNCHRONIZATION USER 文 (Mobile Link) 205
ASA クライアントからの削除 97
ASA クライアントでの作成 92
ASA クライアントでのプロパティの設定 93
ASA クライアントへの追加 92
CREATE SYNCHRONIZATION USER 文 (Mobile Link) 213
DROP SYNCHRONIZATION USER 文 (Mobile Link) 219
作成 11
説明 9
同期の状態 7
名前の共有 12
[Mobile Link ユーザ作成] ウィザード
使用 92
Mobile Link ユーザの概要 10
Mobile Link ユーザの削除
ASA クライアント 97
Mobile Link ユーザの作成
ASA クライアントの説明 92
説明 11
Mobile Link ユーザの認証
説明 9
Mobile Link ユーザのプロパティの設定
ASA クライアント 93
Mobile Link ユーザ名
説明 10
命名規則 10
Mobile Link ユーザ名の共有
説明 12
Mobile Link ユーティリティ
Mobile Link ActiveSync プロバイダ [dbasinst] 34
Mobile Link クライアント・データベース抽出 [mlxtract] 38
MobiLinkPwd dbmlsync 拡張オプション
説明 161
mp dbmlsync 拡張オプション
説明 161
-mp オプション
Mobile Link [dbmlsync] 182
MSGQ_SHUTDOWN_REQUESTED
dbmlsync の DBTools インタフェース 341

MSGQ_SLEEP_THROUGH
dbmlsync の DBTools インタフェース 341
MSGQ_SHUTDOWN_REQUESTED
dbmlsync の DBTools インタフェース 341

N

network_connect_timeout プロトコル・オプション
Mobile Link クライアント 59
network_leave_open プロトコル・オプション
Mobile Link クライアント 59
network_name プロトコル・オプション
Mobile Link クライアント 60
new_password 同期パラメータ
Ultra Light 406
NewMobiLinkPwd dbmlsync 拡張オプション
説明 162
New Password
Ultra Light 同期パラメータ 406
Number of Authentication Parameters
Ultra Light 同期パラメータ 407
-n オプション
Mobile Link [dbasinst] 34
Mobile Link [dbmlsync] 183

O

observer
Ultra Light 同期パラメータ 408
OfflineDirectory dbmlsync 拡張オプション
説明 163
OPTION 句
CREATE SYNCHRONIZATION USER 142
-os オプション
Mobile Link [dbmlsync] 184
-ot オプション
Mobile Link [dbmlsync] 184
-o オプション
Mobile Link [dbmlsync] 183

P

Palm Computing Platform
Mobile Link 用の Ultra Light アプリケーション開発 372
Ultra Light による同期 382
Ultra Light の HotSync 同期 374
Ultra Light の TCP/IP 同期 382
パブリケーションの制限 351
Palm Computing Platform での Ultra Light データベースの同期 372
Palm Computing Platform の HTTP または HTTPS による同期を設定する
Ultra Light クライアント 383
Partial Download Retained
Ultra Light 同期パラメータ 409
Password
Ultra Light 同期パラメータ 410
path プロパティ
dbmlsync 統合コンポーネント 306
p dbmlsync 拡張オプション
説明 146
-pd オプション
Mobile Link [dbmlsync] 186
persistent ストリーム・パラメータ
HTTPS による Ultra Light 同期 441
HTTP による Ultra Light 同期 439
persistent プロトコル・オプション
Mobile Link クライアント 61
Ping
Ultra Light 同期パラメータ 411
pinging
Mobile Link 同期サーバ 186
-pi オプション
Mobile Link [dbmlsync] 186
PollingPeriod dbmlsync 拡張オプション
説明 163
POP3 サーバ
Mobile Link 認証 28
port ストリーム・パラメータ
HTTPS による Ultra Light 同期 441
HTTP による Ultra Light 同期 439
port プロトコル・オプション

Mobile Link クライアント 62
pp dbmlsync 拡張オプション
説明 163
-pp オプション
Mobile Link [dbmlsync] 187
ProgressIndex イベント
dbmlsync 統合コンポーネント 322
ProgressMessage イベント
dbmlsync 統合コンポーネント 323
proxy_hostname プロトコル・オプション
Mobile Link クライアント 63
proxy_host プロトコル・オプション
Mobile Link クライアント 63
proxy_portnumber プロトコル・オプション
Mobile Link クライアント 63
proxy_port プロトコル・オプション
Mobile Link クライアント 63
publication
Ultra Light 同期パラメータ 413
-p オプション
Mobile Link [dbmlsync] 185

Q

-q オプション
Mobile Link [dbmlsync] 188

R

RAS
Ultra Light Palm クライアントの設定 384
Ultra Light 同期 382
同期にシリアル・ポートを使用する場合の Ultra Light Palm クライアントの設定 386
同期にモデムを使用する場合の Ultra Light Palm クライアントの設定 384
-ra オプション
Mobile Link [dbmlsync] 188
-rb オプション
Mobile Link [dbmlsync] 188
Resume Partial Download
Ultra Light 同期パラメータ 414
ROLLBACK 文
イベント・フック・プロシージャ 228
RowOperation プロパティ
dbmlsync 統合コンポーネント 328
RSA
Mobile Link ASA クライアントの暗号を解く鍵 64
rsa_tls
Mobile Link ASA クライアントの暗号を解く鍵 64
rsa_tls_fips
Mobile Link ASA クライアントの暗号を解く鍵 64
run メソッド
dbmlsync 統合コンポーネント 304
-r オプション
Mobile Link [dbmlsync] 188

S

sa dbmlsync 拡張オプション
説明 169
sch dbmlsync 拡張オプション
説明 164
Schedule dbmlsync 拡張オプション
説明 164
scn dbmlsync 拡張オプション
説明 168
ScriptVersion dbmlsync 拡張オプション
説明 167
-sc オプション
Mobile Link [dbmlsync] 190
Security
Ultra Light 同期パラメータ 415
security_parms
Ultra Light の同期パラメータ 416
security_parms 同期パラメータ
Ultra Light 416
Security Parameters
Ultra Light 同期パラメータ 416
security プロトコル・オプション

Mobile Link クライアント　64
send_column_names 同期パラメータ
Ultra Light　418
send_download_ack 同期パラメータ
Ultra Light　419
SendColumnNames
dbmlsync 拡張オプション　168
Send Column Names
Ultra Light 同期パラメータ　418
SendColumnNames dbmlsync 拡張オプション
説明　168
SendDownloadACK dbmlsync 拡張オプション
説明　169
SendDownloadAcknowledgement
dbmlsync 拡張オプション　169
Send Download Acknowledgement
Ultra Light 同期パラメータ　419
SendTriggers dbmlsync 拡張オプション
説明　170
set_cookie プロトコル・オプション
Mobile Link クライアント　67
setNewPassword メソッド
Ultra Light　406
setObserver メソッド
Ultra Light　408
setPassword メソッド
Ultra Light　410
setPing メソッド
Ultra Light　411
setScriptVersion メソッド
Ultra Light　431
setStreamParms メソッド
Ultra Light　425
setStream メソッド
Ultra Light　421
setSynchPublication メソッド
Ultra Light　413
SetTitle イベント
dbmlsync 統合コンポーネント　323
setUploadOnly メソッド
Ultra Light　428
setUserData メソッド
Ultra Light　429
sp_hook_dbmlsync_abort ストアド・プロシージャ
SQL 構文　234
sp_hook_dbmlsync_begin ストアド・プロシージャ
SQL 構文　237
sp_hook_dbmlsync_connect_failed ストアド・プロシージャ
SQL 構文　239
sp_hook_dbmlsync_delay ストアド・プロシージャ
SQL 構文　243
sp_hook_dbmlsync_download_begin ストアド・プロシージャ
SQL 構文　246
sp_hook_dbmlsync_download_com_error ストアド・プロシージャ
SQL 構文　248
sp_hook_dbmlsync_download_end ストアド・プロシージャ
SQL 構文　250
sp_hook_dbmlsync_download_fatal_SQL_error ストアド・プロシージャ
SQL 構文　252
sp_hook_dbmlsync_download_log_ri_violation ストアド・プロシージャ
SQL 構文　254
sp_hook_dbmlsync_download_ri_violation ストアド・プロシージャ
SQL 構文　257
sp_hook_dbmlsync_download_sql_error ストアド・プロシージャ
SQL 構文　260
sp_hook_dbmlsync_download_table_begin ストアド・プロシージャ
SQL 構文　263
sp_hook_dbmlsync_download_table_end ストアド・プロシージャ
SQL 構文　265
sp_hook_dbmlsync_end ストアド・プロシージャ
SQL 構文　268

sp_hook_dbmlsync_log_rescan ストアド・プロシージャ
SQL 構文 271
sp_hook_dbmlsync_logscan_begin ストアド・プロシージャ
SQL 構文 273
sp_hook_dbmlsync_logscan_end ストアド・プロシージャ
SQL 構文 275
sp_hook_dbmlsync_process_return_code ストアド・プロシージャ
SQL 構文 278
sp_hook_dbmlsync_schema_upgrade ストアド・プロシージャ
SQL 構文 281
sp_hook_dbmlsync_set_extended_options ストアド・プロシージャ
SQL 構文 284
sp_hook_dbmlsync_upload_begin ストアド・プロシージャ
SQL 構文 286
sp_hook_dbmlsync_upload_end ストアド・プロシージャ
SQL 構文 288
sp_hook_dbmlsync_validate_download_file ストアド・プロシージャ
SQL 構文 293
SQL Anywhere Studio
マニュアル viii
SQL 文
ALTER PUBLICATION 構文 (Mobile Link) 200
ALTER SYNCHRONIZATION SUBSCRIPTION 構文 (Mobile Link) 202
ALTER SYNCHRONIZATION USER 構文 (Mobile Link) 205
CREATE PUBLICATION 構文 (Mobile Link) 207
CREATE SYNCHRONIZATION SUBSCRIPTION 構文 (Mobile Link) 210
CREATE SYNCHRONIZATION USER 構文 (Mobile Link) 213
DROP PUBLICATION 構文 (Mobile Link) 216
DROP SYNCHRONIZATION SUBSCRIPTION 構文 (Mobile Link) 217
DROP SYNCHRONIZATION USER 構文 (Mobile Link) 219
START SYNCHRONIZATION DELETE 構文 (Mobile Link) 220
STOP SYNCHRONIZATION DELETE 構文 (Mobile Link) 222
START SYNCHRONIZATION DELETE 文
SQL 構文 (Mobile Link) 220
st dbmlsync 拡張オプション
説明 170
STOP SYNCHRONIZATION DELETE 文
ASA クライアント 114
SQL 構文 (Mobile Link) 222
stop メソッド
dbmlsync 統合コンポーネント 304
stream_error 同期パラメータ
ul_stream_error 構造体 420
Ultra Light 420
stream_parms 同期パラメータ
HotSync による Ultra Light 同期 373, 379
Ultra Light 425
Ultra Light での設定 435
Stream Error
Ultra Light 同期パラメータ 420
Stream Parameters
Ultra Light 同期パラメータ 425
Stream Type
Ultra Light 同期パラメータ 421
SUBSCRIBE BY 句 (Mobile Link)
CREATE PUBLICATION 文 207
SUBSCRIBE BY 句
Ultra Light の制限 351
sv dbmlsync 拡張オプション
説明 167
Sybase Central
Ultra Light のパブリケーションの作成 352
Sync Result
Ultra Light 同期パラメータ 426
SYSSERVERS システム・テーブル
コンポーネント統合サービスのリモート・サーバ 214

T

TableName プロパティ
dbmlsync 統合コンポーネント 328
TableOrder dbmlsync 拡張オプション
説明 170
TCP/IP
TCP/IP を使用した Mobile Link ASA クライアント 138
TCP/IP を使用した Mobile Link Ultra Light クライアント 438
TCP/IP 同期
dbmlsync のオプション 138
Ultra Light の Palm Computing Platform 382
Ultra Light のパラメータ 438
TCP/IP、HTTP、HTTPS 同期の設定
Ultra Light クライアント 382
tor dbmlsync 拡張オプション
説明 170
trusted_certificates プロトコル・オプション
Mobile Link クライアント 68
-tu オプション
Mobile Link [dbmlsync] 190
TYPE 句
CREATE SYNCHRONIZATION USER 213

U

UL_DEBUG_CONDUIT_LOG 環境変数
Ultra Light における HotSync コンジットのトラブルシューティング 381
UL_DEBUG_CONDUIT 環境変数
Ultra Light におけるコンジットのトラブルシューティング 378
ul_stream_error 構造体
Ultra Light 420
UL_SYNC_ALL_PUBS マクロ
Ultra Light でのパブリケーション・マスク 413
UL_SYNC_ALL マクロ
Ultra Light でのパブリケーション・マスク 413
ULActiveSyncStream 関数
Ultra Light における同期ストリームの設定 422
ULConduitStream 関数
Ultra Light における同期ストリームの設定 422
ulecc.lib
ULSecureCerticomTLSStream での使用 415
ulecc9.dll
ULSecureCerticomTLSStream での使用 415
UlHTTPSStream オブジェクト
Ultra Light の Java 同期ストリーム 422
ULHTTPSStream 関数
Ultra Light における同期ストリームの設定 422
UlHTTPStream オブジェクト
Ultra Light の Java 同期ストリーム 422
ULHTTPStream 関数
Ultra Light における同期ストリームの設定 422
ulrsa.lib
ULSecureRSATLSStream での使用 415
ulrsa9.dll
ULSecureRSATLSStream での使用 415
ULSecureCerticomTLSStream 関数
セキュリティ 416
説明 415
ULSecureRSASocketStream オブジェクト
Ultra Light のパラメータ 443
ULSecureRSATLSStream 関数
Ultra Light におけるセキュリティ 416
説明 415
UlSecureSocketStream オブジェクト
Ultra Light の Java 同期ストリーム 422
Ultra Light のパラメータ 444
ULSocketStream オブジェクト
Ultra Light の Java 同期ストリーム 422
ULSocketStream 関数
Ultra Light における同期ストリームの設定 422
ULSynchronize ライブラリ関数 5
Ultra Light
Mobile Link クライアント 5

同期の概要　348
Ultra Light アプリケーション
Mobile Link クライアント　5
Ultra Light アプリケーションの同期
概要　348
Ultra Light アプリケーションへの同期の追加
説明　349
Ultra Light クライアント
Mobile Link の説明　347
概要　5
Ultra Light データベースへの非同期テーブルの組み込み　355
Ultra Light データベースへの読み込み専用テーブルの組み込み
Mobile Link アプリケーション　356
Ultra Light データベース用のパブリケーションの作成
Mobile Link アプリケーション　352
Ultra Light 同期ストリーム
Ultra Light クライアント　360
Ultra Light 同期パラメータ
説明　397
Ultra Light ネットワーク・プロトコル
Ultra Light クライアント　360
Ultra Light の setUserName メソッド
Ultra Light　430
uo dbmlsync 拡張オプション
説明　172
-uo オプション
Mobile Link [dbmlsync]　193
upload_ok 同期パラメータ
Ultra Light　427
upload_only 同期パラメータ
Ultra Light　428
UploadAck イベント
dbmlsync 統合コンポーネント　324
UploadEventsEnabled プロパティ
dbmlsync 統合コンポーネント　306
Upload OK
Ultra Light 同期パラメータ　427
Upload Only
Ultra Light 同期パラメータ　428
UploadOnly dbmlsync 拡張オプション
説明　172
UploadRow イベント
dbmlsync 統合コンポーネント　325
-urc オプション
Mobile Link [dbmlsync]　193
url_suffix プロトコル・オプション
Mobile Link クライアント　70
USB
Ultra Light における HotSync のサポート　372
user_data 同期パラメータ
Ultra Light　429
user_name 同期パラメータ
Ultra Light　430
User Data
Ultra Light 同期パラメータ　429
User Name
Ultra Light 同期パラメータ　430
UseVB6Types プロパティ
dbmlsync 統合コンポーネント　309
-u オプション
Mobile Link [dbasinst]　34
Mobile Link [dbmlsync]　192

V

-v+ オプション
Mobile Link [dbmlsync]　194
-vc オプション
Mobile Link [dbmlsync]　194
v dbmlsync 拡張オプション
説明　172
Verbose dbmlsync 拡張オプション
説明　172
VerboseHooks dbmlsync 拡張オプション
説明　173
VerboseMin dbmlsync 拡張オプション
説明　174
VerboseOptions dbmlsync 拡張オプション
説明　175
VerboseRowCounts dbmlsync 拡張オプション
説明　176

VerboseRowValues dbmlsync 拡張オプション
説明 177
VerboseUpload dbmlsync 拡張オプション
説明 178
Version
Ultra Light 同期パラメータ 431
version 同期パラメータ
Ultra Light 431
version プロトコル・オプション
Mobile Link クライアント 71
vm dbmlsync 拡張オプション
説明 174
vn dbmlsync 拡張オプション
説明 176
-vn オプション
Mobile Link [dbmlsync] 194
vo dbmlsync 拡張オプション
説明 175
-vo オプション
Mobile Link [dbmlsync] 194
-vp オプション
Mobile Link [dbmlsync] 194
vr dbmlsync 拡張オプション
説明 177
-vr オプション
Mobile Link [dbmlsync] 194
vs dbmlsync 拡張オプション
説明 173
-vs オプション
Mobile Link [dbmlsync] 194
vu dbmlsync 拡張オプション
説明 178
-vu オプション
Mobile Link [dbmlsync] 194
-v オプション
Mobile Link [dbasinst] 34
Mobile Link [dbmlsync] 194

W

WaitingForUploadAck イベント
dbmlsync 統合コンポーネント 326
WarningMessageEnabled プロパティ
dbmlsync 統合コンポーネント 308
-wc オプション
Mobile Link [dbmlsync] 195
WHERE 句
Mobile Link パブリケーション 86
Windows CE
dbmlsync アプリケーション 186
Mobile Link Ultra Light クライアント 391
Windows CE での Ultra Light データベースの同期
説明 391

X

-x オプション
Mobile Link [dbmlsync] 196

Z

-za オプション
dbmlsrv9 と Ultra Light アプリケーション 418
-ze オプション
dbmlsrv9 と Ultra Light アプリケーション 418

あ

アーティクル
Mobile Link ASA クライアントの削除 88
Mobile Link ASA クライアントの作成 82
Mobile Link ASA クライアントの追加 88
Mobile Link ASA クライアントの変更 88
Mobile Link 同期サブスクリプション 98
Ultra Light データベース 351
Ultra Light の制限 351
[アーティクル作成] ウィザード
Mobile Link での使用 88
アイコン
マニュアルで使用 xiii
新しいユーザ

Mobile Link のユーザ認証 22
新しいユーザからの同期
説明 22
アップロード・ストリーム
アップロード専用同期のための -uo オプション 193
アップロード専用同期
ASA リモート・データベース 172
dbmlsync -uo オプション 193
アップロードのみの同期
Ultra Light データベース 428, 354
Ultra Light 同期パラメータの upload_only 428
Ultra Light の getUploadOnly メソッド 428
Ultra Light の setUploadOnly メソッド 428
アプリケーションからの同期の開始
ASA クライアント 106
暗号
Mobile Link クライアントの security プロトコル・オプション 64
暗号化
HotSync 同期 379
Ultra Light のセキュリティ 349

い

一貫性 vii
イベント
dbmlsync 統合コンポーネント 312
イベント引数
ASA クライアント 231
イベント・フック
#hook_dict テーブル 231
ASA クライアントの同期処理のカスタマイズ 226
sp_hook_dbmlsync_abort 234
sp_hook_dbmlsync_begin 237, 246
sp_hook_dbmlsync_connect_failed 239
sp_hook_dbmlsync_delay 243
sp_hook_dbmlsync_download_begin 246
sp_hook_dbmlsync_download_com_error 248
sp_hook_dbmlsync_download_end 250
sp_hook_dbmlsync_download_fatal_SQL_error 252
sp_hook_dbmlsync_download_log_ri_violation 254
sp_hook_dbmlsync_download_sql_error 260
sp_hook_dbmlsync_download_table_begin 257, 263
sp_hook_dbmlsync_download_table_end 265
sp_hook_dbmlsync_end 268
sp_hook_dbmlsync_log_rescan 271
sp_hook_dbmlsync_logscan_begin 273
sp_hook_dbmlsync_logscan_end 275
sp_hook_dbmlsync_process_return_code 278
sp_hook_dbmlsync_set_extended_options 284
sp_hook_dbmlsync_upload_begin 281, 286
sp_hook_dbmlsync_upload_end 288
sp_hook_dbmlsync_validate_download_file 293
イベント引数 231
エラー処理 228
エラーの無視 233
コミット禁止 228
使用 228
接続 230
説明 225
致命的なエラー 230
同期イベント・フック 225
同期イベント・フックの順序 226
プロシージャの所有者 230
ロールバック禁止 228
イベント・フックの順序
ASA クライアント 226
イベント・フック・プロシージャ内でのエラーの無視
ASA クライアント 233
イベント・フック・プロシージャの使用
ASA クライアント 228
イベント・フック・プロシージャの所有者
ASA クライアント 230
イベント・フック・プロシージャ用の接続
ASA クライアント 230
インクリメンタル・アップロード
Mobile Link 同期 156
インストール
ASA クライアントへの ActiveSync 用 Mobile

Link プロバイダ　110
Ultra Light クライアントへの ActiveSync 用 Mobile Link プロバイダのインストール　391
インタフェース
dbmlsync の DBTools　333

う

ウィザード
Mobile Link の [アーティクル作成]　88
Mobile Link の [パブリケーション作成]　83
[Mobile Link ユーザ作成]　92
[ユーザの追加]　20

え

エンド・ユーザに対するパスワード入力の要求
Mobile Link　24

お

オートインクリメント　vii
オプション
dbmlsync　124
dbmlsync 拡張オプション　135
dbmlsync の拡張オプション　135
Mobile Link ActiveSync プロバイダ [dbasinst]　34
Mobile Link 拡張オプションの優先順位　95
Mobile Link クライアント [dbmlsync]　124
オフセット
Mobile Link ASA クライアント　79

か

開始
ASA クライアントの同期　102
Ultra Light アプリケーションからの Mobile Link 同期　5
外部キー循環
Ultra Light　358
拡張オプション
ASA クライアントでの設定　93
ASA クライアントの優先順位　95
dbmlsync　135
拡張オプションと接続パラメータの優先順位
ASA クライアント　95
カスタマイズ
ASA クライアントの同期処理　226
カスタム認証
Mobile Link クライアント　26
カスタム・ユーザ認証
Mobile Link クライアント　26
カラムワイズ分割
Mobile Link ASA クライアント　84

き

規則
表記　xi
既存のパブリケーションの変更
Mobile Link ASA クライアント　88
起動
Ultra Light アプリケーションからの Mobile Link 同期　5
キャッシュ・サイズ
dbmlsync アップロード・ストリーム　159

く

クライアント
ASA Mobile Link クライアント　75
dbmlsync　124
Mobile Link クライアントとしての ASA　4
Mobile Link としての Ultra Light アプリケーション　5
Ultra Light Mobile Link クライアント　347
クライアント・イベント・フック・プロ

シージャ
Mobile Link ASA クライアント 225
クライアント・データベース
Mobile Link dbmlsync のオプション 123
Mobile Link Ultra Light オプション 397
クライアント・データベース抽出ユーティリティ [mlxtract]
構文 38
クライアント同期処理のカスタマイズ
ASA クライアント 226
クライアントのネットワーク・プロトコルの指定
Mobile Link 6
クラス名
ActiveSync 195
グローバル・オートインクリメント vii
Ultra Light での使用 365
Ultra Light での設定 367
Ultra Light でのデフォルト設定 366
Ultra Light の不足範囲 370
グローバル・データベース識別子
Ultra Light の設定 367
グローバル・データベース識別子の設定
Mobile Link システム内の Ultra Light クライアント 367

こ

構文
Mobile Link ActiveSync プロバイダ [dbasinst] 34
Mobile Link dbmlsync イベント・フック 225
Mobile Link クライアント [dbmlsync] 124
Mobile Link クライアント・データベース抽出 [mlxtract] 38
Mobile Link 同期ユーティリティ 33
SQL 文 199
コマンド・ライン
dbmlsync の起動 124
コマンド・ライン・ユーティリティ
dbasinst コマンド・ライン構文 34
Mobile Link 同期 33
コンジット
Ultra Light アプリケーションの配備 375
Ultra Light 同期用のインストール 376
Ultra Light におけるテスト 378
Ultra Light における配備 381
Ultra Light の dbcond9.exe 374
Ultra Light の HotSync 同期 383
コンジット・インストール・ユーティリティ
Ultra Light 同期 376
コンポーネント
Mobile Link dbmlsync 統合コンポーネント 297

さ

サーバ・ストアド・プロシージャ
Mobile Link dbmlsync イベント・フック 225
再起動可能なダウンロード
dbmlsync -dc オプション 133
sp_hook_dbmlsync_end 268
Ultra Light の Keep Partial Download 405
Ultra Light の Partial Download Retained 409
Ultra Light の Resume Partial Download 414
最後に割り当てられた値の割り出し
Ultra Light プライマリ・キーの一意性 369
削除
ASA クライアントからの Mobile Link サブスクリプションの削除 100
ASA クライアントからの Mobile Link ユーザの削除 97
ASA クライアントのアップロードの停止 114
DROP PUBLICATION 文 (Mobile Link) 216
DROP SYNCHRONIZATION SUBSCRIPTION 文 (Mobile Link) 217
DROP SYNCHRONIZATION USER 文 (Mobile Link) 219
Mobile Link ASA クライアントからのアーティクルの削除 88
Mobile Link ASA クライアントからのパブリケーションの削除 90

START SYNCHRONIZATION DELETE 文 (Mobile Link) 220
STOP SYNCHRONIZATION DELETE 文 (Mobile Link) 222
削除同期の一時停止
ASA クライアント 114
作成
ASA クライアントでの Mobile Link ユーザの作成 92
ASA リモート・データベース 76
CREATE PUBLICATION 文 (Mobile Link) 207
CREATE SYNCHRONIZATION SUBSCRIPTION 文 (Mobile Link) 210
Mobile Link ASA クライアントのアーティクル 82
Mobile Link ASA クライアントのカラムワイズ分割を使用したパブリケーション 84
Mobile Link ASA クライアントのテーブル全体のパブリケーション 83
Mobile Link ASA クライアントのパブリケーション 82
Mobile Link ASA クライアントのローワイズ分割を使用したパブリケーション 86
Mobile Link クライアント・データベース 38
Mobile Link ユーザ 11
Ultra Light パブリケーション 352
サブスクリプション
ALTER SYNCHRONIZATION SUBSCRIPTION 文 (Mobile Link) 202
CREATE SYNCHRONIZATION SUBSCRIPTION 文 (Mobile Link) 210
DROP SYNCHRONIZATION SUBSCRIPTION 文 (Mobile Link) 217
Mobile Link ASA クライアント 98
サポート
ニュースグループ xvi
サポートされているネットワーク・プロトコル
Ultra Light 360
サポートされているプラットフォーム
Ultra Light クライアントの Mobile Link 同期 360
サポートされるプラットフォーム
dbmlsync 統合コンポーネント 298
参照整合性
Mobile Link 参照整合性違反の解決 254

し

システム・プロシージャ
Mobile Link dbmlsync イベント・フック 225
自動ダイヤル
Mobile Link クライアント 60
終了コード
dbmlsync [sp_hook_dbmlsync_abort] 234
dbmlsync [sp_hook_dbmlsync_process_return_code] 278
順序
同期イベント・フック 226
準備
Mobile Link 用リモート・データベース 77
冗長オプション
Mobile Link [dbmlsync] 194
冗長性
Mobile Link での設定 [dbmlsync] 194
使用できるデフォルト値の数の検出
Ultra Light プライマリ・キーの一意性 370
証明書フィールド
Mobile Link TLS certificate_company オプション 45
Mobile Link TLS certificate_name オプション 47
Mobile Link TLS certificate_unit オプション 49
証明書フィールドの確認
Mobile Link TLS certificate_company オプション 45
Mobile Link TLS certificate_name オプション 47
Mobile Link TLS certificate_unit オプション 49
シリアル・ポート接続の場合の RAS の設定
Ultra Light クライアント 386
進行オフセット
Mobile Link ASA クライアント 79

す

スイッチ
Mobile Link ActiveSync プロバイダ [dbasinst] 34
Mobile Link クライアント [dbmlsync] 124
スクリプトの自動化
Ultra Light での同期 418
スクリプト・バージョン
Ultra Light の getScriptVersion メソッド 431
Ultra Light の setScriptVersion メソッド 431
Ultra Light の version 同期パラメータ 431
スケジュール
dbmlsync では無視 155
Mobile Link ASA クライアント 115
Mobile Link スケジュール・オプションの構文 164
ステータス
Mobile Link ASA クライアント 79
ストアド・プロシージャ
Mobile Link dbmlsync イベント・フック 225
Mobile Link クライアント・プロシージャ 225
sp_hook_dbmlsync_abort SQL 構文 234
sp_hook_dbmlsync_begin SQL 構文 237
sp_hook_dbmlsync_connect_failed SQL 構文 239
sp_hook_dbmlsync_delay SQL 構文 243
sp_hook_dbmlsync_download_begin SQL 構文 246
sp_hook_dbmlsync_download_com_error SQL 構文 248
sp_hook_dbmlsync_download_end SQL 構文 250
sp_hook_dbmlsync_download_fatal_SQL_error SQL 構文 252
sp_hook_dbmlsync_download_log_ri_violation 254
sp_hook_dbmlsync_download_ri_violation 257
sp_hook_dbmlsync_download_sql_error SQL 構文 260
sp_hook_dbmlsync_download_table_begin SQL 構文 263
sp_hook_dbmlsync_download_table_end SQL 構文 265
sp_hook_dbmlsync_end SQL 構文 268
sp_hook_dbmlsync_log_rescan SQL 構文 271
sp_hook_dbmlsync_logscan_begin SQL 構文 273
sp_hook_dbmlsync_logscan_end SQL 構文 275
sp_hook_dbmlsync_process_return_code SQL 構文 278
sp_hook_dbmlsync_schema_upgrade SQL 構文 281
sp_hook_dbmlsync_set_extended_options SQL 構文 284
sp_hook_dbmlsync_upload_begin SQL 構文 286
sp_hook_dbmlsync_upload_end SQL 構文 288
sp_hook_dbmlsync_validate_download_file SQL 構文 293
ストリーム・パラメータ
Mobile Link クライアント 44
Ultra Light 同期 433

せ

セキュリティ
ASA クライアントの Mobile Link 同期 102
Mobile Link のカスタム・ユーザ認証 26
Mobile Link パスワードの変更 25
Mobile Link ユーザの認証 9
send_column_names 同期パラメータ 418
Ultra Light アプリケーション 416
Ultra Light での同期 349
Ultra Light の Certicom 415, 416
Ultra Light の security_parms 同期パラメータ 416
新しい Mobile Link ユーザ 22
ユーザ認証パスワード 24
接続
Mobile Link dbmlsync adr オプション 138
Mobile Link dbmlsync ctp オプション 143
Mobile Link dbmlsync -c オプション 132
Mobile Link Ultra Light Stream Type 同期パラメータ 421

Mobile Link Ultra Light プロトコル・オプション 433
Mobile Link クライアント 44
接続オプション
dbmlsync 138
接続パラメータ
Mobile Link ASA クライアントの優先順位 95
Mobile Link クライアント 44
Ultra Light 同期 433
設定
ActiveSync 用の ASA リモート・データベース 109
ASA クライアントの Mobile Link ユーザのプロパティ 93

た

ダイヤルアップ
dbmlsync 接続 138
Mobile Link クライアント・プロトコル・オプション 44
Ultra Light 接続 433
ダイヤルアップ・ネットワーキング
Ultra Light 同期 382
同期にシリアル・ポートを使用する場合の Ultra Light Palm クライアントの設定 386
ダウンロード確認
Ultra Light の send_download_ack 同期パラメータ 419
ダウンロード専用同期
ASA リモート・データベース 148
dbmlsync -ds オプション 134
ダウンロードの続行
dbmlsync -dc オプション 133
ダウンロードのみの同期
Ultra Light データベース 356, 403
Ultra Light での download_only 同期パラメータ 403
Ultra Light の getNewPassword メソッド 406

ち

抽出
Mobile Link クライアント・データベース 38
抽出ユーティリティ
Mobile Link 38

つ

追加
ASA クライアントへの Mobile Link ユーザの追加 92
Mobile Link ASA クライアントのアーティクル 88
通信
Mobile Link dbmlsync adr オプション 138
Mobile Link dbmlsync ctp オプション 143
Mobile Link dbmlsync -c オプション 132
Mobile Link Ultra Light プロトコル・オプション 433
Mobile Link クライアント 44
Mobile Link 用の指定 6
通信プロトコル
Mobile Link の複数設定 214

て

停止
ASA クライアントの削除のアップロード 114
dbmlsync 116
dbmlsync の自動的な停止 181
データの一貫性 vii
データの調整 vii
データのパブリッシュ
Mobile Link ASA クライアント 82
データベース
Mobile Link リモート・データベース 3
データベース抽出ユーティリティ
Mobile Link 38
データベース・ツール・インタフェース vii

dbmlsync 333
dbmlsync 用の設定 336
テーブル
Mobile Link ASA クライアントのカラムワイズ分割 84
Mobile Link ASA クライアントのパブリッシュ 82
Mobile Link ASA クライアントのローワイズ分割 86
Ultra Light を使用したパブリッシュ 352
テーブル全体
Ultra Light を使用したパブリッシュ 352
テーブル全体のパブリッシュ
Mobile Link ASA クライアント 83
テーブル内の一部のカラムのみのパブリッシュ
Mobile Link ASA クライアント 84
テーブル内の一部のローのみのパブリッシュ
Mobile Link ASA クライアント 86
テクニカル・サポート
ニュースグループ xvi
テストとトラブルシューティング
Mobile Link Ultra Light クライアント 389
デフォルト値の選択方法
Ultra Light プライマリ・キーの一意性 368
デフォルトのグローバル・オートインクリメント・カラムの宣言
Mobile Link システム内の Ultra Light クライアント 366
テンポラリ・テーブル
Ultra Light のクライアント固有のデータを使用した同期 355

と

同期 vii
ASA クライアント 75
ASA クライアントの開始 102
Mobile Link ASA クライアントのスケジュール 115
Mobile Link ASA クライアント用の ActiveSync 108
Mobile Link dbmlsync イベント・フック 225
Mobile Link ユーティリティ 33
Mobile Link 用スケジュール・オプションの構文 164
Palm Computing Platform と Ultra Light 382
Ultra Light アプリケーションへの追加 349
Ultra Light クライアント 347
Ultra Light でのアップロードのみ 428
Ultra Light でのキャンセル 408
Ultra Light での停止 408
Ultra Light での同時実行性 402
Ultra Light でのモニタ 408
Ultra Light での優先度の高い変更 354
Ultra Light で無視されるロー 404
Ultra Light の Checkpoint Store 同期パラメータ 401
Ultra Light の download_only パラメータ 403
Ultra Light の HotSync Palm Computing Platform 374
Ultra Light の概要 348
Ultra Light のクライアント固有のデータ 355
Ultra Light のテーブルの除外 355
Ultra Light のパブリケーション 351
Ultra Light のプロトコル 349
カスタマイズ 225
カスタム・ユーザ認証 26
クライアントの接続パラメータ 44
トランザクション 228
パスワードの変更 25
同期イベント・フックの順序
ASA クライアント 226
同期関数の呼び出し
Ultra Light 363
同期コンジット
Ultra Light の HotSync 383
同期サブスクリプション vii
ASA クライアント 98
ASA クライアントからの削除 100
ASA クライアントの変更 99
オプション 95
同期ストリーム

getStream メソッド　421
ULActiveSyncStream　422
Ultra Light における設定　421
Ultra Light の setStreamParms メソッド　425
Ultra Light の setStream メソッド　421
Ultra Light の stream_error 同期パラメータ　420
Ultra Light の stream_parms 同期パラメータ　425
Ultra Light の stream 同期パラメータ　421
Ultra Light の ULHTTPSStream　422
Ultra Light の ULHTTPStream　422
Ultra Light の UlHTTPStream　422
Ultra Light の ULSecureSocketStream　422
Ultra Light の ULSocketStream　422
Ultra Light の UlSocketStream　422

同期中の同時実行性
ASA クライアント　105

同期定義
作成　119
バージョン 7 との違い　117
バージョン 7 の書き換え　121

同期にモデムを使用する場合の RAS の設定
Ultra Light クライアント　384

同期の開始
ASA クライアント　102

同期のカスタマイズ
ASA クライアント　103

同期のスケジュール
ASA クライアント　115

同期のモニタ
Ultra Light の Observer 同期パラメータ　408
Ultra Light の setObserver メソッド　408

同期パラメータ
ping　411
Ultra Light　398
Ultra Light の Authentication Value　400
Ultra Light の Disable Concurrency　402
Ultra Light の download_only　403
Ultra Light の getNewPassword メソッド　406
Ultra Light の getPassword メソッド　410
Ultra Light の getScriptVersion メソッド　431
Ultra Light の getStream メソッド　421
Ultra Light の getUploadOK メソッド　427
Ultra Light の getUploadOnly メソッド　428
Ultra Light の getUserName メソッド　430
Ultra Light の Keep Partial Download　405
Ultra Light の new_password　406
Ultra Light の observer　408
Ultra Light の Partial Download Retained　409
Ultra Light の password　410
Ultra Light の Resume Partial Download　414
Ultra Light の security_parms　416
Ultra Light の send_column_names　418
Ultra Light の send_download_ack　419
Ultra Light の setNewPassword メソッド　406
Ultra Light の setObserver メソッド　408
Ultra Light の setPassword メソッド　410
Ultra Light の setPing メソッド　411
Ultra Light の setScriptVersion メソッド　431
Ultra Light の setStreamParms メソッド　425
Ultra Light の setStream メソッド　421
Ultra Light の setSynchPublication メソッド　413
Ultra Light の setUploadOnly メソッド　428
Ultra Light の setUserData メソッド　429
Ultra Light の setUserName メソッド　430
Ultra Light の stream_error　420
Ultra Light の stream_parms　425
Ultra Light の Stream Type　421
Ultra Light の Sync Result　426
Ultra Light の upload_ok　427
Ultra Light の upload_only　428
Ultra Light の user_data　429
Ultra Light の user_name　430
Ultra Light の Version　431
Ultra Light のセキュリティ　415
Ultra Light のパブリケーション　413
競合　398
必須　398

同期方法の選択
Ultra Light　372

同期ユーザ
ASA クライアントからの削除　97

ASA クライアントでの作成　92
ASA クライアントでのプロパティの設定　93
ASA クライアントへの追加　92
作成　11
説明　9
名前の共有　12
複数の ASA クライアント　102
同期を制御するクライアント固有のデータの使用
Ultra Light Mobile Link アプリケーション　355
統合コンポーネント
dbmlsync　297
統合データベース
Mobile Link ユーザ名　10
同時実行性
Mobile Link ASA クライアント　105
同時同期
Ultra Light の Disable Concurrency 同期パラメータ　402
登録
ActiveSync での Mobile Link ASA アプリケーション　112
ActiveSync に対する Mobile Link Ultra Light アプリケーション　393
トラブルシューティング
ASA クライアントの Mobile Link 配備　79
Mobile Link Ultra Light クライアントの RAS　389
Mobile Link Ultra Light クライアントのダイヤルアップ・ネットワーキング　389
Ultra Light での同期　420, 426
Ultra Light の getUploadOK メソッド　427
Ultra Light の HotSync コンジット　381
Ultra Light の ping 同期パラメータ　411
Ultra Light の setPing メソッド　411
Ultra Light の upload_ok 同期パラメータ　427
Ultra Light のコンジット　378
リモート・データベースをバックアップからリストア　188
トランザクション単位のアップロード　vii
トランザクション・レベルのアップロード
dbmlsync -tu オプション　190
トランザクション・ログ
dbmlsync のロケーション　104
トランザクション・ログ・ファイル
dbmlsync のロケーション　104
トランスポート・レイヤ・セキュリティ
Ultra Light Java アプリケーション　443, 444
Ultra Light の ActiveSync 同期　434
Ultra Light の HotSync 同期　437

に

ニュースグループ
テクニカル・サポート　xvi
認証
Mobile Link　9
認証処理
Mobile Link　17
認証ステータス
Ultra Light 同期パラメータ　399
認証値
Ultra Light 同期パラメータ　400
認証パラメータ
Ultra Light 同期パラメータ　398

ね

ネットワーク・オプション
dbmlsync　138
ネットワーク・パラメータ
Mobile Link クライアント　44
ネットワーク・プロトコル
ActiveSync による Ultra Light 同期　422
dbmlsync 用の指定　143
HTTPS_FIPS による Ultra Light 同期　422
HTTPS_FIPS を使用した dbmlsync　140
HTTPS による Ultra Light 同期　422
HTTPS を使用した dbmlsync　140
HTTP による Ultra Light 同期　422
HTTP を使用した dbmlsync　138
Mobile Link のための指定　6
TCP/IP による Ultra Light 同期　422

TCP/IP を使用した dbmlsync 138
UlSecureRSASocketStream による Ultra Light 同期 422
UlSecureSocketStream による Ultra Light 同期 422
Ultra Light サポート 360
Ultra Light の Sync Result 同期パラメータ 426
Ultra Light の説明 360
ネットワーク・プロトコル・オプション
dbmlsync 138
Mobile Link クライアント 44
ネットワーク・プロトコルのオプション
Ultra Light 同期 433
ネットワーク・プロトコルの選択
Ultra Light 361

は

廃止された機能
Mobile Link バージョン 7 との違い 117
配備
ASA クライアントにおける Mobile Link 配備のトラブルシューティング 79
Mobile Link ASA クライアント 76
Mobile Link 用の Ultra Light Palm アプリケーション 381
Palm 用 Mobile Link 同期コンジット 381
Ultra Light クライアントで ActiveSync を使用するアプリケーション 394
パスワード
Mobile Link での変更 25
Mobile Link のユーザ認証 20, 24
Ultra Light での同期 406, 410
パスワードの変更
Mobile Link 25
バックアップ
リモート・データベースをリストア 188
パフォーマンス
Mobile Link ASA クライアント 103
Ultra Light でのアップロードのみの同期 428
Ultra Light のダウンロードのみの同期 403
パフォーマンス・チューニングのオプション
Mobile Link ASA クライアント 103
パブリケーション
ALTER PUBLICATION 文 (Mobile Link) 200
CREATE PUBLICATION 文 (Mobile Link) 207
DROP PUBLICATION 文 (Mobile Link) 216
Mobile Link ASA クライアント・オフセット 79
Mobile Link ASA クライアントからの削除 90
Mobile Link ASA クライアントのカラムワイズ分割 84
Mobile Link ASA クライアントの簡単なパブリケーション 83
Mobile Link ASA クライアントの作成 82
Mobile Link ASA クライアントの説明 82
Mobile Link ASA クライアントの変更 88
Mobile Link ASA クライアントのローワイズ分割 86
Mobile Link Ultra Light クライアントの説明 351
Mobile Link での WHERE 句の使用 86
Ultra Light データベース 351, 352
Ultra Light での同期 413
Ultra Light の publication 同期パラメータ 413
Ultra Light の setSynchPublication メソッド 413
[パブリケーション作成] ウィザード
ASA クライアント用 Mobile Link パブリケーションの作成 83
Mobile Link ASA クライアントでのローワイズ分割 86
Mobile Link でのカラムワイズ分割 84
Ultra Light のパブリケーションの作成 352
パブリケーションの削除
Mobile Link ASA クライアント 90
パブリケーション・マスク
Ultra Light 413
パブリッシュ
Mobile Link ASA クライアントの選択したカラム 84
Mobile Link ASA クライアントのテーブル 82
Mobile Link ASA クライアントのテーブル全体

83
Mobile Link で選択したロー 86
Ultra Light のテーブル全体 352
汎用シリアル・バス
Ultra Light における HotSync のサポート 372

ひ

非同期のサフィックス
Ultra Light の非同期テーブル 355
表記
規則 xi

ふ

ファイルベースのダウンロード
dbmlsync -bc オプション 130
dbmlsync -be オプション 131
dbmlsync -bg オプション 131
フィードバック
提供 xvi
マニュアル xvi
複数の Mobile Link 同期ユーザ
ASA クライアント 102
フック
dbmlsync イベント・フックの説明 225
sp_hook_dbmlsync_abort 234
sp_hook_dbmlsync_begin 237
sp_hook_dbmlsync_connect_failed 239
sp_hook_dbmlsync_delay 243
sp_hook_dbmlsync_download_begin 246
sp_hook_dbmlsync_download_com_error 248
sp_hook_dbmlsync_download_end 250
sp_hook_dbmlsync_download_fatal_sql_error 252
sp_hook_dbmlsync_download_log_ri_violation 254
sp_hook_dbmlsync_download_ri_violation 257
sp_hook_dbmlsync_download_sql_error 260
sp_hook_dbmlsync_download_table_begin 263
sp_hook_dbmlsync_download_table_end 265
sp_hook_dbmlsync_end 268
sp_hook_dbmlsync_log_rescan 271
sp_hook_dbmlsync_logscan_begin 273
sp_hook_dbmlsync_logscan_end 275
sp_hook_dbmlsync_process_return_code 278
sp_hook_dbmlsync__schema_upgrade 281
sp_hook_dbmlsync_set_extended_options 284
sp_hook_dbmlsync_upload_begin 286
sp_hook_dbmlsync_upload_end 288
sp_hook_dbmlsync_validate_download_file 293
エラーの無視 154
同期イベント・フック 225
同期イベント・フックの順序 226
プライマリ・キーの一意性の管理
Mobile Link システム内の Ultra Light クライアント 365
プライマリ・キー・プール
Ultra Light を使用したユニークな値の生成 365
プロシージャ
Mobile Link dbmlsync イベント・フック 225
プロトコル vii
dbmlsync 用の指定 143
HTTPS_FIPS を使用した dbmlsync 140
HTTPS を使用した dbmlsync 140
HTTP を使用した dbmlsync 138
TCP/IP を使用した dbmlsync 138
プロトコル・オプション
dbmlsync 138
Mobile Link クライアント 44
Ultra Light Mobile Link クライアント 433
Ultra Light 同期 433
プロパティ
dbmlsync 統合コンポーネント 306
文
ALTER PUBLICATION 構文 (Mobile Link) 200
ALTER SYNCHRONIZATION SUBSCRIPTION 構文 (Mobile Link) 202
ALTER SYNCHRONIZATION USER 構文 (Mobile Link) 205
CREATE PUBLICATION 構文 (Mobile Link) 207
CREATE SYNCHRONIZATION

SUBSCRIPTION 構文 (Mobile Link) 210
CREATE SYNCHRONIZATION USER 構文 (Mobile Link) 213
DROP PUBLICATION 構文 (Mobile Link) 216
DROP SYNCHRONIZATION SUBSCRIPTION 構文 (Mobile Link) 217
DROP SYNCHRONIZATION USER 構文 (Mobile Link) 219
START SYNCHRONIZATION DELETE 構文 (Mobile Link) 220
STOP SYNCHRONIZATION DELETE 構文 (Mobile Link) 222
分割
Mobile Link ASA クライアント間のデータ 78
Mobile Link ASA クライアントのカラムワイズ分割 84
Mobile Link ASA クライアントのローワイズ分割 86

へ

別々に同期させるためのデータ・セットの設計
Ultra Light 351
変更
ALTER PUBLICATION 文 (Mobile Link) 200
ASA クライアントのサブスクリプション 99
ASA クライアントの Mobile Link パブリケーション 88
Mobile Link ASA クライアントのアーティクル 88

ほ

ポート番号
Ultra Light の ULSynchronize 引数 425
ポーリング
dbmlsync ログスキャンのポーリング 146
ホスト名
Ultra Light の ULSynchronize 引数 425

ま

マニュアル
SQL Anywhere Studio viii

み

ミラー・ログ
dbmlsync のミラー・ログの削除 159

む

無視されるロー
Ultra Light での同期 404

も

モデム
Ultra Light の Palm Computing Platform 382
モニタリング
Mobile Link 参照整合性違反のロギング 254

ゆ

ユーザ
ALTER SYNCHRONIZATION USER 文 (Mobile Link) 205
CREATE SYNCHRONIZATION USER 文 (Mobile Link) 213
DROP SYNCHRONIZATION USER 文 (Mobile Link) 219
Mobile Link の説明 10
[ユーザ追加] ウィザード
使用 20
ユーザ認証
Ultra Light 同期のステータス 399
Ultra Light 同期のレポート 399
ユーザ認証アーキテクチャ
説明 16
ユーザ認証メカニズムの選択

説明 14
ユーザの最初のパスワードを設定する
Mobile Link 20
ユーザの認証
.NET 同期論理 27
Java 同期論理 27
Mobile Link カスタム・メカニズム 26
Mobile Link でのメカニズムの選択 14
Mobile Link のアーキテクチャ 16
Mobile Link のセキュリティ 9
Mobile Link パスワード 20
Mobile Link パスワードの変更 25
SQL 同期論理 27
Ultra Light の Authentication Value 同期パラメータ 400
ユーザの認 証
Ultra Light の getPassword メソッド 410
ユーザの認証
Ultra Light の getUserName メソッド 430
Ultra Light の new_password 同期パラメータ 406
Ultra Light の password 同期パラメータ 410
Ultra Light の setNewPassword メソッド 406
ユーザの認 証
Ultra Light の setPassword メソッド 410
ユーザの認証
Ultra Light の setUserName メソッド 430
Ultra Light のカスタム Mobile Link 407
Ultra Light パラメータの user_name 同期 430
新しい Mobile Link ユーザ 22
カスタム Mobile Link 398
パスワード 24
ユーザ名
Mobile Link 10
優先度の高い変更
Ultra Light での同期 354
優先度の高い変更の同期
Ultra Light Mobile Link アプリケーション 354
ユーティリティ
Mobile Link ActiveSync プロバイダ [dbasinst] 34
Mobile Link クライアント [dbmlsync] 124
Mobile Link クライアント・データベース抽出 [mlxtract] 38
Mobile Link のユーティリティのリスト 33
ユニークな値
Ultra Light でのデフォルト・グローバル・オートインクリメントの使用 365

よ

読み込み専用テーブル
Ultra Light データベース 356

ら

ライブラリ関数
ULSynchronize 5

り

リストア
バックアップからリモート・データベース 188
リターン・コード
dbmlsync [sp_hook_dbmlsync_abort] 234
dbmlsync [sp_hook_dbmlsync_process_return_code] 278
リモート DBA パーミッション
ASA クライアントの Mobile Link 同期 102
リモート・アクセス・サービス
Ultra Light Palm クライアントの設定 384
Ultra Light 同期 382
同期にシリアル・ポートを使用する場合の Ultra Light Palm クライアントの設定 386
同期にモデムを使用する場合の Ultra Light Palm クライアントの設定 384
リモート・アクセス・サービスの設定
Ultra Light クライアント 384
リモート・データベース
ASA クライアントの作成 76

ASA クライアントの配備　76
Mobile Link ASA クライアント　75
バックアップからリストア　188
リモート・データベース間でのデータの分割
Mobile Link ASA クライアント　78
リモート・データベースのアップグレード
Mobile Link ASA クライアント　79
リモート・データベースの作成
ASA クライアント　76
リモート・データベースの配備
Mobile Link ASA クライアント　76
リモート・データベースへの Mobile Link ユーザの追加　92

れ

レジストリ
Ultra Light の HotSync パラメータ　374

ろ

ローのダウンロード
Mobile Link 参照整合性違反の解決　254
ローワイズ分割
Mobile Link ASA クライアント　86
ロギング
dbmlsync -v オプション　194
Mobile Link 参照整合性違反　254
ログ・オフセット
Mobile Link ASA クライアント　79
ログスキャンのポーリング
説明　146
ログ・ファイル
dbmlsync のミラー・ログの削除　159
Ultra Light Palm 同期　381
ロック
Mobile Link ASA クライアント　105