SQL Anywhere - Ultra Light
文書バージョン: 17 – 2016-05-11

Ultra Light - 管理

目次

1 Ultra Light - データベース管理および開発者ガイド 7

1.1 Ultra Light 概要 8

Ultra Light のアーキテクチャ 9

Ultra Light 同期クライアント 10

Ultra Light のサポート対象プラットフォーム 11

Ultra Light と SQL Anywhere の機能の比較 11

Ultra Light データベースの制限事項 15

CustDB サンプルアプリケーションの概要 17

Windows Mobile 用の Ultra Light ソリューションの考慮事項 20

1.2 Ultra Light でのセキュリティの考慮事項 23

1.3 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 26

1.4 Ultra Light データベースの作成方法 27

データベース作成ウィザードでの Ultra Light データベースの作成 28

コマンドプロンプトでの Ultra Light データベースの作成 29

Mobile Link 同期モデルでの Ultra Light データベースの作成 30

リモートデータベースの集中管理による Ultra Light データベースの作成 30

XML ファイルからの Ultra Light データベースの作成 30

初回接続時の Ultra Light データベースの作成 31

作成オプション値へのアクセス方法 32

Ultra Light 文字セット 32

データベースセキュリティ 35

1.5 SQL Anywhere データベースから Ultra Light データベースへの変換 36

1.6 Ultra Light データベースへの接続 38

Ultra Light 接続文字列とパラメータ 39

Ultra Light 接続パラメータと ULSQLCONNECT 環境変数 40

接続パラメータでの Ultra Light ファイルパスの指定 41

1.7 Ultra Light データベースのタスクと機能 42

データベースプロパティの読み取り 43

データベースオプションへのアクセス 44

Ultra Light のイベント通知 45

独立性レベル 47

Ultra Light データベースの検証 49

Ultra Light データベースのバックアップとリカバリ 50

1.8 Ultra Light データベースのスキーマ 51

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

Ultra Light のテーブルとカラム. 52
Ultra Light インデックス. 59
Ultra Light ユーザ. 64
1.9 Mobile Link クライアントとしての Ultra Light. 71
Ultra Light クライアント. 71
Microsoft ActiveSync 同期の概要. 90
Ultra Light の同期パラメータ. 90
Ultra Light のネットワークプロトコルのオプション. 116
1.10 Ultra Light の配備. 117
Ultra Light アプリケーションのビルドと配備の仕様. 118
Ultra Light データベースの配備方法. 124
Ultra Light データベーススキーマのアップグレードの配備. 125
Ultra Light エンジンの起動. 127
Microsoft ActiveSync Manager を使用したアプリケーションの登録. 128
1.11 チュートリアル:Ultra Light の CustDB サンプルアプリケーションの構築. 129
レッスン 1:CustDB アプリケーションの構築と実行. 130
レッスン 2:Mobile Link サーバの起動と初期同期の実行. 131
レッスン 3:Ultra Light データベース内のデータの更新. 132
レッスン 4:Ultra Light データベースと統合データベースの同期. 134
レッスン 5: Mobile Link 同期スクリプトのブラウズ. 136
1.12 Ultra Light データベースのリファレンス. 137
Ultra Light オプション. 138
Ultra Light 接続パラメータ. 175
Ultra Light データベースプロパティ. 197
Ultra Light データベースオプション. 201
Ultra Light ユーティリティ. 207
Ultra Light のシステムテーブル. 240
1.13 Ultra Light SQL リファレンス. 247
Ultra Light SQL 言語の要素. 247
SQL データ型. 278
空間データ型. 304
ユーザ定義データ型とそれに相当する型. 306
SQL 関数. 307
Ultra Light SQL 文. 475
1.14 Ultra Light のパフォーマンスに関するヒント. 527
Ultra Light データベースのキャッシュサイズの調整. 528
クエリのパフォーマンスに関するヒント. 528
挿入と更新のパフォーマンスのヒント. 540
Ultra Light のベンチマークに関するヒント. 545

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.15 Ultra Light トラブルシューティング . . . 550
Ultra Light エンジンを起動できない . . . 551
アップグレード後にデータベースに接続できない . . . 551
Ultra Light データベースの破損 . . . 552
データベースのサイズが安定しない . . . 553
新しい Ultra Light データベースへの ASCII データのインポート . . . 554
以前のバージョンのユーティリティが動作する . . . 555
結果セットに予測できない変更が生じる . . . 555
Ultra Light エンジンクライアントでエラー -764 が発生する . . . 556

2 Ultra Light - .NET プログラミング . . . 558
2.1 システムの要件とサポート対象プラットフォーム . . . 558
2.2 Ultra Light.NET アプリケーション開発 . . . 559
Microsoft Visual Studio の SQL Anywhere ツール . . . 560
Ultra Light データベースの接続の設定 . . . 560
Ultra Light.NET での SQL 文を使用したデータの作成と修正 . . . 562
Ultra Light.NET での ULTable クラスを使用したデータの作成と修正 . . . 568
Ultra Light.NET でのトランザクション管理 . . . 576
Ultra Light.NET でのスキーマ情報 . . . 576
Ultra Light.NET でのエラー処理 . . . 577
Ultra Light.NET での Mobile Link データ同期 . . . 577
Ultra Light.NET アプリケーションを配備する方法 . . . 579
2.3 チュートリアル: Ultra Light.NET を使用した Windows Mobile アプリケーションの構築 . . . 583
レッスン 1: Microsoft Visual Studio プロジェクトの作成 . . . 584
レッスン 2: Ultra Light データベースの作成 . . . 587
レッスン 3: アプリケーションへのデータベース接続コントロールの追加 . . . 589
レッスン 4: データの挿入、更新、削除 . . . 591
レッスン 5: アプリケーションの構築と配備 . . . 596
C# チュートリアルのコードリスト . . . 597
Microsoft Visual Basic チュートリアルのコードリスト . . . 599
2.4 Ultra Light .NET API リファレンス . . . 601

3 Ultra Light - C++ プログラミング . . . 603
3.1 システムの要件とサポート対象プラットフォーム . . . 603
3.2 アプリケーション開発 . . . 604
Ultra Light C++ アプリケーション開発 . . . 604
Embedded SQL を使用した Ultra Light C++ アプリケーション開発 . . . 635
Windows Mobile 向け Ultra Light アプリケーション開発 . . . 667
3.3 チュートリアル:C++ API を使用した Windows アプリケーションの構築 . . . 675
レッスン 1: データベースの作成とデータベースへの接続 . . . 676

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

レッスン 2:データベースへのデータの挿入. 679
レッスン 3:テーブルのローの選択とリスト. 681
レッスン 4:アプリケーションへの同期の追加. 682
チュートリアルのコードリストの確認. 684
3.4 API リファレンス. 686
Ultra Light C++ 共通 API リファレンス. 687
Ultra Light C++ API リファレンス. 689
Ultra Light Embedded SQL API リファレンス. 689

4 Ultra Light - Java プログラミング. 693
4.1 システムの要件とサポート対象プラットフォーム. 693
4.2 Ultra Light J アプリケーションの開発. 693
Ultra Light J アプリケーション開発のクイックスタートガイド. 694
Android セットアップの考慮事項. 695
Ultra Light データベースの作成と接続アプローチ. 696
スキーマ操作とデータ管理のクイックスタートガイド. 698
Ultra Light J でのスキーマ情報. 707
Ultra Light J でのエラー処理. 708
Ultra Light J を使用した Mobile Link データの同期. 709
Android 用 Ultra Light J アプリケーションの配備. 710
サンプルコード. 711
4.3 チュートリアル:Android アプリケーションの構築. 711
レッスン 1:新しい Android プロジェクトの設定. 713
レッスン 2:Mobile Link サーバの起動. 715
レッスン 3: Android アプリケーションの実行. 716
レッスン 4:Android アプリケーションのテストと同期. 717
レッスン 5:クリーンアップ. 718
4.4 Ultra LightJ API リファレンス. 719

5 Ultra Light - WinRT プログラミング. 721
5.1 システムの要件とサポート対象プラットフォーム. 721
5.2 Ultra Light for WinRT アプリケーション開発. 722
Ultra Light for WinRT アプリケーション開発のクイックスタートガイド. 723
Ultra Light for WinRT の設定の考慮事項. 723
スキーマ操作とデータ管理のクイックスタートガイド. 724
Windows Phone または Windows ストアアプリ用 Ultra Light アプリケーションの配備. 726
5.3 チュートリアル: Windows Phone アプリケーションの構築. 728
レッスン 1: 新しい Windows Phone アプリケーションの設定. 729
レッスン 2: Mobile Link サーバの起動. 730
レッスン 3: Windows Phone アプリケーションの実行と同期. 731

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

レッスン 4: クリーンアップ. 732
5.4 Ultra Light for WinRT API リファレンス. 733
5.5 このマニュアルの印刷、再生、および再配布. 734

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1 Ultra Light - データベース管理および開発者ガイド

このマニュアルでは、小型デバイス用 Ultra Light データベースシステムの概要を説明します。

このセクションの内容:

Ultra Light 概要 [8 ページ]
Ultra Light は、SQL Anywhere と同じ機能を数多く備えた、コンパクトなリレーショナルデータベース管理システムです。このシステムを使用して、スマートフォン、ハンドヘルドコンピュータ、タブレット PC などの小型デバイス用のモバイルデータベースを作成できます。

Ultra Light でのセキュリティの考慮事項 [23 ページ]
データベースには機密の情報や個人的な情報などが含まれている場合があるので、データベース、データ、およびネットワークを介した通信はセキュリティを考慮した設計になっていることが重要です。

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 [26 ページ]
Ultra Light クライアントを、TCP/IP または HTTPS プロトコルでトランスポートレイヤセキュリティを使用するように設定できます。

Ultra Light データベースの作成方法 [27 ページ]
データベースには、次の 3 つの作成方法があります。

SQL Anywhere データベースから Ultra Light データベースへの変換 [36 ページ]
SQL Anywhere リファレンスデータベースから Ultra Light データベースを作成するには、-a オプションを指定して ulinit ユーティリティを実行します。新しいデータベースは、可能な限りリファレンスデータベースと同じ設定で作成されます。

Ultra Light データベースへの接続 [38 ページ]
データベースを使用するアプリケーションでは、データベースへの接続を確立してから、トランザクションが行われます。Ultra Light データベースに接続することで、アプリケーションのすべてのアクティビティが行われるチャネルが作成されます。

Ultra Light データベースのタスクと機能 [42 ページ]
Ultra Light データベースの管理には、実行するタスクや使用できる機能が数多くあります。

Ultra Light データベースのスキーマ [51 ページ]
データベースの論理的なフレームワークをスキーマといいます。

Mobile Link クライアントとしての Ultra Light [71 ページ]
Mobile Link クライアントとして動作するよう、Ultra Light を設定できます。

Ultra Light の配備 [117 ページ]
ほとんどの場合、開発は、Ultra Light をモバイルデバイスとして使用することを最終リリース目標として、Windows デスクトップまたは Mac OS X 上で行われます。

チュートリアル:Ultra Light の CustDB サンプルアプリケーションの構築 [129 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

このチュートリアルでは、Mobile Link サーバを実行して統合データベースと Ultra Light リモート間でデータを同期する方法、SQL Central を使用して Ultra Light 内のデータを参照する方法、および Ultra Light ユーティリティで Ultra Light データベースを管理する方法について説明します。

Ultra Light データベースのリファレンス [137 ページ]
Ultra Light は、Ultra Light データベースの実行、管理、および設定に役立つ、数多くのツールや機能を提供します。

Ultra Light SQL リファレンス [247 ページ]
Ultra Light は、数多くの SQL 言語機能や要素をサポートしています。

Ultra Light のパフォーマンスに関するヒント [527 ページ]
Ultra Light データベースのパフォーマンス向上に役立ついくつかのトピックについて説明します。

Ultra Light トラブルシューティング [550 ページ]
Ultra Light データベースの問題をトラブルシューティングする際に役立ついくつかのトピックについて説明します。

1.1 Ultra Light 概要

Ultra Light は、SQL Anywhere と同じ機能を数多く備えた、コンパクトなリレーショナルデータベース管理システムです。このシステムを使用して、スマートフォン、ハンドヘルドコンピュータ、タブレット PC などの小型デバイス用のモバイルデータベースを作成できます。

Ultra Light には、Ultra Light データベース内で行われた変更を追跡し、ネットワークを介して Mobile Link サーバと更新を交換する同期クライアントが組み込まれています。Ultra Light が Mobile Link クライアントとして動作することで、モバイルアプリケーションと中央のデータベースや他の Ultra Light データベースとの同期を保持することができます。

Ultra Light では、データベースサーバによく見られるデータベース管理システムはインプロセスランタイムライブラリとして実装されます。ランタイムライブラリとアプリケーションは、同じプロセスの一部です。

このセクションの内容:

Ultra Light のアーキテクチャ [9 ページ]
多数のモバイルプラットフォームをサポートする Ultra Light は、API 開発、データベース管理、およびデータベースレイヤから構成されています。

Ultra Light 同期クライアント [10 ページ]
Ultra Light には、Ultra Light データベース内のすべてのデータがデフォルトで同期される双方向同期クライアントが組み込まれています。

Ultra Light のサポート対象プラットフォーム [11 ページ]
Ultra Light はさまざまなモバイルプラットフォームをサポートしています。Ultra Light データベースの開発には、サードパーティソフトウェアが必要です。

Ultra Light と SQL Anywhere の機能の比較 [11 ページ]
Ultra Light と SQL Anywhere では、可用性や機能が異なる場合があります。

Ultra Light データベースの制限事項 [15 ページ]
Ultra Light データベースには、いくつかのハード制限が適用されます。

CustDB サンプルアプリケーションの概要 [17 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

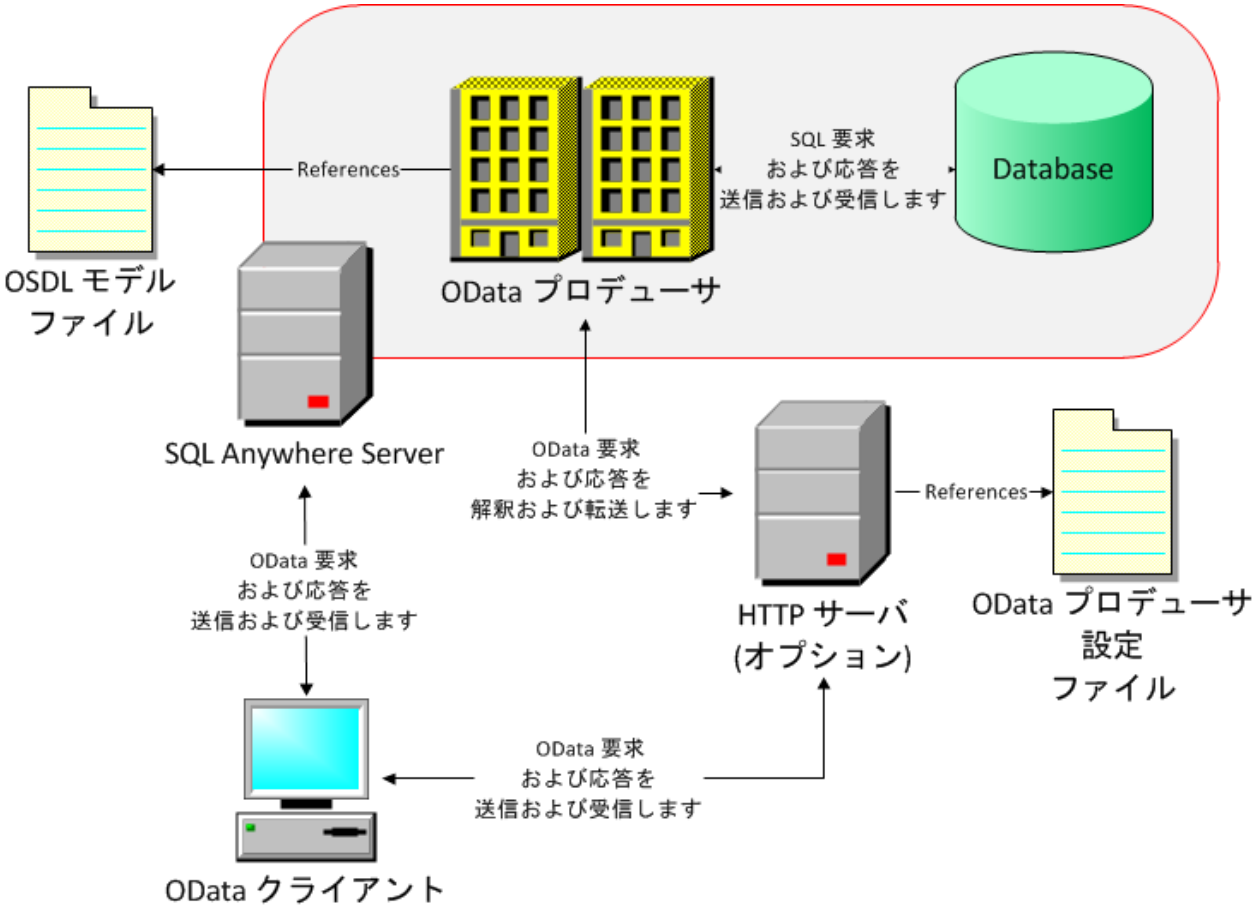
CustDB サンプルは、Mobile Link による SQL Anywhere 統合データベースとの同期を実行する多層データベース管理ソリューションです。

Windows Mobile 用の Ultra Light ソリューションの考慮事項 [20 ページ]

Windos Moble 開発に使用できる Ultra Light の設計オプションについて説明します。

1.1.1 Ultra Light のアーキテクチャ

多数のモバイルプラットフォームをサポートする Ultra Light は、API 開発、データベース管理、およびデータベースレイヤから構成されています。



サポートされるモバイルプラットフォーム

モバイルプラットフォームによって、アプリケーション開発に使用できる Ultra Light API が異なります。

API 開発レイヤ

目的のモバイルプラットフォームにどの API を使用できるかは、上の図を参照してください。

データベース管理レイヤと同期クライアント

Ultra Light API を使用して、Ultra Light データベース管理システムとインタフェースを取ります。このシステムにより、Ultra Light データベースを作成して、接続できます。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

Ultra Light プロジェクトを維持するために、包括的な管理ツールセットが用意されています。これらのツールは、コマンドラインユーティリティとして、または SQL Central の Ultra Light プラグインのウィザードとして実行できます。

データベースレイヤ

このレイヤは、ファイルとして格納されるローカルデータレポジトリです。Ultra Light データベースは UDB ファイルとして保存されます。UDB ファイルは、すべてのモバイルプラットフォームに移植できます。Ultra Light のデータベースには、データベース内のデータの分散に関する情報が含まれません。Ultra Light は、別のログファイルではなく、内部でトランザクションを追跡します。Ultra Light のテンポラリファイルは、データベースファイルと同じディレクトリに保存されます。

関連情報

Ultra Light C++ アプリケーション開発 [604 ページ]
Ultra Light.NET アプリケーション開発 [559 ページ]
Ultra Light for WinRT アプリケーション開発 [722 ページ]
Ultra Light ユーティリティ [207 ページ]
Ultra Light データベースの制限事項 [15 ページ]
Ultra Light 同期クライアント [10 ページ]

1.1.2 Ultra Light 同期クライアント

Ultra Light には、Ultra Light データベース内のすべてのデータがデフォルトで同期される双方向同期クライアントが組み込まれています。

Mobile Link 同期を初めて行う場合は、このデフォルトの動作を使用できます。業務上、必要であれば、カスタム同期設計により、統合データベースと同期させる Ultra Light データを変更できます。SQL Anywhere リモートデータベースとは異なり、同期機能を追加するために Ultra Light の占有容量を増やす必要がありません。

Ultra Light のランタイムに組み込まれている重要な同期機能には、ローのステータスを追跡するメカニズムと同期のステータスを追跡するメカニズムがあります。

ローのステータスを追跡するメカニズム

テーブルとローのステータスの追跡は、データ同期時に特に重要です。Ultra Light データベースの各ローには、ローのステータス構造が関連付けられています。Ultra Light では、ローのステータスは、同期だけでなく、トランザクション処理とデータリカバリの制御にも使用されます。

同期のステータスの追跡

Ultra Light は、進行状況カウンタを使用して堅牢な同期を行います。各アップロードには、識別のためのユニークな ID が付けられます。この ID により、通信エラーが発生した場合にアップロードが正常に行われたかどうかを判別できます。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

新しいデータベースを作成したときは、同期進行状況カウンタは必ず 0 に設定されます。進行状況のカウンタの値 0 は、データベースが新しい Ultra Light データベースなので、Mobile Link サーバでこのクライアントのステータス情報をリセットする必要があることを示します。

> ⚠ 警告
>
> Ultra Light は、同期が発生すると進行状況カウンタを 1 つずつ増やすので、Ultra Light データベースを別の統合データベースに同期することはできません。進行状況カウンタの値が 0 ではなく、統合データベースに格納されているシーケンス番号と一致しない場合は、Mobile Link 同期はオフセット不一致をレポートし、同期は失敗します。進行状況カウンタが現在の値よりも古い場合、Ultra Light データベースをバックアップコピーに置き換えることはできません。

関連情報

Ultra Light データベースのローのステータス管理 [542 ページ]
Mobile Link クライアントとしての Ultra Light [71 ページ]

1.1.3 Ultra Light のサポート対象プラットフォーム

Ultra Light はさまざまなモバイルプラットフォームをサポートしています。Ultra Light データベースの開発には、サードパーティソフトウェアが必要です。

関連情報

Ultra Light のネットワークプロトコルのオプション [116 ページ]
Mobile Link クライアントとしての Ultra Light [71 ページ]
Ultra Light 同期ユーティリティ (ulsync) [231 ページ]

1.1.4 Ultra Light と SQL Anywhere の機能の比較

Ultra Light と SQL Anywhere では、可用性や機能が異なる場合があります。

> ℹ 注記
>
> Ultra Light データベース管理システムは、アプリケーションのサイズを 750 ～ 1500 KB 増やします。一方、SQL Anywhere のデータベース、データベースサーバ、および同期クライアントによる追加分は 6 MB ほどになります。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

機能	SQL Anywhere	Ultra Light	考慮事項
トランザクション処理、マルチテーブルジョイン	X	X	
トリガ、ストアドプロシージャ、ビュー	X		
外部ストアドプロシージャ (呼び出し可能な外部 DLL)	X		
組み込みの参照整合性とエンティティ整合性	X	X	
カスケード更新および削除	X	制限あり	削除と更新がカスケードされる宣言参照整合性は、Ultra Light データベースではサポートされていない機能です。ただし、同期中はこの目的で削除が自動的にカスケードされます。
動的な複数データベースのサポート	X	X	
マルチスレッドアプリケーションサポート	X	X	
ローレベルのロック	X	X	
XML のアンロードとロードのユーティリティ		X	Ultra Light は、ulload、ulunload、uljload、および uljunload 管理ツールを使用して XML のロードとアンロードを行います。
SQLX 機能	X		
SQL 関数	X	X	Ultra Light アプリケーションでは、すべての SQL 関数を使用できるわけではありません。サポートされていない関数を使用すると、エラーが発生します。
SQL 文	X	X	SQL 文のスコープは、SQL Anywhere とは異なります。
統合 HTTP サーバ	X		
データベースファイルとネットワーク通信の強力な暗号化	X	X	
イベントのスケジュールと処理	X	X	Ultra Light イベントモデルは SQL Anywhere とは異なります。
高パフォーマンス、セルフチューニング、コストベースのクエリオプティマイザ	X		Ultra Light にはクエリオプティマイザがありますが、SQL Anywhere のオプティマイザほど機能が豊富ではありません。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

機能	SQL Anywhere	Ultra Light	考慮事項
複数のスレッド対応 API	X	X	Ultra Light には独特の柔軟なアーキテクチャがあるので、アプリケーション開発者は、変化する配備環境や複数の異なる配備環境を対象としたアプリケーションを作成できます。
カーソルのサポート	X	X	
動的キャッシュサイズ決定	X	X	Ultra Light では、CACHE_SIZE、CACHE_MIN_SIZE、および CACHE_MAX_SIZE 接続パラメータを使用することで、キャッシュの初期サイズ、最小サイズ、最大サイズをそれぞれ設定できます。キャッシュのサイズは、Ultra Light によって最大サイズまで継続的に最適化されます (指定した場合)。
システムまたはアプリケーションの障害後のデータベースのリカバリ	X	X	
バイナリラージオブジェクト (BLOB) のサポート	X	X	Ultra Light では、BLOB のインデックス付けまたは比較を実行できません。
Windows パフォーマンスモニタの統合	X		
オンラインのテーブルとインデックスの断片化解除	X		
オンラインバックアップ	X		
デスクトップから Windows Mobile デバイスへの直接デバイス接続		X	SQL Anywhere データベースでは、Windows Mobile デバイスに配備するデータベースにデスクトップ接続できるようになるには、データベースサーバが必要です。Ultra Light では、接続文字列のプレフィクスとして *WCE:¥* を使用します。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

機能	SQL Anywhere	Ultra Light	考慮事項
インデックスの使用による高パフォーマンスの更新と検索	X	X	Ultra Light には、テーブルの検索時にインデックスを使用するか、ローを直接スキャンするかを決定するメカニズムがあります。 また、インデックスをハッシュしてデータ検索を高速化することもできます。 max_hash_size 作成パラメータを使用すると、ハッシュの最大サイズを設定できます。
SAP HANA、Oracle、DB2、SAP Adaptive Server Enterprise、Microsoft SQL Server、MySQL、または SQL Anywhere との同期	X	X	
組み込みの同期		X	SQL Anywhere とは異なり、Ultra Light では、クライアントエージェントによる同期を必要としません。同期は Ultra Light ランタイムに組み込まれているので、配備する必要があるコンポーネントが最小限に抑えられます。
プロセス内実行		X	
計算カラム	X		
宣言されたテンポラリテーブル/グローバルテンポラリテーブル	X		
システム関数	X		
timestamp カラム	X	X	SQL Anywhere は、DEFAULT TIMESTAMP デフォルトをサポートしています。 Ultra Light がサポートしているのは、DEFAULT CURRENT TIMESTAMP デフォルトだけです。したがって、Ultra Light では、ローの更新時にタイムスタンプを自動的に更新することはできません。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

機能	SQL Anywhere	Ultra Light	考慮事項
ユーザベースのパーミッションのスキームによるオブジェクトベースの所有権とアクセスの決定	X		Ultra Light は、認証システムの必要がない、単一ユーザのデータベース向けに設計されています。ただし、最大 4 つのユーザ ID とパスワードを設定し、認証のためだけに使用できます。これらのユーザは、すべてのデータベースオブジェクトにアクセスできます。
空間データ	X	制限あり	Ultra Light は、ポイントデータのみをサポートしています。
完全なテキストデータ	X		

関連情報

外部キー循環に関連する同期の問題の防止 [82 ページ]
Ultra Light SQL 文 [475 ページ]
Microsoft Windows Mobile 用の Ultra Light API の利点 [21 ページ]
Ultra Light データベースの制限事項 [15 ページ]
接続パラメータでの Ultra Light ファイルパスの指定 [41 ページ]
Ultra Light ユーザ [64 ページ]
Ultra Light データベースへの XML のロードユーティリティ (ulload) [227 ページ]
Ultra Light データベースのアンロードユーティリティ (ulunload) [236 ページ]
Ultra Light max_hash_size 作成オプション [155 ページ]

1.1.5 Ultra Light データベースの制限事項

Ultra Light データベースには、いくつかのハード制限が適用されます。

これらの制限はモバイルデバイスの最大容量を超えることもあります。パフォーマンスとデバイスの容量を考慮すると、他の制限が必要になります。

項目	Ultra Light データベースの制限事項
データベースとファイルサイズ	4、8、または 16 KB ページサイズに対して 16 GB。2 KB ページサイズに対して 8 GB。1 KB ページサイズに対して 1 GB。オペレーションシステムにファイルサイズに対する制限がある場合はそれ未満。
テンポラリファイルのサイズ	4、8、または 16 KB ページサイズに対して 16 GB。2 KB ページサイズに対して 8 GB。1 KB ページサイズに対して 1 GB。オペレーションシステムにファイルサイズに対する制限がある場合はそれ未満。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

項目	Ultra Light データベースの制限事項
キャッシュサイズ	デバイス上の使用可能メモリによって制限されます (最大 64000 ページ)。
動的キャッシュサイズ決定	Ultra Light では、データベースのキャッシュの初期サイズ、最小サイズ、最大サイズのいずれかまたはすべてを設定できます。キャッシュのサイズは、Ultra Light によって最大サイズまで継続的に最適化されます (指定した場合)。 CACHE_SIZE、CACHE_MIN_SIZE、CACHE_MAX_SIZE 接続パラメータを使用すると、キャッシュサイズの設定および調整ができます。
データベースでサポートされる同時オープン接続の最大数	最大 14
すべてのデータベースへの同時オープン接続の最大数	メモリの量によってのみ制限されます。
同時に実行できるデータベースの最大数	メモリの量によってのみ制限されます。
データベースに同時に接続できるアプリケーションの最大数	Ultra Light エンジンを使用して、データベースに同時に接続する複数のアプリケーションを処理します。あるいは、一度にデータベースに接続できるアプリケーションを 1 つだけにします。
返される SQL 関数値	場合により、Ultra Light 制限文が 2000 バイトになることがあります。
テーブルあたりのロー数	最大 1600 万 場合によっては、ローに対する変更 (削除および更新) とその他のステータス情報がローデータとともに保持されます。この情報により、変更が同期されます。必然的に、同期間のテーブルのトランザクション数によっては、実際のローの制限が 1600 万より小さくなります。
ローサイズ	パックされた各ローの長さはページサイズ以下である必要があります。 文字列がカラムサイズよりも短い場合は埋め込みなしで格納されます。LONG BINARY と LONG VARCHAR として宣言されたカラムは個別に格納されるため、除外されます。
データベースあたりのロー数	データベースサイズにより制限。
テーブルのサイズ	データベースのサイズによって制限されます。
データベースあたりのテーブル数	データベースのサイズによって制限されます。
テーブルあたりのカラム数	ローサイズはページサイズによって制限されるため、テーブルあたりのカラム数の上限は、実際にはページサイズによって決まります。一般的に、実質的な制限は 4000 よりもかなり小さくなります。
テーブルあたりのインデックス数	データベースのサイズによって制限されます。
パブリケーション数	最大 63
データベースページサイズ	最小 1 KB、最大 16 KB

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

項目	Ultra Light データベースの制限事項
接続ごとのカーソル数	Ultra Light データベースへの特定の接続で使用できるカーソルの最大数は 64 です (すべてのプラットフォーム)。
文字列	ローがページ上に収まる必要があります。
バイナリデータ型	ローがページ上に収まる必要があります。
long binary と long varchar のサイズ	データベースサイズによってのみ制限されます。
blob サイズ	ファイルサイズによって制限されます。
使用できる独立性レベル	0 (コミットされていない読み出し) または 1 (コミットされた読み出し)
カスケード更新および削除	削除と更新がカスケードされる宣言参照整合性は、Ultra Light データベースではサポートされていない機能です。ただし、同期中はこの目的で削除が自動的にカスケードされます。
イベントのスケジュールと処理	Ultra Light イベントモデルは SQL Anywhere とは異なります。

関連情報

Ultra Light でのトランザクション処理 [543 ページ]
ローのパックとテーブル定義 [53 ページ]
Ultra Light CACHE_SIZE 接続パラメータ [181 ページ]
Ultra Light CACHE_MIN_SIZE 接続パラメータ [180 ページ]
Ultra Light CACHE_MAX_SIZE 接続パラメータ [179 ページ]
Ultra Light データベースのアンロードユーティリティ (ulunload) [236 ページ]
Ultra Light データベースへの XML のロードユーティリティ (ulload) [227 ページ]

1.1.6 CustDB サンプルアプリケーションの概要

CustDB サンプルは、Mobile Link による SQL Anywhere 統合データベースとの同期を実行する多層データベース管理ソリューションです。

CustDB は SQL Anywhere とともにインストールされ、下記で構成されます。

- 統合 SQL Anywhere データベースこのデータベースには販売管理データが事前に挿入されています。
- Ultra Light リモートデータベース。このデータベースは最初は空です。
- Ultra Light クライアントアプリケーション。
- Mobile Link サーバ同期スクリプト。

> **注記**
>
> 一度に実行できる CustDB のインスタンスは 1 つだけです。複数のインスタンスを実行すると、最初のインスタンスがフォアグランドに移されます。

CustDB を使用すると、販売担当者は取引を追跡、監視し、次の 2 種類のユーザの情報をプールできます。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

- ユーザ ID 51、52、53 で認証された販売担当者
- ユーザ ID 50 で認証されたモバイル管理者

これらのユーザによって収集された情報は、統合データベースと同期できます。

統合データベースとリモートデータベースの両方に ULOrder というテーブルがあります。統合データベースにはすべての注文 (承認済みと未承認) が含まれますが、Ultra Light リモートデータベースには、認証されたユーザに応じて、ローのサブセットだけが表示されます。

テーブル内のカラムは、クライアントアプリケーションでフィールドとして表示されます。注文を追加するときは、[Customer]、[Product]、[Quantity]、[Price]、[Discount] の各フィールドに入力する必要があります。[Status] や [Notes] などその他の情報を追加することもできます。タイムスタンプカラムにより、ローの同期が必要かどうかが判断されます。

CustDB の同期ロジックは、Mobile Link 同期スクリプトとして統合データベースに格納されています。同期ロジックによって、統合データベースでダウンロードまたはアップロードの対象となる容量を特定できます。タイムスタンプベースの同期やスナップショット同期などの方法で、テーブル全体またはテーブルの一部 (ローまたはカラムのサブセット) をダウンロードできます。

SQL Central を使用すると、統合データベース内の同期スクリプトをブラウズできます。SQL Central は、データベースにスクリプトを追加するためのプライマリツールです。

`custdb.sql` ファイルは、ml_add_connection_script または ml_add_table_script を呼び出して、各同期スクリプトを統合データベースに追加します。接続スクリプトは、特定のテーブルに関連付けられていない高いレベルのイベントを制御します。これらのイベントは、各同期の処理中に必要な全般的なタスクを実行するときに使用します。テーブルスクリプトによって、ローのアップロードの開始や終了、競合の解決、ダウンロードするローの選択など、特定のテーブルの同期に関する特定のイベントでのアクションを実行できます。

SQL Anywhere CustDB データベース

これは統合データベースです。インストール中、このデータベース用に SQL Anywhere 17 CustDB という名前の ODBC データソースが作成されます。データベースファイルは *%SQLANYSAMP17%*`¥UltraLite¥CustDB¥` にあります。

統合された `CustDB.db` ファイルに同期された変更内容を消去できるため、次のスクリプトを使用して新しいバージョンを作成します。*%SQLANYSAMP17%*`¥UltraLite¥CustDB¥makedbs.cmd` と同等です。

Ultra Light CustDB データベース

これは情報のサブセットだけを含む、統合データベースのリモートバージョンです。含まれる情報はデータベースを同期するユーザによって異なります。

ファイル名とロケーションは、プラットフォーム、プログラミング言語、またはデバイスによっても異なることがあります。

- Ultra Light.NET の場合: *%SQLANYSAMP17%*`¥UltraLite.NET¥CustDB¥Common¥custdb.udb`
- その他のすべてのプラットフォームと API の場合: *%SQLANYSAMP17%*`¥UltraLite¥CustDB¥custdb.udb`

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

RDBMS 固有の構築スクリプト

サポートされている任意の RDBMS 用に CustDB 統合データベースを再構築する SQL スクリプトです。

%SQLANYSAMP17%¥MobiLink¥CustDB ディレクトリに次のファイルがあります。

- SQL Anywhere の場合: `custdb.sql`
- Adaptive Server Enterprise の場合: `custase.sql`
- Microsoft SQL Server の場合: `custmss.sql`
- Oracle の場合: `custora.sql`
- IBM DB2 の場合: `custdb2.sql`
- MySQL の場合: `custmys.sql`

Ultra Light CustDB クライアントアプリケーションと ReadMe ファイル

Ultra Light リモートデータベースへの使いやすいインタフェースを提供するエンドユーザアプリケーションです。サポートされているプラットフォームごとにサンプルクライアントがインストールされます。

各クライアントアプリケーションには、`ReadMe.html` ファイルまたは `ReadMe.txt` ファイルが含まれています。各ファイルには、サンプルの構築と実行に必要な手順の概要が含まれています。

アプリケーションと ReadMe のロケーションは、開発環境によって異なります。

同期ロジック

Ultra Light データベース SQL 文と同期の呼び出しは、C++ API の場合 `custdbcpp.cpp` にあります。

このセクションの内容:

Ultra Light の CustDB ファイルのロケーション [20 ページ]
CustDB アプリケーションは、さまざまな開発環境用に構築されています。

関連情報

レッスン 1:CustDB アプリケーションの構築と実行 [130 ページ]

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

1.1.6.1 Ultra Light の CustDB ファイルのロケーション

CustDB アプリケーションは、さまざまな開発環境用に構築されています。

Microsoft Windows 32 ビットデスクトップ用 Ultra Light

CustDB アプリケーションを実行する前に構築する必要はありません。

CustDB 実行ファイルは *%SQLANY17%*¥UltraLite¥Windows¥x86 ディレクトリにあります。

Ultra Light for C++

C++ のすべてのバージョン

C++ には多くの開発環境があるので、C++ CustDB プロジェクトファイルには複数のバージョンが用意されています。ほとんどのバージョンでは汎用ファイルが使用されます。これらのファイルは *C:¥Program Files¥SQL Anywhere 17*¥UltraLite¥Custdb ディレクトリにあります。

C++ CustDB アプリケーションのすべてのバージョンの詳細については、*C:¥Program Files¥SQL Anywhere 17*¥UltraLite¥Custdb¥readme.txt を参照してください。

Microsoft Visual Studio

プロジェクトファイルは、*%SQLANYSAMP17%*¥UltraLite¥CustDB ディレクトリにあります。

Ultra Light.NET

Microsoft Visual Studio 用のプロジェクトファイルは、*C:¥Users¥Public¥Documents¥SQL Anywhere 17¥Samples*¥UltraLite.NET¥CustDB ディレクトリにあります。

1.1.7 Windows Mobile 用の Ultra Light ソリューションの考慮事項

Windos Moble 開発に使用できる Ultra Light の設計オプションについて説明します。

Ultra Light API の選択

Windows Mobile 開発用の各 API を使用した場合の利点が説明されています。

- Ultra Light C++
- Ultra Light Embedded SQL
- Ultra Light.NET

データ管理コンポーネントの選択

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

Windows Mobile 開発用の次のデータ管理コンポーネントを使用した場合の利点が説明されています。

- Ultra Light インプロセスランタイム環境
- Ultra Light データベースエンジン

このセクションの内容:

Microsoft Windows Mobile 用の Ultra Light API の利点 [21 ページ]
Ultra Light C++、Embedded SQL、.NET API には、単純なテーブルベースのデータアクセスインタフェースや、複雑なクエリ用の動的 SQL など、いくつかのデータアクセスモデルがあります。

Windows Mobile 用の Ultra Light データ管理コンポーネント [22 ページ]
Ultra Light では、個別データベースサーバを設定するための追加オーバヘッドを必要とせずに、サイズの小さいリレーショナルデータベースソリューションを構築できます。

1.1.7.1 Microsoft Windows Mobile 用の Ultra Light API の利点

Ultra Light C++、Embedded SQL、.NET API には、単純なテーブルベースのデータアクセスインタフェースや、複雑なクエリ用の動的 SQL など、いくつかのデータアクセスモデルがあります。

これらの利点を組み合わせることによって、Ultra Light はアプリケーション開発者に、さまざまな配備環境向けのアプリケーションを作成できる柔軟なアーキテクチャを提供しています。

Ultra Light.NET API の利点

Ultra Light.NET API は通常、Microsoft Windows Mobile 開発に推奨されます。SQL Anywhere .NET API は、Ultra Light コンポーネントと SQL Anywhere で共有される共通プログラミングモデルを提供し、.NET プログラミングは C/C++ と比較されるからです。

Ultra Light C++ と Embedded SQL API の利点

Ultra Light がさまざまな環境および使用状況で高いパフォーマンスを実現するのに対し、最低レベルの API である Embedded SQL と Ultra Light C++ API は、一般的に最高のパフォーマンスを実現します。

作成するアプリケーションのサイズを最小限に抑えたい場合は、Ultra Light C++ API を使用してください。これらのアプリケーションでは通常、小さなアプリケーションファイルサイズを維持しながら、最良のパフォーマンスを実現できます。

関連情報

Ultra Light C++ アプリケーション開発 [604 ページ]
Embedded SQL を使用した Ultra Light C++ アプリケーション開発 [635 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

Ultra Light.NET アプリケーション開発 [559 ページ]

1.1.7.2 Windows Mobile 用の Ultra Light データ管理コンポーネント

Ultra Light では、個別データベースサーバを設定するための追加オーバヘッドを必要とせずに、サイズの小さいリレーショナルデータベースソリューションを構築できます。

Ultra Light プログラミングインタフェースは、**Ultra Light** インプロセスランタイムライブラリと **Ultra Light** データベースエンジンといった 2 つのアプローチのどちらかを使用します。どちらのアプローチも、接続とデータアクセス要求を制御します。

どちらのコンポーネントにも、Ultra Light データベースと Mobile Link 同期サーバをリンクする双方向の同期クライアントが組み込まれています。

Ultra Light インプロセスランタイムライブラリ

データベースに一度にアクセスする必要があるアプリケーションが 1 つだけの場合は、Ultra Light インプロセスランタイムを推奨します。

ランタイムとアプリケーションは一体のプロセスであり、データベースはアプリケーションに固有のものとなります。ランタイムライブラリによって、Ultra Light データベースと組み込み同期処理が管理されます。

ランタイムにリンクするには、エンジンとは別のインポートライブラリ/DLL ペアを使用する必要があります。

Ultra Light では、静的リンクと動的リンクの両方がサポートされます。

静的リンク

静的リンクは必要なデバイスメモリが少なく、デバイス上で 1 つの Ultra Light アプリケーションだけを使用する場合には、より効率的です。

動的リンク

動的リンクは、デバイス上で複数の Ultra Light アプリケーションを使用する場合には、デバイスメモリをより節約できます。

Ultra Light データベースエンジン (`uleng17.exe` ユーティリティ)

Ultra Light エンジンが使用可能なプラットフォームは、Windows デスクトップと Windows Mobile のみです。エンジンは Ultra Light ランタイムライブラリを使用する別個の実行プログラムであり、複数のアプリケーションからの同時アクセスをサポートします。各アプリケーションは、Ultra Light エンジンと通信するために、クライアントライブラリを使用する必要があります。

Ultra Light エンジンは Ultra Light ランタイムよりも多くのシステムリソースを必要とし、クライアントとデータベース間で大量のデータが移動する場合には、パフォーマンスが低下することがあります。

エンジンに接続するには、ランタイムとは別のインポートライブラリ/DLL ペアを指定する必要があります。

Ultra Light エンジンは、次の状況で必要になります。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

- 複数のプロセスが同じデータベースファイルにアクセスし、しかも同時にアクセスする可能性がある場合 (同時にとは、複数のプロセスに同時に同じデータベースに対する開かれた接続があることを意味します)。
- 中央管理を使用して、Ultra Light アプリケーションデータベースを管理する場合。

関連情報

Ultra Light 同期クライアント [10 ページ]
Ultra Light C++ アプリケーションの構築と配備 [626 ページ]
Ultra Light エンジンの起動 [127 ページ]
Ultra Light の同時実行性 [541 ページ]
Ultra Light と SQL Anywhere の機能の比較 [11 ページ]
Ultra Light データベースの制限事項 [15 ページ]
Ultra Light アプリケーションのビルドと配備の仕様 [118 ページ]
Ultra Light エンジンユーティリティ (uleng17) [217 ページ]

1.2 Ultra Light でのセキュリティの考慮事項

データベースには機密の情報や個人的な情報などが含まれている場合があるので、データベース、データ、およびネットワークを介した通信はセキュリティを考慮した設計になっていることが重要です。

データの安全な環境の構築に役立ついくつかの機能があります。

データベースの暗号化

デフォルトでは、データベースは作成時から暗号化または難読化されていません。テキストカラムやバイナリカラムは、16 進エディタなどの表示ツールを使用して読むことができます。Ultra Light は、次のようなデータベース作成メソッドを提供します。

AES 256 ビットの暗号化

このオプションを使用すると、AES 256 ビットアルゴリズムでデータベースが暗号化されます。強力な暗号化は、巧妙で容赦ないデータへのアクセス試行に対抗するセキュリティを提供します。デバイスで使用するのが AES 暗号化の場合、特別な設定は不要です。

FIPS 140-2 認定 AES 256 ビット暗号化

FIPS 140-2 コンピュータセキュリティ標準である *Security Requirements for Cryptographic Modules* (暗号モジュールのセキュリティ要件) に準拠した暗号化ライブラリは、別途のライセンスのもとで提供されます。FIPS 認定の AES 暗号化を使用するには、デバイスを適切に設定する必要があります。

高い効果を得るために、暗号化キーは、文字、数字、特殊記号を組み合わせて作成する必要があります。暗号化キーは、長くするほど推測されにくくなります。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

> **注記**
>
> データベースを暗号化した後は、暗号化キーは復元できません。

SQL Central ウィザードでは、データベースを暗号化するオプションをクリックしてから強力な暗号化を使用をクリックすることにより、Ultra Light データベース作成中に暗号化オプションを指定できます。AES アルゴリズムの 1 つを選択し、暗号キーを入力します。

ulinit ユーティリティでは、-e オプションを使用して暗号化を指定できます。FIPS 認定の暗号化を使用するかどうかを指定するには、--fips オプションを使用します。-k (--key) オプションとともに暗号キーを指定します。

Ultra Light API 暗号化オプションは、データベース作成時に使用できます。

> **警告**
>
> 暗号化キーはデータベースの作成後に変更できますが、変更する場合は細心の注意が必要です。
>
> この操作は負荷が高く、回復できません。操作が途中で停止すると、データベースが完全に失われる可能性があります。
>
> データベースに強力な暗号化を適用した場合、暗号化キーのコピーを必ず安全な場所に保管してください。暗号化キーがわからなくなった場合は、テクニカルサポートに依頼してもデータにはアクセスできません。アクセスできなくなったデータベースは、廃棄して、新しくデータベースを作成する必要があります。

DBKEY パラメータは、データベースへの接続時に指定される必要があります。指定されないと接続に失敗します。暗号化キーは機密情報として扱う必要があります。

データベースの難読化

このオプションは、データベース内のデータにアクセスしようとする軽い試行を防ぐことはできますが、強力な暗号化と同様の厳重なセキュリティは提供しません。難読化は、パフォーマンスにほとんど影響しません。デバイスで使用するのが単純な難読化の場合、特別な設定は不要です。

> **注記**
>
> データベースを暗号化または難読化する場合は、データベースのキャッシュサイズへの影響を考慮してください。5 ～ 10% オーバヘッドが増大するため、パフォーマンスが低下する可能性があります。パフォーマンスへの影響の度合いは、キャッシュのサイズによって異なります。キャッシュがきわめて小さい場合は、暗号化によって著しいオーバヘッドが追加されることがあります。一方、キャッシュが十分に大きければ、まったく影響を感じない場合もあります。

データを難読化するには、データベースの作成時にデータベース作成パラメータとして `obfuscate=1` を指定します。エンドユーザが対応する接続パラメータを指定する必要はありません。

Ultra Light J API でデータを難読化するには、データベース作成中に ConfigPersistent.enableObfuscation メソッドを使用します。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

トランスポートレイヤセキュリティ (TLS)

Mobile Link トランスポートレイヤセキュリティは、Mobile Link HTTPS プロトコルの本来の機能です。HTTPS および Ultra Light クライアントを使用する場合は、ネットワークプロトコルオプションを使用して、信頼できる証明書と、証明書フィールドを指定できます。信頼できるルート証明書を指定する方法としては、Ultra Light データベース初期化ユーティリティ (ulinit) を使用する方法と、trusted_certificates プロトコルオプションを使用する方法の 2 つがあります。また、Ultra Light Load XML をデータベースユーティリティ (ulload) に使用するとき、クライアント側の認定情報を指定することもできます。

Apple Mac OS X と iOS のエンドツーエンド暗号化

Mobile Link サーバを使用して Mac OS X と iOS の Ultra Light クライアントを同期させるとき、エンドツーエンド暗号化を使用するには、(PEM パブリックキーファイルではなく) パブリックキーを PEM でエンコードされた X509 証明書にカプセル化し、E2EE プライベートキーを指定する必要があります。E2EE プライベートキーを使用して PEM でエンコードされた X509 証明書を作成する場合は、証明書作成ユーティリティ createcert を使用してください。

関連情報

Ultra Light データベースの作成方法 [27 ページ]
Ultra Light データベースのキャッシュサイズの調整 [528 ページ]
Ultra Light のネットワークプロトコルのオプション [116 ページ]
Apple iOS と Mac OS X での考慮事項 [606 ページ]
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 [26 ページ]
Ultra Light obfuscate 作成オプション [159 ページ]
Ultra Light 作成オプション [153 ページ]
Ultra Light データベースプロパティ [197 ページ]
Ultra Light DBKEY 接続パラメータ [187 ページ]
Ultra Light データベース初期化ユーティリティ (ulinit) [221 ページ]
Ultra Light データベースへの XML のロードユーティリティ (ulload) [227 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.3 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

Ultra Light クライアントを、TCP/IP または HTTPS プロトコルでトランスポートレイヤセキュリティを使用するように設定できます。

前提条件

TCP/IP または HTTPS プロトコルを使用している必要があります。

コンテキスト

Mobile Link トランスポートレイヤセキュリティは、Mobile Link HTTPS プロトコルの本来の機能です。HTTPS および Ultra Light クライアントを使用する場合は、信頼できる証明書と、ネットワークプロトコルオプションとして、証明書フィールドを直接指定できます。信頼できるルート証明書を指定する方法としては、Ultra Light データベース初期化ユーティリティを使用する方法と、trusted_certificates プロトコルオプションを使用する方法の 2 つがあります。

手順

1. TCP/IP または HTTPS プロトコルを同期用に指定します。セキュア TCP/IP のキーワードは tls です。
2. TCP/IP または HTTPS プロトコルオプションを指定します。

   certificate_company、certificate_unit、certificate_name の各プロトコルオプションは、証明書のフィールドを確認するために使用されています。

   trusted_certificates HTTPS プロトコルオプションを指定することもできます。このオプションを指定すると、Ultra Light データベースに埋め込まれている、信用された証明書情報が上書きされます。

結果

Ultra Light クライアントが、TCP/IP または HTTPS プロトコルでトランスポートレイヤセキュリティを使用するように設定されます。

例

次の例は C/C++ Ultra Light で記述されています。tls を指定するには、HTTPS を tls に変更します。

```
auto ul_sync_info synch_info;
conn.InitSynchInfo( &synch_info );
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

```
synch_info.user_name = UL_TEXT( "50" );
synch_info.version = UL_TEXT( "ul_default" );
...
synch_info.stream = "https";
...
```

```
auto ul_sync_info synch_info;
...
synch_info.stream = "https";
synch_info.stream_parms = TEXT(
      "port=9999;
       certificate_company=SAP;
       certificate_unit=IAS;
       certificate_name=MobiLink");
```

```
auto ul_sync_info synch_info;
...
synch_info.stream = "https";
synch_info.stream_parms = TEXT(
      "port=9999;
       trusted_certificates=¥rsaroot.crt;
       certificate_company=SAP;
       certificate_unit=IAS;
       certificate_name=MobiLink");
```

1.4 Ultra Light データベースの作成方法

データベースには、次の 3 つの作成方法があります。

- データベース作成用の Ultra Light 管理ツールを使用してデスクトップで作成する方法。
- Ultra Light の API を使用してデバイス上で作成する方法。
  デバイス上で作成する方法については、主に各 API に固有の Ultra Light プログラミングマニュアルで説明しています。
- デバイス上で Ultra Light データベースを作成するように設定された中央管理リモートタスク。

データベースを作成した後は、そのデータベースに接続して、テーブルやその他のデータベースオブジェクトを構築できます。

このセクションの内容:

データベース作成ウィザードでの Ultra Light データベースの作成 [28 ページ]
SQL Central を使用してデータベースを作成します。

コマンドプロンプトでの Ultra Light データベースの作成 [29 ページ]
コマンドプロンプトでデータベースを作成する際に使用できるいくつかのユーティリティが用意されています。

Mobile Link 同期モデルでの Ultra Light データベースの作成 [30 ページ]
開発を簡略化するために、Mobile Link には同期モデル作成ウィザードがあります。このウィザードによって、Ultra Light データベースおよびサーバ側の同期ロジックを作成できます。

リモートデータベースの集中管理による Ultra Light データベースの作成 [30 ページ]
Mobile Link は、Ultra Light データベースを作成できるデータベース作成コマンドを提供します。

XML ファイルからの Ultra Light データベースの作成 [30 ページ]
Ultra Light データベースを管理する中間のフォーマットとして XML を使用します。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

初回接続時の Ultra Light データベースの作成 [31 ページ]
接続時に Ultra Light データベースを検出できなかった場合に新しいデータベースを作成するようにアプリケーションをプログラミングできます。プログラミングすると、アプリケーションは、SQL を使用してテーブル、インデックス、外部キーなどを作成できるようになります。データベースにデータを移植するには、統合データベースと同期します。

Ultra Light 文字セット [32 ページ]
文字列の比較結果とソート順は、データベースの文字セット、照合、エンコードのプロパティによって異なります。

データベースセキュリティ [35 ページ]
データベースを暗号化または難読化することができます。暗号化の場合はデータベースのデータが非常に安全に表現されますが、難読化はデータベースの内容が簡単には読み取れないというだけです。

関連情報

データベースオプションへのアクセス [44 ページ]

1.4.1 データベース作成ウィザードでの Ultra Light データベースの作成

SQL Central を使用してデータベースを作成します。

コンテキスト

使用可能なデータベース作成オプションのナビゲーションを使用して作成する場合は、この方法を使用します。このウィザードを使用すると、選択するターゲットのプラットフォームに基づいて設定できる項目が制限されるので、選択が簡単になります。データベースが作成されると、コマンドライン構文が表示されます。この構文は、記録して、ulinit ユーティリティで使用できます。

手順

1. スタート > プログラム > *SQL Anywhere17* > 管理ツール > *SQL Central* の順にクリックします。
2. ツール > *Ultra Light 17* > データベースの作成 の順にクリックします。
3. データベース作成ウィザードの指示に従います。

結果

データベースが作成されます。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

次のステップ

これで、データベースに接続してテーブルおよびその他のデータベースオブジェクトを作成できます。

関連情報

Ultra Light データベース初期化ユーティリティ (ulinit) [221 ページ]

1.4.2 コマンドプロンプトでの Ultra Light データベースの作成

コマンドプロンプトでデータベースを作成する際に使用できるいくつかのユーティリティが用意されています。

- 空の新しい Ultra Light データベースまたは SQL Anywhere のリファレンスデータベーススキーマに基づく Ultra Light データベースを作成する場合は、ulinit ユーティリティを使用します。このユーティリティをオプションとともに使用すると、データベースを設定できます。
- 新しい Ultra Light データベースのスキーマかデータまたはその両方のソースとして使用する XML ファイルがある場合は、ulload ユーティリティを使用します。
- 集中管理: 配備したすべてのデバイスに Mobile Link エージェントが設定されている場合、またはアプリケーションで初期の Ultra Light データベースを配備できない場合は、この方法を選択します。これらのデバイスに新しい Ultra Light データベースを作成するようにリモートタスクを設定できます。これで、このデータベースを管理者が集中管理できます。

新しい Ultra Light データベースの作成 (コマンドラインの場合)

デフォルトを受け入れる場合は、新しい Ultra Light データベースファイルを指定する次の ulinit ユーティリティを実行してください。

```
ulinit test.udb
```

関連情報

SQL Anywhere データベースから Ultra Light データベースへの変換 [36 ページ]
XML ファイルからの Ultra Light データベースの作成 [30 ページ]
Ultra Light データベース初期化ユーティリティ (ulinit) [221 ページ]

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

1.4.3 Mobile Link 同期モデルでの Ultra Light データベースの作成

開発を簡略化するために、Mobile Link には同期モデル作成ウィザードがあります。このウィザードによって、Ultra Light データベースおよびサーバ側の同期ロジックを作成できます。

複数のリモート Ultra Light データベースと集中型の統合データベースがある同期システムを作成している場合は、この方法を使用します。

モデルを作成したら、SQL Central を Mobile Link モデルモードで操作し、同期モデルを配備する前にカスタマイズできます。モデルの準備ができたら配備できます。このウィザードによって、同期アプリケーションに必要なスクリプトやテーブルが生成されます。

1.4.4 リモートデータベースの集中管理による Ultra Light データベースの作成

Mobile Link は、Ultra Light データベースを作成できるデータベース作成コマンドを提供します。

1.4.5 XML ファイルからの Ultra Light データベースの作成

Ultra Light データベースを管理する中間のフォーマットとして XML を使用します。

前提条件

Ultra Light では任意の XML ファイルは使用できません。*%SQLANY17%*¥Bin32 ディレクトリと *%SQLANY17%*¥Bin64 ディレクトリには、スキーマ定義を含む usm.xsd ファイルがあります。このファイルを使用して、XML フォーマットを確認してください。

コンテキスト

XML を使用すると次のことができます。

- データベースのプロパティやオプションが異なる新しいデータベースにデータをロードします。
- Ultra Light の旧バージョンで作成されたデータベースからスキーマをアップグレードします。
- テキストバージョンの Ultra Light データベースを作成します。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

手順

1. XML ファイルを任意のフォルダに保存します。次のいずれかの操作を行うことができます。
   - データベースを XML ファイルにエクスポートまたはアンロードします。SQL Anywhere データベースをアンロードする場合は、サポートされている任意のエクスポート方法を使用します。
   - 別のソースから XML 出力を取得します。ソースには、別のリレーショナルデータベース、またはフラットファイルにトランザクションが記録される Web サイトを使用できます。常に、XML のフォーマットが Ultra Light の条件を満たしている必要があります。
2. 必要なオプションを指定して ulload ユーティリティを実行します。

結果

データベースが作成されます。

例

`sample.xml` 内のテーブルフォーマットとデータから、新しい Ultra Light データベースを `sample.udb` ファイルに作成するには、次のコマンドを実行します。

```
ulload -c DBF=sample.udb sample.xml
```

次のステップ

これで、データベースに接続してテーブルおよびその他のデータベースオブジェクトを作成できます。

関連情報

コマンドプロンプトでの Ultra Light データベースの作成 [29 ページ]
SQL Anywhere データベースから Ultra Light データベースへの変換 [36 ページ]
データベース作成ウィザードでの Ultra Light データベースの作成 [28 ページ]
Ultra Light データベースへの XML のロードユーティリティ (ulload) [227 ページ]

1.4.6 初回接続時の Ultra Light データベースの作成

接続時に Ultra Light データベースを検出できなかった場合に新しいデータベースを作成するようにアプリケーションをプログラミングできます。プログラミングすると、アプリケーションは、SQL を使用してテーブル、インデックス、外部キーなどを作成できるようになります。データベースにデータを移植するには、統合データベースと同期します。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

考慮事項

追加データベースの作成と SQL コードを追加することで、アプリケーションのサイズが大幅に増大する可能性があります。ただし、アプリケーションをデバイスに配備するだけなので、配備作業は簡単になります。また、運用前の開発サイクルで、テストのためにデバイス上のデータベースを削除し、再構築する場合にもこの方法を使用できます。

1.4.7.1 作成オプション値へのアクセス方法

データベース作成後に作成オプションの値を変更することはできません。ただし、SQL Central で対応するデータベースプロパティを表示することはできます。

Ultra Light J API の場合、getCreationString メソッドを使用して、setCreationString メソッドで登録した作成文字列を表示できます。

その他の Ultra Light API の場合は、その Ultra Light API に適切な GetDatabaseProperty 関数を呼び出すことによって、Ultra Light アプリケーションからデータベースプロパティにプログラム的にアクセスできます。

1.4.8 Ultra Light 文字セット

文字列の比較結果とソート順は、データベースの文字セット、照合、エンコードのプロパティによって異なります。

データベースに適切な文字セット、照合、エンコードのプロパティを決定するには、主に次の点を考慮します。

- 適切なソート順。データベースに格納する文字を適切にソートできる照合を選択してください。
- デバイスのプラットフォーム。サポートされているデバイスによって要件が異なり、一部のデバイスでは、UTF-8 で文字をエンコードする必要があります。複数のデバイスをサポートする必要がある場合は、データベースを共有できるかどうかを確認する必要があります。
- データを同期する場合は、統合データベースでサポートされている言語と文字セット。Ultra Light データベースと統合データベースで使用する文字セットに互換性があることを確認します。互換性がなく、一方のデータベースの文字セットにある文字が他方の文字セットになかった場合、データが失われたり、予期せず変更される可能性があります。Ultra Light を多言語環境に配備する場合も、Ultra Light データベースを UTF-8 でエンコードしてください。
  同期のときに、Mobile Link サーバで次のように文字が変換されます。
  1. Ultra Light データベースの文字が Unicode に変換されます。
  2. Unicode 文字が統合データベースの文字セットに変換されます。

このセクションの内容:

Ultra Light での文字セットのエンコードに関するプラットフォーム要件 [33 ページ]
各プラットフォームには特定の文字セットとエンコードの要件があります。

Ultra Light でサポートされている照合 [33 ページ]
Ultra Light は、いくつかの CHAR 照合をサポートします。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

関連情報

データベースセキュリティ [35 ページ]
Ultra Light の collation 作成オプション [144 ページ]
Ultra Light utf8_encoding 作成オプション [174 ページ]
Ultra Light 接続パラメータ [175 ページ]
Ultra Light の case 作成オプション [142 ページ]

1.4.8.1 Ultra Light での文字セットのエンコードに関するプラットフォーム要件

各プラットフォームには特定の文字セットとエンコードの要件があります。

Windows デスクトップと Windows Mobile

UTF-8 でエンコードされたデータベースを Windows で使用するときは、ワイド文字をデータベースに渡してください。これらのプラットフォームで UTF-8 エンコードを使用している場合、Ultra Light ではワイド文字以外の文字列パラメータが UTF-8 でエンコードされていると見なされますが、これは Windows で標準の文字セットではありません。接続文字列は例外であり、文字列パラメータはアクティブなコードページであると見なされます。ただし、ワイド文字を使用することで、このような複雑な状況を避けることができます。

関連情報

データベースセキュリティ [35 ページ]
Ultra Light utf8_encoding 作成オプション [174 ページ]
Ultra Light 接続パラメータ [175 ページ]

1.4.8.2 Ultra Light でサポートされている照合

Ultra Light は、いくつかの CHAR 照合をサポートします。

次のコマンドを入力してリストを生成することもできます。

```
ulinit -Z
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

照合ラベル	説明
1250LATIN2	Code Page 1250、Windows Latin 2、中央/東ヨーロッパ言語
1250POL	Code Page 1250、Windows Latin 2、ポーランド語
1251CYR	Code Page 1251、Windows キリル語
1252LATIN1	Code Page 1252、Windows Latin 1、西ヨーロッパ言語
1252NOR	Code Page 1252、Windows Latin 1、ノルウェー語
1252SPA	Code Page 1252、Windows Latin 1、スペイン語
1252SWEFIN	Code Page 1252、Windows Latin 1、スウェーデン語/フィンランド語
1253ELL	Code Page 1253、Windows ギリシア語、ISO8859-7 拡張付き
1254TRK	Code Page 1254、Windows トルコ語、ISO8859-9 拡張付き
1254TRKALT	Code Page 1254、Windows トルコ語、拡張付き ISO8859-9、I-dot と I-no-dot は同じ
1255HEB	Code Page 1255、Windows ヘブライ語、ISO8859-8 拡張付き
1256ARA	Code Page 1256、Windows アラビア語、ISO8859-6 拡張付き
1257LIT	Code Page 1257、Windows バルト諸国語、リトアニア語
874THAIBIN	Code Page 874、Windows Thai、ISO8859-11、バイナリ順序
932JPN	Code Page 932、日本語シフト JIS、Microsoft 拡張文字付き
936ZHO	Code Page 936、中国語 (簡体文字)、PRC GBK
949KOR	Code Page 949、韓国語 KS C 5601-1987 コード、完成型
950ZHO_HK	Code Page 950、中国語 (繁体文字)、Big 5 コード (HKSCS を含む)
950ZHO_TW	Code Page 950、中国語 (繁体文字)、Big 5 コード
EUC_CHINA	中国語 (簡体文字)、GB 2312-80 コード
EUC_JAPAN	日本語の EUC JIS X 0208-1990 と JIS X 0212-1990 コード
EUC_KOREA	Code Page 1361、韓国語 KS C 5601-1992 8 ビットコード、Johab (組合型)
EUC_TAIWAN	Code Page 964、EUC-TW エンコード
ISO1LATIN1	ISO8859-1、ISO Latin 1、西ヨーロッパ言語、Latin 1 順序
ISO9LATIN1	ISO8859-15、ISO Latin 9、西ヨーロッパ言語、Latin 1 順序
ISO_1	ISO8859-1、ISO Latin 1、西ヨーロッパ言語
ISO_BINENG	バイナリ順序、英語 ISO/ASCII 7 ビット文字ケースマッピング
UTF8BIN	UTF-8、Unicode 用 8 ビットマルチバイトエンコード、バイナリ順序

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.4.9 データベースセキュリティ

データベースを暗号化または難読化することができます。暗号化の場合はデータベースのデータが非常に安全に表現されますが、難読化はデータベースの内容が簡単には読み取れないというだけです。

デフォルトでは、データベースは暗号化も難読化もされません。テキストカラムやバイナリカラムは、16 進エディタなどの表示ツールを使用して読むことができます。データをプレーンテキストとして格納したくない場合は、次のオプションを検討してください。

難読化

このオプションは、データベース内のデータにアクセスしようとする軽い試行を防ぐことはできますが、強力な暗号化と同様の厳重なセキュリティは提供しません。難読化は、パフォーマンスにほとんど影響しません。デバイスで使用するのが単純な難読化の場合、特別な設定は不要です。

AES 256 ビットの暗号化

このオプションを使用すると、AES 256 ビットアルゴリズムでデータベースが暗号化されます。強力な暗号化は、巧妙で容赦ないデータへのアクセス試行に対抗するセキュリティを提供します。デバイスで使用するのが AES 暗号化の場合、特別な設定は不要です。

FIPS 140-2 認定 **AES 256** ビット暗号化

FIPS 140-2 コンピュータセキュリティ標準である *Security Requirements for Cryptographic Modules (*暗号モジュールのセキュリティ要件*)* に準拠した暗号化ライブラリは、別途のライセンスのもとで提供されます。FIPS 認定の AES 暗号化を使用するには、デバイスを適切に設定する必要があります。

データベースの難読化

データを難読化するには、データベースの作成時にデータベース作成パラメータとして `obfuscate=1` を指定します。エンドユーザが対応する接続パラメータを指定する必要はありません。

Ultra Light J API でデータを難読化するには、データベース作成中に ConfigPersistent.enableObfuscation メソッドを使用します。

データベースの暗号化

高い効果を得るために、暗号化キーは、文字、数字、特殊記号を組み合わせて作成する必要があります。暗号化キーは、長くするほど推測されにくくなります。

> **注記**
>
> データベースを暗号化した後は、暗号化キーは復元できません。

SQL Central ウィザードでは、データベースを暗号化するオプションをクリックしてから強力な暗号化を使用をクリックすることにより、Ultra Light データベース作成中に暗号化オプションを指定できます。AES アルゴリズムの 1 つを選択し、暗号キーを入力します。

ulinit ユーティリティでは、-e オプションを使用して暗号化を指定できます。FIPS 認定の暗号化を使用するかどうかを指定するには、--fips オプションを使用します。-k (--key) オプションとともに暗号キーを指定します。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

Ultra Light API 暗号化オプションは、データベース作成時に使用できます。

> ⚠ 警告
>
> 暗号化キーはデータベースの作成後に変更できますが、変更する場合は細心の注意が必要です。
>
> この操作は負荷が高く、回復できません。操作が途中で停止すると、データベースが完全に失われる可能性があります。
>
> データベースに強力な暗号化を適用した場合、暗号化キーのコピーを必ず安全な場所に保管してください。暗号化キーがわからなくなった場合は、テクニカルサポートに依頼してもデータにはアクセスできません。アクセスできなくなったデータベースは、廃棄して、新しくデータベースを作成する必要があります。

DBKEY パラメータは、データベースへの接続時に指定される必要があります。指定されないと接続に失敗します。暗号化キーは機密情報として扱う必要があります。

関連情報

Ultra Light データベースのキャッシュサイズの調整 [528 ページ]
Ultra Light obfuscate 作成オプション [159 ページ]
Ultra Light DBKEY 接続パラメータ [187 ページ]
Ultra Light 作成オプション [153 ページ]
Ultra Light データベース初期化ユーティリティ (ulinit) [221 ページ]

1.5 SQL Anywhere データベースから Ultra Light データベースへの変換

SQL Anywhere リファレンスデータベースから Ultra Light データベースを作成するには、-a オプションを指定して ulinit ユーティリティを実行します。新しいデータベースは、可能な限りリファレンスデータベースと同じ設定で作成されます。

SQL Anywhere リファレンスデータベースはデータベーステンプレートとして機能し、Ultra Light データベーススキーマの作成に次の設定を使用します。

- データベースの設定 (照合順など)
- テーブル定義
- 同期パブリケーション

リファレンスデータベースのパブリケーションの一部として、データを含めたり、カラム、テーブル、インデックスを選択することもできます。

> ℹ 注記
>
> SQL Anywhere 以外の RDBMS から Ultra Light データベースを初期化するには、SQL Central で同期モデル作成ウィザードを使用します。スキーマ情報の取得を要求されたら、統合データベースに接続してください。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

変換に関する考慮事項

ulinit ユーティリティを実行する前に、次のリファレンスデータベースタスクが必要かどうか検討してください。

必要に応じてテーブル、キー、インデックス、および同期パブリケーションを追加します。

必要に応じてテーブルを追加し、プライマリキーを設定します。Ultra Light アプリケーション内で必要な外部キー関係を指定することもできます。

インデックスを使用すると、特に低速のデバイスでパフォーマンスが大きく向上します。プライマリキーカラムには自動的にインデックスが付きますが、他のカラムタイプには付きません。

> **注記**
>
> Ultra Light アプリケーションで情報を特定の順序で取り出すことが多い場合は、リファレンスデータベースにインデックスを追加することを検討してください。

異なるテーブルを異なる時期に同期する場合は、同期パブリケーションを使用します。複数の同期パブリケーションを使用して、テーブルのサブセットを定義し、サブセットごとに同期の優先度を設定できます。
望ましくない影響を及ぼす可能性のあるデータベースオプションまたはテーブルスキーマを更新します。

たとえば、SQL Anywhere データベースのカラムに含まれる句が Ultra Light ではサポートされていない場合、デフォルト値は無視され、新しいデータベースには Ultra Light のデフォルト値が指定されます。
Ultra Light でサポートされていない場合は照合順を変更します。

Ultra Light では、リファレンスデータベースに定義されている照合順の名前が使用されます。utf8_encoding プロパティを設定すると、UTF-8 を使用してエンコードできます。

Ultra Light でサポートされている照合および対応するコードページのリストを表示するには、コマンドプロンプトで -Z オプションを指定して ulinit を実行します。CHAR に対する UCA 照合順など、サポートされていない照合順がリファレンスデータベースで使用されている場合は、次の手順でサポートされている照合順に変更します。

1. Unload ユーティリティを使用して、SQL Anywhere リファレンスデータベースをアンロードします。
2. 異なる照合を使用する新しい SQL Anywhere データベースを作成し、Interactive SQL で `reload.sql` スクリプトを実行します。

> **例**
>
> 次のコマンドでは、`MySADb` データソースに定義されている既存の SQL Anywhere リファレンスデータベースから、`customer.udb` という名前の新しい Ultra Light データベースが作成されます。リファレンスデータベースのテーブルは TestPublication に定義されています。作成された Ultra Light データベースには、TestPublication 内とすべて同じデータベースオプションとテーブルが含まれ、データベースは mykey 暗号化キーで暗号化されます。
>
> ```
> ulinit -a "DSN=MySADb;UID=JimmyB;PWD=secret" -n TestPublication -k mykey
> customer.udb
> ```
>
> 次のコマンドでは、`MySource.db` という名前の既存の SQL Anywhere データベースから、`customer.udb` という名前の新しい Ultra Light データベースが作成されます。作成されたデータベース内のテーブルとインデックスは、Pub1 スキーマパブリケーションに含まれるものと同じです。Pub2 同期パブリケーションが Ultra Light データベースに作成されます。
>
> ```
> ulinit -a DBF=MySource.db;UID=JimmyB;PWD=secret customer.udb -n Pub1 -s Pub2
> ```


© 2016 SAP SE or an SAP affiliate company. All rights reserved.

関連情報

Ultra Light のテーブルとカラム [52 ページ]
インデックスを使用する場合 [61 ページ]
インデックススキャンの作成と保持 [529 ページ]
Ultra Light データベースの作成方法 [27 ページ]
Ultra Light でのデータのパブリッシュ [80 ページ]
Ultra Light データベース初期化ユーティリティ (ulinit) [221 ページ]

1.6 Ultra Light データベースへの接続

データベースを使用するアプリケーションでは、データベースへの接続を確立してから、トランザクションが行われます。Ultra Light データベースに接続することで、アプリケーションのすべてのアクティビティが行われるチャネルが作成されます。

アプリケーションには、Ultra Light のユーティリティ、接続ウィンドウ、またはカスタムアプリケーションがあります。接続が試行されるたびに、データベースに固有の SQL トランザクションが作成されます。

このセクションの内容:

Ultra Light 接続文字列とパラメータ [39 ページ]
接続文字列は、データベース接続を定義し、確立できるようにアプリケーションから Ultra Light ランタイムに渡す一連の接続パラメータです。一部のパラメータは接続を開くために常に必要ですが、その他のパラメータは単一の接続のデータベース機能を調整するために使用されます。

Ultra Light 接続パラメータと ULSQLCONNECT 環境変数 [40 ページ]
ULSQLCONNECT 環境変数を使用すると、同じ接続パラメータを Ultra Light デスクトップ管理ツールに繰り返し指定する必要がなくなります。Interactive SQL と SQL Central の両方とも、ULSQLCONNECT 環境変数をサポートしています。

接続パラメータでの Ultra Light ファイルパスの指定 [41 ページ]
デバイスの物理記憶領域によって、データベースをファイルとして保存するかどうか、およびデータベースを指定するときの命名規則が決定されます。

関連情報

Ultra Light データベースへの接続 [607 ページ]
Embedded SQL を使用した Ultra Light データベース接続 [641 ページ]
Ultra Light データベースの接続の設定 [560 ページ]
Ultra Light 接続パラメータ [175 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.6.1 Ultra Light 接続文字列とパラメータ

接続文字列は、データベース接続を定義し、確立できるようにアプリケーションから Ultra Light ランタイムに渡す一連の接続パラメータです。一部のパラメータは接続を開くために常に必要ですが、その他のパラメータは単一の接続のデータベース機能を調整するために使用されます。

接続文字列は、セミコロンで区切ったリスト内の `keyword=value` のペアとして定義されます。次に、データベースファイル名、ユーザ ID、パスワードを指定する接続文字列フラグメントの例を示します。

```
DBF=myULdb.udb;UID=JDoe;PWD=token
```

これらのパラメータをデータベースに指定する方法は、Ultra Light ユーティリティから接続するか、Ultra Light アプリケーションから接続するかによって異なります。

Ultra Light のコマンドラインユーティリティでは、データベースへの接続が必要な場合に接続文字列が使用されます。

Ultra Light アプリケーションは、保管されたファイルまたはアプリケーションコード内から接続パラメータ値を読み取る Ultra Light API を使用して開発できます。ユーザ認証が不要な場合は固定接続文字列を提供できます。または、接続時にパラメータ値を提供するようにユーザに要求します。

接続文字列は、アセンブルされると、Ultra Light ランタイムに渡されて処理されます。接続試行が検証されると、データベースへの接続が許可されます。データベースファイルが存在しないか、または認証に失敗すると、接続エラーが発生します。

認識できない接続パラメータがあった場合は、Ultra Light でエラーが発生します。

プレフィクス

接続パラメータとともにプレフィクスを使用して、アプリケーションが特定の出力に関して実行されているときのみパラメータを適用するよう指定できます。Ultra Light の場合、これらのプレフィクスは `desktop:` と `device:` です。

制限事項

接続文字列パラメータ値に含まれる前後のスペースはすべて無視されます。接続パラメータ値に、先頭の一重引用符 (')、先頭の二重引用符 (")、またはセミコロン (;) を含めることはできません。

例

たとえば、ulload ユーティリティを使用した場合、次の接続文字列を使用して新しい XML データが既存のデータベースにロードされます。この文字列を指定するまで、指定のデータベースファイルには接続できません。

```
ulload -c "DBF=sample.udb;UID=DBA;PWD=sql" sample.xml
```

このセクションの内容:

Ultra Light 管理ツールの接続パラメータの優先度 [40 ページ]
Ultra Light 管理ツールでは、次の順位で接続パラメータが優先されます。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

関連情報

Ultra Light データベースへの接続 [607 ページ]
Embedded SQL を使用した Ultra Light データベース接続 [641 ページ]
Ultra Light データベースの接続の設定 [560 ページ]
Ultra Light 接続パラメータ [175 ページ]

1.6.1.1 Ultra Light 管理ツールの接続パラメータの優先度

Ultra Light 管理ツールでは、次の順位で接続パラメータが優先されます。

- デバイス固有のオプションは、OS 固有でないオプションよりも優先されます。たとえば、デバイス DBF は Windows Mobile デバイス上で DBF よりも優先されます。
- デスクトップ固有のオプションは、OS 固有でないオプションよりも優先されます。たとえば、デスクトップ DBF は Windows 上で DBF よりも優先されます。
- 接続文字列で重複するパラメータを指定すると、最後に指定したパラメータが使用されます。他のパラメータはすべて無視されます。
- 接続文字列のパラメータが、ULSQLCONNECT 環境変数または接続オブジェクトで指定されたパラメータより優先されます。
- UID と PWD の両方の値が接続文字列でも ULSQLCONNECT でも指定されていない場合は、デフォルトの *UID=DBA* と *PWD=sql* が想定されます。

1.6.2 Ultra Light 接続パラメータと ULSQLCONNECT 環境変数

ULSQLCONNECT 環境変数を使用すると、同じ接続パラメータを Ultra Light デスクトップ管理ツールに繰り返し指定する必要がなくなります。Interactive SQL と SQL Central の両方とも、ULSQLCONNECT 環境変数をサポートしています。

ULSQLCONNECT 環境変数は、*keyword*=`value` をセミコロンで区切って定義したパラメータのリストとして設定できます。

指定した値が、デスクトップ管理ツールのデフォルトの値になります。追加パラメータが必要な場合や、この環境変数で設定したデフォルト値を変更する必要がある場合は、これらの値を設定してください。ユーザが指定する値が常にこの環境変数より優先されます。

例

この例では、ULSQLCONNECT 環境変数を設定して、`c:¥database¥myfile.udb` というデフォルトデータベースファイル、*demo* というデフォルトユーザ、および *test* というデフォルトパスワードを指定します。

```
set ULSQLCONNECT=DBF=c:¥database¥myfile.udb;UID=demo;PWD=test
```

引数なしで Interactive SQL を起動すると、指定したたクレデンシャルを使用して、指定したデータベースに自動的に接続することになります。SQL Central を起動して Ultra Light データベースに接続する場合、接続ダイアログの自動入力にこれらのパラメータが使用されます。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

関連情報

Ultra Light 管理ツールの接続パラメータの優先度 [40 ページ]

1.6.3 接続パラメータでの Ultra Light ファイルパスの指定

デバイスの物理記憶領域によって、データベースをファイルとして保存するかどうか、およびデータベースを指定するときの命名規則が決定されます。

> **注記**
>
> Ultra Light エンジンを使用して、データベースへのマルチプロセスアクセスに対応する場合は、エンジンが別のロケーションから起動される可能性があるため、絶対ファイルパスを使用してください。

単一のプラットフォームに配備する場合や、Ultra Light デスクトップ管理ツールを使用する場合は、DBF パラメータが最適です。次に例を示します。

```
ulload -c DBF=sample.udb sample.xml
```

> **注記**
>
> Ultra Light 管理ツールを使用して、接続しているデバイスにすでに配備されているデータベースを管理できます。

プラットフォームを問わないアプリケーションを作成している場合は、プラットフォーム固有の DBF 接続パラメータを使用して汎用の接続文字列を組み立てます。例:

```
Connection = DatabaseManager::OpenConnection("UID=JDoe;PWD=ULdb;device:DBF=¥database
¥MobileDB.udb;desktop:DBF=DesktopDB.udb")
```

Windows Mobile 以外のプラットフォーム

絶対パスまたは相対パスのいずれかを使用できます。

Windows Mobile

Windows Mobile デバイスでは、絶対パスが必要です。

Windows Mobile データベースはデスクトップまたは接続しているデバイスで管理できます。Windows Mobile デバイスにあるデータベースを管理するには、絶対パスの前に *wce:¥* を付ける必要があります。たとえば、次のように ulunload ユーティリティを使用します。

```
ulunload -c DBF=wce:¥UltraLite¥myULdb.udb c:¥out¥ce.xml
```


© 2016 SAP SE or an SAP affiliate company. All rights reserved.

この例では、Ultra Light によってデータベースが Windows Mobile デバイスから、Windows デスクトップの `c:¥out` フォルダにある `ce.xml` ファイルにアンロードされます。

関連情報

Ultra Light DBF 接続パラメータ [185 ページ]

1.7 Ultra Light データベースのタスクと機能

Ultra Light データベースの管理には、実行するタスクや使用できる機能が数多くあります。

このセクションの内容:

データベースプロパティの読み取り [43 ページ]
Ultra Light データベースプロパティの設定を検査します。

データベースオプションへのアクセス [44 ページ]
データベースの動作を設定するデータベースオプションを表示し変更します。

Ultra Light のイベント通知 [45 ページ]
Ultra Light はイベントと通知をサポートします。通知は、イベントが発生したときに送信されるメッセージであり、パラメータに関する追加情報が示されます。

独立性レベル [47 ページ]
独立性レベルは、あるトランザクションの操作が、同時に処理されている別のトランザクションの操作からどの程度参照できるかを定義します。

Ultra Light データベースの検証 [49 ページ]
SQL Central のデータベース検証ウィザードや、Ultra Light データベース検証ユーティリティ、または Ultra Light API の ValidateDatabase メソッドなどのツールを使用して、データベースを検証します。

Ultra Light データベースのバックアップとリカバリ [50 ページ]
Ultra Light データベースを使用しているアプリケーションが予期せずに停止した場合、アプリケーションを再起動すると、データベースは自動的に一貫性が保たれた状態にリカバリされます。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.7.1 データベースプロパティの読み取り

Ultra Light データベースプロパティの設定を検査します。

コンテキスト

各 Ultra Light API には、データベースプロパティにアクセスするためにアプリケーション内で使用できる GetDatabaseProperty メソッドが含まれています。

DB_PROPERTY SQL 関数を呼び出す方法でも、データベースプロパティにアクセスできます。

手順

1. SQL Central を使用してデータベースに接続します。
2. データベースを右クリックし、プロパティをクリックします。

結果

データベースプロパティは、データベースプロパティウィンドウの一般タブおよび同期情報タブに表示されます。同期情報タブでは、データベースプロパティはプロパティ名のアルファベット順で表示されます。データベースプロパティを値の順にソートするには、値カラムをクリックします。

関連情報

Ultra Light データベースプロパティ [197 ページ]
DB_PROPERTY 関数 [システム] [368 ページ]
作成オプション値へのアクセス方法 [32 ページ]
Ultra Light 文字セット [32 ページ]
Ultra Light データベース初期化ユーティリティ (ulinit) [221 ページ]
Ultra Light データベースへの XML のロードユーティリティ (ulload) [227 ページ]
Ultra Light DBF 接続パラメータ [185 ページ]
Ultra Light DBKEY 接続パラメータ [187 ページ]
Ultra Light desktop 接続パラメータプレフィックス [189 ページ]
Ultra Light device 接続パラメータプレフィックス [190 ページ]
Ultra Light MIRROR_FILE 接続パラメータ [191 ページ]
Ultra Light PWD 接続パラメータ [192 ページ]
Ultra Light RESERVE_SIZE 接続パラメータ [193 ページ]

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

Ultra Light UID 接続パラメータ [196 ページ]

1.7.2 データベースオプションへのアクセス

データベースの動作を設定するデータベースオプションを表示し変更します。

コンテキスト

SQL Central から、永続データベースオプションの設定を表示および変更できます。一時的に、SQL Central から、Ultra Light データベースオプションを表示または設定できない場合があります。

データベースオプションは、いつでも設定または変更できます。一時的なデータベースオプションは、データベースの実行中のみ保持されます。

オプションの値は、SET OPTION SQL 文を使用して設定します。

手順

1. SQL Central を使用してデータベースに接続します。
2. データベースを右クリックし、オプションをクリックします。
3. オプションを設定またはリセットするには、値フィールドに新しい値を入力します。
4. すぐに設定またはすぐにリセットをクリックして変更をコミットします。

結果

データベースオプションの設定が変更され、保存されます。

関連情報

Ultra Light データベースオプション [201 ページ]
データベースプロパティの読み取り [43 ページ]
SET OPTION 文 [Ultra Light] [519 ページ]
作成オプション値へのアクセス方法 [32 ページ]
Ultra Light 文字セット [32 ページ]
Ultra Light データベース初期化ユーティリティ (ulinit) [221 ページ]
Ultra Light データベースへの XML のロードユーティリティ (ulload) [227 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

Ultra Light DBF 接続パラメータ [185 ページ]
Ultra Light DBKEY 接続パラメータ [187 ページ]
Ultra Light desktop 接続パラメータプレフィックス [189 ページ]
Ultra Light device 接続パラメータプレフィックス [190 ページ]
Ultra Light MIRROR_FILE 接続パラメータ [191 ページ]
Ultra Light PWD 接続パラメータ [192 ページ]
Ultra Light RESERVE_SIZE 接続パラメータ [193 ページ]
Ultra Light UID 接続パラメータ [196 ページ]

1.7.3 Ultra Light のイベント通知

Ultra Light はイベントと通知をサポートします。通知は、イベントが発生したときに送信されるメッセージであり、パラメータに関する追加情報が示されます。

イベント通知によって、接続間または同じデータベースに接続されているアプリケーション間での調整および通知を行うことができます。通知は、接続のデフォルトキューまたは明示的に作成および命名されたキューのいずれかで管理されます。イベントの発生時に、通知が登録されたキュー (または接続) に送信されます。

それぞれの接続では、固有の通知キューが管理されます。名前付きのキューは、任意の接続に対して作成できます。

また、この機能では、事前に定義されたシステムイベントを使用して、テーブルが変更された場合などのデータの変更に対する "トリガ" や同期が行われた場合の通知が提供されます。次のような事前に定義されたイベントがあります。

- Commit
- SyncComplete
- TableModified

Ultra Light にはシステムイベントがありますが、イベントにはユーザ定義のものもあります。ユーザイベントは、アプリケーションによって定義およびトリガすることもできます。

イベントと通知の API は、サポートされている各言語で提供されます。また、API 機能にアクセスするための SQL 関数も提供されています。

イベント

イベント	出現箇所
Commit	コミットの終了時に通知されます。
SyncComplete	同期の終了時に通知されます。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

イベント	出現箇所
TableModified	テーブルのローが挿入、更新、または削除されたときにトリガされます。イベントに登録された要求によって影響を受けるローの数にかかわらず、要求ごとに 1 つのイベントが通知されます。 `object_name` パラメータは、モニタするテーブルを指定します。値 "*" は、データベース内のすべてのテーブルを意味します。 `table_name` 通知パラメータは、変更されたテーブルの名前です。

```
note_info.event_name = "SyncComplete";
note_info.event_name_len = 12;
note_info.parms_type = ul_ev_note_info::P_NONE;
```

```
note_info.event_name = "TableModified";
note_info.event_name_len = 13;
note_info.parms_type = ul_ev_note_info::P_TABLE_NAME;
note_info.parms = table->name->data;
note_info.parms_len = table->name->len;
```

キューの操作

キューは、作成および破棄することができます。

CreateNotificationQueue は、現在の接続のイベント通知キューを作成します。キュー名は、接続ごとにスコープされるため、別々の接続で同じ名前を持つキューを作成できます。イベント通知が送信されると、データベース内で一致する名前を持つすべてのキューが、個別のインスタンスの通知を受け取ります。名前では、大文字と小文字は区別されません。デフォルトのキューは、キューが指定されていない場合に要求に応じて、各接続に対して作成されます。接続に対してその名前がすでに存在する場合や有効でない場合は、エラーが発生して呼び出しが失敗します。

DestroyNotificationQueue は、指定されたイベント通知キューを破棄します。キューに未読の通知がある場合は、警告が表示されます。未読の通知は破棄されます。接続のデフォルトのイベントキューが作成されている場合、接続が閉じると破棄されます。

イベントの操作

DeclareEvent は、登録して、トリガできるイベントを宣言します。Ultra Light では、データベースまたは環境での操作によってトリガされるシステムイベントの一部が事前に定義されています。イベント名は、ユニークにする必要があります。また、イベント名では大文字と小文字は区別されません。イベントが正常に宣言された場合は true、名前がすでに使用されているか、または無効な場合は false を返します。

RegisterForEvent は、イベントの通知を受信するキューを登録します。キュー名が指定されていない場合は、デフォルトの接続キューが暗黙で指定され、必要に応じて作成されます。特定のシステムイベントでは、そのイベントが適用されるオブジェクト名を指定できます。たとえば、TableModified イベントではテーブル名を指定できます。SendNotification とは異なり、登録された特定のキューのみイベントの通知を受信します。別の接続の同じ名前を持つ他のキューは、明示的に登録されている場合を除き、イベントの通知を受信しません。正常に登録された場合は true、キューまたはイベントが存在しない場合は false を返します。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

TriggerEvent は、イベントをトリガし、登録されているすべてのキューに通知を送信します。送信したイベント通知の数を返します。パラメータは、name=value; のペアとして指定できます。

通知の操作

SendNotification は、指定された名前と一致する、データベース内のすべてのキュー (現在の接続におけるキューを含む) に通知を送信します。この呼び出しはブロックしません。特別なキュー名の "*" を使用すると、すべてのキューに送信します。送信した通知の数 (一致するキューの数) を返します。パラメータは、name=value; のペアとして指定できます。

GetNotification は、イベント通知を読み込みます。この呼び出しは、通知が受信されるか、または指定された待機期間が終了するまでブロックします。待機をキャンセルするには、指定したキューに別の通知を送信するか、CancelGetNotification を使用します。通知を読み込んだら、ReadNotificationParameter を使用して追加のパラメータを取得します。イベントが読み込まれた場合は true、待機期間が終了したか、またはキャンセルされた場合は false を返します。

GetNotificationParameter は、GetNotification で読み込まれたイベント通知の名前付きパラメータを取得します。指定されたキューでの最後に読み込まれた通知のパラメータのみ取得できます。パラメータが見つかった場合は true、見つからなかった場合は false を返します。

CancelGetNotification は、指定された名前と一致する名前を持つすべてのキューで保留中の GetNotification 呼び出しをキャンセルします。影響を受けるキューの数を返します (必ずしも、ブロックされた読み込みの数ではありません)。

その他の考慮事項

- 通知キューおよびイベントの名前は 32 文字に制限されています。
- システムリソースを管理するには、通知の数を制限します。この制限を超えると、SQLE_EVENT_NOTIFICATION_QUEUE_FULL が通知され、保留中の通知が廃棄されます。

1.7.4 独立性レベル

独立性レベルは、あるトランザクションの操作が、同時に処理されている別のトランザクションの操作からどの程度参照できるかを定義します。

Ultra Light は、接続に対してデフォルトの独立性レベル、read_committed を使用します。Ultra Light のデフォルトの独立性レベルでは、コミットされていないローを独立させることによって、データの整合性を保持します。

独立性レベル	特性
0 - read_uncommitted	• ダーティリード、繰り返し不可能読み出し、幻ローを許可します。 • 同時トランザクションがローを変更しないこと、またはローに対しての変更がロールバックされないことは保証されません。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

独立性レベル	特性
1 - read_committed	• 繰り返し不可能読み出しと幻ローを許可します。 • ダーティリードを防ぐ。 • トランザクション中にクエリの結果が変更されないという保証はありません。

次のいずれかのアプローチを使用して、独立性レベルを read_committed から read_uncommitted に変更できます。

- SET OPTION SQL 文と isolation_level データベースオプションを使用します。
  たとえば次の文では、コミットされていない読み出しに独立性レベルが設定されます。

  ```
  SET OPTION isolation_level = 'READ_UNCOMMITTED'
  ```

- Ultra Light C++ API の場合は、ULConnection SetDatabaseOption メソッドを使用して独立性レベルを変更します。
  Ultra Light.NET API の場合は、ULConnection.BeginTransaction メソッドまたは ULDatabaseSchema.SetDatabaseOption メソッドを使用して、read_committed 独立性レベルでトランザクションを作成します。
  Ultra Light J API の場合は、Connection.setOption メソッドを使用します。

このセクションの内容:

read_uncommitted 独立性レベルの特性 [48 ページ]
Ultra Light を isolation_level 0 (read_uncommitted) で操作すると、いくつかの影響が生じることがあります。

関連情報

SET OPTION 文 [Ultra Light] [519 ページ]
Ultra Light isolation_level オプション [206 ページ]

1.7.4.1 read_uncommitted 独立性レベルの特性

Ultra Light を isolation_level 0 (read_uncommitted) で操作すると、いくつかの影響が生じることがあります。

- アプリケーションは、コミットされていないデータを読み込むことができます (ダーティリード)。このシナリオでは、コミットされていないデータベースのローにトランザクションがアクセスできるため、別のトランザクションによりロールバックされる可能性があります。これにより幻ローが発生します。幻ローとは元のクエリの後に追加されたローのことで、これにより、繰り返されて重複したクエリで返された結果セットの内容に相違が生じます。
- アプリケーションは、繰り返し不可能読み出しを実行できます。このシナリオでは、アプリケーションがデータベースからローを読み出し、他の操作の実行に移ります。そこへ、別のアプリケーションがローの更新や削除を行い、変更をコミットします。最初のアプリケーションが元のローを再度読み出すと、更新された情報が取得されるか、元のローが削除されたことを発見します。

例

たとえば、A と B という 2 つの接続があり、それぞれにトランザクションがあるとします。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1. 接続 A がクエリの結果セットを処理する場合、Ultra Light によって、現在のローのコピーがフェッチされてバッファに格納されます。

   注記

   ローを読み込んだりフェッチしたりしても、ローはロックされません。接続 A がローをフェッチしても変更はしない場合、接続 B はこのローを変更できます。

2. 接続 A は現在のローを変更するときに、バッファ内のコピーを変更します。接続 A が Update メソッドを呼び出すか結果セットを閉じると、バッファ内のコピーはデータベースに書き戻されます。
3. このローに対して書き込みロックが掛けられ、他のトランザクションがこのローを変更できなくなります。この変更は、接続 A がコミットを実行するまで、コミットされないままになります。
4. 変更によっては、接続 B が現在のローをフェッチすると、次の現象が発生する可能性があります。

接続 A による変更	結果 [1]
ローの削除	接続 B は結果セットの次のローを取得します。
ローの変更	接続 B はローの最新のコピーを取得します。

[1] 接続 A および B で使用されたクエリには、テンポラリテーブルは含まれていません。テンポラリテーブルは、他の副次的な影響を与える可能性があります。

関連情報

SET OPTION 文 [Ultra Light] [519 ページ]
Ultra Light isolation_level オプション [206 ページ]

1.7.5 Ultra Light データベースの検証

SQL Central のデータベース検証ウィザードや、Ultra Light データベース検証ユーティリティ、または Ultra Light API の ValidateDatabase メソッドなどのツールを使用して、データベースを検証します。

コンテキスト

データベースファイルの破損は、データベースサーバがデータベース内の破損部分にアクセスするまで判明しないことがあります。

警告

データベースの検証は、接続がデータベースに変更を加えていない場合に実行してください。これを守らなかった場合、実際に破損していなくても、データベースが破損したことを示すエラーがレポートされる可能性があります。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

Ultra Light データベースは、次のいずれかの方法で検証できます。

- SQL Central のデータベース検証ウィザード
- ulvalid コマンドラインユーティリティ
- Ultra Light API の ValidateDatabase メソッド

手順

1. SQL Central の左ペインで、Ultra Light データベースをクリックします。
2. ファイル > データベースの検証 の順にクリックします。
3. データベース検証ウィザードの指示に従います。

結果

データベースが検証されます。

関連情報

Ultra Light データベース検証ユーティリティ (ulvalid) [239 ページ]

1.7.6 Ultra Light データベースのバックアップとリカバリ

Ultra Light データベースを使用しているアプリケーションが予期せずに停止した場合、アプリケーションを再起動すると、データベースは自動的に一貫性が保たれた状態にリカバリされます。

予期せぬ障害が発生する前にメモリにフラッシュされたすべてのコミット済みトランザクションは、データベースに保持されます。障害発生時にフラッシュされていないすべてのトランザクションは、ロールバックされます。

Ultra Light データベースでは、リカバリの実行にトランザクションログを使用しません。各ローのステータス情報を保存して、リカバリ時のローの処理を決定します。

バックアップ

Ultra Light は、システム障害に対する保護機能は備えていますが、メディア障害に対する保護機能はありません。Ultra Light アプリケーションのバックアップの作成に最適な方法として、統合データベースとの同期を行うことが挙げられます。Ultra Light データベースをリストアするには、空のデータベースを作成し、統合データベースと同期してデータを移植します。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

関連情報

Ultra Light データベースのローのステータス管理 [542 ページ]
単一のトランザクションまたはグループ化されたトランザクションのフラッシュ [544 ページ]
Mobile Link クライアントとしての Ultra Light [71 ページ]

1.8 Ultra Light データベースのスキーマ

データベースの論理的なフレームワークをスキーマといいます。

Ultra Light データベースのスキーマ

Ultra Llight データベースのスキーマを更新するには、適切なデータ定義言語 (DDL) 文を使用します。または、ALTER DATABASE SCHEMA FROM FILE 文を使用し、SQL スクリプトを使用してスキーマ定義を変更します。

スキーマの変更には時間がかかる場合があります。たとえば、カラム型が変更された場合は、関連するテーブルにあるすべてのローを更新する必要があります。DDL 文は、次の場合に正常に実行されます。

- コミットされていないトランザクションがありません。
- 同期、準備されたが解放されていない文、実行中のデータベース操作など、データベースのその他のアクティブな使用がありません。

DDL 文を実行すると、DDL 文によるスキーマの変更が完了するまで、その他のデータベースの操作はブロックされます。

このセクションの内容:

Ultra Light のテーブルとカラム [52 ページ]
テーブルは、データを格納するため、またテーブル内のデータの関係を定義するために使用します。テーブルはローとカラムで構成されます。各カラムは電話番号や名前など情報の種類を特定し、各ローはデータのエントリを保持します。

Ultra Light インデックス [59 ページ]
インデックスとは、1 つ以上のテーブルカラムに含まれているデータ値の順序に基づく、テーブル内のローのポインタセットです。

Ultra Light ユーザ [64 ページ]
一般的な Ultra Light データベースには、1 つのユーザ ID とパスワードが含まれます。Ultra Light データベースは、特に指定しなければ、*DBA* のデフォルトのユーザ ID と *sql* のデフォルトのパスワードを使用して作成されます。

関連情報

Ultra Light データベーススキーマのアップグレードの配備 [125 ページ]

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

ALTER DATABASE SCHEMA FROM FILE 文 [Ultra Light] [479 ページ]

1.8.1 Ultra Light のテーブルとカラム

テーブルは、データを格納するため、またテーブル内のデータの関係を定義するために使用します。テーブルはローとカラムで構成されます。各カラムは電話番号や名前など情報の種類を特定し、各ローはデータのエントリを保持します。

Ultra Light データベースを初めて作成したときには、システムテーブルだけが表示されます。システムテーブルには Ultra Light のスキーマが保管されます。システムテーブルは、必要に応じて SQL Central で表示/非表示を切り替えることができます。

その後、アプリケーションに必要なテーブルを新しく追加できます。また、これらのテーブル内のデータをブラウズしたり、ソースデータベース内の既存のテーブル間や開いている宛先データベース間でデータをコピーアンドペーストしたりできます。

Ultra Light では、ベーステーブルだけを作成できます。ベーステーブルは、永続的なデータを保管するテーブルです。Ultra Light では、グローバルテンポラリテーブルや宣言されたテンポラリテーブルはサポートされていません。

このセクションの内容:

ローのパックとテーブル定義 [53 ページ]
Ultra Light では、アンパックされたローとパックされたローという、2 つの形式のローを扱います。

Ultra Light のテーブルの作成 [53 ページ]
永続的なリレーショナルデータを保持するには、ベーステーブルを作成してください。

Ultra Light テーブルへのカラムの追加 [54 ページ]
カラムの作成後、そのカラムを Ultra Light テーブルに追加します。

Ultra Light のカラム定義の変更 [55 ページ]
テーブルのカラム定義の構造を変更するには、さまざまなカラムの属性を変更するか、またはカラム全体を削除します。

Ultra Light テーブルの削除 [56 ページ]
必要がなくなったテーブルは削除してください。

Ultra Light テーブル内の情報のブラウズ [57 ページ]
Ultra Light データベースのテーブル内のデータを表示します。

Ultra Light データベースからのデータのコピーアンドペースト [58 ページ]
SQL Central では、コピーアンドペーストおよびドラッグアンドドロップを使用できます。したがって、データベース内または複数のデータベース間でオブジェクトを共有または移動できます。コピーアンドペーストまたはドラッグアンドドロップで、次の表に示すデータを共有できます。

SQL Central での ER (実体関連) 図の表示 [59 ページ]
SQL Central を使用して、Ultra Light データベースのテーブルの ER (実体関連) 図を表示および設定します。

関連情報

Ultra Light のシステムテーブル [240 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.8.1.1 ローのパックとテーブル定義

Ultra Light では、アンパックされたローとパックされたローという、2 つの形式のローを扱います。

アンパックされたロー

非圧縮形式です。個々のカラム値を読み込んだり書き込んだりするには、その前に、該当するローをアンパックする必要があります。

パックされたロー

アンパックされたローの圧縮形式です。各カラム値が圧縮され、ロー全体に必要なメモリが最小限に抑えられています。パックされたローのサイズは、各カラムに含まれている値に依存します。たとえば、同じテーブルに属する 2 つのローで、パックされたサイズが大きく異なる場合があります。LONG BINARY カラムと LONG VARCHAR カラムは、パックされたローとは別に格納されます。

Ultra Light には、パックされたローはデータベースページに収まらなければならないという制約があります。LONG BINARY カラムと LONG VARCHAR カラムは、パックされたローには格納されないので、ページサイズを超えてもかまいません。

テーブル定義は Ultra Light ランタイムがデータをパックする前のローについて記述していることを理解することが重要です。パックされたローのサイズは各カラムの値によって異なるので、パックされたローの要件が満たされているかどうかをテーブル定義を基にして事前に判断することはできません。この理由から、Ultra Light ではアンパックされたローがページに収まらないテーブルを定義できるようになっています。ローがページに収まるかどうかを確認するには、ロー自体を挿入または更新する必要があります。ローが収まらなかった場合、Ultra Light はエラーを検出して通知します。

注記

宣言するテーブルのサイズを自由に大きくすることはできません。Ultra Light では、宣言されたテーブルローサイズの上限は 64 KB に固定されています。アンパックされたローのサイズがこの上限を超えることができるテーブルを定義しようとすると、Ultra Light によって SQL エラーコード SQLE_MAX_ROW_SIZE_EXCEEDED (-1132) が生成されます。

関連情報

Ultra Light のシステムテーブル [240 ページ]
Ultra Light page_size 作成オプション [160 ページ]

1.8.1.2 Ultra Light のテーブルの作成

永続的なリレーショナルデータを保持するには、ベーステーブルを作成してください。

前提条件

Ultra Light アプリケーション内のテーブルには、プライマリキーが含まれます。プライマリキーは、Ultra Light データベースのローを統合データベースのローに関連付けるため、Mobile Link 同期を行うときにも必要です。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

手順

1. SQL Central を使用して Ultra Light データベースに接続します。
2. 左ウィンドウ枠で テーブルを右クリックし、 新規 テーブル の順にクリックします。
3. 新しいテーブルの名前を指定してください。フィールドに、新しいテーブルの名前を入力します。
4. 完了をクリックします。
5. ファイル メニューで、保存をクリックします。

結果

テーブルが作成されます。テーブルとそのデータは、それらを明示的に削除しないかぎり維持されます。

次のステップ

カラムを追加するか、またはインデックスを作成します。

関連情報

Ultra Light テーブルへのカラムの追加 [54 ページ]
CREATE TABLE 文 [Ultra Light] [495 ページ]

1.8.1.3 Ultra Light テーブルへのカラムの追加

カラムの作成後、そのカラムを Ultra Light テーブルに追加します。

前提条件

テーブルにすでにデータが格納されている場合、カラムを追加できるのは、そのカラムの定義にデフォルト値が含まれるか、そのカラムに NULL 値が許可される場合だけです。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

手順

1. SQL Central を使用して Ultra Light データベースに接続します。
2. 左ウィンドウ枠で、テーブルをダブルクリックします。
3. テーブルをダブルクリックします。
4. カラムタブでテーブルの下の空白スペースを右クリックし、新規 > カラム を選択します。
5. 新しいカラムの属性を設定し、変更を保存します。

結果

カラムがテーブルに追加されます。

関連情報

SQL データ型 [278 ページ]
CREATE TABLE 文 [Ultra Light] [495 ページ]
ALTER TABLE 文 [Ultra Light] [483 ページ]

1.8.1.4 Ultra Light のカラム定義の変更

テーブルのカラム定義の構造を変更するには、さまざまなカラムの属性を変更するか、またはカラム全体を削除します。

前提条件

変更したカラム定義は、カラムにすでに格納されているデータの条件を満たす必要があります。たとえば、カラムにすでに NULL のエントリがある場合は、カラムを変更して NULL を不許可にすることはできません。

手順

1. SQL Central を使用して Ultra Light データベースに接続します。
2. 左ウィンドウ枠で、テーブルをダブルクリックします。
3. テーブルをダブルクリックします。
4. カラム タブをクリックし、カラムの属性を変更します。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

5. ファイルメニューで、テーブルの保存をクリックします。

結果

新しいカラム属性でテーブルが保存されます。

関連情報

SQL データ型 [278 ページ]
ALTER TABLE 文 [Ultra Light] [483 ページ]

1.8.1.5 Ultra Light テーブルの削除

必要がなくなったテーブルは削除してください。

前提条件

次の条件を満たすテーブルを削除できます。

- パブリケーションでアーティクルとして使用されていない
- 別のテーブルの外部キーで参照されているカラムがない

上記以外の場合は、パブリケーションを変更するか、外部キーを削除してから、テーブルを削除します。

手順

1. SQL Central を使用して Ultra Light データベースに接続します。
2. 左ウィンドウ枠で、テーブルをダブルクリックします。
3. テーブルを右クリックして、削除をクリックします。
4. はいをクリックします。

結果

テーブルが削除されます。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

関連情報

DROP TABLE 文 [Ultra Light] [507 ページ]

1.8.1.6 Ultra Light テーブル内の情報のブラウズ

Ultra Light データベースのテーブル内のデータを表示します。

前提条件

データベースが接続され、選択されている必要があります。

コンテキスト

テーブルには、ユーザテーブルとシステムテーブルがあります。現在のデータベースの表示でシステムテーブルの表示と非表示を切り替えてテーブルをフィルタできます。Ultra Light には所有権の概念がないので、すべてのユーザがすべてのテーブルをブラウズできます。

手順

1. SQL Central を使用して Ultra Light データベースに接続します。
2. テーブルのリストを表示するには、テーブルをダブルクリックします。
3. テーブルデータを表示するには、テーブルを右クリックして、右ウィンドウ枠のデータタブをクリックします。

結果

テーブルとデータが表示されます。

関連情報

Ultra Light のシステムテーブル [240 ページ]

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

1.8.1.7 Ultra Light データベースからのデータのコピーアンドペースト

SQL Central では、コピーアンドペーストおよびドラッグアンドドロップを使用できます。したがって、データベース内または複数のデータベース間でオブジェクトを共有または移動できます。コピーアンドペーストまたはドラッグアンドドロップで、次の表に示すデータを共有できます。

ターゲット	結果
別の Ultra Light/SQL Anywhere データベース	新しいオブジェクトが作成され、元のオブジェクトのコードが新しいオブジェクトにコピーされます。
同じ Ultra Light データベース	オブジェクトのコピーが作成されるので、新しいオブジェクトの名前を変更します。

注記

Mobile Link で開いているデータベースのデータをコピーし、Ultra Light データベースにペーストできます。ただし、Ultra Light のデータを、Mobile Link で開いているデータベースにペーストすることはできません。

SQL Central

Ultra Light プラグインでは、次のいずれかのオブジェクトをコピーすると、そのオブジェクトの SQL もクリップボードにコピーされます。この SQL は、Interactive SQL やテキストエディタなどの他のアプリケーションに貼り付けることができます。たとえば、SQL Central でインデックスをコピーし、テキストエディタにペーストすると、そのインデックスの CREATE INDEX 文が表示されます。Ultra Light プラグインにある次のオブジェクトをコピーできます。

- アーティクル
- カラム
- 外部キー
- インデックス
- パブリケーション
- テーブル
- 一意性制約

Interactive SQL

Interactive SQL では、結果セットから別のオブジェクトにデータをコピーすることもできます。

- 結果を指定のオブジェクトにコピーするには、SELECT 文を使用します。
- データベース内の別の場所にある 1 つのローまたは複数のローをテーブルに挿入するには、INSERT 文を使用します。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

関連情報

INSERT 文 [Ultra Light] [511 ページ]
SELECT 文 [Ultra Light] [516 ページ]

1.8.1.8 SQL Central での ER (実体関連) 図の表示

SQL Central を使用して、Ultra Light データベースのテーブルの ER (実体関連) 図を表示および設定します。

コンテキスト

Ultra Light の場合、レイアウトのインポートやエクスポート、あるいはオーナによって表示されたテーブルのフィルタリングによってユーザが ER 図を設定することはできません。

手順

1. SQL Central で、*Ultra Light 17* プラグインを使用してデータベースに接続します。
2. データベースを選択し、右ウィンドウ枠の *ER 図*タブをクリックして ER 図を表示します。
3. ファイル > *ER 図のテーブルを選択* をクリックし、ER 図に表示するテーブルを指定します。
4. *OK* をクリックします。
5. 必要に応じ、図でオブジェクトを整列させます。

   オブジェクトをドラッグしてレイアウトを変更します。
6. (任意) テーブルをダブルクリックし、そのテーブルのカラム定義を表示します。

結果

SQL Central に ER 図が表示されます。

1.8.2 Ultra Light インデックス

インデックスとは、1 つ以上のテーブルカラムに含まれているデータ値の順序に基づく、テーブル内のローのポインタセットです。

インデックスが作成されると、データベースオブジェクトとして Ultra Light によって自動的に保持されます。Ultra Light は、クエリを最適化するときに、既存のインデックスをスキャンして、クエリで指定されているテーブルにインデックスがあるかどうか

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

を確認します。インデックスによってより短時間でローを返すことができる場合は、インデックスが使用されます。アプリケーションで Ultra Light のテーブル API を使用している場合は、ローをスキャンする順序を決定するインデックスを指定できます。

> **注記**
>
> インデックスによって、特にテーブルが大きい場合は、クエリのパフォーマンスを向上できます。クエリで特定のインデックスが使用されているかどうかを確認するには、Interactive SQL で実行プランを確認します。
>
> また、プランを返すメソッドがある PreparedStatement オブジェクトを Ultra Light アプリケーションに含めることもできます。

Ultra Light は、次のインデックスをサポートしています。インデックスは、シングルカラムまたはマルチカラム (複合インデックス) で作成できます。LONG VARCHAR カラムや LONG BINARY カラムをインデックス付けすることはできません。

インデックス	特性
プライマリキー	必須。ユニークキーのインスタンスです。プライマリキーは 1 つだけ持つことができます。インデックス付けされたカラムの値はユニークである必要があります。NULL は不可です。
外部キー [1]	省略可能です。インデックス付けされたカラムの値は重複していてかまいません。NULL を入力できるかどうかは、NULL を許可するようにカラムが作成されたかどうかで決まります。外部キーカラムの値は、参照されているテーブルに存在している必要があります。
ユニークキー [2]	省略可能です。インデックス付けされたカラムの値はユニークである必要があります。NULL は不可です。
ユニークでないインデックス	省略可能です。インデックス付けされたカラムの値は重複していてかまいません。NULL を指定できます。
ユニークインデックス	省略可能です。インデックス付けされたカラムの値が重複していてはいけません。NULL を指定できます。

[1] 外部キーは、プライマリキーまたはユニークキーを参照できます。

[2] 一意性制約と同じです。

複合インデックスについて

マルチカラムのインデックスは複合インデックスとも呼ばれます。インデックスにカラムを追加すると検索対象を限定できますが、2 カラムのインデックスを使用することと 2 つの別個のインデックスを使用することは異なります。たとえば、次の文では 2 カラムの複合インデックスが作成されます。

```
CREATE INDEX name
ON Employees ( Surname, GivenName )
```

複合インデックスは、最初のカラムだけでは高い選択性が得られない場合に役立ちます。たとえば、Surname と GivenName に対する複合インデックスは、従業員の姓が同じ場合に便利です。各従業員はユニークな ID を持っており、カラム Surname は追加の選択性を提供しないため、EmployeeID と Surname の複合インデックスは役に立ちません。

このセクションの内容:

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

インデックスを使用する場合 [61 ページ]
インデックスは、データをクエリするときのパフォーマンスを向上させます。

インデックスのタイプ [62 ページ]
Ultra Light では、インデックスのタイプにユニークキー、ユニークインデックス、ユニークでないインデックスがあります。これらのタイプでは、インデックスで許可されることが異なります。

Ultra Light インデックスの追加 [63 ページ]
データベースにインデックスを追加すると、検索プロセスが高速になります。

Ultra Light インデックスの削除 [64 ページ]
データベースから インデックスを削除します。

関連情報

インデックススキャンの作成と保持 [529 ページ]
Ultra Light の実行プラン [534 ページ]
Ultra Light.NET での ULTable クラスを使用したデータの作成と修正 [568 ページ]
Ultra Light C++ での ULTable クラスを使用したデータの作成と修正 [617 ページ]

1.8.2.1 インデックスを使用する場合

インデックスは、データをクエリするときのパフォーマンスを向上させます。

インデックスは次の場合に使用します。

Ultra Light で参照整合性を保つ必要がある場合

インデックスは Ultra Light に、テーブル内のローに一意性制約を強制するための手段も提供します。よく似ているデータにインデックスを追加する必要はありません。

特定のクエリのパフォーマンスがアプリケーションにとって重要である場合

インデックスによってクエリのパフォーマンスが向上し、そのクエリのパフォーマンスがアプリケーションにとって重要であり、そのクエリを頻繁に使用する場合は、インデックスを使用します。テーブルが非常に小さい場合を除き、インデックスによって検索のパフォーマンスが大幅に向上します。データを頻繁に検索する場合は一般にインデックスを使用することをおすすめします。

複雑なクエリがある場合

複雑なクエリ (たとえば、JOIN 句、GROUP BY 句、ORDER BY 句を使用するもの) は、インデックスを使用するとパフォーマンスが大幅に向上する可能性があります。ただし、パフォーマンスの向上の程度を確認するのは困難である場合があります。したがって、インデックスを使用した場合と使用しなかった場合でクエリをテストし、パフォーマンスを確認することをおすすめします。

Ultra Light テーブルのサイズが大きい場合

ローを検索する平均時間は、テーブルのサイズに比例して長くなります。したがって、非常に大きなテーブルの検索効率を向上させるには、インデックスの使用を検討してください。インデックスを使用すると、インデックスが付いているカラムについては、Ultra Light でローが短時間で見つかります。インデックスがない場合、Ultra Light ではテーブル内のすべ

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

てのローを検索し、ローが探索条件を満たすかどうかを確認する必要があるので、大きなテーブルの場合は時間がかかります。

Ultra Light クライアントアプリケーションで大量の挿入、更新、または削除の操作を行わない場合

Ultra Light ではインデックスがデータ自体とともに保持されるので、これらの操作では、インデックスによってパフォーマンスが下がります。したがって、インデックスの使用は、前述のように頻繁に問い合わせるデータだけに制限してください。Ultra Light のデフォルトのインデックス (プライマリキーと一意性制約のインデックス) で十分である可能性があります。

WHERE 句か **ORDER BY** 句またはその両方を使用する場合

これらの句の対象となるカラムにインデックスを使用すると、これらの句の評価を高速化できます。特にインデックスによってマルチカラムの ORDER BY 句を最適化できます。ただし、インデックスと ORDER BY 句でのカラムの配置がまったく同じである必要があります。

1.8.2.2 インデックスのタイプ

Ultra Light では、インデックスのタイプにユニークキー、ユニークインデックス、ユニークでないインデックスがあります。これらのタイプでは、インデックスで許可されることが異なります。

インデックスの特性	ユニークキー	ユニークインデックス	ユニークでないインデックス
インデックスカラムに同じ値があるローの重複するインデックスエントリを許可する	なし	なし	あり
インデックスカラムに NULL 値を許可する	なし	あり	あり

注記

ユニークキーには外部キーを作成できますが、ユニークインデックスには作成できません。

また、自分でキーカラムにインデックスを設定する必要はありません。Ultra Light によってユニークキーのインデックスが自動的に作成され、管理されます。

関連情報

Ultra Light インデックスの追加 [63 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.8.2.3 Ultra Light インデックスの追加

データベースにインデックスを追加すると、検索プロセスが高速になります。

前提条件

データベースが接続されている必要があります。

コンテキスト

注記

Ultra Light では重複する、または冗長なインデックスは検出されません。インデックスはデータベース内のデータとともに保持されるので、インデックスは慎重に追加してください。

手順

1. SQL Central を使用して Ultra Light データベースに接続します。
2. インデックスを右クリックし、新規 > インデックス の順にクリックします。
3. ウィザードの指示に従います。

結果

インデックスが作成されます。

例

従業員情報を追跡するデータベースでの従業員の姓の検索を高速化し、このインデックスに対するクエリのパフォーマンスをチューニングするには、次の文で EmployeeNames というインデックスを作成し、ハッシュサイズを 20 バイトに増やします。

```
CREATE INDEX EmployeeNames
ON Employees (Surname, GivenName)
WITH MAX HASH SIZE 20
```

設定したデフォルトの最大ハッシュサイズでインデックスが作成されます。デフォルト以外の値を使用するインデックスを作成するには、WITH MAX HASH SIZE `value` 句を使用してこのインデックスインスタンスの新しい値を設定します。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

関連情報

CREATE INDEX 文 [Ultra Light] [490 ページ]

1.8.2.4 Ultra Light インデックスの削除

データベースから インデックスを削除します。

手順

1. SQL Central を使用して Ultra Light データベースに接続します。
2. 左ウィンドウ枠で、インデックスをダブルクリックします。
3. インデックスを右クリックして、削除を選択します。
4. はいをクリックします。

結果

インデックスはデータベースから削除されます。

関連情報

DROP INDEX 文 [Ultra Light] [504 ページ]

1.8.3 Ultra Light ユーザ

一般的な Ultra Light データベースには、1 つのユーザ ID とパスワードが含まれます。Ultra Light データベースは、特に指定しなければ、*DBA* のデフォルトのユーザ ID と *sql* のデフォルトのパスワードを使用して作成されます。

ユーザスキーマの変更はオプションで、必須ではありません。ほとんどのアプリケーションは、データベースレベルの認証は不要で、アプリケーションとそのデータへのアクセスにはデバイスレベルのパスワード認証で十分であるとみなされます。

ユーザを認証しない主な理由としては、配備先が単一ユーザのデバイスである、ユーザがアプリケーションを起動するたびにプロンプトを表示するのは不便である、などがあります。

データベースレベルの認証が不要であれば、データベース接続文字列にユーザ ID またはパスワードを含める必要はありません。最も単純な Ultra Light 接続文字列は、*DBF*=`filename` です。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

カスタムユーザ認証インタフェースを使用して Ultra Light アプリケーションを開発する場合、Ultra Light データベースに格納されている Ultra Light ユーザ ID とパスワードのハッシュを効果的に使用して、ユーザが提供したクレデンシャルを検証できます。独自のパスワードハッシュアルゴリズムを作成する必要はありません。Ultra Light データベースにユーザを追加すると、そのユーザのユーザ ID とパスワードのハッシュが格納されます。さらに、UID と PWD の接続パラメータ、つまり *UID*=`username` と *PWD*=`password` を使用してデータベースに接続を試みることによって、アプリケーションでユーザ提供のクレデンシャルを検証できます。Ultra Light データベースに正常に接続できれば、そのユーザは認証されたことを意味します。

警告

SQL Anywhere のユーザとは異なり、Ultra Light ユーザの作成と管理は、オブジェクト所有権や特定のデータベースのロールと権限のためではなく、認証のみを目的としたものです。認証されたユーザは、データベースへの完全なアクセス権を取得します。

ユーザ ID とパスワードを作成すると、Ultra Light データベースへの接続を制御できますが、データベースファイル内のデータの機密性が保護されるわけではありません。コンテンツはプレーンテキストとして格納されるので、直接読み取ることができます。

データベースコンテンツの機密性を保護するためにファイルを暗号化する場合は、ユーザ ID とパスワードの代わりに暗号化キーを使用してユーザを認証できます。

データがプレーンテキストとして格納されないように、ファイルを難読化して格納することもできますが、このアプローチでもデータの機密性が完全に保護されるわけではありません。

注記

Ultra Light ユーザ ID は、Mobile Link ユーザ名とは異なります。

制限事項

Ultra Light ユーザ ID には、次の制限事項が適用されます。

- Ultra Light では、データベースごとに最大 4 つのユニークなユーザ ID がサポートされます。
- ユーザ ID とパスワードは、SQL Central、SQL 文、またはアプリケーションの Ultra Light API メソッドを使用して変更できます。
- ユーザ ID の長さは 31 文字までに制限されます。
- ユーザ ID に、先頭の一重引用符 (')、先頭の二重引用符 (")、またはセミコロン (;) を含めることはできません。
- ユーザ ID は常に大文字と小文字が区別されません。パスワードは常に大文字と小文字が区別されます。
- ユーザ ID の名前を変更することはできません。新しいユーザ ID を追加して、既存のユーザ ID を既存のデータベース接続から削除する必要があります。
- Ultra Light API を使用してユーザをプログラムによってリストすることはできません。データベースの既存ユーザをリストするには、データベースツールを使用する必要があります。
- Ultra Light データベースに初めて接続するときの *UID* と *PWD* は、データベースの作成時に設定した値です。これらの接続パラメータが指定されていない場合、Ultra Light は、*DBA* のユーザ ID と *sql* のパスワードを使用して接続を試みます。データベースの作成時にユーザ名とパスワードを明示的に設定しなかった場合は、データベースに接続するときにユーザ名とパスワードを入力する必要はありません。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

このセクションの内容:

Ultra Light ユーザの管理に使用する接続パラメータ [66 ページ]
UID と *PWD* の接続パラメータを使用して、Ultra Light データベースでユーザを作成または認証することができます。

SQL 文を使用した Ultra Light ユーザの管理 [68 ページ]
CREATE USER、ALTER USER、DROP USER の SQL 文を使用して、Ultra Light データベースのユーザを管理できます。

SQL Central を使用した Ultra Light ユーザの作成 [69 ページ]
SQL Central を使用して Ultra Light データベースのユーザを作成できます。

SQL Central を使用した Ultra Light ユーザの削除 [70 ページ]
SQL Central を使用して Ultra Light データベースのユーザを明示的に削除できます。

関連情報

データベースセキュリティ [35 ページ]
Ultra Light DBKEY 接続パラメータ [187 ページ]
Ultra Light UID 接続パラメータ [196 ページ]
Ultra Light PWD 接続パラメータ [192 ページ]

1.8.3.1 Ultra Light ユーザの管理に使用する接続パラメータ

UID と *PWD* の接続パラメータを使用して、Ultra Light データベースでユーザを作成または認証することができます。

注記

接続パラメータの代わりに、アプリケーション内の次の Ultra Light API メソッドを使用して、Ultra Light データベースへのユーザのアクセス権を付与したり、取り消したりすることができます。

- ULConnection.GrantConnectTo メソッド [Ultra Light C++]
- ULConnection.RevokeConnectFrom メソッド [Ultra Light C++]
- ULGrantConnectTo メソッド [Ultra Light Embedded SQL]
- ULRevokeConnectFrom メソッド [Ultra Light Embedded SQL]
- ULConnection.GrantConnectTo メソッド [Ultra Light.NET]
- ULConnection.RevokeConnectFrom メソッド [Ultra Light.NET]

付与と取り消しのメソッドは、Ultra Light J API では使用できません。Android デバイスの場合は、ConfigPersistent.setConnectionString メソッドを使用して、UID と PWD の接続パラメータを設定します。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

Ultra Light データベースでの接続パラメータの使用

ほとんどの Ultra Light API では、DatabaseManager オブジェクトの createDatabase メソッドを使用して、接続パラメータと作成パラメータを指定し、新しい Ultra Light データベースを作成できます。

次に、Ultra Light C++ API の CreateDatabase メソッドに *UID* と *PWD* のパラメータを渡して、新しい Ultra Light データベースのデフォルトユーザを作成する例を示します。

```
ULConnection * conn;
ULError ulerr;
ULDatabaseManager::CreateDatabase("dbf=sample.udb;uid=default-name;pwd=default-
password", &ulerr);
```

次に、Ultra Light C++ API の OpenConnection メソッドに *UID* と *PWD* のパラメータを渡して、既存の Ultra Light データベースでユーザを認証する例を示します。

```
ULConnection * conn;
ULError ulerr;
ULDatabaseManager::OpenConnection("dbf=sample.udb;uid=test-name;pwd=test-password",
&ulerr);
```

Android デバイスの Ultra Light データベースでの接続パラメータの使用

Ultra Light J API の場合、Ultra Light J API の Configuration オブジェクトの setConnectionString メソッドを使用して、ユーザを作成または認証できます。

次に、Ultra Light J API の createDatabase メソッドに *UID* と *PWD* のパラメータを渡して、新しい Ultra Light データベースのデフォルトユーザを作成する例を示します。

```
ConfigFile config =
    DatabaseManager.createConfigurationFileAndroid("DBname.udb",
getApplicationContext());
config.setConnectionString("uid=default-name;pwd=default-password");
Connection conn = DatabaseManager.createDatabase(config);
```

次に、Ultra Light J API の connect メソッドに *UID* と *PWD* のパラメータを渡して、既存の Ultra Light データベースでユーザを認証する例を示します。

```
ConfigFile config =
    DatabaseManager.createConfigurationFileAndroid("DBname.udb",
getApplicationContext());
config.setConnectionString("uid=test-name;pwd=test-password");
Connection conn = DatabaseManager.connect(config);
```

setConnectionString メソッドの代わりに、setPassword または setUserName メソッドを使用して、ユーザを作成または認証することもできます。

関連情報

Ultra Light UID 接続パラメータ [196 ページ]

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

Ultra Light PWD 接続パラメータ [192 ページ]

1.8.3.2 SQL 文を使用した Ultra Light ユーザの管理

CREATE USER、ALTER USER、DROP USER の SQL 文を使用して、Ultra Light データベースのユーザを管理できます。

注記

SQL 文の代わりに、アプリケーション内の次の Ultra Light API メソッドを使用して、Ultra Light データベースへのユーザのアクセス権を付与したり、取り消したりすることができます。

- ULConnection.GrantConnectTo メソッド [Ultra Light C++]
- ULConnection.RevokeConnectFrom メソッド [Ultra Light C++]
- ULGrantConnectTo メソッド [Ultra Light Embedded SQL]
- ULRevokeConnectFrom メソッド [Ultra Light Embedded SQL]
- ULConnection.GrantConnectTo メソッド [Ultra Light.NET]
- ULConnection.RevokeConnectFrom メソッド [Ultra Light.NET]

付与と取り消しのメソッドは、Ultra Light J API では使用できません。Android デバイスの場合は、CREATE USER、ALTER USER、または DROP USER の SQL 文を文字列変数として構成し、これを Connection.prepareStatement メソッドに渡します。

例

次に、Ultra Light J API を使用して Ultra Light データベースに接続し、CREATE USER 文を使用して新しいユーザを作成する例を示します。

```
ConfigFile config =
    DatabaseManager.createConfigurationFileAndroid("DBname.udb",
getApplicationContext());
Connection conn = DatabaseManager.connect(config);
String sql_string = "CREATE USER test-user IDENTIFIED BY test-password";
PreparedStatement authenticator = conn.prepareStatement(sql_string);
authenticator.execute();
authenticator.close();
```

関連情報

CREATE USER 文 [Ultra Light] [502 ページ]
ALTER USER 文 [Ultra Light] [487 ページ]
DROP USER 文 [Ultra Light] [507 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.8.3.3 SQL Central を使用した Ultra Light ユーザの作成

SQL Central を使用して Ultra Light データベースのユーザを作成できます。

コンテキスト

> **注記**
>
> SQL Central の代わりに、アプリケーション内の次の Ultra Light API メソッドを使用して、Ultra Light データベースへのユーザアクセス権を付与できます。
>
> - ULConnection.GrantConnectTo メソッド [Ultra Light C++]
> - ULGrantConnectTo メソッド [Ultra Light Embedded SQL]
> - ULConnection.GrantConnectTo メソッド [Ultra Light.NET]
>
> 付与のメソッドは、Ultra Light J API では使用できません。Android デバイスの場合は、CREATE USER 文を文字列変数として構成し、これを Connection.prepareStatement メソッドに渡します。

手順

1. SQL Central を使用して Ultra Light データベースに接続します。
2. **ユーザ**フォルダを右クリックし、**新規** > **ユーザ** の順にクリックします。
3. ウィザードの指示に従います。

結果

新しいユーザが作成されます。

関連情報

CREATE USER 文 [Ultra Light] [502 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.8.3.4 SQL Central を使用した Ultra Light ユーザの削除

SQL Central を使用して Ultra Light データベースのユーザを明示的に削除できます。

コンテキスト

> **注記**
>
> SQL Central, の代わりに、アプリケーション内の次の Ultra Light API メソッドを使用して、Ultra Light データベースへのユーザのアクセス権を拒否できます。
>
> - ULConnection.RevokeConnectFrom メソッド [Ultra Light C++]
> - ULRevokeConnectFrom メソッド [Ultra Light Embedded SQL]
> - ULConnection.RevokeConnectFrom メソッド [Ultra Light.NET]
>
> 拒否のメソッドは、Ultra Light J API では使用できません。DROP USER 文を文字列変数として構成し、これを Connection.prepareStatement メソッドに渡します。

手順

1. SQL Central を使用して Ultra Light データベースに接続します。
2. 左ウィンドウ枠で、ユーザフォルダをダブルクリックします。
3. ユーザを右クリックし、削除をクリックします。

結果

ユーザがデータベースから削除されます。

関連情報

DROP USER 文 [Ultra Light] [507 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.9 Mobile Link クライアントとしての Ultra Light

Mobile Link クライアントとして動作するよう、Ultra Light を設定できます。

このセクションの内容:

Ultra Light クライアント [71 ページ]
Ultra Light データベースを同期するには、アプリケーションで同期パラメータを設定する必要があります。

Microsoft ActiveSync 同期の概要 [90 ページ]
Microsoft ActiveSync を通じた同期は数ステップに要約できます。

Ultra Light の同期パラメータ [90 ページ]
同期パラメータは、Ultra Light データベースと Mobile Link サーバ間の同期を制御します。

Ultra Light のネットワークプロトコルのオプション [116 ページ]
アプリケーションでネットワークプロトコルを設定する必要があります。

1.9.1 Ultra Light クライアント

Ultra Light データベースを同期するには、アプリケーションで同期パラメータを設定する必要があります。

これらのパラメータにより、Mobile Link サーバのアドレスおよび他の必要な情報の特定、同期機能の呼び出し、または SYNCHRONIZE SQL 文の実行が行われます。選択するオプションは、使用している API によって異なります。

このセクションの内容:

Ultra Light クライアントの同期の動作のカスタマイズ [71 ページ]
Ultra Light にカスタム同期のサポートを追加するには、次に示すいくつかの作業を行います。

Ultra Light でのプライマリキーの一意性 [73 ページ]
Ultra Light では、Mobile Link でサポートされている任意の方法でプライマリキーの一意性を維持できます。

Ultra Light クライアントの同期設計 [77 ページ]
Ultra Light データベース内のすべてのデータは、デフォルトで同期されます。

Mobile Link ファイル転送 [85 ページ]
Ultra Light では、Mobile Link サーバを使用してファイルを転送する機能がサポートされています。

Ultra Light パブリケーション [86 ページ]
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。

1.9.1.1 Ultra Light クライアントの同期の動作のカスタマイズ

Ultra Light にカスタム同期のサポートを追加するには、次に示すいくつかの作業を行います。

複数のリモートクライアントを含む同期モデル内でプライマリキーの一意性を維持します。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

必須。同期システムでは、プライマリキーは、異なるデータベース (リモートと統合) 内の同じローを識別する唯一の方法であり、競合を検出する唯一の方法です。したがって、複数のクライアントがある場合は、以下の規則に従う必要があります。

- 同期される各テーブルには、プライマリキーが存在する必要があります。
- プライマリキーの値は更新しません。
- プライマリキーは、同期されるすべてのデータベース間でユニークでなければなりません。

端数データが失われないように日付カラムを設定します。

SQL Anywhere 統合データベースの場合、概してこれは問題にはなりません。ただし、Oracle などのデータベースでは、互換性の問題を考慮する必要がある場合があります。たとえば、Ultra Light と Oracle のデータベースでは、タイムスタンプ精度が同じである必要があります。また、Oracle データベースに TIMESTAMP を追加して、Ultra Light リモートデータベースから統合データベースにデータをアップロードするときに秒の小数点以下のデータが失われないようにする必要があります。
統合データベースにアップロードするデータサブセットを指定します。

省略可能です。デフォルトですべてのデータを同期しない場合にのみ行います。同期するデータを指定するには、1 つまたは複数のサブセット方法を使用します。

たとえば、優先度の高いデータのパブリケーションを作成することができます。その後、アプリケーションで無線ネットワークを介してこのデータを同期できます。無線ネットワークには使用コストがかかるので、このコストは、業務に必要なものに制限する必要があります。緊急でないデータは後でクレードルから同期できます。
Ultra Light アプリケーションから同期を初期化し、セッションに関するパラメータを指定します。

必須。同期のプログラミング時には、セッションの設定を行ってから、同期操作を開始します。

セッションの設定を行うには、同期の通信ストリーム (ネットワークプロトコル) とそのストリームのパラメータを選択し、同期スクリプトのバージョンを設定し、Mobile Link ユーザを識別します。ただし、他にも設定できるパラメータがあります。たとえば、upload_only パラメータと download_only パラメータを使用して同期の方向をデフォルトの双方向から一方向に変更できます。

その他の重要な同期の動作は、Mobile Link サーバで Mobile Link 同期スクリプトを使用して制御します。これらを以下に示します。

- Ultra Light リモートデータベース内のテーブルに更新または挿入としてダウンロードするデータ
- リモートデータベースからアップロードされた変更内容に必要な処理

自分専用の同期スクリプトを作成すれば、リモートデータベース間でデータを適切に分割できます。

関連情報

Ultra Light でのプライマリキーの一意性 [73 ページ]
Ultra Light クライアントの同期設計 [77 ページ]
Ultra Light アプリケーションの同期の設定 [83 ページ]
Ultra Light precision 作成オプション [162 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.9.1.2 Ultra Light でのプライマリキーの一意性

Ultra Light では、Mobile Link でサポートされている任意の方法でプライマリキーの一意性を維持できます。

方法の一つとして、GLOBAL AUTOINCREMENT カラムを使用できます。GLOBAL AUTOINCREMENT は AUTOINCREMENT と同じですが、ドメインが分割されます。Ultra Light では、データベースのグローバルデータベース ID に割り当てられた分割からのみカラムの値が設定されます。各 Ultra Light データベースには、ユニークな整数のグローバルデータベース ID が割り当てられます。

もう 1 つの方法として、UUID プライマリキーカラムを使用できます。UUID では、必要なデータの量は増えますが、別個のデータベース識別子は不要です。

このセクションの内容:

Ultra Light の GLOBAL AUTOINCREMENT カラム [73 ページ]
Ultra Light 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。

最後に割り当てられた GLOBAL AUTOINCREMENT 値の調べ方 [74 ページ]
最後の挿入操作中に選択された GLOBAL AUTOINCREMENT の値は、取り出すことができます。

分割サイズ [75 ページ]
この分割サイズには任意の正の整数を設定できますが、サイズの値がいずれの分割においても不足しないように設定する必要があります。

関連情報

Ultra Light global_database_id オプション [205 ページ]

1.9.1.2.1 Ultra Light の GLOBAL AUTOINCREMENT カラム

Ultra Light 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。

これらのカラム ID を自動的に増分するには、その前に Ultra Light データベースのグローバルデータベース ID を設定する必要があります。

> ⚠ 警告
>
> Mobile Link 同期を介してダウンロードされた GLOBAL AUTOINCREMENT カラムの値は、GLOBAL AUTOINCREMENT 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。

global_database_id データベースオプションを使用すると、Ultra Light データベースに値を設定できます。Ultra Light 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。

Ultra Light は、Ultra Light のデータベース番号でユニークに識別された分割を使用して、カラムにデフォルト値を設定します。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

Ultra Light は、次の規則に従います。

- 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 です。ここで、p は分割サイズ、n はグローバル ID 番号を表します。
- 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ります。
- 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ません。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ません。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。
  たとえば、Ultra Light データベースにグローバル ID 番号 1 を割り当て、分割サイズが 1000 の場合、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 ID 番号は負の値には設定できないので、GLOBAL AUTOINCREMENT カラムの値は常に正の数です。ID 番号の最大値を制限するのは、カラムデータ型と分割サイズだけです。
- グローバル ID の値を設定しないか、分割内の値がなくなった場合は、カラムに 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。

GLOBAL AUTOINCREMENT と宣言されたカラムに使用可能な値がなくなったか、なくなりそうになったら、新しいグローバルデータベース ID を設定する必要があります。GLOBAL AUTOINCREMENT の値は、グローバル ID 番号によって識別される分割から、最大値に達するまで選択されます。値を超えると、NULL 値が生成され始めます。新しいグローバルデータベース ID 番号を割り当てることで、Ultra Light で別の分割から適切な値を設定できるようになります。

新しいグローバルデータベース ID を選択する方法の 1 つとして、未使用のグローバルデータベース ID の値のプールを管理できます。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。

> **注記**
>
> Ultra Light の API 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。

関連情報

分割サイズ [75 ページ]
Ultra Light global_database_id オプション [205 ページ]
Ultra Light global_database_id オプション [205 ページ]

1.9.1.2.2 最後に割り当てられた GLOBAL AUTOINCREMENT 値の調べ方

最後の挿入操作中に選択された GLOBAL AUTOINCREMENT の値は、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。次のいずれかを使用して値を確認できます。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

Ultra Light for C/C++

ULConnection オブジェクトの GetLastIdentity 関数を使用します。

Ultra Light.NET

ULConnection クラスの LastIdentity プロパティを使用します。

API Ultra Light J

Connection インタフェースの GetLastIdentity メソッドを使用します。

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。

> **注記**
>
> ときには、1 つの INSERT 文に GLOBAL AUTOINCREMENT 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。

1.9.1.2.3 分割サイズ

この分割サイズには任意の正の整数を設定できますが、サイズの値がいずれの分割においても不足しないように設定する必要があります。

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。それ以外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。

Ultra Light アプリケーションと SQL 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。他のデータベースの一貫性を保つ場合は、分割サイズを明示的に宣言します。

このセクションの内容:

GLOBAL AUTOINCREMENT カラムの分割サイズの変更 [76 ページ]
GLOBAL AUTOINCREMENT カラムの分割サイズを変更します。

関連情報

CREATE TABLE 文 [Ultra Light] [495 ページ]
ALTER TABLE 文 [Ultra Light] [483 ページ]

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

1.9.1.2.3.1 GLOBAL AUTOINCREMENT カラムの分割サイズの変更

GLOBAL AUTOINCREMENT カラムの分割サイズを変更します。

前提条件

Ultra Light データベースに接続されている必要があります。

コンテキスト

GLOBAL AUTOINCREMENT カラムの分割サイズを大きくすると、分割内のサイズの不足がほとんどなくなります。

手順

1. カッコ内に分割サイズを指定して、GLOBAL AUTOINCREMENT カラムを持つテーブルを作成します。

   次の SQL コードを実行します。

   ```
   CREATE TABLE customer (
      id   INT          DEFAULT GLOBAL AUTOINCREMENT (5000),
      name VARCHAR(128) NOT NULL,
      PRIMARY KEY (id)
   )
   ```

   2 つのカラム顧客 ID 番号を保持する整数カラムと顧客名を保持する文字列カラム) を持つ単純な参照テーブルが作成されます。ID の分割サイズは 5000 です。
2. SQL Central で Ultra Light データベースに接続します。
3. 顧客テーブルの ID カラムを右クリックして、プロパティをクリックします。
4. 値タブをクリックします。
5. 分割サイズフィールドに、5000 を超える正の整数を入力します。

結果

分割サイズフィールドに入力した値に応じて、ID カラムの分割サイズが更新されます。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.9.1.3 Ultra Light クライアントの同期設計

Ultra Light データベース内のすべてのデータは、デフォルトで同期されます。

Mobile Link リモートデータベースとして Ultra Light を初めて配備する場合、最初はデフォルトの動作を使用することを検討します。

同期処理に慣れてきたら、同期操作の動作をカスタマイズして、より複雑なビジネス論理を展開できます。カスタム同期動作を設計するときは、次の点を確認する必要があります。業務上の要件が単純である場合は、単一の同期機能だけを使用できます。ただし、非常に複雑な配備では、目的の同期動作を設定するために複数の同期機能の使用が必要になる場合もあります。

設計時の確認事項	"はい" の場合の処理
統合データベースから変更をダウンロードするが、ローカルの変更をデータベースにアップロードしないかどうか。	download_only テーブル名サフィックスで、同期をダウンロードするのみのテーブルを識別できます。ローカルテーブルに対して行われた変更は、統合データベースにアップロードされません。
同期からテーブルを除外するかどうか。	テーブル名サフィックス nosync を使用して、同期しないテーブルを指定できます。
データが変更されてないときも含め、常にテーブル全体を同期するかどうか。	テーブル名サフィックス allsync を使用すると、変更が検出されなくてもテーブル全体を同期できます。
テーブル全体を同期するか、特定の条件を満たすローだけを同期するか。重要度または緊急度により、同期の優先度が高いデータがあるかどうか。	パブリケーションには、同期が必要なテーブルをリストするアーティクルが含まれます。アーティクルには、ローが定義した基準を満たすかどうかに基づいてアップロードするローを指定する WHERE 句を含めることができます。 複数のパブリケーションを使用すると、特定の Ultra Light データを他のデータよりも前にアップロードする必要がある場合の優先度の問題に対処できます。
外部キーの循環があるため、同期するテーブルの順序を指定する必要があるかどうか。	TableOrder 同期パラメータを使用すると、外部キーの循環があるときの同期操作の順序を指定できます。ただし、外部キーの循環は一般に Ultra Light にはお奨めしません。
同期の動作を制御するかどうか。たとえば、アップロードと同時にダウンロードを行うかどうか。また、同期の方向を双方向から一方向に変更するかどうか。	次の場所で適切な同期パラメータを使用します。 • アプリケーションの同期構造体 (または同期列挙) • ulsync ユーティリティの -e オプション
Ultra Light クライアントで TLS を有効にするかどうか。	選択した暗号化アルゴリズムにより、デバイスで実行するプラットフォームに従ってデバイスを設定する方法が決まります。

このセクションの内容:

Ultra Light 非同期テーブル [78 ページ]
SYNCHRONIZE OFF を使用してテーブルを作成することで、テーブル全体をアップロード操作から除外することを指定できます。

Ultra Light ダウンロード専用テーブル [79 ページ]
SYNCHRONIZE DOWNLOAD を使用してテーブルを作成することにより、テーブル全体をアップロード操作から除外します。

Ultra Light の allsync (すべて同期) テーブル [80 ページ]

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

SYNCHRONIZE ALL を使用してテーブルを作成または変更すると、前回の同期セッション後に何も変更されていなくてもすべてのテーブルのデータが同期されるように、アップロード時の同期の動作を指定できます。

Ultra Light でのデータのパブリッシュ [80 ページ]
SQL Central または SQL を使用してパブリケーションを Ultra Light データベースに追加します。

Ultra Light のテーブルの順序 [82 ページ]
TableOrder 同期パラメータを設定することで、同期操作の順序を制御できます。

Ultra Light アプリケーションの同期の設定 [83 ページ]
Ultra Light では、設定されている通信ストリーム (ネットワークプロトコル) で Mobile Link サーバとの特定の接続を開始することで同期が開始されます。

関連情報

Ultra Light の同期パラメータ [90 ページ]
Ultra Light のネットワークプロトコルのオプション [116 ページ]

1.9.1.3.1 Ultra Light 非同期テーブル

SYNCHRONIZE OFF を使用してテーブルを作成することで、テーブル全体をアップロード操作から除外することを指定できます。

これらの非同期テーブルは、統合データベースでは不要なクライアント固有の永続的なデータ用に使用できます。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。

> **注記**
>
> テーブルの同期タイプは、アップロードが必要な未同期の変更がない場合にのみ変更できます。

_nosync サフィックスを含むテーブルを作成し、名前を変更する場合は、_nosync サフィックスが保持されるようにする必要があります。たとえば、次の RENAME 句を指定した ALTER TABLE 文は、変更した名前の末尾が nosync ではなくなるので使用できません。

```
ALTER TABLE purchase_comments_nosync
RENAME comments
```

修正するには、次のようにこのサフィックスを含む文に書き換えます。

```
ALTER TABLE purchase_comments_nosync
RENAME comments_nosync
```

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。

> **注記**
>
> SYNCHRONIZE OFF 句でテーブルを作成または変更する代わりに、テーブル名に *_nosync* フレーズを付加して非同期テーブルに変更することができます。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

関連情報

Ultra Light パブリケーション [86 ページ]
Download Only 同期パラメータ [97 ページ]
Upload Only 同期パラメータ [113 ページ]
Additional Parameters 同期パラメータ [92 ページ]
CREATE TABLE 文 [Ultra Light] [495 ページ]
ALTER TABLE 文 [Ultra Light] [483 ページ]
CREATE PUBLICATION 文 [Ultra Light] [492 ページ]

1.9.1.3.2 Ultra Light ダウンロード専用テーブル

SYNCHRONIZE DOWNLOAD を使用してテーブルを作成することにより、テーブル全体をアップロード操作から除外します。

これらのテーブルを統合データベースに同期しないテーブル用に使用できます。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。

> **注記**
>
> テーブルの同期タイプは、アップロードが必要な未同期の変更がない場合にのみ変更できます。

> **注記**
>
> SYNCHRONIZE DOWNLOAD 句でテーブルを作成または変更する代わりに、テーブル名に *_download_only* フレーズを付加してダウンロード専用テーブルに変更することができます。

関連情報

Ultra Light パブリケーション [86 ページ]
Download Only 同期パラメータ [97 ページ]
Upload Only 同期パラメータ [113 ページ]
Additional Parameters 同期パラメータ [92 ページ]
CREATE TABLE 文 [Ultra Light] [495 ページ]
ALTER TABLE 文 [Ultra Light] [483 ページ]
CREATE PUBLICATION 文 [Ultra Light] [492 ページ]

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

1.9.1.3.3 Ultra Light の allsync (すべて同期) テーブル

SYNCHRONIZE ALL を使用してテーブルを作成または変更すると、前回の同期セッション後に何も変更されていなくてもすべてのテーブルのデータが同期されるように、アップロード時の同期の動作を指定できます。

> **注記**
>
> テーブルの同期タイプは、アップロードが必要な未同期の変更がない場合にのみ変更できます。

一部の Ultra Light アプリケーションでは、SYNCHRONIZE ALL TABLES に保管できるユーザ固有またはクライアント固有のデータが必要です。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して、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。たとえば、Ultra Light アプリケーションを使用してチャネルやトピックの中から関連のあるものを示し、この情報を使って適切なローをダウンロードする場合などが該当します。

> **注記**
>
> SYNCHRONIZE ALL 句でテーブルを作成または変更する代わりに、テーブル名に フレーズを付加して *_allsync* テーブルに変更することができます。

関連情報

Ultra Light パブリケーション [86 ページ]
Download Only 同期パラメータ [97 ページ]
Upload Only 同期パラメータ [113 ページ]
Additional Parameters 同期パラメータ [92 ページ]
CREATE TABLE 文 [Ultra Light] [495 ページ]
ALTER TABLE 文 [Ultra Light] [483 ページ]
CREATE PUBLICATION 文 [Ultra Light] [492 ページ]

1.9.1.3.4 Ultra Light でのデータのパブリッシュ

SQL Central または SQL を使用してパブリケーションを Ultra Light データベースに追加します。

コンテキスト

パブリケーションは、同期するデータを指定するアーティクルを定義します。

パブリケーションアーティクルは 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。特定のテーブルのローのサブセットを定義する場合は、オプションで述部 (WHERE 句) を含めることができます。パブリケ

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

ーションは、SYNCHRONIZE OFF を使用してテーブルを作成する方法よりも柔軟です。Ultra Light データベースのデータサブセットを別々に同期させるには、複数のパブリケーションを使用します。パブリケーションをアップロード専用またはダウンロード専用の同期パラメータと組み合わせると、優先度の高い変更を効率よく同期することが可能です。

> **注記**
>
> Ultra Light でのユーザパブリケーションの最大数は 63 です。

Ultra Light パブリケーションでは、カラムのサブセットの定義と、SQL Anywhere で使用できる SUBSCRIBE BY 句がサポートされていません。Ultra Light テーブルのカラムが統合データベースのテーブルと正確に一致しない場合は、Mobile Link スクリプトを使用して、これらの違いを解消してください。

パブリケーションでテーブル同期順序を設定する必要はありません。配備においてテーブル順序が重要な場合は、Ultra Light データベースを同期するときに TableOrder 同期パラメータを設定して、テーブル順序を設定できます。

手順

1. Ultra Light プラグインを使用して、Ultra Light 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右クリックし、新規 > パブリケーション の順にクリックします。
3. 新しい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、一致するテーブルリストから 1 つのテーブルをクリックします。追加をクリックします。

   右側の選択したテーブルリストに、そのテーブルが表示されます。
5. テーブルを追加します。
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。
7. 完了をクリックします。

結果

新しいパブリケーションが作成されます。

関連情報

Ultra Light パブリケーション [86 ページ]
Download Only 同期パラメータ [97 ページ]
Upload Only 同期パラメータ [113 ページ]
Additional Parameters 同期パラメータ [92 ページ]
CREATE PUBLICATION 文 [Ultra Light] [492 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.9.1.3.5 Ultra Light のテーブルの順序

TableOrder 同期パラメータを設定することで、同期操作の順序を制御できます。

同期のテーブル順序を指定する場合は、TableOrder パラメータをプログラムで使用するか、テスト中に ulsync ユーティリティの一部として使用します。TableOrder パラメータは、アップロードするテーブルの順序を指定します。

テーブル順序は、Ultra Light データベースに次のいずれかがある場合にのみ明示的に設定します。

- 外部キー循環。循環に含まれるすべてのテーブルをリストする必要があります。
- 統合データベースとは異なる外部キー関係

このセクションの内容:

外部キー循環に関連する同期の問題の防止 [82 ページ]
テーブル順序は、外部キー循環を使用する Ultra Light データベースに特に重要です。

関連情報

Additional Parameters 同期パラメータ [92 ページ]

1.9.1.3.5.1 外部キー循環に関連する同期の問題の防止

テーブル順序は、外部キー循環を使用する Ultra Light データベースに特に重要です。

循環が形成されるようにテーブルをリンクさせると、循環が発生します。ただし、統合データベースと Ultra Light リモートデータベース間の循環が異なる場合は、外部キー循環は複雑になるので推奨しません。

外部キー循環がある場合は、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テーブルを並べてください。TableOrder パラメータによって、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)。

テーブル順序とは別に、同期の問題を防ぐ方法として、トランザクションがコミットされるまで、参照整合性のチェックを延期できます。統合データベースが SQL Anywhere データベースである場合は、1 つの外部キーに *CHECK ON COMMIT* 句を使用します。このように設定すると、外部キーの参照整合性が、操作の開始時ではなくコミットフェーズでチェックされます。次に例を示します。

```
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)
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

```
    CHECK ON COMMIT;
```

他のデータベースベンダーの統合データベースを使用している場合は、参照総合性のチェックが同様の方法で行われているかどうかを確認します。同じであれば、この方法を実装します。方法が異なる場合は、すべての外部キー循環をなくすようにテーブル関係を再設計してください。

関連情報

Additional Parameters 同期パラメータ [92 ページ]

1.9.1.3.6 Ultra Light アプリケーションの同期の設定

Ultra Light では、設定されている通信ストリーム (ネットワークプロトコル) で Mobile Link サーバとの特定の接続を開始することで同期が開始されます。

ネットワークに直接接続している場合の同期のサポートに加えて、Windows Mobile デバイスでは Microsoft ActiveSync 同期もサポートされています。

接続の定義

各 Ultra Light リモートデータベースは、ネットワークプロトコルによって Mobile Link サーバと同期します。ネットワークプロトコルは同期ストリームパラメータを使用して設定します。サポートされているネットワークプロトコルには、TCP/IP、HTTP、HTTPS、TLS があります。選択するプロトコルについて、Mobile Link サーバのホストやポートなどその他の必要な接続情報を定義するストリームパラメータを指定する必要があります。また、Mobile Link ユーザ情報と同期スクリプトのバージョンも指定してください。

同期動作の定義

同期動作は、さまざまな同期パラメータを設定することで制御できます。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。

検討する必要がある重要な動作は、次のとおりです。

同期の方向

デフォルトでは、同期は双方向です。一方向だけの同期を使用する場合は、upload_only パラメータまたは download_only パラメータを使用します。一方向の同期を実行することによって、同期に必要な時間を最小限に抑えることができます。また、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。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

ダウンロード専用同期を使用する場合は、ダウンロードと重なっているローがローカルレベルで変更さ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、Ultra Light アプリケーションの同期は失敗します。

同期中の変更

アップロードフェーズでは、Ultra Light アプリケーションは、読み込み専用として Ultra Light データベースにアクセスできます。ダウンロードフェーズでは、読み込み/書き込みアクセスが許可されますが、アプリケーションがローを変更してからダウンロードによってこのローが変更されようとすると、ダウンロードが失敗し、ロールバックが行われます。DisableConcurrency 同期パラメータを設定することで、同期中のデータへの同時アクセスを無効にすることができます。

アプリケーションに同期機能を追加するには、通常、次の手順を行います。

1. 同期情報の構造体のフィールドとして、セッションに必要な同期パラメータとプロトコルオプションを指定します。
   たとえば、Ultra Light C++ API を使用する場合は、ul_sync_info 構造体に適切な値を設定して、アプリケーションに同期を追加します。

```
ul_sync_info info;
       // define a sync structure named "info"
        ULEnableTcpipSynchronization( &sqlca );
       // use a TCP/IP stream
       conn->InitSynchInfo( &info );
       // initialize the structure
       info.stream = ULSocketStream();
       // specify the Socket Stream
       info.stream_parms= UL_TEXT( "host=myMLserver;port=2439"  );
       // set the MobiLink host information
       info.version = UL_TEXT( "custdb 11.0" );
       // set the MobiLink version information
       info.user_name = UL_TEXT( "50" );
       // set the MobiLink user name
       info.download_only =ul_true;
       // make the synchronization download-only
```

2. 同期を初期化します。
   直接同期する場合は、API 固有の同期メソッドを呼び出します。これらのメソッドは、同期操作の成功または失敗を示すブール値を返します。同期が失敗した場合は、別の構造体で詳細なエラーステータスのフィールドを検証して、追加のエラー情報を取得できます。
   Microsoft ActiveSync 同期の場合は、Microsoft ActiveSync プロバイダからの同期メッセージを取得し、DoSync メソッドを使用して ULSynchronize メソッドを呼び出す必要があります。
3. 同期の進行状況をユーザにレポートする場合は、observer コールバック関数を使用します。

> **注記**
>
> ダウンロードが非常に大きいか、ネットワーク接続の信頼性が低いため、DLL を正常に実行できない場合は、再開可能なダウンロードを実装できます。

関連情報

Ultra Light の同期パラメータ [90 ページ]
Ultra Light のネットワークプロトコルのオプション [116 ページ]
Ultra Light.NET での Mobile Link データ同期 [577 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

Ultra Light C++ での Mobile Link データの同期 [625 ページ]
Ultra Light C++ での Microsoft ActiveSync 同期 [671 ページ]
Embedded SQL アプリケーションの同期の設定 [658 ページ]

1.9.1.4 Mobile Link ファイル転送

Ultra Light では、Mobile Link サーバを使用してファイルを転送する機能がサポートされています。

その他のすべての API については、次の場合に Mobile Link のファイル転送メカニズムを使用します。

- 特にセキュリティ対策に企業のファイアウォールを使用していて、複数のファイルを複数のデバイスに配備する必要がある場合。Mobile Link は、企業のファイアウォールを介して同期を処理するように設定済みなので、MLFileTransfer メカニズムにより、アップグレードや他のタイプのファイル転送のためのデバイスのプロビジョニングが非常に便利になります。
- 特定の Mobile Link ユーザ ID を対象とするファイルがある場合。ユーザ ID ごとに、Mobile Link サーバで 1 つまたは複数のユーザ固有のディレクトリを作成する必要があります。ファイルのバージョンが 1 つだけの場合は、デフォルトのフォルダを使用できます。

ファイル転送のしくみ

Mobile Link から開始する 2 種類のファイル転送方法のどちらかを使用して、ファイルをデバイスにダウンロードできます。デスクトップ転送用の mlfiletransfer ユーティリティを実行する方法か、使用する API に適切な関数を呼び出して Ultra Light アプリケーションをコーディングする方法です。どちらの方法でも次の処理が必要です。

1. 転送先を指定します。
   デスクトップから mlfiletransfer ユーティリティを使用するか、API に適切な関数を使用するかに関係なく、転送先のデバイスまたはデスクトップコンピュータにあるファイルのローカルパスとファイル名を設定する必要があります。アプリケーションで、またはエンドユーザがこの情報を指定しなかった場合、ファイルは転送元のファイル名で現在のフォルダに保存されます。
   ターゲットの転送先フォルダは、次のようにデバイスのオペレーティングシステムによって異なります。
   - Windows Mobile では、転送先が NULL の場合、ファイルはルートフォルダ (¥) に格納されます。
     ファイル名は、Windows Mobile のファイルの命名規則に従っている必要があります。
   - デスクトップコンピュータでは、転送先が NULL の場合、ファイルは現在のフォルダに格納されます。
     ファイル名は、デスクトップシステムのファイルの命名規則に従っている必要があります。
   - Apple iOS では、アプリケーションのドキュメントフォルダにファイルを格納してください。ドキュメントフォルダのロケーションを取得するには、NSDocumentDirectory パラメータを使用して NSSearchPathForDirectories/uDomains を呼び出します。
2. ユーザが識別され、正しいファイルがダウンロードされるように、Mobile Link ユーザクレデンシャルを設定します。
   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 サーバに対して識別し、認証するために使用されます。
3. 使用するストリームタイプを設定し、目的とするストリームのパラメータを定義します。このパラメータは、Ultra Light で Mobile Link 同期用にサポートされているパラメータと同じです。
   ほとんどの同期ストリームでは、Mobile Link サーバのアドレスを識別したり、その他の動作を制御したりするパラメータが必要です。ストリームタイプを、プラットフォームに適さない無効な値に設定すると、ストリームタイプは TCP/IP に設定されます。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

4. 転送メカニズムに必要な動作を指定します。
   たとえば、転送先にファイルがすでに存在し、変更されていない場合でも、ダウンロードを強制したり、部分的なダウンロードを再開したりするようにプロパティを設定できます。また、進行状況をモニタし、レポートするかどうかも設定できます。
5. Mobile Link サーバが実行中であり、-ftr オプションを指定して起動されたことを確認します。
6. 転送を開始し、必要な場合はダウンロードの進行状況をモニタします。
   ダウンロードの進行状況を表示することで、ユーザはダウンロードをキャンセルし、後で再開できます。

関連情報

接続パラメータでの Ultra Light ファイルパスの指定 [41 ページ]
Ultra Light の同期パラメータ [90 ページ]
Ultra Light のネットワークプロトコルのオプション [116 ページ]

1.9.1.5 Ultra Light パブリケーション

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。

Ultra Light データベース内のすべてのテーブルと、そのテーブルのすべてのローを同期させる場合は、パブリケーションを作成しないでください。

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。各アーティクルはテーブル全体、またはテーブル内のローです。WHERE 句を使用してこのローのセットを定義できます。

各データベースには、必要な同期の論理に応じて、複数のパブリケーションを作成できます。たとえば、優先度の高いデータのパブリケーションを作成することができます。ユーザは高速無線ネットワークを経由してこのデータを同期できます。無線ネットワークには使用料がかかります。使用料を抑えるには、ビジネスに必須のデータのみを同期します。緊急でないデータは、後でクレードルから同期します。

SQL 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。SQL 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。

使用上の注意

- Ultra Light パブリケーションでは、カラムのサブセットの定義と、SUBSCRIBE BY 句がサポートされていません。Ultra Light テーブルのカラムが SQL Anywhere 統合データベースのテーブルと正確に一致しない場合は、Mobile Link スクリプトを使用して、これらの違いを解消してください。
- カラムは、CREATE TABLE 文で定義された順に常に送信されます。
- パブリケーションでテーブル同期順序を設定する必要はありません。配備においてテーブル順序が重要な場合は、Ultra Light データベースを同期するときに TableOrder 同期パラメータを設定して、テーブル順序を設定できます。
- Ultra Light ではオブジェクトの所有権がサポートされていないので、すべてのユーザがパブリケーションを削除できます。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

このセクションの内容:

Ultra Light でのテーブル全体のパブリッシュ [87 ページ]
テーブル全体を Ultra Light にパブリッシュします。

Ultra Light テーブルのローのサブセットのパブリッシュ [88 ページ]
Ultra Light テーブルからローのサブセットをパブリッシュします。

Ultra Light のパブリケーションの削除 [89 ページ]
Ultra Light のパブリケーションを削除します。

関連情報

Ultra Light のテーブルの順序 [82 ページ]
Ultra Light クライアントの同期設計 [77 ページ]

1.9.1.5.1 Ultra Light でのテーブル全体のパブリッシュ

テーブル全体を Ultra Light にパブリッシュします。

コンテキスト

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。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。このアーティクルは、テーブルのすべてのローとカラムで構成されます。

SQL Central または Interactive SQL のいずれかを使用して、このタスクを実行できます。

手順

1. SQL Central を使用して Ultra Light 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右クリックし、新規 > パブリケーション の順にクリックします。
3. 新しいパブリケーションの名前を指定してください。フィールドに、新しい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リストからテーブルを選択します。追加をクリックします。
5. 完了をクリックします。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

結果

パブリケーションが作成されます。

関連情報

Ultra Light クライアント [71 ページ]
CREATE PUBLICATION 文 [Ultra Light] [492 ページ]

1.9.1.5.2 Ultra Light テーブルのローのサブセットのパブリッシュ

Ultra Light テーブルからローのサブセットをパブリッシュします。

コンテキスト

パブリケーションには、特定のテーブルローだけが含まれます。WHERE 句によって、変更されたローのうち WHERE 句の探索条件に一致するローのみをアップロードするよう制限されます。

すべての変更されたローをアップロードする場合は、WHERE 句を指定しないでください。

手順

1. SQL Central を使用して Ultra Light 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右クリックし、新規 > パブリケーション の順にクリックします。
3. 新しいパブリケーションの名前を指定してください。フィールドに、新しいパブリケーションの名前を入力します。
4. 次へをクリックします。
5. 使用可能なテーブルリストでテーブルをクリックし、追加をクリックします。
6. *WHERE* 句タブをクリックし、アーティクルリストでテーブルをクリックします。オプションで、挿入ウィンドウを使用して探索条件をフォーマットできます。
7. 完了をクリックします。

結果

アップロードされるローは、変更されたローのうち WHERE 句の探索条件に一致するローだけに制限されます。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

関連情報

Ultra Light クライアント [71 ページ]
CREATE PUBLICATION 文 [Ultra Light] [492 ページ]

1.9.1.5.3 Ultra Light のパブリケーションの削除

Ultra Light のパブリケーションを削除します。

コンテキスト

テーブルのパブリケーションを削除すると、Ultra Light データベース内のすべてのテーブルとそのテーブルのローを同期できます。

パブリケーションは SQL Central または Interactive SQL を使用して削除できます。

手順

1. SQL Central で Ultra Light データベースに接続します。
2. 左ウィンドウ枠で、パブリケーションフォルダをダブルクリックします。
3. パブリケーションを右クリックして、削除をクリックします。
4. はいをクリックします。

結果

パブリケーションが削除されます。

関連情報

Ultra Light クライアント [71 ページ]
DROP PUBLICATION 文 [Ultra Light] [505 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.9.2 Microsoft ActiveSync 同期の概要

Microsoft ActiveSync を通じた同期は数ステップに要約できます。

1. Microsoft ActiveSync によって同期セッションが開始されます。
2. Microsoft ActiveSync プロバイダは、同期の通知メッセージをデバイス上の最初に登録されているアプリケーションに送信します。アプリケーションが実行中でない場合は、起動されます。
3. 登録されたアプリケーションごとに WndProc が起動されます。
4. アプリケーションは、メッセージが Microsoft ActiveSync からの同期通知メッセージであると判断すると、ULIsSynchronizeMessage を呼び出してデータベース同期プロシージャを起動します。
5. 同期処理が完了すると、アプリケーションは ULSignalSyncIsComplete を呼び出して、同期が終了したことをプロバイダに通知します。
6. プロバイダに登録されている各アプリケーションについて、手順 2 ～ 5 が繰り返されます。

1.9.3 Ultra Light の同期パラメータ

同期パラメータは、Ultra Light データベースと Mobile Link 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。

> **注記**
>
> ここ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のみです。SQL Anywhere リモートデータベースを同期するには、Mobile Link SQL Anywhere クライアントを使用します。

必須パラメータ

次のパラメータは必須です。

- Stream Type
- User Name
- Version

これらのパラメータ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。

パラメータの競合

次のパラメータは、いずれか 1 つのみ指定できます。

- Download Only
- Ping

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

- Upload Only

これらのパラメータの 1 つ以上を true に設定した場合、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。

このセクションの内容:

Additional Parameters 同期パラメータ [92 ページ]
この同期パラメータを使用すると、あまり使用されない、または他の定義済みパラメータでは簡単に指定できない追加パラメータをアプリケーションで指定できます。

Authentication Parameters 同期パラメータ [94 ページ]
Mobile Link イベントの認証パラメータにパラメータを渡します。

Authentication Status 同期パラメータ [95 ページ]
このフィールドは同期によって設定され、Mobile Link のユーザ認証のステータスをレポートます。Mobile Link サーバが、この情報をクライアントに提供します。

Authentication Value 同期パラメータ [96 ページ]
このフィールドは同期によって設定され、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 サーバが、この情報をクライアントに提供します。

Download Only 同期パラメータ [97 ページ]
この同期中は、Ultra Light データベースから変更がアップロードされないようにします。

Ignored Rows 同期パラメータ [98 ページ]
このフィールドは同期によって設定され、同期中にスクリプトがないために Mobile Link サーバによってローが無視されたことを示します。

Keep Partial Download 同期パラメータ [98 ページ]
同期時の通信エラーが原因でダウンロードが失敗したときに、変更をロールバックしないで部分的なダウンロードを保持するかどうかを制御します。

New Password 同期パラメータ [99 ページ]
ユーザ名に対する新しい Mobile Link パスワードを設定します。

Number of Authentication Parameters パラメータ [100 ページ]
Mobile Link イベントの認証パラメータに渡す認証パラメータの数を指定します。

Observer 同期パラメータ [101 ページ]
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を指定します。

Partial Download Retained 同期パラメータ [101 ページ]
同期時の通信エラーが原因でダウンロードが失敗したときに、変更をロールバックしないで、ダウンロードされた変更が適用されたかどうかを示します。

Password 同期パラメータ [102 ページ]
ユーザ名に対する Mobile Link パスワードを指定します。

Ping 同期パラメータ [103 ページ]
Ultra Light クライアントと Mobile Link 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。

Publications 同期パラメータ [105 ページ]
同期させるパブリケーションを指定します。

Resume Partial Download 同期パラメータ [106 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

失敗したダウンロードを再開します。

Send Download Acknowledgement 同期パラメータ [107 ページ]
クライアントがダウンロード確認を提供することを Mobile Link サーバに指示します。

Stream Error 同期パラメータ [107 ページ]
通信エラーレポート情報を保持する構造体です。

Stream Type 同期パラメータ [109 ページ]
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。

Stream Parameters 同期パラメータ [110 ページ]
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。

Sync Result 同期パラメータ [111 ページ]
同期のステータスをレポートします。

Upload OK 同期パラメータ [112 ページ]
このフィールドは同期によって設定され、Mobile Link 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。

Upload Only 同期パラメータ [113 ページ]
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。これにより低速の通信リンクでの通信時間を節約できます。

User Data 同期パラメータ [114 ページ]
アプリケーション固有の情報を同期オブザーバで使用できるようにします。

User Name 同期パラメータ [114 ページ]
必須。Mobile Link サーバが認証用に使用する文字列です。

Version 同期パラメータ [115 ページ]
統合データベースのバージョンを定義します。

1.9.3.1 Additional Parameters 同期パラメータ

この同期パラメータを使用すると、あまり使用されない、または他の定義済みパラメータでは簡単に指定できない追加パラメータをアプリケーションで指定できます。

パラメータは、複数の keyword=value 設定をセミコロンで区切った文字列として指定されます。

構文

構文は、使用する API によって異なります。Ultra Light J では使用できません。

指定可能な値

次のプロパティを追加パラメータ設定の一部として指定できます。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

プロパティ名	説明
AllowDownloadDupRows	同期時に、ダウンロードされたローにプライマリキーの重複が検出された場合にエラーにしないようにします。 このプロパティを 0 に設定すると、エラーが発生し、ダウンロードをロールバックします。1 に設定すると、警告が発生し、ダウンロードを続行します。 このプロパティは Ultra Light C++ でのみ使用できます。
CheckpointStore	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。 このプロパティを 1 に設定するとこの機能が有効になります。多くの更新を伴う大量のダウンロードには最適ですが、同期処理に時間がかかるようになります。デフォルトでは 0 に設定されています。
DisableConcurrency	同期のアップロードフェーズ中は、他のスレッドからのデータベースへのアクセスを禁止します。 このプロパティを 0 に設定すると、データベースへの同時アクセスが許可されます。許可しないためには 1 に設定します。デフォルトでは、このプロパティは 0 に設定されています。
TableOrder	Ultra Light のデフォルトのテーブルの順序が適切でない場合に、優先同期に必要なテーブルの順序を設定します。 このプロパティを設定して、テーブル名をアップロード順に並べたリストを作成します。Ultra Light ではカンマで区切られたリストを使用し、ulsync ではセミコロンで区切られたリストを使用します。デフォルトでは、外部キーの関係に基づいた順序で並べられます。統合データベースの外部キーが Ultra Light リモートに一致し、外部キー循環がない場合は、通常、デフォルトの順序を使用できます。 一重または二重引用符でテーブル名を囲みます。たとえば、"Customer,Sales"と 'Customer,Sales' の両方が Ultra Light でサポートされています。 同期に含まれていないテーブルを指定すると、そのテーブルは無視されます。リストしないテーブルは、リモートデータベースで定義した外部キーに基づいて適切に保存されます。 ダウンロードのテーブルの順序は、アップロードで定義した順序と同じです。 Ultra Light テーブルが次のいずれかに該当する場合は、テーブル順序を明示的に設定するだけですみます。 • 外部キー循環に含まれる場合。循環に含まれるすべてのテーブルをリストする必要があります。 • 統合データベース内の外部キーの関係が異なる場合。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

例

Ultra Light for C++ アプリケーションは、次のように追加パラメータを設定できます。

```
ul_sync_info info;
// ...
info.additional_parms = UL_TEXT(
    "AllowDownloadDupRows=1;
    CheckpointStore=1;
    DisableConcurrency=1;
    TableOrder=Customer,Sales"
);
```

1.9.3.2 Authentication Parameters 同期パラメータ

Mobile Link イベントの認証パラメータにパラメータを渡します。

構文

構文は、使用する API によって異なります。

備考

パラメータには、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。

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。

例

Ultra Light for C/C++ アプリケーションは、次のようにパラメータを設定できます。

```
ul_char * Params[ 3 ] = { UL_TEXT( "parm1" ),
                          UL_TEXT( "parm2" ),
                          UL_TEXT( "parm3" ) };
// ...
info.num_auth_parms = 3;
info.auth_parms = Params;
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

関連情報

Number of Authentication Parameters パラメータ [100 ページ]

1.9.3.3 Authentication Status 同期パラメータ

このフィールドは同期によって設定され、Mobile Link のユーザ認証のステータスをレポートます。Mobile Link サーバが、この情報をクライアントに提供します。

構文

構文は、使用する API によって異なります。

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。たとえば、C/C++ アプリケーションの列挙は次のようになります。

定数	値	説明
UL_AUTH_STATUS_UNKNOWN	0	認証ステータスが不明です。接続がまだ同期していない可能性があります。
UL_AUTH_STATUS_VALID	1	ユーザ ID とパスワードは、同期時には有効でした。
UL_AUTH_STATUS_VALID_BUT_EXPIRES_SOON	2	ユーザ ID とパスワードは、同期時には有効でしたが、まもなく有効期限が切れます。
UL_AUTH_STATUS_EXPIRED	3	認証に失敗しました。ユーザ ID またはパスワードの有効期限が切れています。
UL_AUTH_STATUS_INVALID	4	認証に失敗しました。不正なユーザ ID またはパスワードです。
UL_AUTH_STATUS_IN_USE	5	認証に失敗しました。ユーザ ID はすでに使用されています。

備考

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、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。

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 サーバによって設定されるため、読み込み専用です。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

例

Ultra Light for C/C++ アプリケーションは、次のようにパラメータにアクセスできます。

```
ul_sync_info info;
// ...
returncode = info.auth_status;
```

1.9.3.4 Authentication Value 同期パラメータ

このフィールドは同期によって設定され、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 サーバが、この情報をクライアントに提供します。

構文

構文は、使用する API によって異なります。Ultra Light J では使用できません。

備考

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、authenticate_user 接続イベントと Authentication Status 同期パラメータの項で説明します。

このパラメータは Mobile Link サーバによって設定されるため、読み込み専用です。

例

Ultra Light for C/C++ アプリケーションは、次のようにパラメータにアクセスできます。

```
ul_sync_info info;
// ...
returncode = info.auth_value;
```

関連情報

Authentication Status 同期パラメータ [95 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.9.3.5 Download Only 同期パラメータ

この同期中は、Ultra Light データベースから変更がアップロードされないようにします。

構文

構文は、使用する API によって異なります。

デフォルト

False

指定可能な値

ブール式

競合するパラメータ

Ping と Upload Only

備考

ダウンロードのみの同期の場合、データ変更はアップロードされません。ただし、スキーマに関する情報と進行状況のカウンタに格納されている値はアップロードされます。ダウンロードされたデータが、アップロードされていないリモートでの変更と矛盾する場合、同期は失敗し、ロールバックされます。

例

次の例は、ulsync ユーティリティを使用して DownloadOnly 同期パラメータを設定する方法を示しています。

```
ulsync -c DBF=myuldb.udb
    "MobiLinkUid=remoteA;ScriptVersion=2;DownloadOnly=ON;Stream=http"
```

Ultra Light for C/C++ アプリケーションは、次のようにパラメータを設定できます。

```
ul_sync_info info;
// ...
info.download_only = ul_true;
```


© 2016 SAP SE or an SAP affiliate company. All rights reserved.

関連情報

Ultra Light 同期クライアント [10 ページ]
Upload Only 同期パラメータ [113 ページ]
Ultra Light 同期ユーティリティ (ulsync) [231 ページ]

1.9.3.6 Ignored Rows 同期パラメータ

このフィールドは同期によって設定され、同期中にスクリプトがないために Mobile Link サーバによってローが無視されたことを示します。

構文

構文は、使用する API によって異なります。

指定可能な値

ブール式

備考

このパラメータは読み込み専用です。

例

Ultra Light for C/C++ アプリケーションは、次のようにパラメータにアクセスできます。

```
ul_sync_info info;
// ...
res = info.ignored_rows;
```

1.9.3.7 Keep Partial Download 同期パラメータ

同期時の通信エラーが原因でダウンロードが失敗したときに、変更をロールバックしないで部分的なダウンロードを保持するかどうかを制御します。

構文

構文は、使用する API によって異なります。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

デフォルト

False (ダウンロードが失敗したときにすべての変更をロールバック)

指定可能な値

ブール式

例

Ultra Light for C/C++ アプリケーションは、次のようにパラメータを設定できます。

```
ul_sync_info info;
// ...
info.keep_partial_download = ul_true;
```

関連情報

Resume Partial Download 同期パラメータ [106 ページ]

1.9.3.8 New Password 同期パラメータ

ユーザ名に対する新しい Mobile Link パスワードを設定します。

構文

構文は、使用する API によって異なります。このパラメータは ulsync を使用して設定することもできます。

指定可能な値

文字列

備考

このパラメータはオプションです。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

例

ulsync は、次のようにこのパラメータを拡張同期パラメータとして設定できます。

```
ulsync -c DBF=myuldb.udb
"MobiLinkUid=remoteA;ScriptVersion=2;NewMobiLinkPwd=mynewpassword;Stream=http"
```

Ultra Light for C/C++ アプリケーションは、次のようにパラメータを設定できます。

```
ul_sync_info info;
// ...
info.new_password = UL_TEXT( "mlnewpass" );
```

関連情報

Ultra Light 同期ユーティリティ (ulsync) [231 ページ]

1.9.3.9 Number of Authentication Parameters パラメータ

Mobile Link イベントの認証パラメータに渡す認証パラメータの数を指定します。

構文

構文は、使用する API によって異なります。Ultra Light J では必要ありません。

デフォルト

カスタム認証スクリプトに渡されるパラメータはありません。

備考

Authentication Parameters とともにパラメータを使用して、カスタム認証スクリプトに情報を提供します。

例

Ultra Light for C/C++ アプリケーションは、次のようにパラメータを設定できます。

```
ul_sync_info info;
// ...
info.num_auth_parms = 3;
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

関連情報

Authentication Parameters 同期パラメータ [94 ページ]

1.9.3.10 Observer 同期パラメータ

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を指定します。

使用するために実装する必要があるコールバック関数のシグネチャは ul_sync_observer_fn 型です。

```
typedef void(UL_CALLBACK_FN *ul_sync_observer_fn)( ul_sync_status * status );
```

構文

構文は、使用する API によって異なります。

例

Ultra Light for C/C++ アプリケーションは、次のようにパラメータを設定できます。

```
ul_sync_info info;
// ...
info.observer=callfunction;
```

関連情報

User Data 同期パラメータ [114 ページ]

1.9.3.11 Partial Download Retained 同期パラメータ

同期時の通信エラーが原因でダウンロードが失敗したときに、変更をロールバックしないで、ダウンロードされた変更が適用されたかどうかを示します。

このパラメータは同期によって設定されます。

構文

構文は、使用する API によって異なります。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

指定可能な値

ブール式

備考

ダウンロードエラーが発生して部分的なダウンロードが保持された場合、同期時にパラメータが設定されます。

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。

例

このパラメータにアクセスするには、次のように入力します。

```
ul_sync_info info;
// ...
returncode=info.partial_download_retained;
```

関連情報

Keep Partial Download 同期パラメータ [98 ページ]
Resume Partial Download 同期パラメータ [106 ページ]

1.9.3.12 Password 同期パラメータ

ユーザ名に対する Mobile Link パスワードを指定します。

構文

構文は、使用する API によって異なります。このパラメータは ulsync を使用して設定することもできます。

指定可能な値

文字列

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

備考

このパラメータはオプションです。

この Mobile Link 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 サーバに対して識別し、認証するためだけに使用されます。

Mobile Link クライアントにパスワードが設定されている場合は、New Password パラメータを使用してパスワードを変更します。

例

ulsync は、次のようにこのパラメータを拡張同期パラメータとして設定できます。

```
ulsync -c DBF=myuldb.udb
"MobiLinkUid=remoteA;ScriptVersion=2;MobiLinkPwd=mypassword;Stream=http"
```

Ultra Light for C/C++ アプリケーションは、次のようにパラメータを設定できます。

```
ul_sync_info info;
// ...
info.password = UL_TEXT( "mypassword" );
```

関連情報

User Name 同期パラメータ [114 ページ]
New Password 同期パラメータ [99 ページ]
Ultra Light 同期ユーティリティ (ulsync) [231 ページ]

1.9.3.13 Ping 同期パラメータ

Ultra Light クライアントと Mobile Link 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。

構文

構文は、使用する API によって異なります。このパラメータは ulsync を使用して設定することもできます。

デフォルト

False

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

指定可能な値

ブール式

備考

Mobile Link サーバは、ping 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 ID 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 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
- authenticate_parameters
- end_connection

例

ulsync は、次のようにこのパラメータを拡張同期パラメータとして設定できます。

```
ulsync -c DBF=myuldb.udb
"MobiLinkUid=remoteA;ScriptVersion=2;Ping=True;Stream=http"
```

Ultra Light for C/C++ アプリケーションは、次のようにパラメータを設定できます。

```
ul_sync_info info;
// ...
info.ping = ul_true;
```

関連情報

Ultra Light 同期ユーティリティ (ulsync) [231 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.9.3.14 Publications 同期パラメータ

同期させるパブリケーションを指定します。

構文

構文は、使用する API によって異なります。このパラメータは ulsync で使用することもできます。

デフォルト

すべてのパブリケーションを同期します。

備考

C/C++ での同期時には、publications 同期パラメータをパブリケーションリスト、つまり、パブリケーション名をカンマで区切ったリストに設定します。

例

ulsync は、次のようにこのパラメータを拡張同期パラメータとして設定できます。

```
ulsync -c DBF=myuldb.udb
"MobiLinkUid=remoteA;ScriptVersion=2;Publications=UL_PUB_MYPUB1,UL_PUB_MYPUB2;Stre
am=http"
```

Ultra Light for C/C++ アプリケーションは、次のようにパラメータを設定できます。

```
ul_sync_info info;
// ...
info.publications = UL_TEXT( "Pubs1,Pubs3" );
```

関連情報

Ultra Light パブリケーション [86 ページ]
Ultra Light でのデータのパブリッシュ [80 ページ]
Ultra Light 同期ユーティリティ (ulsync) [231 ページ]

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

1.9.3.15 Resume Partial Download 同期パラメータ

失敗したダウンロードを再開します。

構文

構文は、使用する API によって異なります。このパラメータは ulsync を使用して設定することもできます。

デフォルト

False

指定可能な値

ブール式

備考

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変更のみがダウンロードされます。

例

Ultra Light for C/C++ アプリケーションは、次のようにパラメータを設定できます。

```
ul_sync_info info;
// ...
info.resume_partial_download = ul_true;
```

関連情報

Keep Partial Download 同期パラメータ [98 ページ]
Partial Download Retained 同期パラメータ [101 ページ]
Ultra Light 同期ユーティリティ (ulsync) [231 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.9.3.16 Send Download Acknowledgement 同期パラメータ

クライアントがダウンロード確認を提供することを Mobile Link サーバに指示します。

構文

構文は、使用する API によって異なります。このパラメータは ulsync を使用して設定することもできます。

デフォルト

False

指定可能な値

ブール式

例

ulsync は、次のようにこのパラメータを拡張同期パラメータとして設定できます。

```
ulsync -c DBF=myuldb.udb
"MobiLinkUid=remoteA;ScriptVersion=2;SendDownloadACK=true;Stream=http"
```

Ultra Light for C/C++ アプリケーションは、次のようにパラメータを設定できます。

```
ul_sync_info info;
// ...
info.send_download_ack = ul_true;
```

関連情報

Ultra Light 同期ユーティリティ (ulsync) [231 ページ]

1.9.3.17 Stream Error 同期パラメータ

通信エラーレポート情報を保持する構造体です。

構文

構文は、使用する API によって異なります。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

適用対象

このパラメータは、C++ インタフェースにのみ適用されます。

指定可能な値

このパラメータにはデフォルト値がないので、サポートされているフィールドの一つを使用して明示的に設定する必要があります。ul_stream_error のフィールドは、次のとおりです。

stream_error_code

エラーコードのサフィックスについては、*%SQLANY17%*¥SDK¥Include¥sserror.h を参照してください。

system_error_code

システム固有のエラーコード。エラーコードの詳細については、プラットフォームのマニュアルを参照してください。Windows プラットフォームの場合は、Microsoft Developer Network (MSDN) マニュアルを参照してください。

Windows における一般的なシステムエラーを次に示します。

10048 (WSAADDRINUSE)

アドレスがすでに使用されています。

10053 (WSAECONNABORTED)

ソフトウェアが接続のアボートを引き起こしました。

10054 (WSAECONNRESET)

通信の他方の側がソケットを閉じました。

10060 (WSAETIMEDOUT)

接続がタイムアウトしました。

10061 (WSAECONNREFUSED)

接続が拒否されました。通常、Mobile Link サーバが稼働していないか、指定されたポートで受信していません。

error_string

stream_error_code の追加情報 (ある場合) の文字列。文字列は空の場合もあれば、空でない場合もあります。error_string の値が空でない場合、stream_error_code の値に追加情報が提供されます。たとえば、書き込みエラー (エラーコード 9) の場合、エラー文字列は書き込もうとしたバイト数を示します。

備考

Ultra Light C++ コンポーネント以外の Ultra Light アプリケーションは、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。

stream_error フィールドは ul_stream_error 型の構造体です。

```
typedef struct {
    ss_error_code stream_error_code;
    asa_uint16    alignment;
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

```
    asa_int32     system_error_code;
    char          error_string[UL_STREAM_ERROR_STRING_SIZE];
} ul_stream_error, * p_ul_stream_error;
```

構造体は、*%SQLANY17%*¥SDK¥Include¥sserror.h に定義されています。

SQLE_COMMUNICATIONS_ERROR をチェックします。

```
ULConnection * conn;
ul_sync_info info;
...
conn->InitSynchInfo( &info );
if( !conn->Synchronize( &info ) ) {
    ULError const * error = conn->GetLastError();
    char buf[256];
    if( error->GetSQLCode() == SQLE_MOBILINK_COMMUNICATIONS_ERROR ) {
        error->GetString( buf, sizeof(buf) );
        printf( "%s¥n", buf );
        // more handling for communication error
    }
}
```

関連情報

Sync Result 同期パラメータ [111 ページ]

1.9.3.18 Stream Type 同期パラメータ

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。

構文

構文は、使用する API によって異なります。このパラメータは ulsync を使用して設定することもできます。

備考

このパラメータは必須です。デフォルト値はありません。

ほとんどのネットワークプロトコルでは、Mobile Link 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 Parameters パラメータで指定します。

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。

ストリームのタイプには次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。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

ネットワークプロトコル	説明
HTTP	HTTP を介して同期します。
HTTPS	HTTPS を介して同期します。 HTTPS プロトコルは、基本のセキュリティレイヤとして TLS を使用します。TCP/IP を介して機能します。
TCP/IP	TCP/IP を介して同期します。
TLS	トランスポートレイヤセキュリティ (TLS) 付きの TCP/IP を介して同期します。TLS は、デジタル証明書とパブリックキー暗号方式を使用して、クライアント/サーバ通信をセキュリティ保護します。

例

Ultra Light for C/C++ アプリケーションは、次のようにパラメータを設定します。

```
Connection conn;
ul_sync_info info;
...
conn.InitSynchInfo( &info );
info.stream = "http";
```

関連情報

Ultra Light のネットワークプロトコルのオプション [116 ページ]
Stream Parameters 同期パラメータ [110 ページ]
Ultra Light 同期ユーティリティ (ulsync) [231 ページ]

1.9.3.19 Stream Parameters 同期パラメータ

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。

構文

構文は、使用する API によって異なります。このパラメータは ulsync を使用して設定することもできます。

デフォルト

NULL

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

指定可能な値

文字列

備考

このパラメータは省略可能です。このパラメータは、セミコロンで区切られたネットワークプロトコルオプションのリストを受け入れます。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。

例

Ultra Light for C/C++ アプリケーションは、次のようにパラメータを設定できます。

```
ul_sync_info info;
// ...
info.stream_parms= UL_TEXT( "host=myserver;port=2439" );
```

関連情報

Ultra Light のネットワークプロトコルのオプション [116 ページ]
Ultra Light 同期ユーティリティ (ulsync) [231 ページ]

1.9.3.20 Sync Result 同期パラメータ

同期のステータスをレポートします。

構文

構文は、使用する API によって異なります。

備考

このパラメータは Ultra Light によって設定され、読み込み専用です。

C/C++ インタフェースでは、この情報を ul_sync_info 構造体の中で複数の別個のパラメータで受け取ります。それ以外の場合は、この情報は、さまざまな情報がそれぞれ別個のフィールドに格納された複合パラメータとして定義されます。

Authentication Status

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

認証の成功または失敗をレポートします。
Ignored Rows

無視されるローの数をレポートします。
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。

関連情報

Authentication Status 同期パラメータ [95 ページ]
Ignored Rows 同期パラメータ [98 ページ]
Stream Error 同期パラメータ [107 ページ]
Upload OK 同期パラメータ [112 ページ]

1.9.3.21 Upload OK 同期パラメータ

このフィールドは同期によって設定され、Mobile Link 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。

構文

構文は、使用する API によって異なります。

備考

このパラメータは Ultra Light 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。

例

Ultra Light for C/C++ アプリケーションは、次のようにパラメータにアクセスできます。

```
ul_sync_info info;
// ...
returncode = info.upload_ok;
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.9.3.22 Upload Only 同期パラメータ

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。これにより低速の通信リンクでの通信時間を節約できます。

構文

構文は、使用する API によって異なります。このパラメータは ulsync を使用して設定することもできます。

デフォルト

False

指定可能な値

ブール式

競合するパラメータ

Download Only、Ping、Resume Partial Download

備考

true に設定すると、クライアントは Mobile Link 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。

例

ulsync は、このパラメータを次のように拡張同期パラメータとして設定できます。

```
ulsync -c DBF=myuldb.udb
"MobiLinkUid=remoteA;ScriptVersion=2;UploadOnly=True;Stream=http"
```

Ultra Light for C/C++ アプリケーションは、次のようにパラメータを設定できます。

```
ul_sync_info info;
// ...
info.upload_only = ul_true;
```


© 2016 SAP SE or an SAP affiliate company. All rights reserved.

関連情報

Ultra Light クライアントの同期設計 [77 ページ]
Download Only 同期パラメータ [97 ページ]
Ultra Light 同期ユーティリティ (ulsync) [231 ページ]

1.9.3.23 User Data 同期パラメータ

アプリケーション固有の情報を同期オブザーバで使用できるようにします。

適用対象

C/C++ アプリケーションのみ。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としないので、User Data パラメータはありません。

構文

構文は、使用する API によって異なります。

備考

同期オブザーバ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。

関連情報

Observer 同期パラメータ [101 ページ]

1.9.3.24 User Name 同期パラメータ

必須。Mobile Link サーバが認証用に使用する文字列です。

構文

構文は、使用する API によって異なります。このパラメータは ulsync を使用して設定することもできます。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

備考

このパラメータは必須です。空の文字列と NULL 文字列は一般に拒否されます。

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。

リモート ID が使用されている場合は、ユーザ名はユニークである必要はありません。

この Mobile Link 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 サーバに対して識別し、認証するためだけに使用されます。

同期システムに含まれるユーザの場合は、Mobile Link サーバを使用してユーザ名を登録する必要があります。ユーザ名は、統合データベースの ml_user Mobile Link システムテーブルの name カラムに格納されます。

例

ulsync は、このパラメータを次のように拡張同期パラメータとして設定できます。

```
ulsync -c DBF=myuldb.udb "MobiLinkUid=remoteA;ScriptVersion=2;Stream=http"
```

Ultra Light for C/C++ アプリケーションは、次のようにパラメータを設定できます。

```
ul_sync_info info;
// ...
info.user_name= UL_TEXT( "remoteA" );
```

関連情報

Password 同期パラメータ [102 ページ]
Ultra Light 同期ユーティリティ (ulsync) [231 ページ]

1.9.3.25 Version 同期パラメータ

統合データベースのバージョンを定義します。

構文

構文は、使用する API によって異なります。このパラメータは ulsync を使用して設定することもできます。

指定可能な値

文字列

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

備考

このパラメータは必須です。空の文字列と NULL 文字列は一般に拒否されます。

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、異なるバージョン文字列によって特定される 2 つの download_cursor スクリプトが存在する場合があります。

例

ulsync は、このパラメータを次のように拡張同期パラメータとして設定できます。

```
ulsync -c DBF=myuldb.udb "MobiLinkUid=remoteA;ScriptVersion=2;Stream=http"
```

Ultra Light for C/C++ アプリケーションは、次のようにパラメータを設定できます。

```
ul_sync_info info;
// ...
info.version = UL_TEXT( "default" );
```

関連情報

Ultra Light 同期ユーティリティ (ulsync) [231 ページ]

1.9.4 Ultra Light のネットワークプロトコルのオプション

アプリケーションでネットワークプロトコルを設定する必要があります。

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 サーバと同期します。使用可能なネットワークプロトコルには、TCP/IP、HTTP、HTTPS、TLS があります。

設定したネットワークプロトコルでは、対応するプロトコルオプションのセットから選択することによって、Ultra Light アプリケーションが 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。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します。

このセクションの内容:

同期ストリームのオプション [117 ページ]
Stream Parameters パラメータを設定することによって、アプリケーション内で Mobile Link サーバを特定するのに必要な情報を提供できます。

関連情報

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 [26 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

Stream Parameters 同期パラメータ [110 ページ]
Ultra Light 同期ユーティリティ (ulsync) [231 ページ]

1.9.4.1 同期ストリームのオプション

Stream Parameters パラメータを設定することによって、アプリケーション内で Mobile Link サーバを特定するのに必要な情報を提供できます。

関連情報

Stream Parameters 同期パラメータ [110 ページ]

1.10 Ultra Light の配備

ほとんどの場合、開発は、Ultra Light をモバイルデバイスとして使用することを最終リリース目標として、Windows デスクトップまたは Mac OS X 上で行われます。

ただし、開発環境によっては、さまざまな配備メカニズムを使用して Ultra Light をインストールできます。

Ultra Light アプリケーションプロジェクトは、同じ Ultra Light データベースを開発データベース、テストデータベース、配備された運用データベースなど、さまざまな用途に使用することによって発展させることができます。配備されたデータベースアプリケーションの存続期間の間、変更や改善は、まず開発データベースで行われ、テストデータベースに伝達された後、最終的に運用データベースに配布されます。

Ultra Light アプリケーションで使用する必要があるモジュールは、目標とするプラットフォーム、使用するインタフェース、および使用したい機能によって異なります。

このセクションの内容:

Ultra Light アプリケーションのビルドと配備の仕様 [118 ページ]
Ultra Light データベースの暗号化の要件を含め、サポート対象のすべてのプラットフォームとデバイスにおける Ultra Light アプリケーションのビルドと配備には最小要件があります。

Ultra Light データベースの配備方法 [124 ページ]
デバイス上で初期データベースファイルを取得するには、次の方法を使用します。

Ultra Light データベーススキーマのアップグレードの配備 [125 ページ]
スキーマのアップグレードを実行します。

Ultra Light エンジンの起動 [127 ページ]
Ultra Light エンジンを使用して Windows または Windows Mobile デバイス上のデータを管理すると、Ultra Light アプリケーションにエンジンのディレクトリの場所を明示的に指定する必要がある場合を除き、アプリケーションによってエンジンが自動的に起動します。

Microsoft ActiveSync Manager を使用したアプリケーションの登録 [128 ページ]

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

Microsoft ActiveSync 同期を使用するアプリケーションを登録します。

関連情報

Windows Mobile 用の Ultra Light データ管理コンポーネント [22 ページ]
Ultra Light C++ アプリケーションの構築と配備 [626 ページ]
Ultra Light.NET アプリケーション開発 [559 ページ]

1.10.1 Ultra Light アプリケーションのビルドと配備の仕様

Ultra Light データベースの暗号化の要件を含め、サポート対象のすべてのプラットフォームとデバイスにおける Ultra Light アプリケーションのビルドと配備には最小要件があります。

> **注記**
>
> Ultra Light エンジンのバージョンによっては、_dev サフィックスを含むディレクトリ (`x86_dev` ディレクトリなど) に置かれていることがあります。これらのバージョンには、デバッグを目的としたプラットフォーム上の問題の診断に使用できる開発時のログ機能が含まれています。運用システムでは、_dev ディレクトリにないバージョンのエンジンを使用してください。

最小要件は次の表のとおりです。

プラットフォームまたはデバイス	最小要件	AES 暗号化の要件	FIPS 140-2 AES 暗号化の要件
Windows Mobile とデスクトップ (静的リンクを使用する Ultra Light C++)	リンク先: • `ulrt.lib`[1] • `ulbase.lib`[1]	データベースの作成時または接続時に、DBKEY 作成パラメータを使用して暗号化キーを設定します。 EnableAesDBEncryption メソッドを呼び出します。	データベースの作成時と接続時に、暗号化キーに DBKEY 作成パラメータを設定します。 データベースの作成時に作成パラメータ *fips=yes* を設定します。 EnableAesFipsDBEncryption メソッドを呼び出します。 展開: • `ulfips17.dll`[2] • `libeay32.dll`[2] • `msvcr90.dll`[2]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

プラットフォームまたはデバイス	最小要件	AES 暗号化の要件	FIPS 140-2 AES 暗号化の要件
Windows Mobile とデスクトップ (動的リンクを使用する Ultra Light C++)	リンク先: • ulimp.lib[1,10] • ulbase.lib[1] 展開: • ulrt*17*.dll[1]	データベースの作成時または接続時に、DBKEY 作成パラメータを使用して暗号化キーを設定します。 EnableAesDBEncryption メソッドを呼び出します。	データベースの作成時と接続時に、暗号化キーに DBKEY 作成パラメータを設定します。 データベースの作成時に作成パラメータ *fips=yes* を設定します。 EnableAesFipsDBEncryption メソッドを呼び出します。 展開: • ulfips*17*.dll[2] • libeay32.dll[2] • msvcr90.dll[2]
Windows Mobile とデスクトップ (Ultra Light エンジンを使用する Ultra Light C++)	リンク先: • ulrtc.lib[1] • ulbase.lib[1] 展開: • *uleng17*.exe[2]	データベースの作成時または接続時に、DBKEY 作成パラメータを使用して暗号化キーを設定します。	データベースの作成時と接続時に、暗号化キーに DBKEY 作成パラメータを設定します。 データベースの作成時に作成パラメータ *fips=yes* を設定します。 展開: • ulfips*17*.dll[2] • libeay32.dll[2] • msvcr90.dll[2]
Windows Mobile とデスクトップ (Ultra Light.NET)	参照の追加先: • *Sap.Data.UltraLite* • *Sap.Data.UltraLite.resources* 展開: • Sap.Data.UltraLite.dll[7] • Sap.Data.UltraLite.resources.dll[8] • ulnet*17*.dll[6]	データベースの作成時または接続時に、DBKEY 作成パラメータを使用して暗号化キーを設定します。	データベースの作成時と接続時に、暗号化キーに DBKEY 作成パラメータを設定します。 データベースの作成時に作成パラメータ *fips=yes* を設定します。 展開: • ulfips*17*.dll[2] • libeay32.dll[2] • msvcr90.dll[2]

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

プラットフォームまたはデバイス	最小要件	AES 暗号化の要件	FIPS 140-2 AES 暗号化の要件
Windows Mobile とデスクトップ (Ultra Light エンジンを使用する Ultra Light.NET)	参照の追加先: • *Sap.Data.UltraLite* • *Sap.Data.UltraLite.resources* 展開: • `Sap.Data.UltraLite.dll`[7] • `Sap.Data.UltraLite.resources.dll`[8] • `ulnetclient17.dll`[6] • *uleng17*`.exe`[2]	データベースの作成時または接続時に、DBKEY 作成パラメータを使用して暗号化キーを設定します。	データベースの作成時と接続時に、暗号化キーに DBKEY 作成パラメータを設定します。 データベースの作成時に作成パラメータ *fips=yes* を設定します。 展開: • `ulfips17.dll`[2] • `libeay32.dll`[2] • `msvcr90.dll`[2]
Mac OS X および iOS (Ultra Light C++)	Xcode プロジェクトに追加: • `libulrt.a`[9] • `libulbase.a`[9] (Mac OS X のみ)	データベースの作成時または接続時に、DBKEY 作成パラメータを使用して暗号化キーを設定します。 EnableAesDBEncryption メソッドを呼び出します。	該当なし
Linux (Ultra Light C++)	リンク先: • `libulrt.a`[3] • `libulbase.a`[3]	データベースの作成時または接続時に、DBKEY 作成パラメータを使用して暗号化キーを設定します。 EnableAesDBEncryption メソッドを呼び出します。	該当なし
Windows Phone (Ultra Light for Windows Phone)	Windows Phone プロジェクトに `UltraLite.winmd`[13] を追加します。	データベースの作成時または接続時に、DBKEY 作成パラメータを使用して暗号化キーを設定します。	該当なし
Android (Ultra Light J)	Android プロジェクトに追加: • `UltraLiteJNI17.jar`[5] • `libultralitej17.so`[4]	データベースの作成時または接続時に、DBKEY 作成パラメータまたは setEncryptionKey メソッドを使用して暗号化キーを設定します。 EnableAesDBEncryption メソッドを呼び出します。	該当なし

[1] Windows Mobile の場合、このファイルは `%SQLANY17%¥UltraLite¥CE¥Arm.50¥Lib` にあります。Windows の場合は、`%SQLANY17%¥UltraLite¥Windows¥x64¥Lib¥VS9` または `%SQLANY17%¥UltraLite¥Windows¥x86¥Lib¥VS9` にあります。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

[2] FIPS は、Windows Mobile ではサポートされていません。Windows の場合、これらのファイルは、`%SQLANY17%¥UltraLite¥Windows¥x64`、または `%SQLANY17%¥UltraLite¥Windows¥x86` にあります。

[3] このファイルは */opt/sqlanywhere17*`/ultralite/linux/x64/lib` にあります。

[4] このファイルは `%SQLANY17%¥UltraLite¥UltraLiteJ¥Android¥ARM` にあります。

[5] このファイルは `%SQLANY17%¥UltraLite¥UltraLiteJ¥Android` にあります。

[6] Windows Mobile の場合、このファイルは `%SQLANY17%¥UltraLite¥UltraLite.NET¥CE¥Arm.50` にあります。Windows の場合は、`%SQLANY17%¥UltraLite¥UltraLite.NET¥x64` または `%SQLANY17%¥UltraLite¥UltraLite.NET¥win32` にあります。

[7] このファイルは `%SQLANY17%¥UltraLite¥UltraLite.NET¥Assembly¥V2` にあります。

[8] このファイルは `%SQLANY17%¥UltraLite¥UltraLite.NET¥Assembly¥V2¥en` にあります。

[9] Mac OS X の場合、このファイルは */Applications/SQLAnywhere17/System*`/ultralite/macosx/x86_64` にあります。iOS の場合、Ultra Light ランタイムはインストール後にビルドしてください。`install-dir/ultralite/iphone/readme.txt` の指示に従ってください。

[10] このライブラリにリンクする場合は、コンパイル時に UL_USE_DLL プリプロセッサマクロを定義してください。たとえば、次のように指定します。

```
-DUL_USE_DLL
```

[11] 無線配信 (OTA) での配備にのみ必要です。また、アプリケーションとともに Ultra Light J を配備する独自の `.jad` ファイルを作成することもできます。

[12] ARM、x86、および x64 プロセッサ向け WinRT コンポーネントは、SQL Anywhere インストール環境の `UltraLite¥WinRT¥Windows¥8.0¥` および `UltraLite¥WinRT¥WindowsPhone¥8.0¥` ディレクトリにあります。x86 プロセッサ向け Windows Phone 8 エミュレータは、それぞれのディレクトリ内に含まれています。

> **注記**
>
> いずれかのプラットフォームまたはデバイス上で Ultra Light データベースの難読化を使用するには、データベースの作成時に作成パラメータ *obfuscate=1* を指定する必要があります。

同期および圧縮のための追加のビルドと配備の要件

次の表に、同期を使用する Ultra Light アプリケーションのビルドと配備のためのストリーム、プロトコルオプション、およびコードの各要件を示します。

> **注記**
>
> HTTPS ストリームオプションを Ultra Light J API で有効にするには、SyncParms.HTTPS_STREAM 定数を Connection.createSyncParms メソッドに渡します。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

同期タイプ	ストリームオプション指定	プロトコルオプション要件	Ultra Light C++ のメソッドコール要件
TCP/IP	"tcpip"	なし	• EnableTcpipSynchronization
HTTP	"http"	なし	• EnableHttpSynchronization
RSA TLS	"tls"	なし	• EnableTlsSynchronization • EnableRsaSyncEncryption
RSA HTTPS	"https"	なし	• EnableHttpsSynchronization • EnableRsaSyncEncryption
FIPS 140-2 RSA TLS	"tls"	*fips=yes*	• EnableTlsSynchronization • EnableRsaFipsSyncEncryption
FIPS 140-2 RSA HTTPS	"https"	*fips=yes*	• EnableHttpsSynchronization • EnableRsaFipsSyncEncryption

次の表に、圧縮またはエンドツーエンド暗号化を使用する Ultra Light アプリケーションのビルドと配備のための追加のプロトコルオプションとコードの要件を示します。

圧縮とストリーム暗号化のオプション	プロトコルオプション要件	Ultra Light C++ および Ultra Light J のメソッドコール要件
ZLIB 圧縮	• *compression=zlib*	• **C++:** EnableZlibSyncCompression • **Java:** setZlibCompression
RSA E2EE	• *e2ee_public_key*=`key-file`	• **C++:** EnableRsaE2ee • **Java:** setE2eePublicKey
FIPS 140-2 RSA E2EE	• *e2ee_public_key*=`key-file` • *fips=yes*	• **C++:**EnableRsaFipsE2ee • **Java:**該当なし

次の表に、圧縮および暗号化した同期化のための追加のビルドと配備の要件を示します。

> **注記**
>
> TCP/IP および HTTP 同期には、追加のビルドと配備の要件はありません。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

プラットフォームまたはデバイス	ZLIB 圧縮の要件	RSA TLS、RSA HTTPS、および RSA E2EE の要件	FIPS 140-2 RSA TLS、FIPS 140-2 RSA HTTPS、および FIPS 140-2 RSA E2EE の要件
Windows Mobile とデスクトップ (静的リンクを使用する Ultra Light C++)	なし	リンク先: • ulrsa.lib[1]	展開: • mlcrsafips*17*.dll[2] • libeay32.dll[2] • ssleay32.dll[2] • msvcr90.dll
Windows Mobile とデスクトップ (動的リンクを使用する Ultra Light C++)	展開: • mlczlib*17*.dll[2]	展開: • mlcrsa*17*.dll[2]	展開: • mlcrsafips*17*.dll[2] • libeay32.dll[2] • ssleay32.dll[2] • msvcr90.dll
Windows Mobile とデスクトップ (Ultra Light エンジンを使用する Ultra Light C++)	展開: • mlczlib*17*.dll[2]	展開: • mlcrsa*17*.dll[2]	展開: • mlcrsafips*17*.dll[2] • libeay32.dll[2] • ssleay32.dll[2] • msvcr90.dll[2]
Windows Mobile とデスクトップ (Ultra Light.NET)	展開: • mlczlib*17*.dll[2]	展開: • mlcrsa*17*.dll[2]	展開: • mlcrsafips*17*.dll[2] • libeay32.dll[2] • ssleay32.dll[2] • msvcr90.dll[2]
Windows Mobile とデスクトップ (Ultra Light エンジンを使用する Ultra Light.NET)	展開: • mlczlib*17*.dll[2]	展開: • mlcrsa*17*.dll[2]	展開: • mlcrsafips*17*.dll[2] • libeay32.dll[2] • ssleay32.dll[2] • msvcr90.dll[2]
Mac OS X および iOS (Ultra Light C++)	なし	なし	該当なし
Linux (Ultra Light C++)	なし	リンク先: • libulrsa.a[3]	該当なし
Windows Phone (Ultra Light for Windows Phone)	なし	該当なし	該当なし
Android (Ultra Light J)	なし	展開: • libmlcrsa*17*.so[4]	該当なし

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

[1] Windows Mobile の場合、このファイルは %SQLANY*17*%¥UltraLite¥CE¥Arm.50¥Lib にあります。Windows の場合は、%SQLANY*17*%¥UltraLite¥Windows¥x64¥Lib¥VS9 または %SQLANY*17*%¥UltraLite¥Windows¥x86¥Lib¥VS9 にあります。

[2] FIPS は、Windows Mobile ではサポートされていません。Windows の場合、これらのファイルは、%SQLANY*17*%¥UltraLite¥Windows¥x64、または %SQLANY*17*%¥UltraLite¥Windows¥x86 にあります。

[3] このファイルは */opt/sqlanywhere17*/ultralite/linux/x64/lib にあります。

[4] このファイルは %SQLANY*17*%¥UltraLite¥UltraLiteJ¥Android¥ARM にあります。

関連情報

Ultra Light C++ アプリケーションの構築と配備 [626 ページ]
Ultra Light.NET アプリケーションを配備する方法 [579 ページ]

1.10.2 Ultra Light データベースの配備方法

デバイス上で初期データベースファイルを取得するには、次の方法を使用します。

- アプリケーションの Ultra Light API を使用して、初期データベースファイルを作成します。
- SQL スクリプトからスキーマファイルを作成し、ALTER DATABASE SCHEMA FROM FILE 文を使用します。
- Ultra Light の FileTransfer メソッドを使用して、初期データベースファイルをダウンロードします (デバイス上に既存のファイルがない場合)。
- 初期データベースをアプリケーションとバンドルします。
- Windows または Windows Mobile デバイスに配備する場合は、中央管理を使用して初期 UDB ファイルを送信するか、コマンドを送信して初期データベースファイルを作成し、SQL を実行してスキーマを提供します。

関連情報

Ultra Light データベーススキーマのアップグレードの配備 [125 ページ]
ALTER DATABASE SCHEMA FROM FILE 文 [Ultra Light] [479 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.10.3 Ultra Light データベーススキーマのアップグレードの配備

スキーマのアップグレードを実行します。

前提条件

使用する SQL ファイルには、新しいスキーマ全体が含まれている必要があります。

コンテキスト

Ultra Light データベーススキーマのアップグレードは、次のいずれかの方法で配備できます。

個々の DDL 文

たとえば、次の文を実行して新しいパブリケーションを作成します。

```
dbconnection->ExecuteStatement("CREATE PUBLICATION p (table t)");
```

ALTER DATABASE SCHEMA FROM FILE 文

DDL 文の要件がわからない場合、または個々の DDL 文を指定したくない場合は、この文を使用してスキーマのアップグレードを実行できます。

> ⚠ 警告
>
> スキーマのアップグレード中にデバイスをリセットしないでください。スキーマのアップグレード中にデバイスをリセットすると、データが失われ、Ultra Light データベースが "不正" としてマークされます。

ALTER DATABASE SCHEMA FROM FILE 文を使用して Ultra Light データベーススキーマをアップグレードすると、Ultra Light で次の手順が実行されます。

1. 新しいデータベーススキーマと既存のデータベーススキーマの違いが比較されます。
2. 既存のデータベースのスキーマは変更されます。
3. 新しいスキーマに適合しないローは削除されます。次に例を示します。
   - テーブルに一意性制約を追加し、同じ値を持つローが複数ある場合は、1 つのロー以外すべて削除されます。
   - カラムドメインを変更して変換エラーが発生した場合は、そのローが削除されます。たとえば、VARCHAR カラムを INT カラムに変更した場合、ローの値が ABCD であれば、そのローは削除されます。
   - 新しいスキーマに新しい外部キーがあり、外部ローに一致するプライマリローがない場合は、これらのローが削除されます。
4. ローが削除されると、SQLE_ROW_DROPPED_DURING_SCHEMA_UPGRADE (130) 警告が発行されます。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

手順

1. 完全に新しいスキーマを作成するには、DDL 文の SQL スクリプトを作成します。

   コンピュータ上にマスタスキーマを保持し、アプリケーションの変更に応じてスキーマを更新できます。

   スクリプトに必要な DDL 文を抽出するには、ulinit または ulunload のいずれかのユーティリティを使用します。これらのユーティリティを次のオプションとともに使用して、DDL 文が構文的に正しいことを確認してください。

   - Ultra Light データベースの場合、ulunload ユーティリティを使用し、-n および -s [`schema-file`] オプションを指定します。次に例を示します。

     ```
     ulunload -c dbf=mydatabase.udb -n -s MySchema.sql
     ```

   - SQL Anywhere データベースの場合、ulinit ユーティリティを使用し、-a、-l [`schema-file`]、および -n [`publication-name`] オプションを指定します。次に例を示します。

     ```
     ulinit -a "dsn=mysqlanywheredatabase" -l MySchema.sql -n MyPub Temp.udb
     ```

   ulunload および ulinit ユーティリティを使用しない場合は、スクリプトで次の事項を確認してください。

   - スクリプトに、CREATE 文によって目的のスキーマ全体が宣言されています。
   - 表、カラム、パブリケーションの名前が変更されていません。RENAME 操作はサポートされていません。テーブルの名前を変更すると、DROP TABLE 操作または CREATE TABLE 操作として処理されます。
   - 影響が発生しない可能性がある文も含め、DDL 文以外の文が含まれていません。
   - SQL 文の語はスペースで区切る。
   - 各行に 1 つの SQL 文だけが表示されています。
   - コメントの先頭には、二重ハイフン (-) を付ける。コメントを指定できるのは、行の先頭のみです。
   - 各文が、*GO* という単語を含む 1 行によって区切られています。
2. 新しい SQL スクリプトファイルの配備
3. データベースが同期されていることを確認します。
4. デバイス上で新しい文を実行します。次に例を示します。

   ```
   ALTER DATABASE SCHEMA FROM FILE 'MySchema.sql'
   ```

結果

スキーマが更新されます。

関連情報

Ultra Light データベースのスキーマ [51 ページ]
ALTER DATABASE SCHEMA FROM FILE 文 [Ultra Light] [479 ページ]
Ultra Light データベース初期化ユーティリティ (ulinit) [221 ページ]
Ultra Light データベースのアンロードユーティリティ (ulunload) [236 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.10.4 Ultra Light エンジンの起動

Ultra Light エンジンを使用して Windows または Windows Mobile デバイス上のデータを管理すると、Ultra Light アプリケーションにエンジンのディレクトリの場所を明示的に指定する必要がある場合を除き、アプリケーションによってエンジンが自動的に起動します。

Ultra Light アプリケーションによって Ultra Light エンジンの起動が試行されると、次のディレクトリが検索されます。

クライアントプラットフォーム	ディレクトリの場所
Windows デスクトップ	1. Ultra Light エンジンを起動するアプリケーションのディレクトリ 2. 現在の作業ディレクトリ 3. システムパス 4. SQL Anywhere インストールディレクトリ (`bin32` または `bin64` の下)、クライアントが 32 ビットか 64 ビットかによって異なる
Windows Mobile/CE	1. `¥Windows¥` 2. `¥` (ルートディレクトリ) 3. `¥UltraLiteDB¥`
Linux	1. Ultra Light エンジンを自動起動するアプリケーションのディレクトリ 2. *%SQLANY17%*`/bin32`

Ultra Light エンジンが異なる場所に格納されている場合は、START 接続パラメータを指定してエンジンを起動します。

たとえば、データベースへの接続文字列または Windows Mobile クライアントアプリケーションの接続コードは、次の START パラメータの値を使用することがあります。

```
"START=¥Program Files¥MyApp¥uleng17.exe"
```

関連情報

Ultra Light アプリケーションのビルドと配備の仕様 [118 ページ]
Ultra Light START 接続パラメータ [194 ページ]

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

1.10.5 Microsoft ActiveSync Manager を使用したアプリケーションの登録

Microsoft ActiveSync 同期を使用するアプリケーションを登録します。

コンテキスト

Microsoft 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、Microsoft ActiveSync Manager 自体を使用する方法があります。

Microsoft ActiveSync Manager を使用してアプリケーションを登録する方法は次のとおりです。

手順

1. Microsoft ActiveSync を起動します。
2. [Microsoft ActiveSync] ウィンドウで**オプション**をクリックします。
3. 情報タイプのリストから、*Mobile Link* **クライアント**をクリックし、**設定**をクリックします。
4. *Mobile Link* **同期**ウィンドウで、**新規**をクリックします。
5. 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

   Microsoft 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。

   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

   Microsoft 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。
6. *OK* をクリックしてアプリケーションを登録します。

結果

Microsoft ActiveSync でアプリケーションが登録されます。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

次のステップ

アプリケーションをデバイスにコピーします。

関連情報

アプリケーションに対するクラス名の割り当て [669 ページ]

1.11 チュートリアル:Ultra Light の CustDB サンプルアプリケーションの構築

このチュートリアルでは、Mobile Link サーバを実行して統合データベースと Ultra Light リモート間でデータを同期する方法、SQL Central を使用して Ultra Light 内のデータを参照する方法、および Ultra Light ユーティリティで Ultra Light データベースを管理する方法について説明します。

コンテキスト

サポートされているプログラミングインタフェースとプラットフォームごとに、複数のバージョンのアプリケーションコードがあります。ただし、このチュートリアルでは、Windows デスクトップ用アプリケーションのコンパイル済みバージョンだけを使用します。各バージョンは、各プラットフォームの変換によって変わります。

1. レッスン 1:CustDB アプリケーションの構築と実行 [130 ページ]
   CustDB アプリケーションを構築します。
2. レッスン 2:Mobile Link サーバの起動と初期同期の実行 [131 ページ]
   Mobile Link サーバを起動し、CustDB アプリケーションを使用して、CustDB データベースを Ultra Light データベースと同期します。
3. レッスン 3:Ultra Light データベース内のデータの更新 [132 ページ]
   CustDB アプリケーションを使用して、リモート データベース内のデータの追加、更新、削除を行います。
4. レッスン 4:Ultra Light データベースと統合データベースの同期 [134 ページ]
   データベースを同期し、Interactive SQL または SQL Central を使用して統合データベースに接続して、変更内容が同期されたことを確認します。
5. レッスン 5: Mobile Link 同期スクリプトのブラウズ [136 ページ]
   同期スクリプトをブラウズすると、CustDB の同期ロジックの機能をよく理解できます。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

関連情報

CustDB サンプルアプリケーションの概要 [17 ページ]

1.11.1 レッスン 1:CustDB アプリケーションの構築と実行

CustDB アプリケーションを構築します。

手順

1. 非 Windows 環境の場合、CustDB アプリケーションを構築します。
   a. 適切な環境で CustDB プロジェクトファイルを開きます。
   b. ソースコードをコンパイルします。
2. CustDB アプリケーションをコピーします。

   Windows 32 ビット環境の場合、CustDB アプリケーション *%SQLANY17%*`¥UltraLite¥Windows¥x86¥custdb.exe` を *%SQLANYSAMP17%*`¥UltraLite¥CustDB` ディレクトリにコピーします。

結果

CustDB アプリケーションがコンパイルされます。

> **注記**
>
> Mobile Link サーバが実行中でない状態で CustDB アプリケーションを起動すると、同期ができないためにエラーメッセージが表示されます。

次のステップ

次のレッスンに進みます。

タスクの概要: チュートリアル:Ultra Light の CustDB サンプルアプリケーションの構築 [129 ページ]

次のタスク: レッスン 2:Mobile Link サーバの起動と初期同期の実行 [131 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

関連情報

Ultra Light の CustDB ファイルのロケーション [20 ページ]

1.11.2 レッスン 2:Mobile Link サーバの起動と初期同期の実行

Mobile Link サーバを起動し、CustDB アプリケーションを使用して、CustDB データベースを Ultra Light データベースと同期します。

前提条件

このチュートリアルのこれまでのレッスンを完了している必要があります。

このチュートリアルの冒頭に一覧されているロールと権限を持っている必要があります。

手順

1. スタート > プログラム > *SQL Anywhere17* > *Mobile Link* > 同期サーバのサンプル の順にクリックします。または、次のコマンドを実行します。

   ```
   mlsrv17 -c "DSN=SQL Anywhere 17 CustDB;uid=ml_server;pwd=sql" -vcrs
   ```

   Mac OS X または Linux では、`mobilink.sh` を使用します。サンプルデータベースのパスワードを指定する必要があります。次に例を示します。

   ```
   mobilink.sh sql
   ```

   ウィンドウには、Mobile Link サーバのステータスに関する情報が表示されます。
2. CustDB アプリケーションを実行します。Windows の場合、*%SQLANYSAMP17%*`¥UltraLite¥CustDB¥custdb.exe` を実行します。
3. ファイル メニューで、同期 をクリックします。

   アプリケーションが同期し、Mobile Link サーバのメッセージウィンドウに、同期が行われたことを示すメッセージが表示されます。

   同期スクリプトによって、ユーザ 50 がログインしたときにアプリケーションにダウンロードされる顧客、製品、注文のサブセットが決まります。ここでは、承認されていない注文のみがダウンロードされます。
4. アプリケーションウィンドウに会社名とサンプル注文情報が表示されることを確認してください。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

結果

CustDB アプリケーションが統合データベースと同期します。

次のステップ

次のレッスンに進みます。

タスクの概要: チュートリアル:Ultra Light の CustDB サンプルアプリケーションの構築 [129 ページ]

前のタスク: レッスン 1:CustDB アプリケーションの構築と実行 [130 ページ]

次のタスク: レッスン 3:Ultra Light データベース内のデータの更新 [132 ページ]

1.11.3 レッスン 3:Ultra Light データベース内のデータの更新

CustDB アプリケーションを使用して、リモート データベース内のデータの追加、更新、削除を行います。

前提条件

このチュートリアルのこれまでのレッスンを完了している必要があります。

このチュートリアルの冒頭に一覧されているロールと権限を持っている必要があります。

手順

1. 注文をブラウズします。

   注文のブラウズは、CustDB アプリケーションの各バージョンで同様の方法で行います。注文をブラウズするときは、ローカルの Ultra Light データベースのデータをスクロールします。顧客はアルファベット順でソートされているので、簡単にリストをスクロールして顧客の名前を見つけることができます。

   a. 顧客のリストを下にスクロールするには、*Next* をクリックします。
   b. 顧客のリストを上にスクロールするには、*Previous* をクリックします。

2. 注文を追加します。

   注文の追加は、CustDB アプリケーションの各バージョンで同様の方法で行います。注文を追加すると、ローカルの Ultra Light データベースでデータが修正されます。このデータは、同期が行われるまで統合データベースとは共有されません。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

a. *Order* > *New* の順にクリックします。
b. *Customer* リストで、方向キーを使用して下へスクロールし、**Basements R Us** をクリックします。
c. *Product* リストで、方向キーを使用して下へスクロールし、**Screwmaster Drill** をクリックします。この製品の価格は *Price* フィールドに自動的に設定されます。
d. *Quantity* フィールドに **20** と入力します。
e. *Discount* フィールドに、**5** (パーセント) と入力し、*OK* をクリックします。

3. 注文の承認、拒否、削除を行います。

ユーザ ID 50 として認証されたユーザはマネージャで、販売担当者と同じ操作に加えて、注文を承認または拒否する権限があります。注文を承認または拒否する場合、注文のステータスを変更し、販売担当者の確認用にメモを追加します。ただし、統合データベースのデータは、同期が行われるまで変更されません。

a. **Apple Street Builders** からの注文を承認します。
   1. *Previous* をクリックしてこの顧客を探します。
   2. *Order*、*Approve* をクリックして、注文を承認します。
   3. *Note* リストで、**Good** をクリックし、*OK* をクリックします。
      注文情報には *Approved* のステータスが表示されます。
b. **Art's Renovations** からの注文を拒否します。
   1. リストの次の項目である **Art's Renovations** からの注文を表示します。
   2. *Order*、*Deny* をクリックして、注文を拒否します。
   3. *Note* リストで、*Discount Is Too High* をクリックして、*OK* をクリックします。
      注文情報には [却下] のステータスが表示されます。
c. **Awnings R Us** に対する注文を削除します。
   1. リストの次の項目である **Awnings R Us** からの注文を表示します。
   2. *Order* > *Delete* の順に選択し、この注文を削除します。
   3. *OK* をクリックして削除を確定します。
      注文に削除済みのマークが付けられます。ただし、変更内容を統合データベースと同期するまでは、現在のデータが Ultra Light リモートデータベースに残ります。

結果

Ultra Light データベースのデータへの変更は保存されますが、CustDB データベースとは同期されません。

次のステップ

次のレッスンに進みます。

タスクの概要: チュートリアル:Ultra Light の CustDB サンプルアプリケーションの構築 [129 ページ]

前のタスク: レッスン 2:Mobile Link サーバの起動と初期同期の実行 [131 ページ]

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

次のタスク: レッスン 4:Ultra Light データベースと統合データベースの同期 [134 ページ]

1.11.4 レッスン 4:Ultra Light データベースと統合データベースの同期

データベースを同期し、Interactive SQL または SQL Central を使用して統合データベースに接続して、変更内容が同期されたことを確認します。

前提条件

このチュートリアルのこれまでのレッスンを完了している必要があります。

このチュートリアルの冒頭に一覧されているロールと権限を持っている必要があります。

コンテキスト

CustDB アプリケーションで同期を行うと、承認された注文情報がデータベースから削除されます。

手順

1. Ultra Light データベースを同期します。

   ファイルメニューで、**データベースの同期**をクリックします。
2. 同期されたことを確認します。

   リモートデータベースでは、**`Awnings R Us`** の注文が削除されたことを確認することで、必要なトランザクションがすべて完了したことを確認できます。このエントリがないことを確認するには、注文をブラウズします。

   統合データベースでは、データを調べて、必要な処理がすべて実行されたことを確認できます。
   - SQL Central を使用し、同期が実行されたことを確認します。
     1. **スタート** > **プログラム** > *SQL Anywhere17* > **管理ツール** > *SQL Central* の順にクリックします。
     2. **接続** > *SQL Anywhere17* **に接続** をクリックします。
     3. **アクション**ドロップダウンメニューで、*ODBC* **データソースを使用した接続**をクリックします。
     4. *ODBC* **データソース名**をクリックします。
     5. **参照**をクリックし、*SQL Anywhere 17 CustDB* をクリックします。
     6. *OK* をクリックします。
     7. **接続**をクリックします。
     8. **テーブル**をダブルクリックします。
     9. **`ULOrder`** をダブルクリックします。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

10. **データ**タブをクリックし、注文 5100 が承認され、注文 5101 が拒否され、注文 5102 が削除されていることを確認します。

- Interactive SQL を使用して、同期が実行されたことを確認します。
  1. Interactive SQL から統合データベースに接続します。
     1. **スタート** > **プログラム** > *SQL Anywhere17* > **管理ツール** > *Interactive SQL* の順にクリックします。
     2. **アクション**ドロップダウンリストで、*ODBC* **データソースを使用した接続**をクリックします。
     3. *ODBC* **データソース名**をクリックし、*SQL Anywhere 17 CustDB* をクリックします。
     4. **接続**をクリックします。
  2. 承認や拒否が同期されたことを確認するため、次の文を実行します。

     ```
     SELECT order_id, status
     FROM ULOrder
     WHERE status IS NOT NULL;
     ```

     この文の結果として、注文 5100 は承認され、5101 は拒否されたことがわかります。
  3. 削除された注文情報の order_id は 5102 です。次のクエリを実行してもローは返りません。これは、その注文情報がシステムから削除されたことを示します。

     ```
     SELECT *
     FROM ULOrder
     WHERE order_id = 5102
     ```

結果

承認された注文はデータベースから削除され、この削除を確認しました。

次のステップ

次のレッスンに進みます。

タスクの概要: チュートリアル:Ultra Light の CustDB サンプルアプリケーションの構築 [129 ページ]

前のタスク: レッスン 3:Ultra Light データベース内のデータの更新 [132 ページ]

次のタスク: レッスン 5: Mobile Link 同期スクリプトのブラウズ [136 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.11.5 レッスン 5: Mobile Link 同期スクリプトのブラウズ

同期スクリプトをブラウズすると、CustDB の同期ロジックの機能をよく理解できます。

前提条件

このチュートリアルのこれまでのレッスンを完了している必要があります。

このチュートリアルの冒頭に記載されているロールと権限を持っている必要があります。

手順

1. *SQL Central* を開きます。

   スタート > プログラム > *SQL Anywhere17* > 管理ツール > *SQL Central* の順にクリックします。
2. 統合データベースに接続します。
   1. コンテキストフィールドで、*SQL Central* を選択し、*Mobile Link 17* をダブルクリックします。
   2. *SQL Central* の右ウィンドウ枠で、*CustDB* をダブルクリックし、統合データベースをダブルクリックします。
   3. *CustDB* を右クリックして、次に接続をクリックします。
   4. アクションドロップダウンメニューで、*ODBC データソースを使用した接続*をクリックします。
   5. *ODBC データソース名*をクリックします。
   6. 参照をクリックし、*SQL Anywhere 17 CustDB* をクリックします。
   7. *OK* をクリックします。
   8. 接続をクリックします。
3. Mobile Link システムを設定します。

   CustDB を右クリックし、*Mobile Link システム設定のチェック*をクリックして、**OK** をクリックします。
4. フォルダオプションが設定されていない場合、ビュー > フォルダ の順にクリックします。
5. 左ウィンドウ枠で、統合データベース > *CustDB* の順に展開します。
6. 接続スクリプトをクリックします。

   右ウィンドウ枠には、同期スクリプトと、それらが関連付けられているイベントセットがリストされています。Mobile Link サーバが同期処理を実行すると、一連のイベントがトリガされます。このときに、イベントに関連付けられた同期スクリプトが実行されます。同期スクリプトを作成し、同期イベントを割り当てることによって、同期の際に実行されるアクションを制御できます。
7. 同期テーブルをクリックします。

   このテーブル固有のスクリプトセットと、対応するイベントが表示されます。これらのスクリプトは、データがリモートデータベースと同期される方法を制御します。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

結果

同期スクリプトを確認しました。

タスクの概要: チュートリアル:Ultra Light の CustDB サンプルアプリケーションの構築 [129 ページ]

前のタスク: レッスン 4:Ultra Light データベースと統合データベースの同期 [134 ページ]

関連情報

Ultra Light クライアント [71 ページ]

1.12 Ultra Light データベースのリファレンス

Ultra Light は、Ultra Light データベースの実行、管理、および設定に役立つ、数多くのツールや機能を提供します。

このセクションの内容:

Ultra Light オプション [138 ページ]
Ultra Light データベースを作成するときに管理できるオプションがいくつかあります。これらのオプションは Ultra Light の多様な用途に役立つように設計されています。作成時に指定したオプションのほとんどは後で変更できません。

Ultra Light 接続パラメータ [175 ページ]
Ultra Light は、Ultra Light データベースに接続する際、下記の接続パラメータをサポートします。

Ultra Light データベースプロパティ [197 ページ]
Ultra Light データベースのプロパティの値は、データベースの最初の作成時に定義されます。

Ultra Light データベースオプション [201 ページ]
Ultra Light データベースのオプション値は、データベースの最初の作成時に定義されますが、データベースへの接続中に変更できます。

Ultra Light ユーティリティ [207 ページ]
Ultra Light は、コマンドプロンプトでデータベースの基本的な管理作業を行うことができる一連のユーティリティを提供します。これらのユーティリティの多くは、SQL Anywhere サーバのユーティリティと機能が似ています。ただし、オプションの使用方法が異なる場合があります。

Ultra Light のシステムテーブル [240 ページ]
Ultra Light データベースのスキーマは独自フォーマットで格納されます。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.12.1 Ultra Light オプション

Ultra Light データベースを作成するときに管理できるオプションがいくつかあります。これらのオプションは Ultra Light の多様な用途に役立つように設計されています。作成時に指定したオプションのほとんどは後で変更できません。

データベースの作成時に作成オプションを指定するには、ulinit または ulload の各ユーティリティを使用するか、サポートされているクライアントインタフェースから指定します。

ブール作成オプションは、YES、Y、ON、TRUE、1 のいずれかによってオンになり、NO、N、OFF、FALSE、0 のいずれかによってオフになります。このオプションでは大文字と小文字は区別されません。

Ultra Light 作成オプションは、プログラムインタフェースからデータベースを作成する際に、セミコロン区切りのリストとして指定されます。

オプションとともにプレフィクスを使用して、アプリケーションが特定のタイプのプラットフォームに関して実行されているときのみオプションを適用するよう指定できます。

オプションとともにプレフィックス `desktop:` を使用することで、アプリケーションがデスクトップで実行中であるときのみオプションが適用されることを示すことができます。

オプションとともにプレフィックス `device:` を使用することで、アプリケーションがモバイルデバイスで実行中であるときのみオプションが適用されることを示すことができます。

> **注記**
>
> オプションがデスクトップとモバイルデバイスの両方で利用可能な場合は、このプレフィックスを使用しないでください。

```
device:DBF=¥Documents¥sample.udb;desktop:DBF=c:¥Databases
¥sample.udb;UID=DBA;device:DBKEY=secret
```

名前	説明	構文
case	Ultra Light データベースの文字列を比較するときに大文字と小文字を区別するかどうかを設定します。	*case*=value
checksum_level	データベースのチェックサム検証のレベルを設定します。	*checksum_level*=value
collation	Ultra Light データベースで使用される照合順を設定します。このプロパティを UTF-8 プロパティと組み合わせて設定するかどうかで、データベースの文字セットが決まります。	*collation*=value
date_format	データベースから取り出した日付のデフォルトの文字列形式を設定します。	*date_format*=value
date_order	年、月、日の日付の順序を解釈する方法を制御します。	*date_order*=value
dbf	Ultra Light データベースのパスとファイル名を指定します。	*dbf*=database-file
dbkey	データベースの暗号化キーを提供します。	*dbkey*=string

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

名前	説明	構文
fips	FIPS 認定の AES 暗号化の使用を制御します。	*fips*=value
kdf_iterations	攻撃の試みを長引かせることで、暗号化されたデータベースへのアクセスをより困難にします。	*kdf_iterations*=value
max_hash_size	デフォルトのインデックスハッシュサイズをバイト単位で設定します。	*max_hash_size*=value
mirror_file	(メインデータベースファイルへの書き込みが発行されると同時に) データベースへのすべての書き込みが発行される、データベースミラーファイルの名前を指定します。	*mirror_file*=mirror-file
nearest_century	文字列から日付への変換で、2 桁の年の解釈を制御します。	*nearest_century*=value
obfuscate	データベースのデータを難読化するかどうか制御します。難読化は、巧妙で容赦ないデータへのアクセス試行に対してセキュアではありません。	*obfuscate*=value
page_size	データベースページサイズを定義します。	*page_size*=sizek
precision	計算結果が小数の場合の最大桁数を指定します。	*precision*=value
pwd	ユーザのパスワードを設定します。	*pwd*=password
reserve_size	実際にデータを挿入することなく、Ultra Light データベースが必要とするファイルシステム領域を事前に割り付けます。	*reserve_size*=number{ *k* | *m* | *g* }
scale	計算結果が最大精度でトランケートされる場合の、小数点以下の最小桁数を指定します。	*scale*=value
time_format	データベースから取り出した時刻の表示形式を設定します。	*time_format*=value
timestamp_format	データベースから取り出したタイムスタンプの表示形式を設定します。	*timestamp_format*=value
timestamp_increment	Ultra Light でのタイムスタンプのトランケート方法を指定します。	*timestamp_increment*=value
timestamp_with_time_zone_format	このオプションでは、データベースから取り出した TIMESTAMP WITH TIME ZONE 値の表示形式を設定します。	*timestamp_with_time_zone_format*=value

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

名前	説明	構文
uid	データベースに対してデフォルトのユーザ ID を設定します。	*uid*=user
utf8_encoding	Unicode 用の 8 ビットマルチバイトエンコードである UTF-8 フォーマットでデータをエンコードします。	*utf8_encoding*=value

このセクションの内容:

Ultra Light の case 作成オプション [142 ページ]
Ultra Light データベースの文字列を比較するときに大文字と小文字を区別するかどうかを設定します。

Ultra Light checksum_level 作成オプション [143 ページ]
データベースのチェックサム検証のレベルを指定します。

Ultra Light の collation 作成オプション [144 ページ]
データベースの照合を設定します。

Ultra Light の date_format 作成オプション [145 ページ]
日付値を文字列に変換するために使用されるフォーマットを指定します。

Ultra Light の date_order 作成オプション [147 ページ]
日付文字列の解釈時における日付部分のデフォルトの順番を指定します。

Ultra Light の DBF 作成オプション [149 ページ]
Ultra Light データベースのパスとファイル名を指定します。

Ultra Light の DBKEY 作成オプション [150 ページ]
新しい Ultra Light データベースの作成時に、この作成オプションでデータベースの暗号化キーを設定します。

Ultra Light desktop 作成オプションプレフィックス [151 ページ]
Ultra Light オプションとともにプレフィックス desktop: を使用することで、アプリケーションがデスクトップで実行中であるときのみオプションが適用されることを示すことができます。

Ultra Light device 作成オプションプレフィックス [152 ページ]
Ultra Light オプションとともにプレフィックス device: を使用することで、アプリケーションがモバイルデバイスで実行中であるときのみオプションが適用されることを示すことができます。

Ultra Light 作成オプション [153 ページ]
新しいデータベースで AES または AES_FIPS の強力な暗号化方式のどちらを使用するかを制御します。

Ultra Light の kdf_iterations 作成オプション [154 ページ]
DBKEY オプションによって提供されるパスフレーズを実際の暗号化キーに変換するキー導出機能の繰り返し回数を 1000 単位で指定します。

Ultra Light max_hash_size 作成オプション [155 ページ]
デフォルトの最大プライマリキーまたはインデックスハッシュサイズをバイト単位で指定します。

Ultra Light nearest_century 作成オプション [157 ページ]
文字列から日付への変換で、2 桁の年の解釈を指定します。

Ultra Light の PWD 作成オプション [158 ページ]
デフォルトユーザのパスワードを指定します。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

Ultra Light obfuscate 作成オプション [159 ページ]
データベース内のデータに対して単純難読化を指定します。

Ultra Light page_size 作成オプション [160 ページ]
データベースページサイズをキロバイト単位で指定します。

Ultra Light precision 作成オプション [162 ページ]
計算結果が小数の場合の最大桁数を指定します。

Ultra Light scale 作成オプション [164 ページ]
計算結果が最大精度でトランケートされる場合の、小数点以下の最小桁数を指定します。

Ultra Light time_format 作成オプション [165 ページ]
時間値を文字列に変換するために使用されるフォーマットを指定します。

Ultra Light timestamp_format 作成オプション [167 ページ]
タイムスタンプ値を文字列に変換するために使用されるフォーマットを指定します。

Ultra Light timestamp_increment 作成オプション [170 ページ]
タイムスタンプ値の精度に関する制限を指定します。データベースにタイムスタンプが挿入されると、この増分に合わせてタイムスタンプはトランケートされます。

Ultra Light timestamp_with_time_zone_format 作成オプション [171 ページ]
TIMESTAMP WITH TIME ZONE 値を文字列に変換するために使用されるフォーマットを指定します。

Ultra Light UID 作成オプション [173 ページ]
データベースに対してデフォルトのユーザ ID を指定します。

Ultra Light utf8_encoding 作成オプション [174 ページ]
データベースに対して UTF-8 エンコード (Unicode 用の 8 ビットマルチバイトエンコード) を指定します。

関連情報

作成オプション値へのアクセス方法 [32 ページ]
Ultra Light 文字セット [32 ページ]
Ultra Light データベース初期化ユーティリティ (ulinit) [221 ページ]
Ultra Light データベースへの XML のロードユーティリティ (ulload) [227 ページ]
Ultra Light DBF 接続パラメータ [185 ページ]
Ultra Light DBKEY 接続パラメータ [187 ページ]
Ultra Light desktop 接続パラメータプレフィックス [189 ページ]
Ultra Light device 接続パラメータプレフィックス [190 ページ]
Ultra Light MIRROR_FILE 接続パラメータ [191 ページ]
Ultra Light PWD 接続パラメータ [192 ページ]
Ultra Light RESERVE_SIZE 接続パラメータ [193 ページ]
Ultra Light UID 接続パラメータ [196 ページ]
データベースオプションへのアクセス [44 ページ]
データベースプロパティの読み取り [43 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.12.1.1 Ultra Light の case 作成オプション

Ultra Light データベースの文字列を比較するときに大文字と小文字を区別するかどうかを設定します。

構文

```
case=value
```

指定可能な値

Ignore、Respect

デフォルト

Ignore

備考

データの大文字と小文字を区別するかしないかは、テーブル、インデックスなどに反映されます。Ultra Light データベースは、デフォルトでは、データは常に入力されたとおりの大文字と小文字で保持されていますが、比較において大文字と小文字を区別しません。識別子 (テーブル名、カラム名など) とユーザ ID は、データベースの大文字と小文字の区別に関係なく、常に大文字と小文字が区別されません。データベースの大文字と小文字の区別の設定に関係なく、パスワードは常に大文字と小文字が区別されます。

文字列の比較結果とソート順は、データベースで大文字と小文字を区別するかどうかによって異なります。

ただし、一部の照合では、識別子の大文字と小文字の区別を前提とする場合、特別な注意が必要です。特に、トルコ語の照合では、予期できない複雑なエラーが発生するような大文字と小文字の変換動作があります。最も一般的なエラーは、i または I という文字を含むシステムオブジェクトが見つからないというものです。

既存のデータベースの大文字と小文字を区別するかどうかは変更できません。変更する必要がある場合は、新しいデータベースを作成してください。

SQL Central では、データベースを作成するウィザードで大文字と小文字を区別するかどうかを設定できます。新しいデータベースの照合と文字セットページで文字列の比較で大文字と小文字を区別するオプションをクリックします。

クライアントアプリケーションから、DatabaseManager/ULDatabaseManager クラスでの CreateDatabase メソッドに対する作成オプションのいずれかとして、このオプションを設定します。*case=respect* をプログラミングインタフェースの CreateDatabase メソッドの作成文字列パラメータに渡します (または、大文字と小文字が区別されるデータベースの場合は *case=ignore*)。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

関連情報

Ultra Light の文字列 [249 ページ]
作成オプション値へのアクセス方法 [32 ページ]
Ultra Light データベース初期化ユーティリティ (ulinit) [221 ページ]
Ultra Light データベースへの XML のロードユーティリティ (ulload) [227 ページ]

1.12.1.2 Ultra Light checksum_level 作成オプション

データベースのチェックサム検証のレベルを指定します。

構文

```
checksum_level=value
```

指定可能な値

0、1、2

デフォルト

0

備考

チェックサムは、ディスク、フラッシュ、またはメモリに保管されているページのオフラインの破損を検出するために使用します。チェックサムによって、重要なページの破損が原因で他のデータが破損するのを防ぐことができます。選択するレベルによって、Ultra Light によって各データベースページのチェックサムが計算、記録されてから、ページが記憶領域に書き込まれます。

計算されたチェックサムが、記憶領域から読み取られたページのチェックサムと一致しない場合は、ページの保管または取り出し中にページが変更されたか、壊れた可能性があります。チェックサムが一致しなかった場合、データベースでページがロードされるときに、Ultra Light によってデータベースが停止され、致命的なエラーがレポートされます。このエラーは解決できません。Ultra Light データベースを再作成し、データベースのエラーを SAP にレポートしてください。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

チェックサムが有効になっている Ultra Light データベースをアンロードしてから再ロードした場合、チェックサムのレベルは保持され、リストアされます。

checksum_level では、次の値がサポートされています。

0

データベースページにチェックサムを追加しません。

1

インデックスや同期ステータスのページなど、重要なデータベースページにチェックサムを追加し、ローページには追加しません。

2

すべてのデータベースページにチェックサムを追加します。

SQL Central では、データベースを作成するウィザードでチェックサムの使用を設定できます。データベース作成ウィザードの新しいデータベースの記憶領域設定ページで、データベースページのチェックサムレベルオプションをクリックします。

クライアントアプリケーションから、DatabaseManager/ULDatabaseManager クラスでの CreateDatabase メソッドに対する作成オプションのいずれかとして、このオプションを設定します。

関連情報

Ultra Light のパフォーマンスに関するヒント [527 ページ]
Ultra Light データベースへの接続 [38 ページ]
Ultra Light データベースプロパティ [197 ページ]
Ultra Light データベース初期化ユーティリティ (ulinit) [221 ページ]
Ultra Light データベースへの XML のロードユーティリティ (ulload) [227 ページ]
Ultra Light page_size 作成オプション [160 ページ]

1.12.1.3 Ultra Light の collation 作成オプション

データベースの照合を設定します。

構文

```
collation=value
```

指定可能な値

文字列

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

デフォルト

932JPN

備考

クライアントアプリケーションから、DatabaseManager/ULDatabaseManager クラスでの CreateDatabase メソッドに対する作成オプションのいずれかとして、このオプションを設定します。

次のコマンドを使用することで、Ultra Light によってサポートされている照合のリストを表示できます。

```
ulinit -Z
```

SQL Central を使用することで照合を設定できます。新しいデータベースの照合と文字セットページで、デフォルトの照合 (932JPN) を選択するか、リストから別の照合を選択します。

関連情報

Ultra Light 文字セット [32 ページ]
作成オプション値へのアクセス方法 [32 ページ]
Ultra Light でサポートされている照合 [33 ページ]
Ultra Light データベース初期化ユーティリティ (ulinit) [221 ページ]
Ultra Light データベースへの XML のロードユーティリティ (ulload) [227 ページ]
Ultra Light utf8_encoding 作成オプション [174 ページ]

1.12.1.4 Ultra Light の date_format 作成オプション

日付値を文字列に変換するために使用されるフォーマットを指定します。

構文

```
date_format=value
```

指定可能な値

文字列

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

デフォルト

YYYY-MM-DD (ISO 日付形式仕様に準拠)

備考

Android デバイスの場合は、代わりに Connection.setOption(OPTION_DATE_FORMAT, value) を使用して、この作成オプションを設定できます。

デフォルトの日付フォーマット YYYY-MM-DD は、ISO 8601 に準拠しています。たとえば、"January 7, 2006" の場合、このフォーマットでは "2006-01-07" として表されます。年、月、日に関して、異なるフォーマットと順序を指定できます。

形式は次の記号を組み合わせた文字列です。

記号	説明
YY	2 桁の年。
YYYY	4 桁の年。
MM	2 桁の月、または 2 桁の分 (hh:mm のように、コロンの後ろに続く場合)
MMM[m...]	月を示す略式文字"m" の数と同数の文字。大文字の M の場合、出力も大文字になります。
D	1 桁の曜日 (0 = 日曜日、6 = 土曜日)。
DD	2 桁の指定月の日。先頭の 0 は必要ありません。
DDD[d...]	曜日を示す略式文字。大文字の D の場合、出力も大文字になります。
JJJ	ユリウス日付 (1 ～ 366)。

既存のデータベースの日付形式は変更できません。変更する必要がある場合は、新しいデータベースを作成してください。

指定可能な値は、上の表に示す記号から構成されます。各記号は、その形式を適用した日付のデータで置き換えられます。

出力の大文字と小文字の制御

文字データを表す記号 (MMM など) では、出力の文字を次のように制御できます。

- 記号を大文字で入力すると、大文字で表記されます。たとえば、MMM と入力すると、JAN と表記されます。
- 記号を小文字で入力すると、小文字で表記されます。たとえば、mmm と入力すると、jan と表記されます。
- 大文字と小文字を混ぜて入力すると、使用される言語に適切な文字が Ultra Light により選択されます。たとえば、Mmm と入力すると、英語では May、フランス語では mai と表記されます。

0 埋め込みの制御

数値データを表す記号では、記号に大文字を使用するか小文字を使用するかで 0 埋め込みを制御できます。

- 記号をすべて大文字または小文字 (MM や mm など) で入力すると、0 埋め込みが行われます。たとえば、yyyy/mm/dd と入力すると、2002/01/01 と表記されます。
- 大文字と小文字を混ぜて入力すると (Mm など)、0 埋め込みは行われません。たとえば、yyyy/Mm/Dd と入力すると、2002/1/1 と表記されます。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

SQL Central では、データベースを作成するウィザードで日付形式を設定できます。新しいデータベース作成パラメータページで日付形式オプションをクリックします。

クライアントアプリケーションから、DatabaseManager/ULDatabaseManager クラスでの CreateDatabase メソッドに対する作成オプションのいずれかとして、このオプションを設定します。

例

次の表は、date_format の設定と、2001 年 5 月 21 日 (木) に実行された SELECT CURRENT DATE 文からの出力を示します。

使用した **date_format** の構文	返された結果
YYYY/MM/DD/ddd	2001/05/21/thu
JJJ	141
mmm YYYY	may 2001
MM-YYYY	05-2001

関連情報

作成オプション値へのアクセス方法 [32 ページ]
Ultra Light の date_order 作成オプション [147 ページ]
Ultra Light データベース初期化ユーティリティ (ulinit) [221 ページ]
Ultra Light データベースへの XML のロードユーティリティ (ulload) [227 ページ]

1.12.1.5 Ultra Light の date_order 作成オプション

日付文字列の解釈時における日付部分のデフォルトの順番を指定します。

構文

```
date_order=value
```

指定可能な値

MDY、YMD、DMY

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

デフォルト

YMD (ISO 日付形式仕様に準拠)

備考

Android デバイスの場合は、代わりに Connection.setOption(OPTION_DATE_ORDER, value) を使用して、この作成オプションを設定できます。

年、月、日のデフォルトの順序は、ISO 8601 日付フォーマットに対応しています。たとえば、"06-01-07" は、2006 年 1 月 7 日として解釈されます。

日付部分の解釈のために、さまざまな順番を指定できます。たとえば、"06-01-07" を 2007 年 6 月 1 日として解釈させるには、日付の順序を "MDY" と指定します。

新しいデータベースに関しては、日付の順序のみ指定できます。データベースが作成された後に、日付の順序を変更することはできません。

SQL Central では、データベースを作成するウィザードで日付の順序を設定できます。新しいデータベース作成パラメータページで、日付順オプションをクリックします。

クライアントアプリケーションから、DatabaseManager/ULDatabaseManager クラスでの CreateDatabase メソッドに対する作成オプションのいずれかとして、このオプションを設定します。

例

値によって、日付 10/11/12 をどう解釈するかが決まります。

日付順	意味
MDY	October 11 2012
YMD	November 12 2010
DMY	November 10 2012

nearest_century オプションを使用して、文字列から日付への変換で、2 桁の年の解釈を制御します。

次の SQL クエリは、デフォルトの date_order と nearest_century 設定を使用して、2010-11-12 を返します。

```
SELECT CAST( CAST( '10/11/12' AS DATE ) AS VARCHAR(15) );
```

関連情報

作成オプション値へのアクセス方法 [32 ページ]
Ultra Light の date_format 作成オプション [145 ページ]
Ultra Light データベース初期化ユーティリティ (ulinit) [221 ページ]
Ultra Light データベースへの XML のロードユーティリティ (ulload) [227 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.12.1.6 Ultra Light の DBF 作成オプション

Ultra Light データベースのパスとファイル名を指定します。

この作成オプションは、新しいデータベースファイルに対してパスとファイル名を指定するとき、または既存のデータベースファイルに接続するときに使用します。

構文

```
DBF=ul-db
```

動作

1. 接続時に、データベースがすでに実行されているか確認します。DBN が指定されている場合、一致するデータベースを探して、見つかったら接続し、見つからない場合は自動起動に進みます。
2. DBF が指定されている場合、一致するデータベース (同一ファイル名) を探して、見つかったら接続し、見つからない場合は自動起動に進みます。
3. DBN も DBF も指定されておらず、1 つのデータベースが実行されている場合、そのデータベースに接続します。
4. DBF が指定されている場合、必要であれば、データベースは自動起動されます。DBN も指定されている場合、DBN が実行するデータベースの名前になり、それ以外の場合、基本のファイル名から名前が生成されます。

備考

単一の接続文字列から、異なるデバイスにある複数のデータベースに接続する場合は、次のオプションを使用して、プラットフォーム固有の代替パラメータを指定できます。

- desktop:DBF
- device:DBF

プラットフォーム固有の作成オプションを指定した場合は、そのパラメータが DBF より優先されます。

DBF の値は、プラットフォームのファイル名の要件を満たしている必要があります。

Windows Mobile

Windows Mobile デバイスに配備する場合は、Windows デスクトップ上で実行される Ultra Light 管理ツールを使用して、接続している Windows Mobile デバイス上の Ultra Light データベースに接続できます。Windows Mobile デバイス上のファイルを特定するには、必要な絶対パスを指定し、*wce:* ファイルプレフィクスを使用する必要があります。

Windows Mobile デバイス上で実行しているアプリケーションで、*wce:* ファイルプレフィックスを使用することはできません。

オプション値に含まれる前後のスペースはすべて無視されます。値に、先頭の一重引用符、先頭の二重引用符、またはセミコロンを含めることはできません。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

関連情報

Ultra Light 接続文字列とパラメータ [39 ページ]
Ultra Light データベースへの接続 [607 ページ]
Embedded SQL を使用した Ultra Light データベース接続 [641 ページ]
Ultra Light データベースの接続の設定 [560 ページ]
Ultra Light オプション [138 ページ]

1.12.1.7 Ultra Light の DBKEY 作成オプション

新しい Ultra Light データベースの作成時に、この作成オプションでデータベースの暗号化キーを設定します。

構文

```
DBKEY=string
```

デフォルト

キーは指定されません。

備考

暗号化キーを使用してデータベースを作成すると、データベースファイルは 256 ビット AES または FIPS 認定の 256 ビット AES アルゴリズムを使用して強力に暗号化されます。強力な暗号化を使用すると、巧妙で容赦ないデータへのアクセス試行に対抗するセキュリティが提供されます。

オプション値に含まれる前後のスペースはすべて無視されます。値に、先頭の一重引用符、先頭の二重引用符、またはセミコロンを含めることはできません。

関連情報

Ultra Light 接続文字列とパラメータ [39 ページ]
データベースセキュリティ [35 ページ]
Ultra Light データベースへの接続 [607 ページ]
Embedded SQL を使用した Ultra Light データベース接続 [641 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

Ultra Light データベースの接続の設定 [560 ページ]
Ultra Light オプション [138 ページ]
Ultra Light obfuscate 作成オプション [159 ページ]

1.12.1.8 Ultra Light desktop 作成オプションプレフィックス

Ultra Light オプションとともにプレフィックス desktop: を使用することで、アプリケーションがデスクトップで実行中であるときのみオプションが適用されることを示すことができます。

構文

```
desktop:option=value
```

備考

オプションがデスクトップとモバイルデバイスの両方で利用可能な場合は、このプレフィックスを使用しないでください。

さまざまなデバイス上で実行される Ultra Light クライアントアプリケーションに対しては、`desktop` オプションプレフィックスを使用します。

`desktop` または `device` プレフィックスのあるオプションは、プレフィックスのないオプションよりも優先されます。

例

次の例は、デスクトップおよびモバイルデバイス用のさまざまなデータベースファイル、デスクトップ上のテンポラリディレクトリの場所、およびモバイルデバイスに対する cache_size を示します。

```
"desktop:DBF=C:¥dir¥db.udb;device:DBF=¥SD Card¥db.udb;desktop:temp_dir=
¥Temp;device:cache_size=4M"
```

関連情報

Ultra Light 接続文字列とパラメータ [39 ページ]
接続パラメータでの Ultra Light ファイルパスの指定 [41 ページ]
Ultra Light 管理ツールの接続パラメータの優先度 [40 ページ]
Ultra Light データベースへの接続 [607 ページ]
Embedded SQL を使用した Ultra Light データベース接続 [641 ページ]
Ultra Light データベースの接続の設定 [560 ページ]

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

Ultra Light オプション [138 ページ]
Ultra Light DBF 接続パラメータ [185 ページ]

1.12.1.9 Ultra Light device 作成オプションプレフィックス

Ultra Light オプションとともにプレフィックス device: を使用することで、アプリケーションがモバイルデバイスで実行中であるときのみオプションが適用されることを示すことができます。

構文

```
device:option=value
```

備考

オプションがデスクトップとモバイルデバイスの両方で利用可能な場合は、このプレフィックスを使用しないでください。

さまざまなデバイス上で実行される Ultra Light クライアントアプリケーションに対しては、`device` オプションプレフィックスを使用します。

`desktop` または `device` プレフィクスのあるオプションは、プレフィクスのないオプションよりも優先されます。

例

次の例は、デスクトップおよびモバイルデバイス用のさまざまなデータベースファイル、モバイルデバイス上のテンポラリディレクトリの場所、およびモバイルデバイスに対する cache_size を示します。

```
"desktop:DBF=C:¥dir¥db.udb;device:DBF=¥SD Card¥db.udb;device:temp_dir=
¥Temp;device:cache_size=4M"
```

関連情報

Ultra Light 接続文字列とパラメータ [39 ページ]
接続パラメータでの Ultra Light ファイルパスの指定 [41 ページ]
Ultra Light 管理ツールの接続パラメータの優先度 [40 ページ]
Ultra Light データベースへの接続 [607 ページ]
Embedded SQL を使用した Ultra Light データベース接続 [641 ページ]
Ultra Light データベースの接続の設定 [560 ページ]
Ultra Light オプション [138 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

Ultra Light DBF 接続パラメータ [185 ページ]

1.12.1.10 Ultra Light 作成オプション

新しいデータベースで AES または AES_FIPS の強力な暗号化方式のどちらを使用するかを制御します。

構文

```
fips=value
```

指定可能な値

Yes (AES_FIPS を使用)、No (AES を使用)

デフォルト

Yes

備考

このオプションは、Ultra Light J または Apple iOS 用の Ultra Light ではサポートされていません。

データベースの暗号化タイプを変更するには、fips 作成パラメータまたは obfuscate 作成オプションを使用してデータベースを作成し直す必要があります。データベースの暗号化キーを変更するには、新しい暗号化キーを Connection オブジェクトで指定します。データベースに接続するユーザは、接続するたびにキーを指定する必要があります。

SQL Central では、データベースを作成するウィザードで暗号化を設定できます。データベース作成ウィザードの新しいデータベースの記憶領域設定ページで、データベースを暗号化するおよび強力な暗号化を使用オプションをクリックし、AES 暗号化のタイプを選択します。暗号化キーを指定し、確認する必要もあります。

クライアントアプリケーションから、DatabaseManager/ULDatabaseManager クラスでの CreateDatabase メソッドに対する作成オプションのいずれかとして、このオプションを設定します。

FIPS オプションを有効にしたデータベースを配備するには、プラットフォームに適切なライブラリをすべてコピーします。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

関連情報

データベースセキュリティ [35 ページ]
作成オプション値へのアクセス方法 [32 ページ]
Ultra Light obfuscate 作成オプション [159 ページ]
Ultra Light DBKEY 接続パラメータ [187 ページ]
Ultra Light データベース初期化ユーティリティ (ulinit) [221 ページ]
Ultra Light データベースへの XML のロードユーティリティ (ulload) [227 ページ]

1.12.1.11 Ultra Light の kdf_iterations 作成オプション

DBKEY オプションによって提供されるパスフレーズを実際の暗号化キーに変換するキー導出機能の繰り返し回数を 1000 単位で指定します。

構文

```
kdf_iterations=value
```

指定可能な値

1 ～ 1000

デフォルト

Apple Mac OS X や iOS のデフォルト値は 30 で、結果として繰り返し回数は 30000 になります。

デスクトップやデバイスプラットフォームを含め、他のプラットフォームのデフォルト値は 5 で、結果として繰り返し回数は 5000 になります。

備考

このパラメータはデータベースの作成時のみ指定されます。

キー導出機能は、攻撃の試みを長引かせることで、暗号化されたデータベースへのアクセスをより困難にします。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

> **注記**
>
> 繰り返しの数を大きくすることで、総当たり攻撃によるパスワードの解読を困難にできますが、データベースの起動時間が長くなります。

明示的な設定が必要なケースは 2 つあります。

1. デバイスの速度が非常に遅いため、暗号化によって Ultra Light の起動に時間がかかりすぎる。例:

   ```
   kdf_iterations=1
   ```

2. ハイエンドなコンピュータを使用していて、セキュリティの強化が必要である。たとえば、Windows または Linux デスクトップ上で Ultra Light を実行しているなど。

   ```
   kdf_iterations=100
   ```

1.12.1.12 Ultra Light max_hash_size 作成オプション

デフォルトの最大プライマリキーまたはインデックスハッシュサイズをバイト単位で指定します。

構文

```
max_hash_size=value
```

指定可能な値

0 ～ 32

デフォルト

4

備考

ハッシュは、インデックスエントリのオプション部分で、インデックスのページに格納されます。ハッシュによって、インデックスカラムの実際のローの値が、インデックスの順序を保持したまま、それに相当する数値 (キー) に変換されます。キーのサイズと、実際の値がハッシュされる長さは、設定するハッシュのサイズで決まります。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

Ultra Light では、ロー ID によって、テーブル内の実際のデータのローを見つけることができます。ロー ID は、常にインデックスエントリの一部です。ハッシュサイズを 0 に設定する (インデックスのハッシュを無効にする) と、インデックスエントリにはこのロー ID だけが含まれます。ハッシュサイズを 0 以外に設定すると、そのローの変換されたデータすべてまたはその一部を含むハッシュキーがロー ID とともにインデックスのページに格納されます。Ultra Light で必ずしもデータを検索、ロード、アンパックしてから実際のローの値を比較する必要がないため、これらのインデックスカラムに対するクエリのパフォーマンスを向上できます。

データベースのデフォルトのハッシュサイズを決定するには、クエリの効率とデータベースのサイズのトレードオフを評価する必要があります。最大ハッシュ値が大きいほど、データベースのサイズが大きくなります。

Ultra Light では、このオプションで指定されている最大値を上限として、カラムのデータ型に必要なバイト数のみが使用されます。デフォルトのハッシュサイズは、インデックスを作成するときにサイズを設定しなかった場合にだけ使用されます。デフォルトのハッシュサイズを 0 に設定すると、Ultra Light では、ローの値はハッシュされません。

既存のインデックスのハッシュサイズは変更できません。プライマリキーまたは新しいインデックスを作成するとき、デフォルトの値を無効化するには、SQL Central の Ultra Light プライマリキー設定ウィザードまたはインデックス作成ウィザードを使用するか、CREATE INDEX または CREATE TABLE 文の WITH MAX SIZE 句を使用します。

カラムを DOUBLE、FLOAT、または REAL として宣言した場合は、ハッシュは使用されません。ハッシュサイズは常に無視されます。

SQL Central では、データベースを作成するウィザードで最大ハッシュサイズを設定できます。データベース作成ウィザードの新しいデータベースの記憶領域設定ページで、インデックスのデフォルトの最大ハッシュサイズオプションをクリックします。

クライアントアプリケーションから、DatabaseManager/ULDatabaseManager クラスでの CreateDatabase メソッドに対する作成オプションのいずれかとして、このオプションを設定します。

関連情報

Ultra Light のパフォーマンスに関するヒント [527 ページ]
Ultra Light インデックス [59 ページ]
最適なハッシュサイズ制限 [531 ページ]
作成オプション値へのアクセス方法 [32 ページ]
CREATE INDEX 文 [Ultra Light] [490 ページ]
CREATE TABLE 文 [Ultra Light] [495 ページ]
Ultra Light データベース初期化ユーティリティ (ulinit) [221 ページ]
Ultra Light データベースへの XML のロードユーティリティ (ulload) [227 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.12.1.13 Ultra Light nearest_century 作成オプション

文字列から日付への変換で、2 桁の年の解釈を指定します。

構文

```
nearest_century=value
```

指定可能な値

整数 (0 ～ 100)

デフォルト

50

備考

Android デバイスの場合は、代わりに Connection.setOption(OPTION_NEAREST_CENTURY, value) を使用して、この作成オプションを設定できます。

Ultra Light では、期待値が日付の値であれば、年が文字列内で 2 桁だけで表されている場合でも、文字列を日付に自動的に変換します。年が 2 桁の場合は、適切なロールオーバー値を設定する必要があります。この値より小さい 2 桁の年は 20yy に変換され、この値以上の年は 19yy に変換されます。

適切なロールオーバー値を決定するには、次の点を考慮します。

2 桁の年の使用

年が 2 桁ではない場合は、基準年への変換は適用されません。設定した nearest_century 値よりも小さい 2 桁の年は 20yy に変換され、nearest_century 値以上の年は 19yy に変換されます。

間違った変換の問題を防止するため、4 桁の年を格納してください。

統合データベースの互換性

たとえば、従来の SQL Anywhere では、年に 1900 を加算していました。Adaptive Server Enterprise では基準年を使用するので、yy が 50 より小さい場合は 20yy になります。

日付の意味: 過去または未来

誕生年は、過去のことなので、小さいロールオーバー値が必要です。したがって、yy が 20 未満の年は 20yy に設定してください。これに対して、日付が有効期限の場合は、未来のことなので、大きい値の方が理にかなっています。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

既存のデータベースの基準年は変更できません。変更する必要がある場合は、新しいデータベースを作成してください。

SQL Central では、データベースを作成するウィザードで基準年を設定できます。新しいデータベース作成パラメータページで基準年オプションをクリックします。

クライアントアプリケーションから、DatabaseManager/ULDatabaseManager クラスでの CreateDatabase メソッドに対する作成オプションのいずれかとして、このオプションを設定します。

関連情報

作成オプション値へのアクセス方法 [32 ページ]
Ultra Light データベース初期化ユーティリティ (ulinit) [221 ページ]
Ultra Light データベース初期化ユーティリティ (ulinit) [221 ページ]
Ultra Light データベースへの XML のロードユーティリティ (ulload) [227 ページ]

1.12.1.14 Ultra Light の PWD 作成オプション

デフォルトユーザのパスワードを指定します。

構文

```
PWD=password
```

デフォルト

ユーザ ID とパスワードの両方を設定しなかった場合、Ultra Light では *UID=DBA* と *PWD=sql* で接続が確立されます。

備考

Android デバイスの場合、代わりに Configuration.setPassword を使用して、この作成オプションを設定できます。

すべてのデータベースユーザにはパスワードがあります。Ultra Light では、最大で 4 つのユーザ ID とパスワードの組み合わせがサポートされています。

パスワードは NULL または空の文字列に設定できます。

新しいユーザが作成されるか、または既存のユーザが各自のパスワードを変更すると、ランダムな 4 バイトのソルト値が生成されます。ソルト値は、パスワードハッシュを計算するときにユーザのパスワードに追加され、ハッシュとともにデータベースに

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

格納されます。ソルト値を適用することによって、辞書攻撃に対する脆弱性が低下するだけでなく、同じパスワードを使用するユーザのパスワードハッシュが確実に異なるようになります。

この作成オプションは暗号化されません。ただし、Ultra Light では、パスワードがハッシュされてから保存されるため、パスワードは SQL Central のみを使用して変更できます。

関連情報

Ultra Light 接続文字列とパラメータ [39 ページ]
Ultra Light ユーザ [64 ページ]
ユーザ認証 [656 ページ]
Ultra Light データベースへの接続 [607 ページ]
ユーザ認証 [656 ページ]
Embedded SQL を使用した Ultra Light データベース接続 [641 ページ]
Ultra Light データベースの接続の設定 [560 ページ]
Ultra Light UID 接続パラメータ [196 ページ]
Ultra Light UID 接続パラメータ [196 ページ]

1.12.1.15 Ultra Light obfuscate 作成オプション

データベース内のデータに対して単純難読化を指定します。

構文

```
obfuscate=value
```

指定可能な値

ブール式

デフォルト

0 (データベースの難読化なし)

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

備考

難読化により、第三者はディスクユーティリティを使用してファイルを表示し、データベースのデータを解読することが困難になります。しかしながら難読化は、巧妙で容赦ないデータへのアクセス試行に対してセキュアではありません。単純難読化では、データベースをエンコードするためのキーは不要です。

正しい暗号化キーがないとデータベースにアクセスできないようにするには、強力な暗号化方式を使用する必要があります。

SQL Central では、データベースを作成するウィザードで難読化を設定できます。データベース作成ウィザードの新しいデータベースの記憶領域設定ページで、データベースを暗号化するおよび単純暗号化を使用 (難読化) オプションを選択します。

クライアントアプリケーションから、DatabaseManager/ULDatabaseManager クラスでの CreateDatabase メソッドに対する作成オプションのいずれかとして、このオプションを設定します。

関連情報

データベースセキュリティ [35 ページ]
作成オプション値へのアクセス方法 [32 ページ]
Ultra Light 作成オプション [153 ページ]
Ultra Light DBKEY 接続パラメータ [187 ページ]
Ultra Light データベース初期化ユーティリティ (ulinit) [221 ページ]
Ultra Light データベースへの XML のロードユーティリティ (ulload) [227 ページ]

1.12.1.16 Ultra Light page_size 作成オプション

データベースページサイズをキロバイト単位で指定します。

構文

```
page_size=sizeK
```

指定可能な値

1K、2K、4K、8K、16K

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

デフォルト

4K

備考

ページサイズは、数字の後に "K" または "k" を付けて入力するか、またはバイトと等しい数字 (1024、2048、4096、8192、または 16384) を入力します。

Ultra Light データベースはページ単位で保管され、I/O 操作はすべてページ単位で行われます。選択したページサイズが、データベースのパフォーマンスまたはサイズに影響することがあります。

前述の値以外の値を使用すると、サイズはその値より大きい次の値に変更されます。単位を指定しない場合、デフォルトはバイトです。

プラットフォームの動的メモリの制約が大きい場合は、ページサイズを小さくして、同期メモリ要件による影響を低減させることを検討してください。

ページサイズを選択するときは、次のガイドラインに従ってください。

データベースサイズ

大規模なデータベースでは、通常、より大きなページサイズの方が有利です。より大きなページではより多くの情報を保持できるため、特に、ページの半分以上のサイズのローを挿入する場合に領域を効率よく使用できます。ページが大きいほど、必要なページのスワップが少なくなります。

ローの数

ロー (BLOB を除く) はページに収まる必要があるので、ページサイズによって、パックされたローの最大サイズと、各ページに格納できるローの数が決まります。1 ページを読み込んで 1 つのローの値を取得すると、次のいくつかのローの内容をメモリにロードできるという 2 次的な効果がある場合があります。

クエリのタイプ

通常、ページサイズが小さいと、ランダムな場所から比較的少ない数のローを取り出すクエリに有利な傾向があります。これに対して、ページサイズが大きいと、テーブルの逐次スキャンを実行するクエリに有利です。

キャッシュサイズ

ページサイズが大きいと、必要なキャッシュサイズも大きくなる場合があります。ダイナミックキャッシュサイジングを使用すると、Ultra Light で必要に応じてキャッシュ容量が設定されます。

インデックスエントリ

ページサイズは、インデックスにも影響を与えます。データベースのページが大きいほど、保持できるインデックスエントリ数が多くなります。

デバイスメモリ

特に、メモリが限られている小型のデバイスでデータベースを実行する場合に便利です。たとえば、1 MB のメモリには、1 ページを 1 KB とすると 1000 ページ格納できますが、1 ページ 4 KB であれば 250 ページしか格納できません。

既存のデータベースのページサイズは変更できません。変更する必要がある場合は、新しいデータベースを作成してください。

SQL Central では、データベースを作成するウィザードでページサイズを設定できます。データベース作成ウィザードの新しいデータベースの記憶領域設定ページで、適切なページサイズをクリックします。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

クライアントアプリケーションから、DatabaseManager/ULDatabaseManager クラスでの CreateDatabase メソッドに対する作成オプションのいずれかとして、このオプションを設定します。

例

データベースのページサイズを 8 KB に設定するには、*page_size=8k* または *page_size=8192* を指定します。

```
ulinit test.udb --page_size=8k
```

関連情報

ローのパックとテーブル定義 [53 ページ]
Ultra Light インデックス [59 ページ]
作成オプション値へのアクセス方法 [32 ページ]
Ultra Light の case 作成オプション [142 ページ]
Ultra Light CACHE_SIZE 接続パラメータ [181 ページ]
Ultra Light CACHE_MIN_SIZE 接続パラメータ [180 ページ]
Ultra Light CACHE_MAX_SIZE 接続パラメータ [179 ページ]
Ultra Light RESERVE_SIZE 接続パラメータ [193 ページ]
Ultra Light データベース初期化ユーティリティ (ulinit) [221 ページ]
Ultra Light データベースへの XML のロードユーティリティ (ulload) [227 ページ]

1.12.1.17 Ultra Light precision 作成オプション

計算結果が小数の場合の最大桁数を指定します。

構文

```
precision=value
```

指定可能な値

整数 (1 ～ 127)

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

デフォルト

30

備考

Android デバイスの場合は、代わりに Connection.setOption(OPTION_PRECISION, value) を使用して、この作成オプションを設定できます。

小数点の位置は数値の精度 (precision) と位取り (scale) で決まります。精度は小数点の左右の合計桁数です。位取りは、計算結果が最大精度にトランケートされるときの小数点以下の最小桁数です。

適切な小数点の位置を決定するには、次の点を考慮します。

実行する計算のタイプ

掛け算、割り算、足し算、引き算、集合関数はすべて結果が最大精度を超える可能性があります。

たとえば、DECIMAL(8,2) と DECIMAL(9,2) を掛ける場合、結果には最大 DECIMAL(17,4) が必要です。precision が 15 の場合、15 桁のみが結果に保持されます。scale が 4 の場合、結果は DECIMAL(15.4) になります。scale が 2 の場合、結果は DECIMAL(15,2) になります。どちらの場合も、オーバーフローエラーの可能性があります。

scale と precision の値の関係

scale は、小数点以下の桁数を設定し、負の数や precision より大きい値にはできません。

既存のデータベースの精度は変更できません。変更する必要がある場合は、新しいデータベースを作成してください。

統合データベースとして Oracle データベースを使用している場合、すべての Ultra Light リモートと Oracle 統合データベースが精度として同じ値を使用する必要があります。

SQL Central では、データベースを作成するウィザードで精度を設定できます。新しいデータベース作成パラメータページで、精度オプションをクリックします。

クライアントアプリケーションから、DatabaseManager/ULDatabaseManager クラスでの CreateDatabase メソッドに対する作成オプションのいずれかとして、このオプションを設定します。

関連情報

作成オプション値へのアクセス方法 [32 ページ]
Ultra Light データベース初期化ユーティリティ (ulinit) [221 ページ]
Ultra Light データベース初期化ユーティリティ (ulinit) [221 ページ]
Ultra Light データベースへの XML のロードユーティリティ (ulload) [227 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.12.1.18 Ultra Light scale 作成オプション

計算結果が最大精度でトランケートされる場合の、小数点以下の最小桁数を指定します。

構文

```
scale=value
```

指定可能な値

整数 (0 ～ 127)

デフォルト

6

備考

Android デバイスの場合は、代わりに Connection.setOption(OPTION_SCALE, value) を使用して、この作成オプションを設定できます。

小数点の位置は数値の精度 (precision) と位取り (scale) で決まります。精度は小数点の左右の合計桁数です。位取りは、計算結果が最大精度にトランケートされるときの小数点以下の最小桁数です。

適切な小数点の位置を決定するには、次の点を考慮します。

実行する計算のタイプ

掛け算、割り算、足し算、引き算、集合関数はすべて結果が最大精度を超える可能性があります。

たとえば、DECIMAL(8,2) と DECIMAL(9,2) を掛ける場合、結果には最大 DECIMAL(17,4) が必要です。precision が 15 の場合、15 桁のみが結果に保持されます。scale が 4 の場合、結果は DECIMAL(15,4) になります。scale が 2 の場合、結果は DECIMAL(15,2) になります。どちらの場合も、オーバーフローエラーの可能性があります。

scale と precision の値の関係

scale は、小数点以下の桁数を設定し、負の数や precision より大きい値にはできません。

既存のデータベースの位取りは変更できません。変更する必要がある場合は、新しいデータベースを作成してください。

SQL Central では、データベースを作成するウィザードで位取りを設定できます。新しいデータベース作成パラメータページで、位取りオプションをクリックします。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

クライアントアプリケーションから、DatabaseManager/ULDatabaseManager クラスでの CreateDatabase メソッドに対する作成オプションのいずれかとして、このオプションを設定します。

例

DECIMAL(8,2) と DECIMAL(9,2) を掛けると、結果には DECIMAL(17,4) が必要です。precision が 15 の場合、15 桁のみが結果に保持されます。scale が 4 の場合、結果は DECIMAL(15,4) になります。scale が 2 の場合、結果は DECIMAL(15,2) になります。どちらの場合も、オーバーフローが発生する可能性があります。

関連情報

作成オプション値へのアクセス方法 [32 ページ]
Ultra Light precision 作成オプション [162 ページ]
Ultra Light データベース初期化ユーティリティ (ulinit) [221 ページ]
Ultra Light データベースへの XML のロードユーティリティ (ulload) [227 ページ]

1.12.1.19 Ultra Light time_format 作成オプション

時間値を文字列に変換するために使用されるフォーマットを指定します。

構文

```
time_format=value
```

指定可能な値

文字列 (下記の記号の組み合わせ)

デフォルト

HH:NN:SS.SSS

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

備考

Android デバイスの場合は、代わりに Connection.setOption(OPTION_TIME_FORMAT, value) を使用して、この作成オプションを設定できます。

Ultra Light では time_format 作成オプションで設定した時間の単位を使用して時間が記述されます。時間の単位には、時、分、秒、ミリ秒があります。

時間の値は、文字列で表すこともできます。時間の値は、文字列変数に割り当ててから取り出します。

Ultra Light では、デフォルトの時間標準として ISO 8601 を使用します。この国際時間規格では、時間が 24 時間制で表されます。たとえば、この国際規格で "真夜中" は、00:00:00 と表されます。デフォルトの時間標準を使用しない場合は、これらの時間の単位に別の表示形式と順序を指定します。

形式は次の記号を組み合わせた文字列です。

記号	説明
HH	2 桁の時間 (24 時間表記)。
NN	2 桁の分。
MM	2 桁の分 (HH:MM のようにコロンの後ろに続く場合)。
SS[.ssssss]	秒と小数点第 6 位までの秒の小数。小数点以下 6 桁のタイムスタンプをサポートしないプラットフォームもあります。

既存のデータベースの time_format 作成オプションは変更できません。変更する必要がある場合は、新しいデータベースを作成してください。

各記号は、その形式を適用した時刻のデータで置き換えられます。数字ではなく文字を表す表示形式の記号は、大文字で入力でき、その場合、置き換えられる文字も大文字になります。フォーマット文字列で大文字と小文字を混ぜて使用すると、数字の前に 0 が付きません。

記号に大文字を使用するか小文字を使用するかで 0 埋め込みを制御します。

- 記号をすべて大文字または小文字 (HH や hh など) で入力すると、0 埋め込みが行われます。たとえば HH:NN:SS と入力すると、01:01:01 と表記されます。
- 大文字と小文字を混ぜて入力すると (Hh など)、0 埋め込みは行われません。たとえば Hh:Nn:Ss と入力すると、1:1:1 と表記されます。
- 大文字と小文字を混ぜて入力すると、使用される言語に適切な文字が Ultra Light により選択されます。たとえば、Mmm と入力すると、英語では May、フランス語では mai と表記されます。
- 小数点以下の秒の最初の 2 桁の大文字と小文字を混ぜて入力すると (Ss または sSssss)、後ろの 0 が削除されます。たとえば、hh:nn:ss.Sss と入力すると、12:34:56.1 と表記されます。

SQL Central では、データベースを作成するウィザードで time_format オプションを設定できます。新しいデータベース作成パラメータページで時間形式オプションをクリックします。

クライアントアプリケーションから、DatabaseManager/ULDatabaseManager クラスでの CreateDatabase メソッドに対する作成オプションのいずれかとして、このオプションを設定します。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

例

下記のコマンドは、データベースを作成するとともに、データベースからの時間値取得時に秒の小数が除外されるよう time_format 作成オプションを設定します。

```
ulinit --time_format=HH:NN:SS example.udb
```

作成されたデータベースで下記のクエリを実行します。

```
SELECT CAST(CAST('3:30:12.345 PM' AS TIME) AS CHAR(32))
```

クエリは 15:30:12 を返します。

関連情報

作成オプション値へのアクセス方法 [32 ページ]
Ultra Light timestamp_format 作成オプション [167 ページ]
Ultra Light データベース初期化ユーティリティ (ulinit) [221 ページ]
Ultra Light データベースへの XML のロードユーティリティ (ulload) [227 ページ]

1.12.1.20 Ultra Light timestamp_format 作成オプション

タイムスタンプ値を文字列に変換するために使用されるフォーマットを指定します。

構文

```
timestamp_format=value
```

指定可能な値

文字列

デフォルト値

YYYY-MM-DD HH:NN:SS.SSS

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

備考

Android デバイスの場合は、代わりに Connection.setOption(OPTION_TIMESTAMP_FORMAT, value) を使用して、この作成オプションを設定できます。

デフォルトタイムスタンプフォーマット YYYY-MM-DD HH:NN:SS.SSS は、ISO 8601 に準拠しています。たとえば、"January 7, 2006 12:34 AM" の場合、このフォーマットでは "2006-01-07 00:34:00.000" として表されます。年、月、日、および時間の部分に関して、異なるフォーマットと順序を指定できます。

形式は次の記号から成る文字列です。

記号	説明
YY	2 桁の年。
YYYY	4 桁の年。
MM	2 桁の月、または 2 桁の分 (HH:MM のように、コロンの後ろに続く場合)
MMM[m...]	月を示す略式文字"m" の数と同数の文字。大文字の M の場合、出力も大文字になります。
D	1 桁の曜日 (0 = 日曜日、6 = 土曜日)。
DD	2 桁の指定月の日。先頭の 0 は必要ありません。
DDD[d...]	曜日を示す略式文字。大文字の D の場合、出力も大文字になります。
HH	2 桁の時間。先頭の 0 は必要ありません。
NN	2 桁の分。先頭の 0 は必要ありません。
SS[.ssssss]	秒と秒の小数位。
AA	12 時間表記。正午前の時刻は AM で表します。
PP	12 時間表記。正午後の時刻は PM で表します。
JJJ	指定年の日 (1 ～ 366)。

既存のデータベースの timestamp_format 作成オプションは変更できません。変更する必要がある場合は、新しいデータベースを作成してください。

指定可能な値は、上の表に示す記号から構成されます。各記号は、その形式を適用した日付のデータで置き換えられます。

略式文字については、指定した文字数が数えられます。必要な場合は、A.M. または P.M. のインジケータ (ローカライズ可能) が、指定した文字数に対応するバイト数までトランケートされます。

文字データを表す記号 (MMM など) では、出力の文字を次のように制御できます。

- 記号をすべて大文字で入力すると、すべて大文字で表記されます。たとえば、MMM と入力すると、JAN と表記されます。
- 記号をすべて小文字で入力すると、すべて小文字で表記されます。たとえば、mmm と入力すると、jan と表記されます。
- 大文字と小文字を混ぜて入力すると、使用される言語に適切な文字が Ultra Light により選択されます。たとえば、Mmm と入力すると、英語では May、フランス語では mai と表記されます。

数値データを表す記号では、記号に大文字を使用するか小文字を使用するかで 0 埋め込みを制御できます。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

- 記号をすべて大文字または小文字 (MM や mm など) で入力すると、0 埋め込みが行われます。たとえば、yyyy/mm/dd と入力すると、2002/01/01 と表記されます。
- 大文字と小文字を混ぜて入力すると (Mm など)、0 埋め込みは行われません。たとえば、yyyy/Mm/Dd と入力すると、2002/1/1 と表記されます。
- 大文字と小文字を混ぜて入力すると、使用される言語に適切な文字が Ultra Light により選択されます。たとえば、Mmm と入力すると、英語では May、フランス語では mai と表記されます。
- 小数点以下の秒の最初の 2 桁の大文字と小文字を混ぜて入力すると (Ss または sSssss)、後ろの 0 が削除されます。たとえば、hh:nn:ss.Sss と入力すると、12:34:56.1 と表記されます。

SQL Central では、データベースを作成するウィザードで timestamp_format オプションを設定できます。新しいデータベース作成パラメータページでタイムスタンプ形式オプションをクリックします。

クライアントアプリケーションから、DatabaseManager/ULDatabaseManager クラスでの CreateDatabase メソッドに対する作成オプションのいずれかとして、このオプションを設定します。

例

下記のコマンドは、データベースを作成するとともに、年が 2 桁で表示され、秒の小数がデータベースからのタイムスタンプ値取得時に除外されるよう timestamp_format 作成オプションを設定します。

```
ulinit --timestamp_format="YY-MM-DD HH:NN:SS" example.udb
```

作成されたデータベースで下記のクエリを実行します。

```
SELECT CAST(CAST('Friday May 12, 2006 3:30 PM' AS TIMESTAMP) AS CHAR(32))
```

クエリは 06-05-12 15:30:00 を返します。

関連情報

作成オプション値へのアクセス方法 [32 ページ]
Ultra Light の同時実行性 [541 ページ]
Ultra Light timestamp_increment 作成オプション [170 ページ]
Ultra Light データベース初期化ユーティリティ (ulinit) [221 ページ]
Ultra Light データベースへの XML のロードユーティリティ (ulload) [227 ページ]

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

1.12.1.21 Ultra Light timestamp_increment 作成オプション

タイムスタンプ値の精度に関する制限を指定します。データベースにタイムスタンプが挿入されると、この増分に合わせてタイムスタンプはトランケートされます。

構文

```
timestamp_increment=value
```

指定可能な値

1 ～ 60000000 マイクロ秒

デフォルト

1 マイクロ秒

備考

Android デバイスの場合は、代わりに Connection.setOption(OPTION_TIMESTAMP_INCREMENT, value) を使用して、この作成オプションを設定できます。

1000000 マイクロ秒は 1 秒と等しい点に注意してください。

既存のデータベースの timestamp_increment 作成オプションは変更できません。変更する必要がある場合は、新しいデータベースを作成してください。

DEFAULT TIMESTAMP カラムがプライマリキーまたはロー識別子として使用されている場合、このインクリメントが役に立ちます。

SQL Central では、データベースを作成するウィザードでタイムスタンプインクリメントを設定できます。新しいデータベース作成パラメータページでタイムスタンプインクリメントオプションをクリックします。

クライアントアプリケーションから、DatabaseManager/ULDatabaseManager クラスでの CreateDatabase メソッドに対する作成オプションのいずれかとして、このオプションを設定します。

例

'2000/12/05 10:50:53:700' などの値を格納するには、この作成オプションを 100000 に設定します。この値に設定すると、秒の部分は小数点以下 1 桁の後でトランケートされます。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

関連情報

作成オプション値へのアクセス方法 [32 ページ]
Ultra Light の同時実行性 [541 ページ]
Ultra Light timestamp_format 作成オプション [167 ページ]
Ultra Light データベース初期化ユーティリティ (ulinit) [221 ページ]
Ultra Light データベースへの XML のロードユーティリティ (ulload) [227 ページ]

1.12.1.22 Ultra Light timestamp_with_time_zone_format 作成オプション

TIMESTAMP WITH TIME ZONE 値を文字列に変換するために使用されるフォーマットを指定します。

構文

```
timestamp_with_time_zone_format=value
```

指定可能な値

文字列 (下記の記号の組み合わせ)

デフォルト

YYYY-MM-DD HH:NN:SS.SSS+HH:NN

備考

デフォルトフォーマット YYYY-MM-DD HH:NN:SS.SSS+HH:NN は、ISO 8601 に準拠しています。年、月、日、時、およびタイムゾーンの部分に関して、異なるフォーマットと順序を指定できます。

形式は次の記号を組み合わせた文字列です。

記号	説明
YY	2 桁の年

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

記号	説明
YYYY	4 桁の年
MM	2 桁の月、または 2 桁の分 (HH:MM のように、コロンの後ろに続く場合)
MMM[m...]	月を略した文字 ("m" の数だけ文字を使用)。
DD	2 桁の指定月の日。
DDD[d...]	曜日を示す略式文字。
HH	2 桁の時間。
NN	2 桁の分。
SS[.ssssss]	秒と小数点第 6 位までの秒の小数。小数点以下 6 桁のタイムスタンプをサポートしないプラットフォームもあります。
AA	A.M. または P.M. (12 時間表記)。24 時間表記の場合、AA と PP は省略します。
PP	必要であれば P.M. (12 時間表記)。24 時間表記の場合、AA と PP は省略します。
HH	2 桁の時間 (タイムゾーンオフセット)。
NN	2 桁の分 (タイムゾーンオフセット)。

各記号は、その形式を適用した日付のデータで置き換えられます。

文字データを表す記号 (MMM など) では、出力の文字を次のように制御できます。

- 記号をすべて大文字で入力すると、すべて大文字で表記されます。たとえば、MMM と入力すると、JAN と表記されます。
- 記号をすべて小文字で入力すると、すべて小文字で表記されます。たとえば、mmm と入力すると、jan と表記されます。
- 大文字と小文字を混ぜて入力すると、使用される言語に適切な文字が Ultra Light により選択されます。たとえば、Mmm と入力すると、英語では May、フランス語では mai と表記されます。
- 小数点以下の秒の最初の 2 桁の大文字と小文字を混ぜて入力すると (Ss または sSssss)、後ろの 0 が削除されます。たとえば、hh:nn:ss.Sss と入力すると、12:34:56.1 と表記されます。

文字データがマルチバイトである場合、各記号の長さはバイト数ではなく文字数を表します。たとえば MMM は 3 文字の月名を示しています。

数値データを表す記号では、記号に大文字を使用するか小文字を使用するかで 0 埋め込みを制御できます。

- 記号をすべて大文字または小文字 (MM や mm など) で入力すると、0 埋め込みが行われます。たとえば、yyyy/mm/dd と入力すると、2002/01/01 と表記されます。
- 大文字と小文字を混ぜて入力すると (Mm など)、0 埋め込みは行われません。たとえば、yyyy/Mm/Dd と入力すると、2002/1/1 と表記されます。
- 小数点以下の秒の最初の 2 桁の大文字と小文字を混ぜて入力すると (Ss または sSssss)、後ろの 0 が削除されます。たとえば、hh:nn:ss.Sss と入力すると、12:34:56.1 と表記されます。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

注記

日付形式の順序を変更するように timestamp_with_time_zone_format オプションの設定を変更する場合は、同じ変更を反映するように date_order オプションも変更し、同様に date_order を変更する場合は timestamp_with_time_zone_format も変更してください。

例

下記のコマンドは、データベースを作成するとともに、年が 2 桁で表示され、秒の小数がデータベースからの TIMESTAMP WITH TIME ZONE 値取得時に除外されるよう timestamp_with_time_zone_format 作成オプションを設定します。

```
ulinit --timestamp_with_time_zone_format="YY-MM-DD HH:NN:SS+HH:NN" example.udb
```

作成されたデータベースで下記のクエリを実行します。

```
SELECT CAST(CAST('Friday May 12, 2006 3:30 PM -04:00' AS TIMESTAMP WITH TIME
ZONE) AS CHAR(32))
```

クエリは 06-05-12 15:30:00-04:00 を返します。

1.12.1.23 Ultra Light UID 作成オプション

データベースに対してデフォルトのユーザ ID を指定します。

構文

```
UID=user
```

デフォルト

接続時に UID と PWD を設定しなかった場合、Ultra Light では *UID=DBA* と *PWD=sql* で接続が確立されます。

備考

すべてのデータベースユーザはユーザ ID を持ちます。Ultra Light では、最大で 4 つのユーザ ID とパスワードの組み合わせがサポートされています。

Ultra Light のユーザ ID は、Mobile Link のユーザ名やその他の SQL Anywhere のユーザ ID とは別です。ユーザ ID は一度作成すると変更できません。変更する必要がある場合は、ユーザ ID を削除し、新しい ID を追加してください。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

UID は NULL または空の文字列に設定できません。ユーザ ID の最大長は 31 文字です。ユーザ ID の大文字と小文字は区別されません。

オプション値に含まれる前後のスペースはすべて無視されます。この作成オプションの値に、先頭の一重引用符、先頭の二重引用符、またはセミコロンを含めることはできません。

関連情報

Ultra Light 接続文字列とパラメータ [39 ページ]
Ultra Light ユーザ [64 ページ]
ユーザ認証 [656 ページ]
Ultra Light データベースへの接続 [607 ページ]
ユーザ認証 [656 ページ]
Embedded SQL を使用した Ultra Light データベース接続 [641 ページ]
Ultra Light データベースの接続の設定 [560 ページ]
Ultra Light オプション [138 ページ]

1.12.1.24 Ultra Light utf8_encoding 作成オプション

データベースに対して UTF-8 エンコード (Unicode 用の 8 ビットマルチバイトエンコード) を指定します。

構文

```
utf8_encoding=value
```

値

ブール式

デフォルト

1 (データベースの UTF-8 エンコードあり)

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

備考

UTF-8 文字は 1 ～ 4 バイトで表されます。他のマルチバイト照合の場合、1 または 2 バイトが使用されます。すべてのマルチバイト照合で、2 バイト以上の文字はアルファベットと同様に扱われます。これらの文字は、二重引用符を使用しないで識別子で使用できます。

Ultra Light データベース内の文字は、選択した照合において暗黙的に決定されるコードページ内の文字であるか、または UTF-8 でエンコードされています。UTF8BIN 照合を使用する Ultra Light データベースは、自動的に UTF-8 でエンコードされます。Ultra Light アプリケーションを配備するオペレーティングシステムで (ほとんどの Linux ディストリビューション、Microsoft Windows Mobile、Apple iOS のように) UTF-8 または Unicode が使用される場合、またはデータベースに複数の言語の文字を格納する予定がある場合は、UTF-8 エンコードを使用してデータベースを作成してください。UTF-8 でエンコードされている文字を、Unicode がサポートされていない統合テーブルに同期しようとすると、ユーザエラーがレポートされます。

SQL Central では、データベースを作成するウィザードで UTF-8 エンコーディングを選択できます。新しいデータベースの照合と文字セットページでデータベース内の文字に *UTF-8* エンコードを使用するオプションをクリックします。

クライアントアプリケーションから、DatabaseManager/ULDatabaseManager クラスでの CreateDatabase メソッドに対する作成オプションのいずれかとして、このオプションを設定します。

関連情報

Ultra Light での文字セットのエンコードに関するプラットフォーム要件 [33 ページ]
Ultra Light 文字セット [32 ページ]
作成オプション値へのアクセス方法 [32 ページ]
Ultra Light データベース初期化ユーティリティ (ulinit) [221 ページ]
Ultra Light データベースへの XML のロードユーティリティ (ulload) [227 ページ]

1.12.2 Ultra Light 接続パラメータ

Ultra Light は、Ultra Light データベースに接続する際、下記の接続パラメータをサポートします。

接続パラメータとともにプレフィクスを使用して、アプリケーションが特定のタイプのプラットフォームに関して実行されているときのみパラメータを適用するよう指定できます。

接続パラメータとともにプレフィックス `desktop:` を使用することで、アプリケーションがデスクトップで実行中であるときのみパラメータが適用されることを示すことができます。

接続パラメータとともにプレフィックス `device:` を使用することで、アプリケーションがモバイルデバイスで実行中であるときのみパラメータが適用されることを示すことができます。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

> **注記**
>
> パラメータがデスクトップとモバイルデバイスの両方で利用可能な場合は、このプレフィックスを使用しないでください。

```
device:DBF=¥Documents¥sample.udb;desktop:DBF=c:¥Databases
¥sample.udb;UID=DBA;device:DBKEY=secret
```

パラメータ名	説明
CACHE_MAX_SIZE	ファイルキャッシュに割り当てる最大メモリ量を指定します。
CACHE_MIN_SIZE	ファイルキャッシュに割り当てる最小メモリ量を指定します。
CACHE_SIZE	ファイルキャッシュに割り当てる初期メモリ量を指定します。
COMMIT_FLUSH	ハードウェア障害またはクラッシュの後に、どのトランザクションをリカバリするかを定義します。
CON	接続に名前を付け、マルチ接続アプリケーションで簡単に切り替えができるようにします。
DBF	作成時には、このパラメータはデータベースのロケーションを設定します。 その後の接続で、このパラメータは Ultra Light にデータベースファイルを見つける場所を示します。 単一プラットフォームのアプリケーションを作成している場合や、Ultra Light 管理ツールからデータベースに接続している場合は、DBF を使用できます。さまざまなプラットフォーム固有データベースに接続する Ultra Light クライアントをプログラミングしている場合は、desktop: または device: プレフィックスを使用します。
DBKEY	データベースの作成時には、このパラメータでデータベースの暗号化に使用する暗号化キーを設定します。 その後の接続では、データベースの暗号化に使用する暗号化キーを指定します。
DBN	アプリケーションが複数のデータベースに接続した場合に、データベースを名前で識別します。
MIRROR_FILE	(メインデータベースファイルへの書き込みが発行されると同時に)データベースへのすべての書き込みが発行される、データベースミラーファイルの名前を指定します。
PWD	新しい Ultra Light データベースの作成時に、この接続パラメータでデフォルトユーザのパスワードを設定します。既存のデータベースに接続する場合は、認証に使用するユーザ ID のパスワードを識別します。
RESERVE_SIZE	実際にデータを挿入することなく、Ultra Light データベースが必要とするファイルシステム領域を事前に割り付けます。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

パラメータ名	説明
START	Ultra Light エンジンの実行プログラムを起動します。
TEMP_DIR	Ultra Light でテンポラリファイル (データベース名から導出される名前を持つ) を配置するフォルダ (すでに存在している必要がある) の名前を指定します。
UID	新しい Ultra Light データベースの作成時に、この接続パラメータでデータベースのデフォルトのユーザ ID を設定します。既存のデータベースに接続する場合は、そのデータベースに接続に使用するユーザ ID を指定します。

このセクションの内容:

Ultra Light 接続パラメータプレフィックス [178 ページ]
接続パラメータとともにプレフィクスを使用することで、アプリケーションが特定のタイプのプラットフォームに関して実行されているときのみパラメータを適用するよう指定できます。

Ultra Light CACHE_MAX_SIZE 接続パラメータ [179 ページ]
データベースキャッシュの最大サイズを定義します。Ultra Light ではキャッシュサイズは自動的に管理されるため、このパラメータを設定することは必須ではありません。

Ultra Light CACHE_MIN_SIZE 接続パラメータ [180 ページ]
データベースキャッシュの最小サイズを定義します。Ultra Light ではキャッシュサイズは自動的に管理されるため、このパラメータを設定することは必須ではありません。

Ultra Light CACHE_SIZE 接続パラメータ [181 ページ]
データベースキャッシュの初期サイズを定義します。Ultra Light ではキャッシュサイズは自動的に管理されるため、このパラメータを設定することは必須ではありません。

Ultra Light COMMIT_FLUSH 接続パラメータ [183 ページ]
コミットされたトランザクションが、コミット呼び出しの後にいつ記憶領域にフラッシュされるかを規定します。Ultra Light アプリケーションからコミットが呼び出されないと、フラッシュされません。

Ultra Light CON 接続パラメータ [184 ページ]
接続に名前を付け、マルチ接続アプリケーションで簡単に切り替えができるようにします。

Ultra Light DBF 接続パラメータ [185 ページ]
Ultra Light データベースのパスとファイル名を指定します。

Ultra Light DBKEY 接続パラメータ [187 ページ]
新しい Ultra Light データベースの作成時に、この接続パラメータでデータベースの暗号化キーを設定します。

Ultra Light DBN 接続パラメータ [188 ページ]
アプリケーションが複数のデータベースに接続した場合に、データベースを名前で識別します。

Ultra Light desktop 接続パラメータプレフィックス [189 ページ]
接続パラメータとともにプレフィックス desktop: を使用することで、アプリケーションがデスクトップで実行中であるときのみパラメータが適用されることを示すことができます。

Ultra Light device 接続パラメータプレフィックス [190 ページ]

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

接続パラメータとともにプレフィックス device: を使用することで、アプリケーションがモバイルデバイスで実行中であるときのみパラメータが適用されることを示すことができます。

Ultra Light MIRROR_FILE 接続パラメータ [191 ページ]
(メインデータベースファイルへの書き込みが発行されると同時に) データベースへのすべての書き込みが発行される、データベースミラーファイルの名前を指定します。

Ultra Light PWD 接続パラメータ [192 ページ]
新しい Ultra Light データベースの作成時に、この接続パラメータでデフォルトユーザのパスワードを設定します。

Ultra Light RESERVE_SIZE 接続パラメータ [193 ページ]
実際にデータを挿入することなく、Ultra Light データベースが必要とするファイルシステム領域を事前に割り付けます。ファイルシステム領域を予約すると、領域が他のファイルによって使用されることはありません。

Ultra Light START 接続パラメータ [194 ページ]
Ultra Light エンジンの実行プログラムを起動します。このパラメータは、Android 用 Ultra Light ではサポートされていません。このパラメータは、エンジンが予測されるロケーションにない場合にのみ必要です。

Ultra Light TEMP_DIR 接続パラメータ [195 ページ]
Ultra Light でテンポラリファイル (データベース名から導出される名前を持つ) を配置するフォルダ (すでに存在している必要がある) の名前を指定します。

Ultra Light UID 接続パラメータ [196 ページ]
新しい Ultra Light データベースの作成時に、この接続パラメータでデータベースのデフォルトのユーザ ID を設定します。

関連情報

Ultra Light 接続文字列とパラメータ [39 ページ]

1.12.2.1 Ultra Light 接続パラメータプレフィックス

接続パラメータとともにプレフィクスを使用することで、アプリケーションが特定のタイプのプラットフォームに関して実行されているときのみパラメータを適用するよう指定できます。

接続パラメータとともにプレフィックス `desktop:` を使用することで、アプリケーションがデスクトップで実行中であるときのみパラメータが適用されることを示すことができます。

接続パラメータとともにプレフィックス `device:` を使用することで、アプリケーションがモバイルデバイスで実行中であるときのみパラメータが適用されることを示すことができます。

> **注記**
>
> 接続パラメータがデスクトップとモバイルデバイスの両方で利用可能な場合は、このプレフィックスを使用しないでください。

```
device:DBF=¥Documents¥sample.udb;desktop:DBF=c:¥Databases
¥sample.udb;UID=DBA;device:CACHE_SIZE=100k
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

関連情報

Ultra Light 接続文字列とパラメータ [39 ページ]
Ultra Light 接続パラメータ [175 ページ]

1.12.2.2 Ultra Light CACHE_MAX_SIZE 接続パラメータ

データベースキャッシュの最大サイズを定義します。Ultra Light ではキャッシュサイズは自動的に管理されるため、このパラメータを設定することは必須ではありません。

構文

```
CACHE_MAX_SIZE=number{ k | m }
```

デフォルト

デフォルトの最大キャッシュサイズはデバイスでは 20 MB、デスクトップでは 50 MB です。

備考

cache_max_size 接続パラメータで、ファイルキャッシュに割り当てる最大メモリ容量を指定します。デフォルトでは、size の値はバイト単位となります。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 k、m を使用します。

最大キャッシュサイズを超える場合、代わりにプラットフォームのキャッシュサイズの上限値が使用されます。Ultra Light では、データベースの実ファイルサイズを超える値にキャッシュサイズを拡大することはできません。

キャッシュサイズの上限値としてデータベースのサイズを超える値を指定すると、ローのキャッシュに余分なスペースが使用されることがあります。

接続パラメータ値に含まれる前後のスペースはすべて無視されます。この接続パラメータの値に、先頭の一重引用符、先頭の二重引用符、またはセミコロンを含めることはできません。

例

次の接続文字列フラグメントによって、最大キャッシュサイズが 100 MB に設定されます。

```
"CACHE_MAX_SIZE=100m"
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

関連情報

Ultra Light 接続文字列とパラメータ [39 ページ]
Ultra Light データベースへの接続 [607 ページ]
Embedded SQL を使用した Ultra Light データベース接続 [641 ページ]
Ultra Light データベースの接続の設定 [560 ページ]
Ultra Light データベースのキャッシュサイズの調整 [528 ページ]
Ultra Light CACHE_MIN_SIZE 接続パラメータ [180 ページ]
Ultra Light CACHE_SIZE 接続パラメータ [181 ページ]
Ultra Light cache_allocation オプション [202 ページ]
Ultra Light page_size 作成オプション [160 ページ]
Ultra Light RESERVE_SIZE 接続パラメータ [193 ページ]

1.12.2.3 Ultra Light CACHE_MIN_SIZE 接続パラメータ

データベースキャッシュの最小サイズを定義します。Ultra Light ではキャッシュサイズは自動的に管理されるため、このパラメータを設定することは必須ではありません。

構文

```
CACHE_MIN_SIZE=number{ k | m }
```

デフォルト

デバイスのデフォルトのキャッシュサイズは 256 KB です。デスクトップのデフォルトのキャッシュサイズは 512 KB です。

備考

cache_max_size 接続パラメータで、ファイルキャッシュに割り当てる最小メモリ容量を指定します。デフォルトでは、size の値はバイト単位となります。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 k、m を使用します。

最小キャッシュサイズが最大キャッシュサイズよりも大きくなるように設定した場合、Ultra Light からエラーメッセージが返され、接続が失敗します。

接続パラメータ値に含まれる前後のスペースはすべて無視されます。この接続パラメータの値に、先頭の一重引用符、先頭の二重引用符、またはセミコロンを含めることはできません。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

例

次の接続文字列フラグメントによって、最小キャッシュサイズが 1 MB に設定されます。

```
"CACHE_MIN_SIZE=1m"
```

関連情報

Ultra Light 接続文字列とパラメータ [39 ページ]
Ultra Light データベースへの接続 [607 ページ]
Embedded SQL を使用した Ultra Light データベース接続 [641 ページ]
Ultra Light データベースの接続の設定 [560 ページ]
Ultra Light CACHE_MAX_SIZE 接続パラメータ [179 ページ]
Ultra Light CACHE_SIZE 接続パラメータ [181 ページ]
Ultra Light cache_allocation オプション [202 ページ]
Ultra Light page_size 作成オプション [160 ページ]
Ultra Light RESERVE_SIZE 接続パラメータ [193 ページ]

1.12.2.4 Ultra Light CACHE_SIZE 接続パラメータ

データベースキャッシュの初期サイズを定義します。Ultra Light ではキャッシュサイズは自動的に管理されるため、このパラメータを設定することは必須ではありません。

Android のデバイスの場合、代わりに Configuration.setPageSize を使用して、この接続パラメータを設定できます。

構文

```
CACHE_SIZE=number{ k | m }
```

デフォルト

デフォルトの初期キャッシュサイズは、システムで使用可能なメモリ容量とデータベースの容量によって決まります。

備考

cache_max_size 接続パラメータで、ファイルキャッシュに割り当てる初期メモリ容量を指定します。このキャッシュは、データベースから最近使用したページをメモリに保持し、次に必要なときにこれらのページにすぐにアクセスできるようするために使

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

用され、またページへの複数の変更を記憶領域に書き込む前に収集するためにも使用されます。キャッシュからのページへのアクセスは、記憶領域から読み込むよりも何倍も速くなります。さらに、記憶領域への書き込みはコストがかさむため、複数の変更を 1 つの書き込みにグループ化することがパフォーマンスのためには重要になります。暗号化されたデータベースにはキャッシュによる利点もあります。復号化はページがキャッシュにロードされた場合にのみ発生し、暗号化はページが記憶領域に書き込まれる前に発生するからです。キャッシュが十分に大きい場合は、暗号化によるオーバヘッドは無視できる程度のものになります。

キャッシュの使用例として、同期を考えてみてください。Ultra Light がダウンロードを受け取っている間、データベースに行が挿入され、参照整合性検査が実行されます。挿入されると、行にはインデックスが付けられ、テーブル上の各インデックスに追加されます。このため、同期が行われている間、キャッシュは新しい行が保存されているページと現在のテーブルのインデックスページを保持する傾向があります。同期のパフォーマンスは、キャッシュのサイズが同期中のテーブルのページの適切な "ワーキングセット" を含むのに十分な大きさかどうかによって左右されます。キャッシュが小さすぎると、行を挿入するために記憶領域から繰り返しインデックスページを読み込む必要が生じる場合があり、必要なインデックスページがキャッシュに収まる場合に比べて明らかにパフォーマンスが低下します。

デフォルトでは、size の値はバイト単位となります。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 k、m を使用します。

許容される最大キャッシュサイズを超える値を指定すると、自動的にプラットフォームのキャッシュサイズ上限が設定されます。

パラメータ値に含まれる前後のスペースはすべて無視されます。この接続パラメータの値に、先頭の一重引用符、先頭の二重引用符、またはセミコロンを含めることはできません。

例

次の接続文字列フラグメントによって、キャッシュサイズが 20 MB に設定されます。

```
"CACHE_SIZE=20m"
```

関連情報

Ultra Light 接続文字列とパラメータ [39 ページ]
Ultra Light データベースへの接続 [607 ページ]
Embedded SQL を使用した Ultra Light データベース接続 [641 ページ]
Ultra Light データベースの接続の設定 [560 ページ]
Ultra Light CACHE_MAX_SIZE 接続パラメータ [179 ページ]
Ultra Light CACHE_MIN_SIZE 接続パラメータ [180 ページ]
Ultra Light cache_allocation オプション [202 ページ]
Ultra Light page_size 作成オプション [160 ページ]
Ultra Light RESERVE_SIZE 接続パラメータ [193 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.12.2.5 Ultra Light COMMIT_FLUSH 接続パラメータ

コミットされたトランザクションが、コミット呼び出しの後にいつ記憶領域にフラッシュされるかを規定します。Ultra Light アプリケーションからコミットが呼び出されないと、フラッシュされません。

構文

COMMIT_FLUSH={ *immediate* | *grouped* | *on_checkpoint* }

デフォルト

immediate

備考

この接続パラメータは、ハードウェア障害またはクラッシュの後に、どのトランザクションをリカバリするかを定義します。論理オートコミット操作を単一のリカバリポイントとしてグループ化できます。

これらの操作をグループ化することで、Ultra Light のパフォーマンスを向上させることができますが、データのリカバリ性が低下します。トランザクションがコミットされた後であっても、トランザクションが記憶領域にフラッシュされる前にハードウェア障害またはクラッシュが発生すると、トランザクションが失われる可能性がわずかながらあります。

次のパラメータがサポートされます。

immediate

コミットされたトランザクションは、コミット操作が完了する前でも、コミット呼び出しが実行されるとすぐに記憶領域にフラッシュされます。

grouped

コミットされたトランザクションは、設定したスレッショルドに達している場合にかぎり、コミット呼び出しがあった時点で記憶領域にフラッシュされます。トランザクションカウントスレッショルドを commit_flush_count データベースオプションを使用して設定するか、時間ベースのスレッショルドを commit_flush_timeout データベースオプションを使用して設定できます。

設定された場合、commit_flush_count オプションと commit_flush_timeout オプションはどちらもコミットフラッシュのトリガとして動作し、先にスレッショルドに達した方がフラッシュをトリガします。フラッシュが実行されると、Ultra Light によってカウンタとタイマーはどちらも 0 に戻されます。その後、どちらか一方がスレッショルドに達するまで、両方ともモニタリングされます。

on_checkpoint

コミットされたトランザクションは、チェックポイント操作の時点で記憶領域にフラッシュされます。次のいずれかを使用してチェックポイントを実行できます。

- CHECKPOINT 文。チェックポイントメソッドのない API では、この SQL 文を使用する必要があります。
- Ultra Light Embedded SQL 用の ULCheckpoint 関数。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

- C++ コンポーネントの接続オブジェクトの Checkpoint メソッド。

関連情報

単一のトランザクションまたはグループ化されたトランザクションのフラッシュ [544 ページ]
Ultra Light commit_flush_count オプション [テンポラリ] [203 ページ]
Ultra Light commit_flush_timeout オプション [テンポラリ] [204 ページ]
CHECKPOINT 文 [Ultra Light] [488 ページ]

1.12.2.6 Ultra Light CON 接続パラメータ

接続に名前を付け、マルチ接続アプリケーションで簡単に切り替えができるようにします。

構文

```
CON=name
```

デフォルト

接続名なし

備考

CON 接続パラメータは、アプリケーション全体に対して設定されます。

複数の同時接続を確立し、その接続間で切り替えを行わない場合は、この接続パラメータは使用しないでください。

接続名はデータベース名とは異なります。

パラメータ値に含まれる前後のスペースはすべて無視されます。この接続パラメータの値に、先頭の一重引用符、先頭の二重引用符、またはセミコロンを含めることはできません。

例

次の接続文字列フラグメントによって、最初の接続名が MyFirstCon に設定されます。

```
"CON=MyFirstCon"
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

関連情報

Ultra Light 接続文字列とパラメータ [39 ページ]
Ultra Light データベースへの接続 [607 ページ]
Embedded SQL を使用した Ultra Light データベース接続 [641 ページ]
Ultra Light データベースの接続の設定 [560 ページ]
Ultra Light DBN 接続パラメータ [188 ページ]

1.12.2.7 Ultra Light DBF 接続パラメータ

Ultra Light データベースのパスとファイル名を指定します。

この接続パラメータは、新しいデータベースファイルに対してパスとファイル名を指定するとき、または既存のデータベースファイルに接続するときに使用します。

構文

```
DBF=ul-db
```

動作

1. 接続時に、データベースがすでに実行されているか確認します。DBN が指定されている場合、一致するデータベースを探して、見つかったら接続し、見つからない場合は自動起動に進みます。
2. DBF が指定されている場合、一致するデータベース (同一ファイル名) を探して、見つかったら接続し、見つからない場合は自動起動に進みます。
3. DBN も DBF も指定されておらず、1 つのデータベースが実行されている場合、そのデータベースに接続します。
4. DBF が指定されている場合、必要であれば、データベースは自動起動されます。DBN も指定されている場合、DBN が実行するデータベースの名前になり、それ以外の場合、基本のファイル名から名前が生成されます。

備考

単一の接続文字列から、異なるデバイスにある複数のデータベースに接続する場合は、次のパラメータを使用して、プラットフォーム固有の代替パラメータを指定できます。

- desktop:DBF
- device:DBF

プラットフォーム固有の接続パラメータを指定した場合は、そのパラメータが DBF より優先されます。

DBF の値は、プラットフォームのファイル名の要件を満たしている必要があります。

Windows Mobile

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

Windows Mobile デバイスに配備する場合は、Windows デスクトップ上で実行される Ultra Light 管理ツールを使用して、接続している Windows Mobile デバイス上の Ultra Light データベースに接続できます。Windows Mobile デバイス上のファイルを特定するには、必要な絶対パスを指定し、*wce:* ファイルプレフィクスを使用する必要があります。

Windows Mobile デバイス上で実行しているアプリケーションで、*wce:* ファイルプレフィックスを使用することはできません。

パラメータ値に含まれる前後のスペースはすべて無視されます。値に、先頭の一重引用符、先頭の二重引用符、またはセミコロンを含めることはできません。

例

デスクトップのディレクトリ `c:¥Databases` にインストールされているデータベース `MyULdb.udb` に接続するには、次の接続文字列を使用します。

```
"DBF=c:¥Databases¥MyULdb.udb"
```

このファイルパスは Windows デスクトップ固有であり、モバイルデバイスには適していない点に注意してください。

下の例は、プラットフォーム固有のファイルパスを使用して、アプリケーションから `MyULdb.udb` データベースに接続する方法を示します。

```
"desktop:DBF=c:¥databases¥MyULdb.udb;device:DBF=¥Documents¥MyULdb.udb"
```

アプリケーションがデスクトッププラットフォーム上で実行されている場合、`desktop:DBF` 接続パラメータが使用されます。アプリケーションがモバイルデバイス上で実行されている場合、`device:DBF` 接続パラメータが使用されます。

接続している Windows Mobile デバイスの `Documents` フォルダに配備されている `MyULdb.udb` データベースに、デスクトップから管理ツールを使用して接続するには、次の接続文字列を使用します。

```
"DBF=wce:¥Documents¥MyULdb.udb"
```

デスクトップからモバイルデバイス上のデータベースへの接続は、Windows/Windows Mobile に対してのみサポートされています。

関連情報

Ultra Light 接続文字列とパラメータ [39 ページ]
接続パラメータでの Ultra Light ファイルパスの指定 [41 ページ]
Ultra Light 管理ツールの接続パラメータの優先度 [40 ページ]
Ultra Light データベースへの接続 [607 ページ]
Embedded SQL を使用した Ultra Light データベース接続 [641 ページ]
Ultra Light データベースの接続の設定 [560 ページ]
Ultra Light DBN 接続パラメータ [188 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.12.2.8 Ultra Light DBKEY 接続パラメータ

新しい Ultra Light データベースの作成時に、この接続パラメータでデータベースの暗号化キーを設定します。

既存のデータベースへの接続を確立する場合は、この接続パラメータで、データベースの暗号化キーを指定します。

構文

```
DBKEY=string
```

デフォルト

キーは指定されません。

備考

データベースの正しい暗号化キーを指定しないと、接続が失敗します。

暗号化キーを使用してデータベースを作成すると、データベースファイルは 256 ビット AES または FIPS 認定の 256 ビット AES アルゴリズムを使用して強力に暗号化されます。強力な暗号化を使用すると、巧妙で容赦ないデータへのアクセス試行に対抗するセキュリティが提供されます。

パラメータ値に含まれる前後のスペースはすべて無視されます。値に、先頭の一重引用符、先頭の二重引用符、またはセミコロンを含めることはできません。

関連情報

Ultra Light 接続文字列とパラメータ [39 ページ]
データベースセキュリティ [35 ページ]
Ultra Light データベースへの接続 [607 ページ]
Embedded SQL を使用した Ultra Light データベース接続 [641 ページ]
Ultra Light データベースの接続の設定 [560 ページ]
Ultra Light obfuscate 作成オプション [159 ページ]

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

1.12.2.9 Ultra Light DBN 接続パラメータ

アプリケーションが複数のデータベースに接続した場合に、データベースを名前で識別します。

構文

```
DBN=db-name
```

デフォルト

なし。

動作

1. 接続時に、データベースがすでに実行されているか確認します。DBN が指定されている場合、一致するデータベースを探して、見つかったら接続し、見つからない場合は自動起動に進みます。
2. DBF が指定されている場合、一致するデータベース (同一ファイル名) を探して、見つかったら接続し、見つからない場合は自動起動に進みます。
3. DBN も DBF も指定されておらず、1 つのデータベースが実行されている場合、そのデータベースに接続します。
4. DBF が指定されている場合、必要であれば、データベースは自動起動されます。DBN も指定されている場合、DBN が実行するデータベースの名前になり、それ以外の場合、基本のファイル名から名前が生成されます。

備考

Ultra Light では、データベースを開いてからデータベース名が設定されます。データベースが開いたら、クライアントアプリケーションでファイルではなく名前を使用してデータベースに接続できます。

パラメータ値に含まれる前後のスペースはすべて無視されます。この接続パラメータの値に、先頭の一重引用符、先頭の二重引用符、またはセミコロンを含めることはできません。

例

次のパラメータを使用して Kitchener という実行中の Ultra Light データベースに接続することもできます。

```
DBN=Kitchener;DBF=cities.udb
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

関連情報

Ultra Light 接続文字列とパラメータ [39 ページ]
Ultra Light データベースへの接続 [607 ページ]
Embedded SQL を使用した Ultra Light データベース接続 [641 ページ]
Ultra Light データベースの接続の設定 [560 ページ]
Ultra Light DBF 接続パラメータ [185 ページ]

1.12.2.10 Ultra Light desktop 接続パラメータプレフィックス

接続パラメータとともにプレフィックス desktop: を使用することで、アプリケーションがデスクトップで実行中であるときのみパラメータが適用されることを示すことができます。

接続パラメータがデスクトップとモバイルデバイスの両方で利用可能な場合は、このプレフィックスを使用しないでください。

構文

```
desktop:connection-parameter=value
```

備考

さまざまなデバイス上で実行される Ultra Light クライアントアプリケーションに対しては、`desktop` 接続パラメータプレフィックスを使用します。

`desktop` または `device` プレフィクスのある接続パラメータは、プレフィクスのないパラメータよりも優先されます。

例

次の例は、デスクトップおよびモバイルデバイス用のさまざまなデータベースファイル、デスクトップ上のテンポラリディレクトリの場所、およびモバイルデバイスに対する cache_size を示します。

```
"desktop:DBF=C:¥dir¥db.udb;device:DBF=¥SD Card¥db.udb;desktop:temp_dir=
¥Temp;device:cache_size=4M"
```

関連情報

Ultra Light 接続文字列とパラメータ [39 ページ]
接続パラメータでの Ultra Light ファイルパスの指定 [41 ページ]
Ultra Light 管理ツールの接続パラメータの優先度 [40 ページ]
Ultra Light データベースへの接続 [607 ページ]

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

Embedded SQL を使用した Ultra Light データベース接続 [641 ページ]
Ultra Light データベースの接続の設定 [560 ページ]
Ultra Light DBF 接続パラメータ [185 ページ]

1.12.2.11 Ultra Light device 接続パラメータプレフィックス

接続パラメータとともにプレフィックス device: を使用することで、アプリケーションがモバイルデバイスで実行中であるときのみパラメータが適用されることを示すことができます。

接続パラメータがデスクトップとモバイルデバイスの両方で利用可能な場合は、このプレフィックスを使用しないでください。

構文

```
device:connection-parameter=value
```

備考

さまざまなデバイス上で実行される Ultra Light クライアントアプリケーションに対しては、`device` 接続パラメータプレフィックスを使用します。

`desktop` または `device` プレフィックスのある接続パラメータは、プレフィックスのないパラメータよりも優先されます。

例

次の例は、デスクトップおよびモバイルデバイス用のさまざまなデータベースファイル、モバイルデバイス上のテンポラリディレクトリの場所、およびモバイルデバイスに対する cache_size を示します。

```
"desktop:DBF=C:¥dir¥db.udb;device:DBF=¥SD Card¥db.udb;device:temp_dir=
¥Temp;device:cache_size=4M"
```

関連情報

Ultra Light 接続文字列とパラメータ [39 ページ]
接続パラメータでの Ultra Light ファイルパスの指定 [41 ページ]
Ultra Light 管理ツールの接続パラメータの優先度 [40 ページ]
Ultra Light データベースへの接続 [607 ページ]
Embedded SQL を使用した Ultra Light データベース接続 [641 ページ]
Ultra Light データベースの接続の設定 [560 ページ]
Ultra Light DBF 接続パラメータ [185 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.12.2.12 Ultra Light MIRROR_FILE 接続パラメータ

(メインデータベースファイルへの書き込みが発行されると同時に) データベースへのすべての書き込みが発行される、データベースミラーファイルの名前を指定します。

構文

```
MIRROR_FILE=path¥mirrorfile-db
```

デフォルト

なし。

備考

Ultra Light には、潜在的に信頼性が低いストレージシステムのフォールトトレランスを改善する基本的なデータベースファイルミラーリング機能があります。ミラーリングには、ミラーファイルが使用されます。データベースへのすべての書き込みは、メインデータベースファイルに対して発行されると同時にミラーファイルにも発行されます。したがって書き込みのオーバヘッドは 2 倍になり、読み込みのオーバヘッドには影響ありません。データベースファイルから破損したページが読み込まれた場合、ミラーファイルから読み込むことでページをリカバリできます。

ミラーリングは、ファイルベースのストアを使用するすべてのプラットフォームでサポートされています。

データベースの起動時に mirror_file オプションが指定された場合、Ultra Light は指定されたファイルを開き、メインデータベースファイルと一致することを確認してから、処理を続行します。ミラーファイルがない場合は、この時点でメインファイルをコピーして作成されます。ミラーがデータベースファイルではないか破損していると、エラーが報告されます。この場合、ファイルが削除されるか、別のミラーが指定されるまで、データベースは起動しません。ミラーがデータベースと一致しない場合は、SQLE_MIRROR_FILE_MISMATCH が生成され、データベースは起動しません。破損したページがリカバリされると、警告 SQLE_CORRUPT_PAGE_READ_RETRY が生成されます。ミラーリングを使用していないか、ミラーファイルも破損している場合は、エラー SQLE_DEVICE_ERROR が生成され、データベースが停止します。

メディア障害から効果的に保護するために、ミラーファイルを使用しているときはページチェックサムを有効にしてください。ミラーリングを行っているかどうかに関係なく、ページチェックサムによって、ページのロード後すぐに Ultra Light でページの破損が検出可能になり、破損したデータの参照を回避できます。チェックサムを有効にするには checksum_level データベース作成オプションを指定します。ミラーファイルを使用しているときにチェックサムが有効になっていなかった場合、警告 SQLE_MIRROR_FILE_REQUIRES_CHECKSUMS が生成されます。

ミラーは、データベースファイルとまったく同じコピーなので、データベースとして直接起動できます。破損したページは ulvalid ユーティリティによって報告されます。

例

次の例では、新しい接続を作成し、ミラーファイルを作成します。

```
Connection = DatabaseMgr.OpenConnection("DBF=c:¥Dbfile.udb; UID=JDoe;PWD=ULdb;
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

```
MIRROR_FILE=c:¥test¥MyMirrorDB.udb")
```

関連情報

Ultra Light 接続文字列とパラメータ [39 ページ]
接続パラメータでの Ultra Light ファイルパスの指定 [41 ページ]
Ultra Light 管理ツールの接続パラメータの優先度 [40 ページ]
Ultra Light データベースへの接続 [607 ページ]
Embedded SQL を使用した Ultra Light データベース接続 [641 ページ]
Ultra Light データベースの接続の設定 [560 ページ]
Ultra Light checksum_level 作成オプション [143 ページ]
Ultra Light データベース検証ユーティリティ (ulvalid) [239 ページ]

1.12.2.13 Ultra Light PWD 接続パラメータ

新しい Ultra Light データベースの作成時に、この接続パラメータでデフォルトユーザのパスワードを設定します。

既存のデータベースに接続する場合は、認証に使用するユーザ ID のパスワードを識別します。

Android のデバイスの場合、代わりに Configuration.setPassword を使用して、この接続パラメータを設定できます。

構文

```
PWD=password
```

デフォルト

UID と PWD の両方を設定しなかった場合、Ultra Light では *UID=DBA* と *PWD=sql* で接続が確立されます。

備考

すべてのデータベースユーザにはパスワードがあります。Ultra Light では、最大で 4 つのユーザ ID とパスワードの組み合わせがサポートされています。

パスワードは NULL または空の文字列に設定できます。

新しいユーザが作成されるか、または既存のユーザが各自のパスワードを変更すると、ランダムな 4 バイトのソルト値が生成されます。ソルト値は、パスワードハッシュを計算するときにユーザのパスワードに追加され、ハッシュとともにデータベースに

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

格納されます。ソルト値を適用することによって、辞書攻撃に対する脆弱性が低下するだけでなく、同じパスワードを使用するユーザのパスワードハッシュが確実に異なるようになります。

この接続パラメータは暗号化されません。ただし、Ultra Light では、パスワードがハッシュされてから保存されるため、パスワードは SQL Central を使用してだけ変更できます。

例

次の部分的な接続文字列は、ユーザ ID DBA とパスワード sql を指定します。

```
"UID=DBA;PWD=sql"
```

次の部分的な接続文字列は、ユーザ ID DBA と空のパスワードを指定します。

```
"UID=DBA;PWD=''"
```

関連情報

Ultra Light 接続文字列とパラメータ [39 ページ]
Ultra Light ユーザ [64 ページ]
ユーザ認証 [656 ページ]
Ultra Light データベースへの接続 [607 ページ]
ユーザ認証 [656 ページ]
Embedded SQL を使用した Ultra Light データベース接続 [641 ページ]
Ultra Light データベースの接続の設定 [560 ページ]
Ultra Light UID 接続パラメータ [196 ページ]

1.12.2.14 Ultra Light RESERVE_SIZE 接続パラメータ

実際にデータを挿入することなく、Ultra Light データベースが必要とするファイルシステム領域を事前に割り付けます。ファイルシステム領域を予約すると、領域が他のファイルによって使用されることはありません。

構文

```
RESERVE_SIZE= number{ k | m | g }
```

デフォルト

0 (予約サイズなし)。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

備考

0 からデータベースの最大サイズまでの任意の値を指定できます。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 k、m、g のいずれかを使用してください。単位を指定しない場合、デフォルトはバイトです。

データベースをテストデータを使用して実行することによってデータベースサイズを確認し、Ultra Light の配備に適した予約サイズを選択してください。

RESERVE_SIZE の値がデータベースのサイズよりも小さい場合、このパラメータは無視されます。

ファイルシステム領域を予約すると、次の理由によりパフォーマンスが多少向上します。

- 段階的なサイズの増加と比較して、断片化レベルが低減されます。
- メモリ不足障害を防ぐ。

Ultra Light データベースはデータとメタデータで構成されているので、データベースサイズが大きくなるのは必要がある場合 (アプリケーションがデータベースを更新する場合) のみです。

例

次の接続文字列フラグメントによって、予約サイズが 128 KB に設定され、起動時にシステムによってデータベース用にこのシステム領域が予約されるようになります。

```
"RESERVE_SIZE=128K"
```

関連情報

Ultra Light 接続文字列とパラメータ [39 ページ]
Ultra Light データベースへの接続 [607 ページ]
Embedded SQL を使用した Ultra Light データベース接続 [641 ページ]
Ultra Light データベースの接続の設定 [560 ページ]
Ultra Light CACHE_SIZE 接続パラメータ [181 ページ]
Ultra Light page_size 作成オプション [160 ページ]

1.12.2.15 Ultra Light START 接続パラメータ

Ultra Light エンジンの実行プログラムを起動します。このパラメータは、Android 用 Ultra Light ではサポートされていません。このパラメータは、エンジンが予測されるロケーションにない場合にのみ必要です。

構文

```
START=path¥uleng17
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

備考

現在実行中でないエンジンに接続する場合は、StartLine (START) 接続パラメータだけを指定します。

スペースが含まれているパスには、引用符が必要です。パスにスペースが含まれていない場合、クライアントは SQLE_UNABLE_TO_CONNECT_OR_START を返します。

例

次のコマンドは、`Program Files` ディレクトリの Ultra Light エンジンを起動します。

```
Start="¥Program Files¥uleng17.exe"
```

関連情報

Ultra Light エンジンの起動 [127 ページ]
Windows Mobile 用の Ultra Light データ管理コンポーネント [22 ページ]
Ultra Light データベースへの接続 [607 ページ]
Embedded SQL を使用した Ultra Light データベース接続 [641 ページ]
Ultra Light データベースの接続の設定 [560 ページ]
Ultra Light エンジンユーティリティ (uleng17) [217 ページ]

1.12.2.16 Ultra Light TEMP_DIR 接続パラメータ

Ultra Light でテンポラリファイル (データベース名から導出される名前を持つ) を配置するフォルダ (すでに存在している必要がある) の名前を指定します。

構文

```
TEMP_DIR=path
```

備考

データベースファイルに加えて、Ultra Light では、データベース操作時にテンポラリファイルが作成され、管理されます。このファイルの操作や管理は必要ありません。

デフォルトでは、テンポラリファイルは Ultra Light データベース自体と同じフォルダ (存在する場合) で管理されます。テンポラリファイルのファイル名はデータベースと同じになりますが、ファイルベースのプラットフォームの場合、ファイルの拡張子にチルダが含まれます。たとえば、サンプルデータベース `CustDB.udb` を実行すると、このデータベースファイルと同じディレクトリに `CustDB.~db` というテンポラリファイルが作成されます。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

高速 I/O 特性を持つテンポラリフォルダを指定すると、キャッシュサイズよりも大きなテンポラリテーブルなどのパフォーマンスが向上することがあります。また、実行時間が長いトランザクションは、テンポラリファイル内のかなりの領域を消費する可能性があります。

スペースが含まれているパスには、引用符が必要です。パスにスペースが含まれていない場合、クライアントは SQLE_UNABLE_TO_CONNECT_OR_START を返します。

例

次の接続文字列フラグメントは、テンポラリファイルを ¥Temp ディレクトリに配置します。

```
temp_dir=¥Temp;
```

関連情報

Ultra Light データベースへの接続 [607 ページ]
Embedded SQL を使用した Ultra Light データベース接続 [641 ページ]
Ultra Light データベースの接続の設定 [560 ページ]

1.12.2.17 Ultra Light UID 接続パラメータ

新しい Ultra Light データベースの作成時に、この接続パラメータでデータベースのデフォルトのユーザ ID を設定します。

既存のデータベースに接続する場合は、そのデータベースに接続に使用するユーザ ID を指定します。値には、データベースで認証されたユーザを指定します。

構文

```
UID=user
```

デフォルト

接続時に UID と PWD を設定しなかった場合、Ultra Light では *UID=DBA* と *PWD=sql* で接続が確立されます。

備考

すべてのデータベースユーザはユーザ ID を持ちます。Ultra Light では、最大で 4 つのユーザ ID とパスワードの組み合わせがサポートされています。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

Ultra Light のユーザ ID は、Mobile Link のユーザ名やその他の SQL Anywhere のユーザ ID とは別です。ユーザ ID は一度作成すると変更できません。変更する必要がある場合は、ユーザ ID を削除し、新しい ID を追加してください。

UID は NULL または空の文字列に設定できません。ユーザ ID の最大長は 31 文字です。ユーザ ID の大文字と小文字は区別されません。

パラメータ値に含まれる前後のスペースはすべて無視されます。この接続パラメータの値に、先頭の一重引用符、先頭の二重引用符、またはセミコロンを含めることはできません。

例

次の接続文字列フラグメントは、データベースのユーザ ID DBA とパスワード sql を指定します。

```
"UID=DBA;PWD=sql"
```

関連情報

Ultra Light 接続文字列とパラメータ [39 ページ]
Ultra Light ユーザ [64 ページ]
ユーザ認証 [656 ページ]
Ultra Light データベースへの接続 [607 ページ]
ユーザ認証 [656 ページ]
Embedded SQL を使用した Ultra Light データベース接続 [641 ページ]
Ultra Light データベースの接続の設定 [560 ページ]

1.12.3 Ultra Light データベースプロパティ

Ultra Light データベースのプロパティの値は、データベースの最初の作成時に定義されます。

プロパティは、Ultra Light データベースを再作成するか、または対応するデータベースオプションを使用できる場合にはオプションを編集することによって、変更できます。

Ultra Light では、次のデータベースプロパティがサポートされます。

プロパティ	説明
cache_allocation	現在のキャッシュサイズを最小および最大設定のパーセンテージとして返します。このプロパティはデータベースの cache_allocation オプションに相当します。
CaseSensitive	大文字と小文字の区別のステータスを返します。データベースで大文字と小文字が区別される場合は、On を返します。それ以外の場合は、Off を返します。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

プロパティ	説明
CharSet	データベースの CHAR 文字セットを返します。データベースで使用される文字セットは、データベースの照合順と、データが UTF-8 でエンコードされているかどうかで決まります。
ChecksumLevel	データベースのチェックサム検証のレベルを返します。値は、0 (チェックサムを追加しない)、1 (重要なページにチェックサムを追加する)、または 2 (すべてのページにチェックサムを追加する) のいずれかです。このプロパティはデータベースの checksum_level 作成オプションに相当します。
Collation	データベースの照合順の名前を返します。このプロパティはデータベースの collation 作成オプションに相当します。
commit_flush_count	コミットカウントスレッショルドを設定する commit_flush_count オプションの値を返します。このプロパティはデータベースの commit_flush_count オプション [テンポラリ] に相当します。
commit_flush_timeout	時間間隔スレッショルドを設定する commit_flush_timeout オプションの値を返します。このプロパティはデータベースの commit_flush_timeout オプション [テンポラリ] に相当します。
ConnCount	データベースとの接続の数を返します。返される値は動的です。現在存在している接続の数によって変わります。Ultra Light では、最大 14 の同時データベース接続をサポートできます。
date_format	データベースで文字列変換に使用される日付形式を返します。このプロパティはデータベースの date_format 作成オプションに相当します。
date_order	データベースで文字列変換に使用される日付順を返します。このプロパティはデータベースの date_order 作成オプションに相当します。
Encryption	データベースエンコーディングのタイプを返します。値は、None、Simple、AES、AES_FIPS のいずれかです。 データベースで使用されるエンコーディングは、強力な暗号化 (AES か AES_FIPS) と DBKEY 作成パラメータ、または単純難読化のどちらが設定されているかによって判断されます。 このプロパティを変更できるのは、最初の値が None (fips も難読化も使用されていない) の場合だけです。その場合は、使用している API の適切な関数またはメソッドを呼び出して、Connection オブジェクトに新しい暗号化キーを指定することで、暗号化キーを変更できます。この場合、値は AES に変更されます。これは、データベースの作成後に fips 作成パラメータを設定することができないためです。 また、API メソッドを使用して暗号化キーを変更することもできます。
File	現在の接続のデータベースルートファイル名をパスを含めて返します。この値は、DBF 接続パラメータ値で指定されている値です。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

プロパティ	説明
global_database_id	グローバルオートインクリメントカラムに使用される global_database_id オプションの値を返します。このプロパティはデータベースの global_database_id オプションに相当します。
isolation_level	データベースの現在の独立性レベルを返します。この値は *read_committed* または *read_uncommitted* のいずれかになります。このプロパティはデータベースの isolation_level オプションに相当します。
MaxHashSize	インデックスのハッシュに使用する、デフォルトの最大バイト数を返します。このプロパティは、インデックス単位で指定できます。このプロパティはデータベースの max_hash_size 作成オプションに相当します。
ml_remote_id	Mobile Link 同期のデータベースをユニークに識別する ml_remote_id オプションの値を返します。このプロパティはデータベースの ml_remote_id オプションに相当します。
Name	現在の接続におけるデータベースの名前 (またはエイリアス) を返します。返される名前は、DBN 接続パラメータ値と一致します。DBN 接続パラメータを使用しなかった場合、返される名前はパスと拡張子のないデータベースファイルになります。
nearest_century	データベースで文字列変換に使用される基準年を返します。このプロパティはこのデータベースの nearest_century 作成オプションに相当します。
PageSize	データベースのページサイズ (バイト単位) を返します。このプロパティはデータベースの page_size 作成オプションに相当します。
PartialDownload	データベースが部分的なダウンロードを含むかどうかを示す値として、Yes または No を返します。
precision	データベースで文字列変換に使用される浮動小数点の精度を返します。このプロパティはデータベースの precision 作成オプションに相当します。
scale	データベースによる文字列変換中に、計算結果が最大 *precision* にトランケートされるときの小数点以下の最大桁数を返します。このプロパティはデータベースの scale 作成オプションに相当します。
time_format	データベースで文字列変換に使用される時間形式を返します。このプロパティはデータベースの time_format 作成オプションに相当します。
timestamp_format	データベースで文字列変換に使用されるタイムスタンプ形式を返します。このプロパティはデータベースの timestamp_format 作成オプションに相当します。
timestamp_increment	2 つのユニークなタイムスタンプ間のマイクロ秒単位の最小差を返します。このプロパティはデータベースの timestamp_increment 作成オプションに相当します。
timestamp_with_time_zone_format	TIMESTAMP WITH TIME ZONE 値のタイムスタンプ形式を返します。このプロパティはデータベースの timestamp_with_time_zone_format 作成オプションに相当します。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

プロパティ	説明
UploadUnknown	前回の同期の失敗が、アップロードの送信後かつ同期サーバからのアップロード確認受領前に発生した場合、true を返します。このプロパティが true である場合、次の同期で Ultra Light はまず Mobile Link に対し、新しい同期へと進む前に、前回のアップロードを受信したかどうかを尋ねます。

関連情報

Ultra Light データベースオプション [201 ページ]
独立性レベル [47 ページ]
データベースセキュリティ [35 ページ]
データベースプロパティの読み取り [43 ページ]
データベースオプションへのアクセス [44 ページ]
Ultra Light オプション [138 ページ]
Ultra Light cache_allocation オプション [202 ページ]
Ultra Light の case 作成オプション [142 ページ]
Ultra Light utf8_encoding 作成オプション [174 ページ]
Ultra Light の collation 作成オプション [144 ページ]
Ultra Light checksum_level 作成オプション [143 ページ]
Ultra Light の collation 作成オプション [144 ページ]
Ultra Light commit_flush_count オプション [テンポラリ] [203 ページ]
Ultra Light commit_flush_timeout オプション [テンポラリ] [204 ページ]
Ultra Light の date_format 作成オプション [145 ページ]
Ultra Light の date_order 作成オプション [147 ページ]
Ultra Light DBF 接続パラメータ [185 ページ]
Ultra Light global_database_id オプション [205 ページ]
Ultra Light max_hash_size 作成オプション [155 ページ]
Ultra Light ml_remote_id オプション [206 ページ]
Ultra Light DBN 接続パラメータ [188 ページ]
Ultra Light DBF 接続パラメータ [185 ページ]
Ultra Light nearest_century 作成オプション [157 ページ]
Ultra Light page_size 作成オプション [160 ページ]
Ultra Light precision 作成オプション [162 ページ]
Ultra Light scale 作成オプション [164 ページ]
Ultra Light time_format 作成オプション [165 ページ]
Ultra Light timestamp_format 作成オプション [167 ページ]
Ultra Light timestamp_increment 作成オプション [170 ページ]
Ultra Light timestamp_with_time_zone_format 作成オプション [171 ページ]
Ultra Light 作成オプション [153 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

Ultra Light obfuscate 作成オプション [159 ページ]
Ultra Light DBKEY 接続パラメータ [187 ページ]

1.12.4 Ultra Light データベースオプション

Ultra Light データベースのオプション値は、データベースの最初の作成時に定義されますが、データベースへの接続中に変更できます。

このセクションの内容:

Ultra Light cache_allocation オプション [202 ページ]
明示的にキャッシュのサイズを変更します。この値は、最小から最大までの範囲のパーセンテージです。値 0 は最小サイズを表し、値 100 は最大サイズを表します。

Ultra Light commit_flush_count オプション [テンポラリ] [203 ページ]
コミットカウントスレッショルドを設定します。このスレッショルドに達すると、コミットフラッシュが実行されます。

Ultra Light commit_flush_timeout オプション [テンポラリ] [204 ページ]
時間間隔スレッショルドを設定します。このスレッショルドに達すると、グループ化されたコミットフラッシュが実行されます。

Ultra Light global_database_id オプション [205 ページ]
データベースの ID 番号を設定します。

Ultra Light isolation_level オプション [206 ページ]
独立性レベルは、あるトランザクションの操作が、同時に処理されている別のトランザクションの操作からどの程度参照できるかを定義します。Ultra Light では、オートコミットモードでの接続には、デフォルトの独立性レベルである read_committed (コミットされた呼び出し) が使用されます。

Ultra Light ml_remote_id オプション [206 ページ]
リモート ID は、Ultra Light データベースのユニークな識別子で、Mobile Link で同期用のデータベースを識別するために使用されます。

関連情報

データベースオプションへのアクセス [44 ページ]
データベースプロパティの読み取り [43 ページ]
Ultra Light オプション [138 ページ]

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

1.12.4.1 Ultra Light cache_allocation オプション

明示的にキャッシュのサイズを変更します。この値は、最小から最大までの範囲のパーセンテージです。値 0 は最小サイズを表し、値 100 は最大サイズを表します。

指定可能な値

整数

デフォルト

なし。

備考

cache_allocation プロパティは、現在のキャッシュサイズを最小キャッシュサイズと最大キャッシュサイズのパーセンテージ値として返します。

関連情報

データベースオプションへのアクセス [44 ページ]
Ultra Light データベースプロパティ [197 ページ]
SET OPTION 文 [Ultra Light] [519 ページ]
Ultra Light CACHE_SIZE 接続パラメータ [181 ページ]
Ultra Light CACHE_MIN_SIZE 接続パラメータ [180 ページ]
Ultra Light CACHE_MAX_SIZE 接続パラメータ [179 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.12.4.2 Ultra Light commit_flush_count オプション [テンポラリ]

コミットカウントスレッショルドを設定します。このスレッショルドに達すると、コミットフラッシュが実行されます。

指定可能な値

整数

デフォルト

10

備考

トランザクションカウントを無効にする場合は、0 を使用します。トランザクションカウントを無効にすると、フラッシュがトリガされた際のコミット数に制限がなくなります。

commit_flush_count と commit_flush_timeout は、どちらも一時的なデータベースオプションです。これらのオプションは、データベースを起動するたびに設定する必要があります。オプションは、データベースが実行され続けている間は保持されます。これらは、COMMIT_FLUSH=grouped が接続文字列の一部として設定される場合にのみ必要です。

このオプションを設定し、接続文字列にグループ化されるように COMMIT_FLUSH 接続パラメータを設定すると、どちらかのスレッショルドに達するとフラッシュがトリガされます。フラッシュが実行されると、Ultra Light によってカウンタとタイマーはどちらも 0 に戻されます。その後、どちらか一方がスレッショルドに達するまで、両方ともモニタリングされます。

コミットフラッシュオプションを設定するうえで重要な考慮事項は、コミットされたトランザクションがフラッシュされるまでの遅延によって、データのリカバリ性はどれ程のリスクを受けるかということです。トランザクションがコミットされた後であっても、トランザクションが失われる可能性がわずかながら存在します。コミット後に深刻なハードウェア障害が発生した場合、それがトランザクションが記憶領域にフラッシュされる前ならば、トランザクションはリカバリ時にロールバックされます。遅延を長くするほど、Ultra Light のパフォーマンスは向上します。適切なカウントのスレッショルドの選択は慎重に行ってください。

クライアントアプリケーションから commit_flush_count オプションを設定するには、オプションを設定するときに、使用しているプログラミングインタフェースのデータベースオプション設定機能を使用するか、SET OPTION SQL 文を使用します。

関連情報

単一のトランザクションまたはグループ化されたトランザクションのフラッシュ [544 ページ]
データベースオプションへのアクセス [44 ページ]
SET OPTION 文 [Ultra Light] [519 ページ]
Ultra Light commit_flush_timeout オプション [テンポラリ] [204 ページ]

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

Ultra Light COMMIT_FLUSH 接続パラメータ [183 ページ]

1.12.4.3 Ultra Light commit_flush_timeout オプション [テンポラリ]

時間間隔スレッショルドを設定します。このスレッショルドに達すると、グループ化されたコミットフラッシュが実行されます。

指定可能な値

整数 (ミリ秒単位)

デフォルト

10000 ミリ秒

備考

時間スレッショルドを無効にする場合は、0 を使用します。

commit_flush_count と commit_flush_timeout は、どちらも一時的なデータベースオプションです。これらのオプションは、データベースを起動するたびに設定する必要があります。オプションは、データベースが実行され続けている間は保持されます。これらは、COMMIT_FLUSH=grouped が接続文字列の一部として設定される場合にのみ必要です。

このオプションを commit_flush_timeout オプションとともに設定し、COMMIT_FLUSH 接続パラメータを grouped に設定した場合は、どちらかのスレッショルドに達するとフラッシュがトリガされます。フラッシュが実行されると、Ultra Light によってカウンタとタイマーはどちらも 0 に戻されます。その後、どちらか一方がスレッショルドに達するまで、両方ともモニタリングされます。

コミットフラッシュオプションを設定するうえで重要な考慮事項は、コミットされたトランザクションがフラッシュされるまでの遅延によって、データのリカバリ性はどれ程のリスクを受けるかということです。トランザクションがコミットされた後であっても、トランザクションが失われる可能性がわずかながら存在します。コミット後に深刻なハードウェア障害が発生した場合、それがトランザクションが記憶領域にフラッシュされる前ならば、トランザクションはリカバリ時にロールバックされます。遅延を長くするほど、Ultra Light のパフォーマンスは向上します。適切なタイムアウトのスレッショルドの選択は慎重に行ってください。

クライアントアプリケーションから commit_flush_timeout オプションを設定するには、オプションを設定するときに、使用しているプログラミングインタフェースのデータベースオプション設定機能を使用するか、SET OPTION SQL 文を使用します。

関連情報

データベースオプションへのアクセス [44 ページ]

 © 2016 SAP SE or an SAP affiliate company.All rights reserved.

SET OPTION 文 [Ultra Light] [519 ページ]
Ultra Light commit_flush_count オプション [テンポラリ] [203 ページ]
Ultra Light COMMIT_FLUSH 接続パラメータ [183 ページ]

1.12.4.4 Ultra Light global_database_id オプション

データベースの ID 番号を設定します。

指定可能な値

負でない一意の整数

デフォルト

特定のグローバルオートインクリメントカラムのデフォルト値の範囲は、pn + 1 ～ p(n + 1) です。ここで、p はカラムの分割サイズを、n はグローバルデータベース ID 番号を示します。

備考

グローバル ID は、Mobile Link サーバとの同期時にプライマリキーの一意性を維持するために GLOBAL AUTOINCREMENT カラムの開始値を設定します。グローバル ID を設定してからデフォルト値を割り当ててください。テーブルにローが追加され、まだ値が設定されていない場合は、Ultra Light によって、global_database_id の値と分割サイズを組み合わせてカラムの値が生成されます。

このオプションが設定されると、Ultra Light はコミット操作を実行します。

アプリケーションを配備するときには、Mobile Link サーバとの同期のために、各データベースに異なる ID 番号を割り当てる必要があります。既存のデータベースのグローバル ID はいつでも変更できます。

クライアントアプリケーションから global_database_id オプションを設定するには、使用しているプログラミングインタフェースのデータベースオプション設定機能を使用するか、SET OPTION SQL 文を使用します。

例

Ultra Light データベースのカラムを 3001 から 4000 までオートインクリメントするには、グローバル ID を 3 に設定します。

```
SET OPTION global_database_id=3
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

関連情報

Ultra Light の GLOBAL AUTOINCREMENT カラム [73 ページ]
データベースオプションへのアクセス [44 ページ]
SET OPTION 文 [Ultra Light] [519 ページ]

1.12.4.5 Ultra Light isolation_level オプション

独立性レベルは、あるトランザクションの操作が、同時に処理されている別のトランザクションの操作からどの程度参照できるかを定義します。Ultra Light では、オートコミットモードでの接続には、デフォルトの独立性レベルである read_committed (コミットされた呼び出し) が使用されます。

関連情報

独立性レベル [47 ページ]
データベースオプションへのアクセス [44 ページ]
SET OPTION 文 [Ultra Light] [519 ページ]

1.12.4.6 Ultra Light ml_remote_id オプション

リモート ID は、Ultra Light データベースのユニークな識別子で、Mobile Link で同期用のデータベースを識別するために使用されます。

リモート ID は、すべてのリモート Mobile Link クライアント間でユニークであれば、意味のある任意の文字列にできます。この ID は NULL に設定することもできます (NULL は初期値です)。同期中、リモート ID が NULL の場合、Ultra Light はそれを生成された GUID に割り当てます。

複数のデバイスへの配布用の同期を使用して、Ultra Light データベースを事前に移植する場合は、各データベースが一意の ID を持つようにリモート ID を NULL にリセットする必要があります。配布時に新しい一意のリモート ID を明示的に設定するか、NULL のままにして、Ultra Light で自動的に新しい一意の値が生成されるようにすることができます。

指定可能な値

Mobile Link による同期時にデータベースをユニークに識別する任意の値

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

デフォルト

NULL

備考

Mobile Link ではリモート ID を使用して、リモートデータベースの同期情報を保存できます。リモート ID が設定されると、Mobile Link ユーザ名は一意である必要がなくなります。リモート ID は、複数の Mobile Link ユーザが同じ Ultra Light データベースを同期する場合に特に便利です。この場合、同期スクリプトでは、ユーザ名だけでなく、リモート ID を参照してください。

このオプションが設定されると、Ultra Light はコミット操作を実行します。

クライアントアプリケーションから ml_remote_id オプションを設定するには、使用しているプログラミングインタフェースのデータベースオプション設定機能を使用するか、SET OPTION SQL 文を使用して設定します。

関連情報

データベースオプションへのアクセス [44 ページ]
SET OPTION 文 [Ultra Light] [519 ページ]
User Name 同期パラメータ [114 ページ]

1.12.5 Ultra Light ユーティリティ

Ultra Light は、コマンドプロンプトでデータベースの基本的な管理作業を行うことができる一連のユーティリティを提供します。これらのユーティリティの多くは、SQL Anywhere サーバのユーティリティと機能が似ています。ただし、オプションの使用方法が異なる場合があります。

Ultra Light でのオプションの実装方法については、必ず Ultra Light のリファレンスを参照してください。

> **注記**
>
> ユーティリティのオプションでは、特に明記されていないかぎり、大文字と小文字が区別されます。オプションはここに示すとおり正確に入力してください。

このセクションの内容:

サポートされている終了コード [208 ページ]
ulload、ulsync、ulunload の各ユーティリティは、処理が正常に終了したかどうかを示す終了コードを返します。0 は処理が成功したことを示します。他の値は処理が失敗したことを示します。

Ultra Light 用 Interactive SQL ユーティリティ (dbisql) [209 ページ]

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

SQL 文を実行し、データベースに対してスクリプトファイルを実行します。

Ultra Light SQL プリプロセッサユーティリティ (sqlpp) [213 ページ]
Embedded SQL (ESQL) を含む C/C++ のプログラムをプリプロセッサで処理し、コンパイラを実行する前に、このプログラムに必要なコードを生成できるようにします。

Ultra Light エンジンユーティリティ (uleng17) [217 ページ]
アプリケーションからの Ultra Light データベースへの同時接続を管理し、Ultra Light エンジンが -ud オプションを使用するデーモンとして稼動できるようにします。

Ultra Light エンジン停止ユーティリティ (ulstop) [218 ページ]
Ultra Light エンジンの停止

Ultra Light ユーティリティ消去 (ulerase) [219 ページ]
Ultra Light データベースを消去します。

Ultra Light 情報ユーティリティ (ulinfo) [220 ページ]
Ultra Light データベースに関する情報を表示します。

Ultra Light データベース初期化ユーティリティ (ulinit) [221 ページ]
新しい Ultra Light データベースを作成します。

Ultra Light データベースへの XML のロードユーティリティ (ulload) [227 ページ]
XML ファイルから新しいデータベースまたは既存のデータベースにデータをロードします。

Ultra Light 同期ユーティリティ (ulsync) [231 ページ]
Ultra Light データベースを Mobile Link サーバと同期させます。このツールを使用して、アプリケーション開発中に同期をテストできます。

Ultra Light データベース同期プロファイルオプション [234 ページ]
同期プロファイルオプションは、ulsync ユーティリティのコマンドラインで、他のすべてのコマンドラインオプションを定義した後に指定します。キーワードは、大文字と小文字を区別しません。

Ultra Light データベースのアンロードユーティリティ (ulunload) [236 ページ]
Ultra Light データベース全体あるいは Ultra Light データの一部またはすべてを、XML か SQL にアンロードします。

Ultra Light データベース検証ユーティリティ (ulvalid) [239 ページ]
Ultra Light データベースのフル (通常の) 検証を実行します。

1.12.5.1 サポートされている終了コード

ulload、ulsync、ulunload の各ユーティリティは、処理が正常に終了したかどうかを示す終了コードを返します。0 は処理が成功したことを示します。他の値は処理が失敗したことを示します。

終了コード	ステータス	説明
0	EXIT_OKAY	処理は成功した。
1	EXIT_FAIL	処理は失敗した。
3	EXIT_FILE_ERROR	データベースが見つかりません。
4	EXIT_OUT_OF_MEMORY	デバイスの動的メモリが足りません。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

終了コード	ステータス	説明
6	EXIT_COMMUNICATIONS_FAIL	Ultra Light エンジンとの通信中に通信エラーが発生した。
9	EXIT_UNABLE_TO_CONNECT	指定された UID または PWD が無効であるため、データベースに接続できません。
12	EXIT_BAD_ENCRYPT_KEY	暗号化キーがないか、無効です。
13	EXIT_DB_VER_NEWER	データベースのバージョンに互換性がないことが検出されました。データベースを新しいバージョンにアップグレードする必要があります。
255	EXIT_USAGE	コマンドラインオプションが無効です。

1.12.5.2 Ultra Light 用 Interactive SQL ユーティリティ (dbisql)

SQL 文を実行し、データベースに対してスクリプトファイルを実行します。

構文

```
dbisql -c "connection-string" [ options ] [ dbisql-statement | dbisql-script-file ]
```

```
dbisql -c "connection-string" -ul [ options ] [ dbisql-statement | dbisql-script-file ]
```

```
dbisql-statement: 1 つの SQL 文、またはコマンドデリミタによって区切られる一連の SQL 文。
```

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みます。 同じ名前の環境変数と設定ファイルが存在する場合は、環境変数が使用されます。 設定ファイル内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 (dbfhide) を使用して、設定ファイルの内容をエンコードします。Interactive SQL では、暗号化された設定ファイルをサポートしていません。
-c"keyword=value; ..."	接続パラメータを指定します。Interactive SQL が接続できない場合は、接続パラメータを入力するウィンドウが表示されます。ユーザ ID とパスワードの両方を指定しなかった場合は、DBA のデフォルトの UID と sql の PWD が使用されます。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

オプション	説明
*-d*delimiter	コマンドデリミタを指定します。デリミタを囲む引用符は省略可能ですが、コマンドシェル自体がデリミタを特別な方法で解釈するときは必ず指定します。 このオプションは、Interactive SQL の command_delimiter オプションの設定を上書きします。
-d1	ユーザが明示的に実行するすべての文をコマンドウィンドウ (STDOUT) にエコーします。これによって、SQL スクリプトのデバッグ、または Interactive SQL が長文の SQL スクリプトを処理しているときに有用なフィードバックが提供されます (最後の文字は数値の 1 であり、L の小文字ではありません)。このオプションは、Interactive SQL をコマンドラインプログラムとして実行している場合のみ使用できます。
*-datasource*DSN-name	接続先の ODBC データソースを指定します。
*-f*filename	[SQL 文] ウィンドウ枠で filename というファイルを (実行しないで) 開きます。 -f オプションが指定されている場合、-c オプションは無視されます。つまり、データベースへの接続は確立されません。 ファイル名は引用符で囲んでも囲まなくてもかまいませんが、ファイル名にスペースが含まれている場合は引用符で囲む必要があります。 そのファイルが存在しない場合、またはファイルではなく実際にはディレクトリである場合は、Interactive SQL がエラーメッセージを出力して終了します。 ファイル名に完全なドライブとパスの仕様が含まれていないときは、現在のディレクトリが基準として想定されます。 このオプションは、Interactive SQL をウィンドウベースのアプリケーションとして実行している場合のみサポートされます。
*-host*hostname	データベースサーバを実行するコンピュータの hostname または IP アドレスを指定します。現在のコンピュータを表す localhost という名前を使用できます。
-nogui	Interactive SQL をコンソールアプリケーションとして実行します。このとき、ウィンドウベースのユーザインタフェースは使用しません。これは、バッチ処理に便利です。 または dbisql-statement、dbisql-script-file のいずれかを指定すると、-nogui と見なされます。 このモードのとき、Interactive SQL は、処理の成功または失敗をプログラム終了コードを設定することで示します。Windows オペレーティングシステムでは、プログラム終了コードに対して環境変数 ERRORLEVEL が設定されます。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

オプション	説明
-onerror { *continue* | *exit* }	スクリプトファイルから文を読み込み中にエラーが発生した場合の処置を制御します。これは、Interactive SQL をバッチ処理で使用するときに便利です。このオプションは、Interactive SQL の on_error オプションの設定を上書きします。 `behavior` には次のいずれかの値を定義できます。 **Continue** エラーは無視され、Interactive SQL は文の実行を継続します。 **Exit** Interactive SQL は終了します。
-q	出力メッセージを表示しません。ユーティリティをクワイエットモードで実行するように設定します。これは、文またはスクリプトファイルを使って Interactive SQL を起動したときにのみ便利です。このオプションを指定した場合、エラーメッセージは表示されますが、次の情報は表示されません。 • 警告およびその他の致命的でないメッセージ • 結果セットの出力
-ul	デフォルトとして Ultra Light データベースを指定します。Interactive SQL では、接続するデータベースのタイプに応じて使用可能なオプションがカスタマイズされます。 Interactive SQL では、デフォルトで SQL Anywhere データベースに接続すると見なされます。-ul オプションを指定すると、デフォルトが Ultra Light データベースに変更されます。デフォルトに設定されているデータベースのタイプに関係なく、接続ウィンドウのデータベースタイプを変更ドロップダウンリストからデータベースのタイプを選択することで、SQL Anywhere または Ultra Light のデータベースに接続できます。
-version	Interactive SQL のバージョン番号を表示します。Interactive SQL 内のヘルプメニューから *Interactive SQL について*をクリックして、バージョン番号を参照することもできます。
-x	文をスキャンしますが、実行しません。長いスクリプトファイルの構文エラーをチェックする場合に有用です。
`dbisql-statement` | `dbisql-script-file`	SQL 文または指定された `dbisql-script-file` を実行します。 `dbisql-statement` または `dbisql-script-file` が指定されていないと、Interactive SQL は対話型モードになります。このモードでは、文をコマンドウィンドウに入力できます。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

備考

Interactive SQL を使用して、データベースのブラウズ、SQL 文の実行、およびスクリプトファイルの実行を行うことができます。また、次の項目に関するフィードバックも提供します。

- 影響を受けたローの数
- 各文に必要な時間
- クエリの実行プラン
- エラーメッセージ

Interactive SQL を使用して、SQL Anywhere データベース、Ultra Light データベース、SAP Sybase IQ データベース、SAP HANA データベース、または汎用の ODBC データベースに接続できます。

Windows の場合、実行プログラムが 2 つあります。

1. バッチスクリプトでは、`dbisql.exe` ではなく、`dbisql` または `dbisql.com` を呼び出す必要があります。実行プログラム `dbisql.com` は、コンソールアプリケーションとしてリンクされています。
2. 実行プログラム `dbisql.exe` はウィンドウアプリケーションとしてリンクされており、起動元のコマンドシェルがブロックされません。`dbisql.exe` をバッチファイルから実行した場合、標準出力または標準のエラーファイルに出力が送信されません。

Interactive SQL のデフォルトのエンコードは、Interactive SQL の default_isql_encoding オプションでも一時的に設定できます。

INPUT、OUTPUT、または READ 文の ENCODING 句を使用して、ファイルの読み込みまたは書き込みをするときに使用するエンコードを指定できます。

- INPUT 文
- OUTPUT 文
- READ 文

終了コードは、0 (成功) または 0 以外の値 (失敗) です。0 以外の終了コードが設定されるのは、(SQL 文またはスクリプトファイル名を指定したコマンドラインによって) Interactive SQL をバッチモードで実行した場合だけです。

コマンドプロンプトモードのとき、Interactive SQL は、処理の成功または失敗を、プログラム終了コードを設定することで示します。Windows オペレーティングシステムでは、プログラム終了コードに対して環境変数 ERRORLEVEL が設定されます。

Interactive SQL で `reload.sql` ファイルを実行しており、データベースが暗号化されている場合は、パラメータとして暗号化キーを指定する必要があります。READ 文にキーを指定していない場合、Interactive SQL でキーの入力を要求するプロンプトが表示されます。

次の方法で Interactive SQL を開始できます。

- SQL Central で ファイル > *Interactive SQL を開く* をクリックします。
- スタートメニューから、 スタート > プログラム > *SQL Anywhere17* > 管理ツール > *Interactive SQL* をクリックします。
- コマンドプロンプトで dbisql コマンドを使用します。

例

次のコマンドは、Ultra Light の `CustDB.udb` データベースに対してスクリプトファイル `mycom.sql` を実行します。ユーザ ID とパスワードが定義されていないため、デフォルトのユーザ ID である DBA とパスワード sql が使用されます。-

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

onerror オプションは終了として定義されています。このため、スクリプトファイルにエラーがあった場合、処理は終了します。

```
dbisql -ul -c DBF=CustDB.udb -onerror exit mycom.sql
```

関連情報

Ultra Light 接続パラメータ [175 ページ]
サポートされている終了コード [208 ページ]

1.12.5.3 Ultra Light SQL プリプロセッサユーティリティ (sqlpp)

Embedded SQL (ESQL) を含む C/C++ のプログラムをプリプロセッサで処理し、コンパイラを実行する前に、このプログラムに必要なコードを生成できるようにします。

次の表では、万全を期すためにすべてのオプションについて説明していますが、Ultra Light に関連するオプションは -eu と -wu のみです。

構文

```
sqlpp -u [ options ] esql-filename [ output-filename ]
```

オプション	説明
-d	データ領域サイズを減らし、コードサイズを増加させるコードを生成します。データ構造体を再利用し、実行時に初期化してから使用します。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

オプション	説明
-*e*flag	このオプションは、指定した規格に含まれない静的 Embedded SQL をエラーとして通知します。level 値は、使用する標準を表します。たとえば、sqlpp -e c03 ... は、コア SQL/2003 規格に含まれない構文を通知します。 level で使用できる値は次のとおりです。 **c03** コア SQL/2003 構文でない構文を通知します。 **p03** 上級レベル SQL/2003 構文でない構文を通知します。 **c99** コア SQL/1999 構文でない構文を通知します。 **p99** 上級レベル SQL/1999 構文でない構文を通知します。 **e92** 初級レベル SQL/1992 構文でない構文を通知します。 **i92** 中級レベル SQL/1992 構文でない構文を通知します。 **f92** 上級レベル SQL/1992 構文でない構文を通知します。 **t** 標準ではないホスト変数型を通知します。 **u** Ultra Light がサポートしていない構文を通知します。 以前のバージョンの SQL Anywhere と互換性を保つために、e、I、f を指定することもできます。これらはそれぞれ e92、i92、f92 に対応します。
-*h*width	sqlpp によって出力される .c ファイル内の分割された行の最大長を width に制限します。分割された行が C コンパイラで 1 行として解析されるように、分割された行の末尾には円記号が追加されます。デフォルト値はありません (出力の行はデフォルトで分割されません)。
-*k*	コンパイルされるプログラムが SQLCODE のユーザ宣言をインクルードすることをプリプロセッサに通知します。
-*m*mode	カーソルの更新可能性モード。*HISTORICAL* または *READONLY*。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

オプション	説明
-n	コード内の適切な場所で `#line` ディレクティブを使用して、C ファイル内に行番号情報を生成します。 このオプションを使用して、ソースのエラーをレポートし、また `output-filename` ファイルではなく、`esql-filename` ファイル内の行番号にあるソースをデバッグできます。
-oO/S spec	Ultra Light には不適用。
-q	ユーティリティをクワイエットモードで実行するように設定します。情報のバナー、バージョン番号、ステータスメッセージが非表示になります。エラーメッセージは引き続き表示されます。
-r-	Ultra Light には不適用。
-Sstring-length	プリプロセッサが C ファイルに出力する文字列の最大サイズを設定します。この値より長い文字列は、文字のリスト (`'a'`、`'b'`、`'c'` など) を使用して初期化されます。ほとんどの C コンパイラには、処理できる文字列リテラルのサイズに制限があります。このオプションを使用して上限を設定します。デフォルト値は 500。
-u	Ultra Light で必須です。Ultra Light データベースに必要な出力を生成します。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

オプション	説明
-*W*level	SQL 構文に準拠しないものを警告として通知します。level 値は、使用する標準を表します。たとえば sqlpp -w c03 ... は、コア SQL/2003 構文に含まれない SQL 構文を通知します。 level で使用できる値は次のとおりです。 **c03** コア SQL/2003 構文でない構文を通知します。 **p03** 上級レベル SQL/2003 構文でない構文を通知します。 **c99** コア SQL/1999 構文でない構文を通知します。 **p99** 上級レベル SQL/1999 構文でない構文を通知します。 **e92** 初級レベル SQL/1992 構文でない構文を通知します。 **i92** 中級レベル SQL/1992 構文でない構文を通知します。 **f92** 上級レベル SQL/1992 構文でない構文を通知します。 **t** 標準ではないホスト変数型を通知します。 **u** Ultra Light がサポートしていない構文を通知します。 以前のバージョンの SQL Anywhere と互換性を保つために、e、I、f を指定することもできます。これらはそれぞれ e92、i92、f92 に対応します。
-*X*	マルチバイト文字列をエスケープシーケンスに変更し、コンパイラをパススルーできるようにします。
-Zcollation-sequence	照合順を指定します。

備考

このプリプロセッサは、入力ファイル内の SQL 文を C/C++ に変換します。結果は output-filename に書き込みます。Embedded SQL を含むソースプログラムの拡張子は通常は sqc です。デフォルトの output-filename は拡張子 c が付いた esql-filename ベースの名前です。ただし、esql-filename に拡張子 .c がすでに付いている場合、デフォルトの出力ファイル拡張子は .cc になります。

照合順は、プリプロセッサにプログラムのソースコードで使用されている文字を理解させるために使用します。たとえば、識別子に使用できるアルファベット文字の識別などに使用されます。Ultra Light では、照合にコードページとソート順が含まれま

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

す。-z を指定しなかった場合は、プリプロセッサが、オペレーティングシステムに基づいて、使用する合理的な照合順を決定しようとします。

ulinit -Z を実行すると、サポートされている照合と対応するコードページのリストが表示されます。

注記

SQL プリプロセッサ (sqlpp) には、Embedded SQL アプリケーション内の静的 SQL 文をコンパイル時に通知する機能があります。この機能は、Ultra Light アプリケーションを開発するときに特に便利です。この機能で、SQL 文に Ultra Light との互換性があることを確認できます。SQL Anywhere アプリケーションと Ultra Light アプリケーションの両方に対する SQL の互換性をテストするには、-e オプションか -w オプションまたはその両方を使用します。

例

次のコマンドは、Ultra Light アプリケーション用に Embedded SQL ファイル `srcfile.sqc` をクワイエットモードで前処理します。

```
sqlpp -u -q MyEsqlFile.sqc
```

関連情報

Ultra Light 文字セット [32 ページ]

1.12.5.4 Ultra Light エンジンユーティリティ (uleng17)

アプリケーションからの Ultra Light データベースへの同時接続を管理し、Ultra Light エンジンが -ud オプションを使用するデーモンとして稼動できるようにします。

構文

```
uleng17 [ -ud ]
```

オプション	説明
-ud	エンジンを実行して、現在のユーザセッションが終了しても実行を継続できるようにします。-ud オプションを使用してデーモンを直接起動したときは、uleng17 コマンドがデーモンプロセスを作成し、(終了して次のコマンドを実行できるように) すぐに返します。その後、デーモンがそれ自体を初期化するか、コマンドで指定されたデータベースを開こうとします。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

備考

Ultra Light エンジンでは、起動時にメッセージウィンドウが表示されません。

Ultra Light エンジンは、複数のプロセスで同じデータベースに同時にアクセスする場合にアプリケーションで使用されます。Ultra Light デスクトップ管理ツールがこのエンジンを使用してデータベースに接続するため、エンジンは SQL Anywhere の `bin32` または `bin64` ディレクトリにインストールされます。

-ud オプションを使用して、Ultra Light エンジンを実行し、コンピュータをログオフするときに、データベースエンジンが動作し続けるようにします (通常、コンピュータからログオフすると、そのセッションに関連するすべてのアプリケーションは停止します)。

関連情報

Ultra Light の配備 [117 ページ]
Ultra Light C++ アプリケーションの構築と配備 [626 ページ]
Windows Mobile 用の Ultra Light データ管理コンポーネント [22 ページ]
Ultra Light エンジン停止ユーティリティ (ulstop) [218 ページ]
Ultra Light START 接続パラメータ [194 ページ]

1.12.5.5 Ultra Light エンジン停止ユーティリティ (ulstop)

Ultra Light エンジンの停止

構文

ulstop

備考

開発中にエンジンを手動でシャットダウンする場合は ulstop を使用してください。本配備で ulstop を使用する必要は一般的にはありません。

関連情報

Windows Mobile 用の Ultra Light データ管理コンポーネント [22 ページ]
Ultra Light エンジンユーティリティ (uleng17) [217 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.12.5.6 Ultra Light ユーティリティ消去 (ulerase)

Ultra Light データベースを消去します。

構文

```
ulerase [ options ] [ db-file-name ]
```

オプション	説明
@data	これを使用すると、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ことができます。同じ名前の環境変数と設定ファイルが両方存在する場合は、環境変数が使用されます。
-kkey または --ek=key	暗号化されたデータベースで使用する暗号化キーを指定します。
-p または --ep	暗号化キーの入力を求めるプロンプトを表示するよう指定します。
--log	指定したファイルに操作のログを記録します。
-q または --quiet	ユーティリティをクワイエットモードで実行するように設定します。情報のバナー、バージョン番号、ステータスメッセージが非表示になります。エラーメッセージは引き続き表示されます。
-uuid,pwd または --dba=uid,pwd	データベースへのアクセスに必要な userid と password を指定します。
-? または --help	ユーティリティの使用法を表示し、終了します。
db-file-name	指定したデータベースを消去します。

備考

データベースにアクセスできる必要があります。ユーザ ID とパスワードの組み合わせを使用して接続が許可される必要があります。そうでない場合、データベースは消去されません。

暗号化されたデータベースでは、接続文字列に指定されるキーが必要です。つまり、-kkey または -p のどちらかを使用します。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.12.5.7 Ultra Light 情報ユーティリティ (ulinfo)

Ultra Light データベースに関する情報を表示します。

構文

```
ulinfo -c options
```

オプション	説明
@data	これを使用すると、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ことができます。同じ名前の環境変数と設定ファイルが両方存在する場合は、環境変数が使用されます。
-c "connection-string" または --connect="connection-string"	データベース接続パラメータを指定します。必須。
-q または --quiet	ユーティリティをクワイエットモードで実行するように設定します。情報のバナー、バージョン番号、ステータスメッセージが非表示になります。エラーメッセージは引き続き表示されます。
--log=filename	指定したファイルに操作のログを記録します。
-? または --help	ユーティリティの使用法を表示し、終了します。

備考

Ultra Light データベースを開くときに生成される警告メッセージは、-q オプションを使用しないかぎり、常に表示されます。

例

すでに同期されている cv_dbattr.udb というファイルの基本的なデータベース内部構成を表示します。

```
ulinfo -c DBF=cv_dbattr.udb
```

関連情報

Ultra Light 接続パラメータ [175 ページ]
Ultra Light global_database_id オプション [205 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

Ultra Light ml_remote_id オプション [206 ページ]

1.12.5.8 Ultra Light データベース初期化ユーティリティ (ulinit)

新しい Ultra Light データベースを作成します。

このユーティリティは、次のいずれかのモードで機能します。

空モード

コマンドライン引数で指定された特性を持つ空のデータベースを作成します。

抽出モード

SQL Anywhere データベースに基づいてデータベースを作成します。

SQL Anywhere 参照データベースのテーブルおよびインデックスと一致する初期スキーマが作成されます。多くの参照データベース特性が抽出され、新しい Ultra Light データベースに使用されます。

構文

```
ulinit options dbname
```

オプション	説明
@data	これを使用すると、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ことができます。同じ名前の環境変数と設定ファイルが両方存在する場合は、環境変数が使用されます。
-a "keyword=value;..." または --SAconnect="keyword=value;..."	ユーティリティを抽出モードに設定して、指定された接続パラメータを使用して、既存のデータベースに接続します。 このオプションがない場合は、ユーティリティは指定された接続パラメータ (空モード) を使用して、新しいデータベースを作成します。
-c または --case	空モード。 すべての文字列比較で大文字と小文字の区別を強制します。
-d または --datacopy	抽出モード。 新しい Ultra Light データベースの各テーブルに、SQL Anywhere データベースの対応するテーブルからデータをコピーします。このオプションを使用しない場合、新しいデータベースは最初は空になります。 デフォルトでは、その後の同期時にこのデータはアップロードされません。次の同期時にデータをアップロードするには、-i と同時に -d を使用します。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

オプション	説明
--date_format=format	空モード。 データベースから取り出した日付の形式を設定します。
--date_order=date-format-interpretation	空モード。 日付形式の解釈を設定します。
*-e*value または *--fips*=value	空モード。 On または Off、1 または 0 など。このオプションは、FIPS 認定の AES 暗号化の使用を制御します。
-f または *--exactschema*	抽出モード。 同じスキーマが Ultra Light でサポートされていない場合は失敗します。スキーマが異なる場合は、警告が表示されます。
-g id または *--databaseid*=id	初期グローバル ID を、指定する INTEGER 値に設定します。この初期値と分割サイズが、グローバルオートインクリメントカラムがある新しいローに使用されます。アプリケーションを配備するときには、Mobile Link サーバとの同期のために、各データベースに対して異なる ID 番号の範囲を割り当てます。
-i または *--insertforupload*	抽出モード。 *-d* と共に使用します。挿入されたローを次の同期時にアップロードします。デフォルトでは、このユーティリティによって挿入されたローは、同期時にアップロードされません。
--identity-file=file	クライアントの TLS、PEM、または PKCS12 ID を含むファイルを指定します。
--identity-password=password	クライアント TLS ID のパスワードを指定します。
*-k*key または *--key*=key	抽出モード。 暗号化されたデータベースで使用する暗号化キーを指定します。
-K または *--prompt*	空モード。 暗号化キーの入力を求めるプロンプトを表示するよう指定します。
*-l*filename または *--sql*=filename	抽出モード。 実行時に、指定されたファイルに DDL データベーススキーマ作成の SQL 文のログを取ります。
--log=filename	空モード。 指定したファイルに操作のログを記録します。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

オプション	説明
-m`filename` または *--mirror_file*=`filename`	抽出モード。 データベースミラーファイルを指定します。
--max_hash_size=`size`	空モード。 デフォルトのインデックスハッシュサイズをバイト単位で設定します。
-n`pubname` または *--publication*=`pubname`	抽出モード。 必須。テーブルを Ultra Light データベーススキーマに追加します。 `pubname` は、リファレンスデータベース内のパブリケーションを指定します。パブリケーション内のテーブルは、Ultra Light データベースに追加されます。 複数のパブリケーションからのテーブルを Ultra Light データベースに追加するには、オプションを複数回指定します。リファレンスデータベース内のすべてのテーブルを Ultra Light データベースに追加するには、-n* を指定します。
--nearest_century=`yy`	空モード。 文字列から日付への変換で、2 桁の年の解釈を制御します。
-o`value` または *--obfuscate*=`value`	空モード。 On または Off、1 または 0 など。データベースのデータを難読化するかどうか制御します。難読化は、巧妙で容赦ないデータへのアクセス試行に対してセキュアではありません。
-p`size` または *--page_size*=`size`	空モード。 データベースページサイズを指定します。
--precision=`precision`	空モード。 計算結果が小数の場合の最大桁数を指定します。
-q または *--quiet*	ユーティリティをクワイエットモードで実行するように設定します。情報のバナー、バージョン番号、ステータスメッセージが非表示になります。エラーメッセージと警告は引き続き表示されます。
-r`size` または *--reserve_size*	データベースの接続のみ。 サイズを予約します。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

オプション	説明
-spubname または --sync_publication	抽出モード。 リファレンスデータベースの pubname と同じ定義を持つパブリケーションを Ultra Light データベースに作成します。パブリケーションは、同期の設定に使用されます。複数の同期パブリケーションを指定するには、複数の -s オプションを指定します。 このパブリケーション内のテーブルが -n オプションでリストされたパブリケーションに含まれている必要があります。 -s を指定しないと、Ultra Light リモートデータベースにパブリケーションが指定されません。
-Schecksum_level または --checksum_level=checksum_level	空モード。 0、1、または 2。データベースページに対するチェックサム検証のレベルを指定します。
--scale=scale	空モード。 計算結果が最大精度でトランケートされる場合の、小数点以下の最小桁数を指定します。
-tfile または --rootcert=file	信頼できるルート証明書を含むファイルを指定します。この証明書は、サーバ認証に必要です。
--time_format=format	空モード。 データベースから取り出した時刻の表示形式を設定します。
--timestamp_format=format	空モード。 データベースから取り出したタイムスタンプの表示形式を設定します。
--timestamp_increment=increment	空モード。 Ultra Light でのタイムスタンプのトランケート方法を指定します。
--timestamp_with_time_zone_format= format	空モード。 このオプションでは、データベースから取り出した TIMESTAMP WITH TIME ZONE 値の表示形式を設定します。
-u<uid>,<pwd> または --dba=<uid>,<pwd>	データベースの接続のみ。 ユーザ ID とパスワードを指定します。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

オプション	説明
--utf8_encoding=value	空モード。 On または Off、1 または 0 など。Unicode 用の 8 ビットマルチバイトエンコードである UTF-8 文字セットフォーマットでデータをエンコードします。
-w または *--nowarnings*	抽出モード。 警告を表示しません。
*-x*table または *--exclude*	抽出モード。 リスト内に指定されたテーブルを除外します。
-y または *--overwrite*	既存のデータベースファイルを上書きします。
*-z*collation-sequence または *--collation*=collation-sequence	空モード。 照合順を指定します。
-Z または *--listcollation*	空モード。 使用可能な照合順をリストして終了します。
-? または *--help*	ユーティリティの使用法を表示し、終了します。

注記

いずれのモードでも説明に指定されていないオプションは使用できません。

備考

抽出モードで実行する場合、ulinit ユーティリティは、SQL Anywhere データベースとできるだけ厳密に一致する Ultra Light データベースを作成しようとします。たとえば、SQL Anywhere データベースのカラムに含まれる句が Ultra Light ではサポートされていない場合、デフォルト値は無視され、代わりに Ultra Light のデフォルト値が使用されます。警告が生成されて、作成は続行されます。これは、SQL Anywhere テーブルを変更できなくても、Ultra Light の妥当な代替手段を使用できるような

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

場合をサポートしています。スキーマの完全一致を強制するには、-f オプションを使用します。スキーマが Ultra Light の妥当な代替値をサポートしていない場合、ulinit ユーティリティは失敗します。

例

TestPublication で定義されているテーブルを含む customer.udb というファイルを作成します。

```
ulinit -a "DSN=MySADb;UID=JimmyB;PWD=secret" -n TestPublication -k mykey
customer.udb
```

この例では、MySADb データソースで定義された SQL Anywhere データベースに接続します。これにより、そのデータベースからのすべてのデータベースオプションと TestPublication パブリケーションに含まれるすべてのテーブルを含む Ultra Light データベースが作成されます。新しい Ultra Light データベースは customer.udb と呼ばれ、mykey で暗号化されます。

2 つの異なるパブリケーションを含む customer.udb というファイルを作成します。Pub1 には、優先的に同期するデータのサブセットが含まれ、Pub2 には残りのデータが含まれます。

```
ulinit -a "DSN=MySADb;UID=JimmyB;PWD=secret" --exactschema -n Pub1 -n Pub2 -s
Pub1 -s Pub2 customer.udb
```

この例では、MySADb データソースで定義された SQL Anywhere データベースに接続します。これにより、そのデータベースからのすべてのデータベースオプションと Pub1 および Pub2 パブリケーションに含まれるすべてのテーブルを含む Ultra Light データベースが作成されます。また、Pub1 および Pub2 パブリケーションで新しい Ultra Light データベースも作成されます。--exactschema オプションが設定されているので、ulinit は正確なスキーマをすべて抽出できないと失敗します。

別の customer.udb ファイルがあれば、それを上書きする新しい空のデータベースを作成します。新しいデータベースにはスキーマがなく、データベースオプションはすべてデフォルト値に設定されています。

```
ulinit -y customer.udb
```

関連情報

SQL Anywhere データベースから Ultra Light データベースへの変換 [36 ページ]
データベースセキュリティ [35 ページ]
データベースセキュリティ [35 ページ]
Ultra Light 文字セット [32 ページ]
Ultra Light でのデータのパブリッシュ [80 ページ]
Ultra Light 接続パラメータ [175 ページ]
Ultra Light の date_format 作成オプション [145 ページ]
Ultra Light の date_order 作成オプション [147 ページ]
Ultra Light 作成オプション [153 ページ]
Ultra Light global_database_id オプション [205 ページ]
Ultra Light MIRROR_FILE 接続パラメータ [191 ページ]
Ultra Light max_hash_size 作成オプション [155 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

Ultra Light nearest_century 作成オプション [157 ページ]
Ultra Light obfuscate 作成オプション [159 ページ]
Ultra Light precision 作成オプション [162 ページ]
Ultra Light RESERVE_SIZE 接続パラメータ [193 ページ]
Ultra Light checksum_level 作成オプション [143 ページ]
Ultra Light scale 作成オプション [164 ページ]
Ultra Light time_format 作成オプション [165 ページ]
Ultra Light timestamp_format 作成オプション [167 ページ]
Ultra Light timestamp_increment 作成オプション [170 ページ]
Ultra Light utf8_encoding 作成オプション [174 ページ]
Ultra Light timestamp_with_time_zone_format 作成オプション [171 ページ]

1.12.5.9 Ultra Light データベースへの XML のロードユーティリティ (ulload)

XML ファイルから新しいデータベースまたは既存のデータベースにデータをロードします。

構文

```
ulload  -c "connection-string" [ options ] xml-file
```

オプション	説明
@data	これを使用すると、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ことができます。同じ名前の環境変数と設定ファイルが両方存在する場合は、環境変数が使用されます。
-a または --append	データとスキーマの定義を既存のデータベースに追加します。
-c "connection-string" または --connect="connection-string"	データベース接続パラメータを指定します。
-d または --dataonly	データだけをロードし、XML ファイル入力内のメタデータは無視します。 -d または --dataonly スイッチは -a が指定されている場合にだけ使用できます (データのみをロードするため、データをロードする UDB がロードされるデータをサポートするスキーマと共に存在する必要があります)。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

オプション	説明
*-e*value または *--fips*= value	On または Off、1 または 0 などを指定します。このオプションは、FIPS 認定の AES 暗号化の使用を制御します。
*-E*behavior または *--onerror*=behavior	XML ファイルからデータを読み出し中にエラーが起こった場合の事象を制御します。behavior には次のいずれかの値を指定します。 **continue** エラーは無視され、XML のロードが継続されます。 **prompt** 続行するように要求されます。 **quit** XML のロードが停止し、エラーで終了します。この動作はデフォルトの動作です。 **exit** ulload が終了します。
*-f*directory または *--filedir*=directory	ロードする追加データを含むファイルが格納されているフォルダを設定します。
*-g*ID または *--databaseid*=ID	初期グローバル ID を、指定する INTEGER 値に設定します。この初期値と分割サイズが、グローバルオートインクリメントカラムがある新しいローに使用されます。アプリケーションを配備するときには、Mobile Link サーバとの同期のために、各データベースに対して異なる ID 番号の範囲を割り当てます。
-i または *--insertforsync*	挿入されたローを次の同期時にアップロードします。デフォルトでは、このユーティリティによって挿入されたローは、同期時にアップロードされません。
--identity-file = file	クライアントの TLS、PEM、または PKCS12 ID を含むファイルを指定します。
--identity-password = password	クライアント TLS ID のパスワードを指定します。
*-l*filename または *--log*=filename	指定したファイルに操作のログを記録します。
-n または *--schemaonly*	メタデータだけをロードし、XML 入力ファイル内のデータは無視します。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

オプション	説明
-o*value* または *--obfuscate*=*value*	On または Off、1 または 0 など。データベースのデータを難読化するかどうか制御します。難読化は、巧妙で容赦ないデータへのアクセス試行に対してセキュアではありません。
-p*page-size* または *--page_size*=*page-size*	データベースページサイズを定義します。
-q または *--quiet*	ユーティリティをクワイエットモードで実行するように設定します。情報のバナー、バージョン番号、ステータスメッセージが非表示になります。エラーメッセージは引き続き表示されます。
-s*file* または *--sql*=*file*	データベースのロードに使用された SQL 文を、指定する *file* に記録します。
-t*file* または *--rootcert*=*file*	信頼できるルート証明書を含むファイルを指定します。この証明書は、サーバ認証に必要です。
--utf8_encoding=*value*	On または Off、1 または 0 など。Unicode 用の 8 ビットマルチバイトエンコードである UTF-8 文字セットフォーマットでデータをエンコードします。
-v または *--verbose*	冗長メッセージを表示します。
-y または *--overwrite*	確認メッセージを表示しないでデータベースファイルを上書きします。このオプションは、ulload を使用して新しいデータベースを作成する場合にのみ適用されます。
-? または *--help*	ユーティリティの使用法を表示し、終了します。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

備考

ulload ユーティリティは、ulunload、ulunloadold (SQL Anywhere 10 で提供)、または ulxml (Ultra Light バージョン 8 と 9) で生成された入力 XML ファイルを取得します。ulunload とこのユーティリティを同時に使用すると、データベースを再構築できます。データベースを再構築するもう 1 つの手段としては、ulunload を使用して SQL 文を生成し、DBISQL を使用して新しいデータベースに読み込む方法があります。

XML ファイルには、スキーマのメタデータかデータベースデータのメタデータまたはその両方が含まれている可能性があります。-d を使用すると、スキーマのメタデータは無視され、データのみが .udb ファイルに追加されます。-n を使用すると、データとメタデータが無視され、スキーマのみが .udb ファイルに追加されます。

コマンドラインでオプションを設定するか、証明書を指定すると、ulload で処理される xml-file 内の設定より優先されます。

ulload ユーティリティは、XML の読み込み時にデータベースに同期プロファイルをリストアします。

このユーティリティはエラーコードを返します。0 以外の値は処理に失敗したことを示します。

例

新しい Ultra Light データベースファイル sample.udb を作成し、sample.xml 内のデータをロードします。

```
ulload -c DBF=sample.udb sample.xml
```

sample.xml から既存のデータベース sample.udb にデータをロードし、エラーが発生した場合はプロンプトを表示します。

```
ulload -d -c DBF=sample.udb --onerror=prompt sample.xml
```

sample.udb データベースの test_data.xml に保存されるスキーマとデータを作成します。-a スイッチが指定されているため、このコマンドを実行する前に sample.udb が存在する必要があります。さらに、sample.udb にすでにあるスキーマやデータと競合するスキーマやデータがあると、ULLOAD コマンドが失敗します。

```
ulload -c DBF=sample.udb -a test_data.xml
```

関連情報

データベースセキュリティ [35 ページ]
データベースセキュリティ [35 ページ]
Ultra Light 文字セット [32 ページ]
Ultra Light 接続パラメータ [175 ページ]
Ultra Light データベースのアンロードユーティリティ (ulunload) [236 ページ]
サポートされている終了コード [208 ページ]
Ultra Light 作成オプション [153 ページ]
Ultra Light データベースのアンロードユーティリティ (ulunload) [236 ページ]
Ultra Light global_database_id オプション [205 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

Ultra Light obfuscate 作成オプション [159 ページ]
Ultra Light page_size 作成オプション [160 ページ]
Ultra Light global_database_id オプション [205 ページ]
Ultra Light utf8_encoding 作成オプション [174 ページ]

1.12.5.10 Ultra Light 同期ユーティリティ (ulsync)

Ultra Light データベースを Mobile Link サーバと同期させます。このツールを使用して、アプリケーション開発中に同期をテストできます。

構文

```
ulsync [ options ]  [synchronization-parameters]
```

オプション	説明
@data	これを使用すると、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ことができます。同じ名前の環境変数と設定ファイルが両方存在する場合は、環境変数が使用されます。
-c "connection-string" または --connect="connection-string"	必須。connection-string の DBF パラメータまたは file_name パラメータで指定するデータベースに接続します。ユーザ ID とパスワードの両方を指定しなかった場合は、DBA のデフォルトの UID と sql の PWD が使用されます。

-pprofile-name
または
--profile=profile
	指定した同期プロファイルを使用して同期します。次の構文と同義です。
	`SYNCHRONIZE profileName MERGE syncOptions`
	ここでは、同期オプションは後続の ulsync オプションから取得されます。次に例を示します。
	`ulsync -p profileName "MobiLinkUid=ml;ScriptVersion=Version001...syncOptions"`

-q または --quiet	ユーティリティをクワイエットモードで実行するように設定します。情報のバナー、バージョン番号、ステータスメッセージが非表示になります。エラーメッセージは引き続き表示されます。
-r または --result	前回の同期の結果を表示し、終了します。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

オプション	説明
-v または *--verbose*	同期の進行状況のメッセージを表示します。C++ API または SQL SYNCHRONIZE PROFILE 文を使用して、同期の進捗状況を表示するかどうかも指定します。
--log=`filename`	指定したファイルに操作のログを記録します。
-? または *--help*	ユーティリティの使用法を表示し、終了します。

備考

証明書は、Xcode プロジェクトに含めることで、アプリケーションにバンドルできます。証明書ファイルオプションに対して拡張子付きのファイル名を指定すると、Ultra Light は自動的にバンドルを確認し、証明書ファイルを見つけます。必要に応じて、フルパスを指定することもできます。

バージョン 10 以前に有効であった次のオプションは今はサポートされていません。-a `authenticate-parameters`、-e `sync-parms`、-k `stream-type`、-n (no sync)、および -x `protocol options`。-e `keyword=value` は同期パラメータ文字列の一部となり、-k および -x は *Stream*=`stream`{`stream-parms`} 同期パラメータ文字列の一部になっています。

次に、ulsync の例と、同義の SQL 文を示します。

```
ulsync -p profile "parms"
```

このコマンドは、次の SQL 文と同義です。

```
SYNCHRONIZE PROFILE profile MERGE parms
```

次は別の例です。

```
ulsync "parms"
```

このコマンドは、次の SQL 文と同義です。

```
SYNCHRONIZE USING parms
```

安全な同期のために、Ultra Light アプリケーションはパブリック証明書にアクセスする必要があります。次の方法で証明書を参照できます。

- ulinit または ulload で *-t*`file` オプションを使用して Ultra Light データベースの作成時に証明書情報を組み込みます。
- *trusted_certificate*=`file` ストリームオプションを使用して同期時に外部の証明書ファイルを参照します。

このユーティリティはエラーコードを返します。0 以外の値は処理に失敗したことを示します。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

例

次のコマンドは、remoteA という Mobile Link ユーザの `myuldb.udb` というデータベースファイルを同期させます。

```
ulsync -c DBF=myuldb.udb "MobiLinkUid=remoteA;Stream=http;ScriptVersion=2"
```

次のコマンドは、`myuldb.udb` というデータベースファイルを HTTPS を通じ *C:¥Users¥Public¥Documents¥SQL Anywhere 17¥Samples*`¥Certificates¥rsaroot.crt` 証明書を使用して同期させます。データベースの作成時に信頼できる証明書ファイルがデータベースに追加されなかったため、*trusted_certificate*`=file` オプションを使用する必要があります。さらに、Mobile Link ユーザ名は remoteB です。

```
ulsync -c DBF=myuldb.udb "Stream=https{trusted_certificate=C:¥Users¥Public
¥Documents¥SQL Anywhere 17¥Samples¥Certificates
¥rsaroot.crt};MobiLinkUid=remoteB;ScriptVersion=2;UploadOnly=ON"
```

次のコマンドは、`synced.udb` というデータベースファイルについて、前回の同期の結果を表示します。

```
ulsync -r -c dbf=synced.udb
```

前回の同期の結果は、次のようにリストされます。

```
SQL Anywhere UltraLite Synchronization Utility Version 17.0.4.1691
    Results of last synchronization:
    Succeeded
        Download timestamp: 2006-07-25 16:39:36.708000
        Upload OK
        No ignored rows
        Partial download retained
        Authentication value: 1000 (0x3e8)
```

次の例は、CustDB データベースをユーザ名 50 で TCP/IP 接続によってポート 2439 で同期させる場合に使用するコマンドラインを示します。これは冗長進行メッセージを使用します。

```
ulsync -c "dbf=C:¥Users¥Public¥Documents¥SQL Anywhere 17¥Samples¥UltraLite¥CustDB
¥custdb.udb" "MobiLinkUid=50;ScriptVersion=custdb 17.0.4;Stream=tcpip{port=2439}"
```

次のコマンドは、E2EE (エンドツーエンド暗号化) とともに TLS 暗号化を使用する方法を示します。

```
ulsync -c "uid=dba;pwd=sql;dbf=myudb.db"
"MobiLinkUid=rem1;MobiLinkPwd=password;ScriptVersion=v1;Stream=tls{host=myServer;p
ort=2439;trusted_certificate=clientcert.pem;e2ee_public_key=e2eepublic.pem}"
```

関連情報

Ultra Light クライアント [71 ページ]
Ultra Light データベース同期プロファイルオプション [234 ページ]
Ultra Light 接続パラメータ [175 ページ]
サポートされている終了コード [208 ページ]
Ultra Light データベース同期プロファイルオプション [234 ページ]
CREATE SYNCHRONIZATION PROFILE 文 [Ultra Light] [493 ページ]

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

1.12.5.11 Ultra Light データベース同期プロファイルオプション

同期プロファイルオプションは、ulsync ユーティリティのコマンドラインで、他のすべてのコマンドラインオプションを定義した後に指定します。キーワードは、大文字と小文字を区別しません。

同期プロファイルオプション	有効な値	説明
AllowDownloadDupRows	ブール式	このオプションを指定すると、プライマリキーの同じローが複数ダウンロードされたときにエラーが発生しません。このオプションによって、同期を失敗させることなく、不整合なデータを同期できます。デフォルト値は "no" です。
AuthParms	文字列 (カンマ区切り)	Mobile Link サーバに送信される認証パラメータのリストを指定します。認証パラメータを使用すると、Mobile Link スクリプトでカスタム認証を実行できます。
CheckpointStore	ブール式	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。
ContinueDownload	ブール式	以前に失敗したダウンロードを再起動します。ダウンロードを続行すると、失敗した同期でダウンロードするように選択されていた変更のみを受信します。デフォルトでは、Ultra Light はダウンロードを続行しません。
DisableConcurrency	ブール式	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。
DownloadOnly	ブール式	ダウンロード専用同期を実行します。
KeepPartialDownload	ブール式	接続エラーが発生した場合に、部分的なダウンロードを維持するかどうかを制御します。デフォルトでは、Ultra Light は、部分的にダウンロードされた変更をロールバックしません。
MobiLinkPwd	文字列	ユーザ名に対する既存の Mobile Link パスワードを指定します。
MobiLinkUid	文字列	Mobile Link ユーザ名を指定します。
NewMobiLinkPwd	文字列	Mobile Link ユーザの新しい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。
Ping	ブール式	サーバとの通信の確認のみを行います。同期は実行しません。
Publications	文字列 (カンマ区切り)	同期させるパブリケーションを指定します。パブリケーションによって、同期に関与するリモートデータベース上のテーブルが決定されます。このパラメータが空白 (デフォルト) の場合は、すべてのテーブルが同期されます。このパラメータがアスタリスク (*) の場合は、すべてのパブリケーションが同期されます。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

同期プロファイルオプション	有効な値	説明
ScriptVersion	文字列	Mobile Link スクリプトのバージョンを指定します。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。スクリプトバージョンを指定しない場合、"デフォルト" が使用されます。
SendDownloadACK	ブール式	クライアントからサーバにダウンロード確認が送信されるように指定します。デフォルトでは、Mobile Link サーバはダウンロード確認を提供しません。
Stream	文字列 (サブリスト付き)	Mobile Link のネットワーク同期プロトコルを指定します。
TableOrder	文字列 (カンマ区切り)	アップロードでのテーブルの順序を指定します。デフォルトでは、Ultra Light は、外部キーの関係に基づいて順序を選択します。
UploadOnly	文字列	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。

ブール値に指定できるのは、Yes/No、1/0、True/False、On/Off です。ブール値の場合はどれであっても、デフォルトは No です。他の値の場合、デフォルトは未指定になります。

関連情報

Ultra Light でのデータのパブリッシュ [80 ページ]
Additional Parameters 同期パラメータ [92 ページ]
Authentication Parameters 同期パラメータ [94 ページ]
Download Only 同期パラメータ [97 ページ]
Keep Partial Download 同期パラメータ [98 ページ]
Ping 同期パラメータ [103 ページ]
Send Download Acknowledgement 同期パラメータ [107 ページ]
Stream Type 同期パラメータ [109 ページ]
Upload Only 同期パラメータ [113 ページ]
ALTER SYNCHRONIZATION PROFILE 文 [Ultra Light] [481 ページ]
DROP SYNCHRONIZATION PROFILE 文 [Ultra Light] [506 ページ]
SYNCHRONIZE 文 [Ultra Light] [522 ページ]
Ultra Light オプション [138 ページ]

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

1.12.5.12 Ultra Light データベースのアンロードユーティリティ (ulunload)

Ultra Light データベース全体あるいは Ultra Light データの一部またはすべてを、XML か SQL にアンロードします。

構文

```
ulunload -c "connection-string" [ options ] output-file
```

オプション	説明
@data	これを使用すると、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ことができます。同じ名前の環境変数と設定ファイルが両方存在する場合は、環境変数が使用されます。
-bmax-size または --maxblob=max-size	XML ファイルに格納するカラムデータの最大サイズを設定します。デフォルトは 10 KB です。すべてのデータを XML ファイルに保存するには (最大サイズなし)、-b -1 を使用します。
-c "connection-string" または --connect="connection-string"	必須。connection-string の DBF パラメータまたは file_name パラメータで指定するデータベースに接続します。ユーザ ID とパスワードの両方を指定しなかった場合は、DBA のデフォルトの UID と sql の PWD が使用されます。
-d または --dataonly	データベースのデータのみを出力ファイルにアンロードします。スキーマ情報はアンロードしません。
-etable,... または --exclude=table,...	データベースのアンロード時に指定した table を除外します。複数のテーブルをカンマで区切って指定できます。次に例を示します。 `-e mydbtable1,mydbtable5`
-fdirectory または --filedir=directory	-b で指定した最大サイズよりも大きいデータを格納するフォルダを設定します。デフォルトは、出力ファイルと同じフォルダです。
-lfilename または --log=filename	指定したファイルに操作のログを記録します。
-n または --schemaonly	スキーマだけをアンロードし、データベース内のデータは無視します。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

オプション	説明
-q または *--quiet*	ユーティリティをクワイエットモードで実行するように設定します。情報のバナー、バージョン番号、ステータスメッセージが非表示になります。エラーメッセージは引き続き表示されます。
-s または *--sql*	SQL Anywhere 互換の SQL 文としてアンロードします。SQL ファイルの出力は、DBISQL. を使用して Ultra Light または SQL Anywhere で読み込むことができます。
-t`table`,... または *--include*=`table`,...	指定した `table` 内のデータだけをアンロードします。複数のテーブルをカンマで区切って指定できます。次に例を示します。 `-t mydbtable2,mydbtable6`
-v または *--verbose*	冗長メッセージを表示します。
-x`owner` または *--owner*=`owner`	テーブルが特定のユーザ ID の所有になるように出力します。このオプションは、-s オプションと同時に使用できます。
-y または *--overwrite*	確認メッセージを表示しないで `output-file` を上書きします。
-? または *--help*	ユーティリティの使用法を表示し、終了します。
`output-file`	必須。データベースのアンロード先のファイルの名前を設定します。-s オプションを使用すると、データベースは SQL 文としてアンロードされます。-s オプションを使用しなかった場合は、データベースは XML としてアンロードされます。

備考

デフォルトでは、データベース内のスキーマとデータを表す XML が出力されます。出力は、アーカイブに使用するか、Ultra Light データベースをすべてのリリース間で移植するために使用できます。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

同期プロファイルがあるデータベースを保存すると、以前のバージョンの Ultra Light ユーティリティと互換性がない XML になります。対処方法として、XML を編集し、次のようにマーク付けされたテキストセクションを削除します。

```
<syncprofiles>...</syncprofiles>
```

データベースのアンロード時に次のデータは保持されません。

- 同期ステータス、同期数、ロー削除。アンロードの前にデータベースを同期させる必要があります。
- Ultra Light ユーザのエントリ。

保持されたデータベースオプションまたはプロパティを確認するには、ulload ユーティリティでデータベースを再ロードした後に ulinfo を実行します。

カラムデータが、-b で指定した最大サイズを超えていた場合、オーバーフローは `*.bin` ファイルに保存されます。このファイルは、次のいずれかのフォルダにあります。

- XML ファイルと同じフォルダ
- -f で指定したフォルダ

ファイルは次の命名規則に従います。

```
tablename-columnname-rownumber.bin
```

-x オプションで Ultra Light テーブルに所有権を割り当てることができます。テーブルに所有者を割り当てるのは、出力の SQL 文を、SQL Anywhere データベースの作成または変更に使用する場合だけです。Ultra Light で読み込まれるとき、所有者名は通知されることなく無視されます。

このユーティリティはエラーコードを返します。0 以外の値は処理に失敗したことを示します。

このユーティリティを使用して Windows Mobile デバイス上で直接データベースをアンロードすると、Ultra Light はアンロードまたは操作の実行前にデータベースをバックアップできません。したがって、これらの操作を手動で行ってから、これらのウィザードを実行してください。

例

`sample.udb` データベースを `sample.xml` ファイルにアンロードします。

```
ulunload -c DBF=sample.udb sample.xml
```

`sample.udb` データベースのデータを `sample.sql` という SQL ファイルにアンロードします。SQL ファイルが存在する場合は、上書きします。

```
ulunload -c DBF=sample.udb -d -y -s sample.sql
```

関連情報

Ultra Light 接続パラメータ [175 ページ]
サポートされている終了コード [208 ページ]
Ultra Light データベースへの XML のロードユーティリティ (ulload) [227 ページ]
Ultra Light 情報ユーティリティ (ulinfo) [220 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.12.5.13 Ultra Light データベース検証ユーティリティ (ulvalid)

Ultra Light データベースのフル (通常の) 検証を実行します。

構文

```
ulvalid -c "connection-string" [ options ]
```

オプション	説明
@data	これを使用すると、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ことができます。同じ名前の環境変数と設定ファイルが両方存在する場合は、環境変数が使用されます。
-c "connection-string" または --connect="connection-string"	必須。connection-string で指定するデータベースに接続します。ユーザ ID とパスワードの両方を指定しなかった場合は、DBA のデフォルトの UID と sql の PWD が使用されます。
-e または --express	エクスプレス検証。テーブルの検証のみを実行します。このオプションを使用すると、通常の検証よりも速く検証できます。
-q または --quiet	ユーティリティをクワイエットモードで実行するように設定します。情報のバナー、バージョン番号、ステータスメッセージが非表示になります。エラーメッセージは引き続き表示されます。
-v または --verbose	冗長メッセージを表示します。
--log=filename	指定したファイルに操作のログを記録します。
-? または --help	ユーティリティの使用法を表示し、終了します。

備考

データベースの検証によってテーブルメタデータの精度を検証し、ファイルが破損していないことを確認します。

検証には次のものがあります。

データベースページ

チェックサム (有効な場合) を使用してすべてのデータベー スページを検証します。特定の重要なページには必ずチェックサムが含まれており、チェックサムのないページにも基本的な妥当性検査が実行されます。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

テーブル

テーブルのロー数が各インデックスの数と一致していることをチェックしてテーブルを検証します。

インデックス

エントリが有効なローを参照することをチェックしてインデックスを検証します。ulvalid -e では、テーブルの検証のみを行うエクスプレスチェックを実行します。

例

データベース `sample.udb` のエクスプレス検証の例は、クワイエットモードで動作します。

```
ulvalid -c DBF=sample.udb -e -q
```

関連情報

Ultra Light データベースの検証 [49 ページ]
Ultra Light checksum_level 作成オプション [143 ページ]

1.12.6 Ultra Light のシステムテーブル

Ultra Light データベースのスキーマは独自フォーマットで格納されます。

以前のバージョンの Ultra Light データベースは、複数のシステムテーブルに格納されていました。これらのシステムテーブル (システムビュー) は下位互換性のために引き続き問い合わせ可能になっていますが、その中にはユーザスキーマ (テーブル、カラム、インデックスなど) に関する情報のみが含まれており、システムスキーマに関する情報は含まれていません。たとえば、systable に対する問い合わせによって systable 自体のプロパティを検索することはできません。systable に対する問い合わせでは、ユーザが作成したテーブルのプロパティのみを検索できます。

各 Ultra Light プログラミング API では、データベースのスキーマに関する問い合わせに使用できるオブジェクトとメソッドがサポートされています。スキーマについて調べる場合は、システムビューを問い合わせるのではなく、これらのオブジェクトと API を使用してください。

これらのシステムビューに対して実行するすべてのクエリは、完全なテーブルスキャンに相当します。これらのシステムビューでは、インデックススキャンはサポートされていません。

このセクションの内容:

sysarticle システムテーブル [241 ページ]
sysarticle システムテーブルの各ローは、パブリケーションに属するテーブルを示します。

syscolumn システムテーブル [241 ページ]
syscolumn システムテーブルの各ローは、カラム 1 つを示します。

sysindex システムテーブル [242 ページ]
sysindex システムテーブルの各ローは、データベース内のインデックス 1 つを示します。

sysixcol システムテーブル [243 ページ]

 © 2016 SAP SE or an SAP affiliate company.All rights reserved.

sysixcol システムテーブルの各ローは、sysindex にリストされているインデックスのカラム 1 つを示します。

syspublication システムテーブル [244 ページ]
syspublication システムテーブルの各ローは、パブリケーションを示します。

syssyncresult システムテーブル [245 ページ]
syssyncresult システムテーブルローにはいずれも最新の同期についての情報が含まれています。

systable システムテーブル [246 ページ]
systable システムテーブルの各ローは、データベース内のテーブル 1 つを示します。

1.12.6.1 sysarticle システムテーブル

sysarticle システムテーブルの各ローは、パブリケーションに属するテーブルを示します。

カラム名	カラム型	説明
publication_id	UNSIGNED INT	このアーティクルが属するパブリケーションの識別子。
table_id	UNSIGNED INT	パブリケーションに属するテーブルの識別子。
where_expr	VARCHAR(2048)	ローをフィルタする述部 (オプション)。

制約

PRIMARY KEY (publication_id, table_id)

FOREIGN KEY (publication_id) REFERENCES syspublication (publication_id)

FOREIGN KEY (table_id) REFERENCES systable (object_id)

関連情報

syspublication システムテーブル [244 ページ]

1.12.6.2 syscolumn システムテーブル

syscolumn システムテーブルの各ローは、カラム 1 つを示します。

カラム名	カラム型	説明
column_name	VARCHAR(128)	カラム名。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

カラム名	カラム型	説明
default	VARCHAR(128)	このカラムのデフォルト値。例: autoincrement。
domain	UNSIGNED INT	カラムのドメインを示す列挙値。
domain_info	UNSIGNED INT	サイズが可変のドメインで使用されます。
nulls	VARCHAR(1)	カラムのデフォルトに NULL が許可されるかどうかを決定します。
object_id	UNSIGNED INT	カラムのユニークな識別子。
table_id	UNSIGNED INT	カラムが属するテーブルの識別子。

制約

PRIMARY KEY (table_id, object_id)

FOREIGN KEY (table_id) REFERENCES systable (object_id)

1.12.6.3 sysindex システムテーブル

sysindex システムテーブルの各ローは、データベース内のインデックス 1 つを示します。

カラム名	カラム型	説明
check_on_commit	BIT	すべての外部キーに一致するプライマリローがあることを確認するために参照整合性をいつチェックするかを示します。type が *foreign* の場合にのみ必要です。
index_name	VARCHAR(128)	インデックスの名前。
ixcol_count	UNSIGNED INT	インデックス内のカラム数。
nullable	BIT	type が *foreign* の場合にのみ必要です。NULL が許可されるかどうかを示します。
object_id	UNSIGNED INT	インデックスのユニークな識別子。
primary_index_id	UNSIGNED INT	type が *foreign* の場合にのみ必要です。プライマリインデックスの識別子をリストします。
primary_table_id	UNSIGNED INT	type が *foreign* の場合にのみ必要です。プライマリテーブルの識別子をリストします。
root_handle	UNSIGNED INT	内部でのみ使用。
table_id	UNSIGNED INT	インデックスが適用されるテーブルのユニークな識別子。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

カラム名	カラム型	説明
type	VARCHAR(10)	インデックスの型。次のいずれかです。 • *primary* • *foreign* • *key* • *unique* • *index*
hash_size	UNSIGNED SHORTINT	インデックスのハッシュに使用されるハッシュサイズ。

制約

PRIMARY KEY (table_id, object_id)

FOREIGN KEY (table_id) REFERENCES systable (object_id)

関連情報

sysixcol システムテーブル [243 ページ]

1.12.6.4 sysixcol システムテーブル

sysixcol システムテーブルの各ローは、sysindex にリストされているインデックスのカラム 1 つを示します。

カラム名	カラム型	説明
column_id	UNSIGNED INT	インデックス対象カラムのユニークな識別子。
index_id	UNSIGNED INT	このインデックスカラムが属するインデックスのユニークな識別子。
order	VARCHAR(1)	インデックス内のカラムが昇順 (A) か、降順 (D) かを示します。
sequence	UNSIGNED INT	インデックス内のカラムの順序。
table_id	UNSIGNED INT	インデックスが適用されるテーブルのユニークな識別子。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

制約

PRIMARY KEY (table_id, index_id, sequence)

FOREIGN KEY (table_id, index_id) REFERENCES sysindex (table_id, object_id)

FOREIGN KEY (table_id, column_id) REFERENCES syscolumn (table_id, object_id)

関連情報

sysindex システムテーブル [242 ページ]

1.12.6.5 syspublication システムテーブル

syspublication システムテーブルの各ローは、パブリケーションを示します。

カラム名	カラム型	説明
download_timestamp	TIMESTAMP	最後のダウンロードの時刻。
last_sync	UNSIGNED BIGINT	アップロードの進行状況を追跡するために使用します。
publication_id	UNSIGNED INT	パブリケーションのユニークな識別子。
publication_name	VARCHAR(128)	パブリケーションの名前。

制約

PRIMARY KEY (publication_id)

関連情報

sysarticle システムテーブル [241 ページ]

 © 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.12.6.6 syssyncresult システムテーブル

syssyncresult システムテーブルローにはいずれも最新の同期についての情報が含まれています。

カラム名	カラム型	説明
sql_code	INTEGER	最後の同期の SQL コード。
error_string	CHAR(200)	前回の同期でのエラーメッセージ。
stream_error_code	SMALLINT	特定のストリームエラー。想定される値については、ss_error_code 列挙を参照してください。
system_error_code	INTEGER	システム固有のエラーコード。エラーコードの詳細については、プラットフォームのマニュアルを参照してください。
stream_error_string	CHAR(80)	stream_error_code 値のための文字列です。利用可能な場合は、追加情報が含まれます。
upload_ok	BIT	アップロードが成功した場合は true、それ以外の場合は false。
ignored_rows	BIT	アップロードされたローが無視された場合は true、それ以外の場合は false。
auth_status	UNSIGNED SMALLINT	同期認証ステータス。
auth_value	INTEGER	Mobile Link サーバが auth_status の結果を判断するために使用する値。
auth_info	CHAR(1024)	Mobile Link ユーザ認証スクリプトから返された認証メッセージ。
partial_download_retained	BIT	部分的なダウンロードが保持されたことを通知する値。
timestamp	TIMESTAMP	最後の同期の日付と時刻。
sent_bytes	UNSIGNED INT	アップロードでこれまでに送信されたバイト数。
sent_inserts	UNSIGNED INT	アップロードでこれまでに挿入されたローの数。
sent_updates	UNSIGNED INT	アップロードでこれまでに送信された更新済みローの数。
sent_deletes	UNSIGNED INT	アップロードでこれまでに送信された削除済みローの数。
received_bytes	UNSIGNED INT	ダウンロードでこれまでに送信されたバイト数。
received_inserts	UNSIGNED INT	ダウンロードでこれまでに挿入されたローの数。
received_updates	UNSIGNED INT	ダウンロードでこれまでに送信された更新済みローの数。
received_ignored_updates	UNSIGNED INT	ダウンロードで受信された重複ローの数。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

カラム名	カラム型	説明
received_deletes	UNSIGNED INT	ダウンロードでこれまでに送信された削除済みローの数。
received_ignored_deletes	UNSIGNED INT	すでに削除されているローのダウンロードで受信された削除済みローの数。
received_truncate_deletes	UNSIGNED INT	ダウンロード時にトランケート操作によって削除されたローの数。

関連情報

Keep Partial Download 同期パラメータ [98 ページ]

1.12.6.7 systable システムテーブル

systable システムテーブルの各ローは、データベース内のテーブル 1 つを示します。

カラム名	カラム型	説明
column_count	UNSIGNED INT	テーブル内のカラム数。
index_count	UNSIGNED INT	テーブル内のインデックス数。
ixcol_count	UNSIGNED INT	テーブル内のすべてのインデックスのカラムの合計数。
table_name	VARCHAR(128)	テーブル名。
object_id	UNSIGNED INT	テーブルのユニークな識別子。
sync_type	VARCHAR(32)	Mobile Link による同期に使用されます。*no_sync* (非同期)、*all_sync* (すべてのローの同期)、または *normal_sync* (変更されたローのみ同期) のいずれか。
table_type	VARCHAR(32)	*user* (ユーザが作成したテーブルを示す)。

制約

PRIMARY KEY (object_id)

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.13 Ultra Light SQL リファレンス

Ultra Light は、数多くの SQL 言語機能や要素をサポートしています。

このセクションの内容:

Ultra Light SQL 言語の要素 [247 ページ]
Ultra Light は、数多くの SQL 要素をサポートしています。

SQL データ型 [278 ページ]
Ultra Light は、SQL Anywhere で使用できるデータ型のサブセットをサポートします。

空間データ型 [304 ページ]
空間データは、定義された空間内のオブジェクトの位置、形、および方向を記述するデータです。Ultra Light には、ポイント形式の空間データ用の記憶領域およびデータ管理機能が用意されており、これらの機能を使用して、地理的ロケーションに関する情報やルート指定情報などを格納できます。

ユーザ定義データ型とそれに相当する型 [306 ページ]
SQL Anywhere データベースとは異なり、Ultra Light ではユーザ定義データ型はサポートされていません。

SQL 関数 [307 ページ]
関数は、データベースの情報を返すために使用します。関数は、式を使用できるところではどこからでも呼び出すことができます。

Ultra Light SQL 文 [475 ページ]
Ultra Light SQL でサポートされている SQL 文は、SQL Anywhere データベースでサポートされている文のサブセットです。

1.13.1 Ultra Light SQL 言語の要素

Ultra Light は、数多くの SQL 要素をサポートしています。

このセクションの内容:

Ultra Light のキーワード [248 ページ]
各 SQL 文には 1 つまたは複数のキーワードが含まれています。SQL 文のキーワードでは大文字と小文字を区別しませんが、このマニュアルではキーワードを大文字で表記します。キーワードの中には、二重引用符で囲まないと識別子として使用できないものがあります。このようなキーワードを予約語と呼びます。

Ultra Light の識別子 [248 ページ]
識別子は、ユーザ ID、テーブル、カラムなど、データベースのオブジェクト名を表します。

Ultra Light の文字列 [249 ページ]
文字列を使用して、文字データをデータベースに格納します。Ultra Light は、SQL Anywhere と同じ文字列の規則をサポートしています。

Ultra Light のコメント [249 ページ]
コメントは、SQL 文または文ブロックに説明テキストを付加するために使用します。Ultra Light ランタイムは、コメントを実行しません。

Ultra Light の数値 [250 ページ]

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

数値を使用して、数値データをデータベースに格納します。

Ultra Light の NULL 値 [251 ページ]
SQL Anywhere と同様に、NULL はすべてのデータ型のすべての有効な値とは異なる特殊な値です。ただし、NULL 値はすべてのデータ型で使用できます。NULL は、情報が不明 (値なし) であるか、該当しないことを表すために使用します。

Ultra Light の特別値 [251 ページ]
特別値は、式や、テーブル作成時のカラムのデフォルトに使用できます。

Ultra Light の日付と時刻 [255 ページ]
日付と時刻の関数の多くは、日付と時刻の単位で構成される日付を使用します。Ultra Light と SQL Anywhere では同じ日付の単位がサポートされています。

Ultra Light の式 [256 ページ]
式は、多くの場合、カラム参照の形式でデータを関数や演算子と組み合わせることによって形成されます。

Ultra Light の探索条件 [263 ページ]
探索条件は、WHERE 句、HAVING 句、ジョインの ON フレーズ、または IF 式の条件です。探索条件は、述部とも言います。

Ultra Light の演算子 [275 ページ]
演算子は値の計算に使用します。計算された値は、さらに上位の式でオペランドとして使用できます。

Ultra Light の変数 [278 ページ]
Ultra Light アプリケーションでは、SQL 変数 (グローバル変数を含む) は使用できません。

1.13.1.1 Ultra Light のキーワード

各 SQL 文には 1 つまたは複数のキーワードが含まれています。SQL 文のキーワードでは大文字と小文字を区別しませんが、このマニュアルではキーワードを大文字で表記します。キーワードの中には、二重引用符で囲まないと識別子として使用できないものがあります。このようなキーワードを予約語と呼びます。

> **注記**
>
> Ultra Light では、SQL Anywhere キーワードのサブセットだけがサポートされています。ただし、将来のリリースで問題が発生しないように、SQL Anywhere のすべての予約語が Ultra Light の予約語でもあると考えてください。

1.13.1.2 Ultra Light の識別子

識別子は、ユーザ ID、テーブル、カラムなど、データベースのオブジェクト名を表します。

識別子は、最大長が 128 バイトであり、アルファベット文字や数字、アンダースコア文字 (_)、アットマーク (@) で構成されます。先頭に数字を使用することも可能ですが、識別子を引用符で囲む必要があります。他の特殊文字を使用することも可能ですが、識別子を引用符で囲む必要があります。データベースの照合順は、どの文字がアルファベットまたは数字として扱われるかを指定します。

識別子に二重引用符 (") を使用することはできません。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

引用符付き識別子

次のいずれかの条件と一致した場合は、識別子を二重引用符で囲みます。

- 識別子が、前後のスペースや埋め込みスペースを含んでいる。
- 識別子の最初の文字を、アルファベット文字、アンダースコア (_) 、またはアットマーク (@) にすることはできません。たとえば、最初の文字を数字にします。
- 識別子が、アルファベット文字、数字、アンダースコア (_) 、アットマーク (@) 以外の文字を含んでいる。
- 識別子が予約語である。

他のデータベース管理システムとの互換性を確保するため、識別子名に下記をはじめとする特殊文字を使用しないことを推奨します。

- 先頭または末尾のスペース
- 先頭の一重引用符
- セミコロン

1.13.1.3 Ultra Light の文字列

文字列を使用して、文字データをデータベースに格納します。Ultra Light は、SQL Anywhere と同じ文字列の規則をサポートしています。

文字列の比較結果や文字列のソート順は、データベース、文字セット、照合順における大文字と小文字の区別によって決まります。これらのプロパティは、データベースの作成時に設定されます。

関連情報

Ultra Light 文字セット [32 ページ]

1.13.1.4 Ultra Light のコメント

コメントは、SQL 文または文ブロックに説明テキストを付加するために使用します。Ultra Light ランタイムは、コメントを実行しません。

Ultra Light では次のコメントインジケータを使用できます。

-- (二重ハイフン)

データベースサーバは、この行の残りの文字を無視します。このインジケータは、ANSI/ISO SQL 標準のコメントインジケータです。

// (二重スラッシュ)

二重スラッシュは、二重ハイフンと同じ意味です。

/* ...*/ (スラッシュ - アスタリスク)

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

2 つのコメントマーカの間にある文字は、すべて無視されます。2 つのコメントマーカは、同じ行にあっても、別の行にあってもかまいません。このスタイルで示されたコメントは、ネストできます。このスタイルは、C スタイルコメントとも呼ばれます。

> **注記**
>
> パーセント記号 (%) は、Ultra Light ではサポートされていません。

例

- 次に、二重ハイフンのコメントの使用例を示します。

```
CREATE TABLE borrowed_book (
   loaner_name CHAR(100)      PRIMARY KEY,
   date_borrowed DATE NOT NULL DEFAULT CURRENT DATE,
   date_returned       DATE,
   book                CHAR(20)
   FOREIGN KEY book REFERENCES library_books (isbn),
)
--This statement creates a table for a library database to hold information on
borrowed books.
--The default value for date_borrowed indicates that the book is borrowed on
the day the entry is made.
--The date_returned column is NULL until the book is returned.
```

- 次に、C スタイルコメントの使用例を示します。

```
CREATE TABLE borrowed_book (
   loaner_name CHAR(100)      PRIMARY KEY,
   date_borrowed DATE NOT NULL DEFAULT CURRENT DATE,
   date_returned       DATE,
   book                CHAR(20)
   FOREIGN KEY book REFERENCES library_books (isbn),
)
/* This statement creates a table for a library database to hold information
on borrowed books.
The default value for date_borrowed indicates that the book is borrowed on the
day the entry is made.
The date_returned column is NULL until the book is returned. */
```

1.13.1.5 Ultra Light の数値

数値を使用して、数値データをデータベースに格納します。

数値は次のようなデータです。

- 任意の数字の並び
- 小数部が続く
- オプションでマイナス記号 (-) またはプラス記号 (+) を含む
- E の後に指数値が続く

たとえば、次に示す数値はすべて Ultra Light でサポートされています。

42

-4.038

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

.001

3.4e10

1e-10

1.13.1.6 Ultra Light の NULL 値

SQL Anywhere と同様に、NULL はすべてのデータ型のすべての有効な値とは異なる特殊な値です。ただし、NULL 値はすべてのデータ型で使用できます。NULL は、情報が不明 (値なし) であるか、該当しないことを表すために使用します。

1.13.1.7 Ultra Light の特別値

特別値は、式や、テーブル作成時のカラムのデフォルトに使用できます。

このセクションの内容:

CURRENT DATE 特別値 - Ultra Light [252 ページ]
現在の年、月、日を返します。

CURRENT TIME 特別値 - Ultra Light [252 ページ]
現在の時、分、秒 (小数位あり) で構成される時刻を返します。

CURRENT TIMESTAMP 特別値 - Ultra Light [253 ページ]
CURRENT DATE と CURRENT TIME を結合して、TIMESTAMP 値を形成します。年、月、日、時、分、秒、秒の小数位で構成されます。

CURRENT UTC TIMESTAMP 特別値 - Ultra Light [254 ページ]
年、月、日で構成される現在の UTC 時間を示す TIMESTAMP WITH TIME ZONE 値を返します。

SQLCODE 特別値 [255 ページ]
SQLCODE 特別値が評価された時点の値です。

関連情報

SQL データ型 [278 ページ]
SQL 関数 [307 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.13.1.7.1 CURRENT DATE 特別値 - Ultra Light

現在の年、月、日を返します。

データ型

DATE

備考

返される日付は、SQL 文が Ultra Light ランタイムで実行されたときのシステムクロックの読み取り値が基準になります。CURRENT DATE を次のいずれかの状態で使用すると、すべての値は、個別のクロック読み取り値が基準になります。

- 同一文で CURRENT DATE を複数回使用
- 同一文で CURRENT DATE を CURRENT TIME または CURRENT TIMESTAMP と組み合わせて使用
- 同一文で CURRENT DATE を NOW 関数または GETDATE 関数と組み合わせて使用

関連情報

Ultra Light の式 [256 ページ]
GETDATE 関数 [日付と時刻] [377 ページ]
NOW 関数 [日付と時刻] [415 ページ]

1.13.1.7.2 CURRENT TIME 特別値 - Ultra Light

現在の時、分、秒 (小数位あり) で構成される時刻を返します。

データ型

TIME

備考

秒は小数第 6 位まで格納されます。現在時刻の精度はシステムクロックの精度によって制限されます。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

返される日付は、SQL 文が Ultra Light ランタイムで実行されたときのシステムクロックの読み取り値が基準になります。CURRENT TIME を次のいずれかと組み合わせて使用すると、すべての値は、個別のクロック読み取り値が基準になります。

- 同一文で CURRENT TIME を複数回使用
- 同一文で CURRENT TIME を CURRENT DATE または CURRENT TIMESTAMP と組み合わせて使用
- 同一文で CURRENT TIME を NOW 関数または GETDATE 関数と組み合わせて使用

関連情報

Ultra Light の式 [256 ページ]
GETDATE 関数 [日付と時刻] [377 ページ]
NOW 関数 [日付と時刻] [415 ページ]

1.13.1.7.3 CURRENT TIMESTAMP 特別値 - Ultra Light

CURRENT DATE と CURRENT TIME を結合して、TIMESTAMP 値を形成します。年、月、日、時、分、秒、秒の小数位で構成されます。

データ型

TIMESTAMP

備考

秒は小数第 3 位まで格納されます。精度はシステムクロックの精度によって制限されます。

DEFAULT CURRENT TIMESTAMP で宣言されたカラムには、ユニークな値が入るとはかぎりません。

CURRENT TIMESTAMP が返す情報は、GETDATE 関数と NOW 関数が返す情報と同じです。

CURRENT_TIMESTAMP は、CURRENT TIMESTAMP と同じです。

返される日付は、SQL 文が Ultra Light ランタイムで実行されたときのシステムクロックの読み取り値が基準になります。CURRENT TIMESTAMP を次のいずれかと組み合わせて使用すると、すべての値は、個別のクロック読み取り値が基準になります。

- 同一文で CURRENT TIMESTAMP を複数回使用
- 同一文で CURRENT TIMESTAMP を CURRENT DATE または CURRENT TIME と組み合わせて使用
- 同一文で CURRENT TIMESTAMP を NOW 関数または GETDATE 関数と組み合わせて使用

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

関連情報

CURRENT TIME 特別値 - Ultra Light [252 ページ]
Ultra Light の式 [256 ページ]
NOW 関数 [日付と時刻] [415 ページ]
GETDATE 関数 [日付と時刻] [377 ページ]

1.13.1.7.4 CURRENT UTC TIMESTAMP 特別値 - Ultra Light

年、月、日で構成される現在の UTC 時間を示す TIMESTAMP WITH TIME ZONE 値を返します。

データ型

DATE

備考

返される日付は、SQL 文が Ultra Light ランタイムで実行されたときのシステムクロックの読み取り値が基準になります。

- 同一文で CURRENT DATE を複数回使用
- 同一文で CURRENT DATE を CURRENT TIME または CURRENT TIMESTAMP と組み合わせて使用
- 同一文で CURRENT DATE を NOW 関数または GETDATE 関数と組み合わせて使用

関連情報

CURRENT TIMESTAMP 特別値 - Ultra Light [253 ページ]
Ultra Light の式 [256 ページ]
GETDATE 関数 [日付と時刻] [377 ページ]
NOW 関数 [日付と時刻] [415 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.13.1.7.5 SQLCODE 特別値

SQLCODE 特別値が評価された時点の値です。

データ型

文字列

備考

SQLCODE 値は各文の後に設定されます。SQLCODE をチェックして、文の実行が成功したかどうかを確認できます。

例

SELECT 文を使用して、結果セットから新しいローをフェッチしようとするたびにエラーコードを生成します。次に例を示します。SELECT a, b, SQLCODE FROM MyTable.

関連情報

Ultra Light の式 [256 ページ]

1.13.1.8 Ultra Light の日付と時刻

日付と時刻の関数の多くは、日付と時刻の単位で構成される日付を使用します。Ultra Light と SQL Anywhere では同じ日付の単位がサポートされています。

関連情報

日付の単位の指定 [312 ページ]

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

1.13.1.9 Ultra Light の式

式は、多くの場合、カラム参照の形式でデータを関数や演算子と組み合わせることによって形成されます。

構文

```
expression:
 case-expression
| constant
| [correlation-name.]column-name
| - expression
| expression operator expression
| ( expression )
| function-name ( expression, ... )
| if-expression
| special value
| input-parameter
```

パラメータ

```
case-expression:
CASE expression
WHEN expression
THEN expression,...
[ ELSE expression ]
END
```

```
alternative form of case-expression:
CASE
WHEN search-condition
THEN expression,...
[ ELSE expression ]
END
```

```
constant:
 integer  | number  | string | host-variable
```

```
special-value:
 CURRENT { DATE  |  TIME  |  TIMESTAMP  }
| NULL
| SQLCODE
| SQLSTATE
```

```
if-expression:
IF condition
THEN expression
[ ELSE expression ]
ENDIF
```

```
input-parameter:
{ ? | :name [ : indicator-name ] }
```

```
operator:
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

```
{ + | - | * | / | || | % }
```

このセクションの内容:

式内の定数 [257 ページ]
Ultra Light における定数とは、数値または文字列リテラルです。

式内のカラム名 - Ultra Light [258 ページ]
式内の識別子です。

IF 式 - Ultra Light [258 ページ]
データの特定サブセットを返す探索条件を設定します。

CASE 式 - Ultra Light [259 ページ]
SQL の条件式です。

集合式 - Ultra Light [260 ページ]
Ultra Light ランタイムでは提供されない集計の計算を実行します。

式のサブクエリ - Ultra Light [261 ページ]
別の SELECT 文の内部でネストされた SELECT 文です。

入力パラメータ [262 ページ]
エンドユーザが準備文に値を指定できるようにするためのプレースホルダとして機能します。文は、ユーザが指定する値を使用して実行されます。

関連情報

SQL データ型 [278 ページ]
SQL 関数 [307 ページ]

1.13.1.9.1 式内の定数

Ultra Light における定数とは、数値または文字列リテラルです。

構文

```
' constant '
```

使用法

文字列定数は、一重引用符 (') で囲まれています。

文字列内にアポストロフィを表すには、一重引用符を 2 つ続けて使用します ('')。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

例

所有を表すフレーズを使用するには、次のような文字列リテラルを入力します。

'John''s database'

1.13.1.9.2 式内のカラム名 - Ultra Light

式内の識別子です。

構文

```
correlation-name.column-name
```

備考

カラム名は、オプションの相関名 (通常はテーブルの名前) の後に続きます。

カラム名がキーワードであるか、カラム名に英字、数字、アンダースコア以外の文字が使用されている場合は、二重引用符 ("") で囲んでください。次の例は、有効なカラム名です。

```
Employees.Name
address
"date hired"
"salary"."date paid"
```

関連情報

FROM 句 [Ultra Light] [509 ページ]

1.13.1.9.3 IF 式 - Ultra Light

データの特定サブセットを返す探索条件を設定します。

構文

```
IF search-condition
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

```
THEN expression1
[ ELSE expression2 ]
ENDIF
```

備考

互換性を保つために、この式は ENDIF または END IF で終了できます。

この式は次のように返します。

- search-condition が TRUE の場合、IF 式は expression1 を返します。
- search-condition が FALSE で、ELSE 句が指定されている場合、IF 式は expression2 を返します。
- search-condition が FALSE で expression2 がない場合、IF 式は NULL を返します。
- search-condition が UNKNOWN の場合、IF 式は NULL を返します。

1.13.1.9.4 CASE 式 - Ultra Light

SQL の条件式です。

構文 1

```
CASE  expression1
WHEN expression2 THEN expression3, ...
[ ELSE expression4 ]
END
```

```
SELECT ID,
   ( CASE name
      WHEN 'Tee Shirt' THEN 'Shirt'
      WHEN 'Sweatshirt' THEN 'Shirt'
      WHEN 'Baseball Cap' THEN 'Hat'
      ELSE 'Unknown'
   END ) as Type
FROM Product
```

構文 2

```
CASE
WHEN search-condition
THEN expression1, ...
[ ELSE expression2 ]
END
```


© 2016 SAP SE or an SAP affiliate company. All rights reserved.

備考

互換性を保つために、この式は ENDCASE または END CASE で終了できます。

CASE 式は、正規表現が使用できればどこでも使用できます。

構文 1

CASE キーワードに続く式が最初の WHEN キーワードに続く式と等しい場合、対応する THEN キーワードの後の式が返されます。それ以外の場合、ELSE キーワードがあればそれに続く式が返されます。

たとえば、下記のコードでは CASE 式が SELECT 文の 2 番目の句として使用されています。この式によって、name カラムの値が Sweatshirt であるローが Product テーブルから選択されます。

構文 2

最初の WHEN キーワードに続く search-condition が TRUE の場合、対応する THEN キーワードに続く式が返されます。それ以外の場合、ELSE 句があればそれに続く式が返されます。

省略形 CASE 式の NULLIF 関数

NULLIF 関数は、CASE 文を省略形で記述する方法の 1 つです。NULLIF の構文は、次のとおりです。

```
NULLIF ( expression-1, expression-2 )
```

NULLIF は 2 つの式の値を比較します。1 番目の式と 2 番目の式が等しい場合、NULLIF は NULL を返します。1 番目の式と 2 番目の式が異なる場合、NULLIF は 1 番目の式を返します。

例

次の文では、CASE 式が SELECT 文の 3 番目の句として使用され、探索条件と文字列を関連付けています。name カラムの値が **Tee Shirt** の場合は、このクエリによって **Sale** が返されます。name カラムの値が **Tee Shirt** ではなく、quantity が 50 より大きい場合は、**Big Sale** が返されます。ただし、それ以外の場合は、このクエリによって **Regular price** が返されます。

```
SELECT ID, name,
   ( CASE
      WHEN name='Tee Shirt' THEN 'Sale'
      WHEN quantity >= 50  THEN 'Big Sale'
      ELSE 'Regular price'
   END ) as Type
FROM Product
```

1.13.1.9.5 集合式 - Ultra Light

Ultra Light ランタイムでは提供されない集計の計算を実行します。

構文

```
SUM( expression )
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

備考

集合式は、一連のローから単一の値を計算します。

集合式は、1 つの集合関数が使用される式か、1 つ以上のオペランドがある式です。

SELECT 文に GROUP BY 句がない場合、SELECT リストの式にすべての集合式が含まれているか、または集合式がまったく含まれていない必要があります。SELECT 文に GROUP BY 句がある場合、SELECT リストの非集合式を GROUP BY リストに記載します。

例

たとえば、次のクエリは、従業員テーブル内の従業員の給与合計を計算します。このクエリでは、`SUM( salary )` が集合式です。

```
SELECT SUM( salary )
FROM employee
```

1.13.1.9.6 式のサブクエリ - Ultra Light

別の SELECT 文の内部でネストされた SELECT 文です。

構文

サブクエリは、通常のクエリのように構成されます。

備考

Ultra Light のサブクエリの参照は、次の場合にのみ使用できます。

- FROM 句内のテーブル式として。この形式のテーブル式 (抽出テーブル) には、SELECT リスト内の値をフェッチする抽出テーブル名とカラム名が必要です。
- EXISTS、ANY、ALL、IN の各探索条件に値を指定するため。

サブクエリの前 (左) に指定されている名前に関するサブクエリを作成できます。これは、左側への外部参照とも呼ばれます。サブクエリ内の項目は参照できません。これは、内部参照とも呼ばれます。

例

次のサブクエリは、在庫数の少ない項目のすべての製品 ID をリストするのに使用します。

```
FROM SalesOrderItems
 ( SELECT ID
   FROM Products
   WHERE Quantity < 20 )
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

関連情報

SELECT 文 [Ultra Light] [516 ページ]
Ultra Light の探索条件 [263 ページ]

1.13.1.9.7 入力パラメータ

エンドユーザが準備文に値を指定できるようにするためのプレースホルダとして機能します。文は、ユーザが指定する値を使用して実行されます。

構文

```
{ ? | :name [ : indicator-name ]  }
```

備考

式でプレースホルダ文字 ? または名前形式を使用します。入力パラメータは、カラム名または定数を使用できる任意の場所で使用できます。

文に値が渡される具体的なメカニズムは、Ultra Light クライアントの作成に使用する API によって異なります。

名前形式の使用

入力パラメータの名前形式には特別な意味があります。一般に、常に `name` を使用して、実際の値を渡す複数のロケーションを指定します。

Embedded SQL アプリケーションの場合にのみ、`indicator-name` が、NULL インジケータが配置される変数を指定します。名前形式と他のコンポーネントを同時に使用した場合、`indicator-name` は無視されます。

データ型の抽出

入力パラメータのデータ型は、文が次のいずれかのパターンから準備されるときに抽出されます。

- *CAST* (*?AS*`type`)
  ここで、`type` は CHAR(32) などのデータベースの型指定です。
- バイナリ演算子の 1 つのオペランドが入力パラメータです。型が抽出され、オペランドの型になります。

Ultra Light で型を抽出できなかった場合は、エラーが発生します。次に例を示します。

- *-?*: オペランドが単項です。
- *? + ?*: 両方共入力パラメータです。

例

次の Embedded SQL 文には 2 つの入力パラメータがあります。

```
INSERT INTO MyTable VALUES ( :v1, :v2, :v1)
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

v1 の最初のインスタンスが、文内の v2 と v1 の両方のロケーションにその値を渡します。

関連情報

ホスト変数 [642 ページ]
Ultra Light C++ での INSERT、UPDATE、DELETE を使用したデータ修正 [610 ページ]
Ultra Light.NET での INSERT、UPDATE、DELETE を使用したデータ修正 [563 ページ]

1.13.1.10 Ultra Light の探索条件

探索条件は、WHERE 句、HAVING 句、ジョインの ON フレーズ、または IF 式の条件です。探索条件は、述部とも言います。

構文

```
search-condition:
expression comparison-operator expression
| expression IS [ NOT ] NULL
| expression [ NOT ] BETWEEN expression AND expression
| expression [ NOT ] IN ( expression, ... )
| expression [ NOT ] IN ( subquery )
| expression [ NOT ] { ANY | ALL } ( subquery )
| expression [ NOT ] EXISTS ( subquery )
| expression [ NOT ] LIKE ( pattern )
| NOT search-condition
| search-condition AND search-condition
| search-condition OR search-condition
| ( search-condition IS [ NOT ] {TRUE | FALSE | UNKNOWN }
```

```
comparison-operator :
 =
| >
| <
| >=
| <=
| <>
| !=
| !<
| !>
```

パラメータ

Ultra Light では、次のタイプの探索条件がサポートされています。

- ALL 条件
- ANY 条件

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

- BETWEEN 条件
- EXISTS 条件
- IN 条件
- LIKE 探索条件

備考

Ultra Light では、次の箇所で探索条件を指定します。

- WHERE 句
- HAVING 句
- ON フレーズ
- SQL クエリ

探索条件は、SELECT 文の FROM 句で使用してテーブル内のローのサブセットを選択するか、IF や CASE などの式で使用して特定の値を選択できます。Ultra Light では、すべての条件が TRUE、FALSE、または UNKNOWN のいずれかに評価されます。これらを組み合わせた場合、3 値的論理と呼ばれます。比較される値のいずれかが NULL の場合、比較結果は UNKNOWN になります。探索条件は、条件の結果が TRUE の場合にのみ満たされます

このセクションの内容:

比較演算子 - Ultra Light [265 ページ]
探索条件で、複数の式を比較できる演算子です。

論理演算子 - Ultra Light [266 ページ]
論理演算子は、検索条件として使用したり (AND、OR、NOT など)、IS 式の真または NULL 値をテストしたりできます。

ALL 検索条件 - Ultra Light [268 ページ]
ALL 条件を比較演算子と組み合せて使用すると、1 つの値を、サブクエリが生成するデータ値と比較できます。

ANY 検索条件 - Ultra Light [269 ページ]
ANY 条件を比較演算子と組み合せて使用すると、1 つの値を、サブクエリが生成するデータ値のカラムと比較できます。

BETWEEN 検索条件 - Ultra Light [270 ページ]
包括的範囲を指定します。包括的範囲には、その範囲の間の値だけではなく、上限値と下限値も含まれます。

EXISTS 検索条件 - Ultra Light [270 ページ]
サブクエリがクエリ結果のローを生成するかどうかを調べます。

IN 検索条件 - Ultra Light [271 ページ]
メインクエリの値からサブクエリの別の値を探索することでメンバーシップをチェックします。

LIKE 検索条件 - Ultra Light [272 ページ]
式の中にパターンが見つかるかどうかを確認します。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

関連情報

SQL データ型 [278 ページ]
SQL 関数 [307 ページ]
式のサブクエリ - Ultra Light [261 ページ]

1.13.1.10.1 比較演算子 - Ultra Light

探索条件で、複数の式を比較できる演算子です。

構文

```
expression operator expression
```

パラメータ

演算子	意味
=	等価
>	より大きい
<	より小さい
>=	大きいかまたは等価
<=	小さいかまたは等価
!=	等価ではない
<>	等価ではない
!>	以下
!<	以上

備考

日付の比較

日付を比較するときには、< はより古いことを意味し、> はより新しいことを意味します。

LONG VARCHAR 値や LONG BINARY 値の比較

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

Ultra Light は、LONG VARCHAR 値や LONG BINARY 値を使用した比較をサポートしていません。

大文字と小文字の区別

Ultra Light では、比較処理は、該当のデータベースの文字の区別に合わせて実行されます。デフォルトでは、Ultra Light データベースは大文字と小文字を区別しないで作成されます。

NOT 演算子

NOT 演算子は式を否定します。

例

次の 2 つのクエリはどちらも、単価が $10 以下の T シャツ (Tee Shirt) と野球帽 (BaseBall Cap) をすべて検索します。ただし、否定の論理演算子 (NOT) と否定の比較演算子 (!>) では、位置が異なることに注意してください。

```
SELECT ID, Name, Quantity
FROM Products
WHERE (name = 'Tee Shirt' OR name = 'BaseBall Cap')
AND NOT UnitPrice > 10
```

```
SELECT ID, Name, Quantity
FROM Products
WHERE (name = 'Tee Shirt' OR name = 'BaseBall Cap')
AND UnitPrice !> 10
```

関連情報

論理演算子 - Ultra Light [266 ページ]

1.13.1.10.2 論理演算子 - Ultra Light

論理演算子は、検索条件として使用したり (AND、OR、NOT など)、IS 式の真または NULL 値をテストしたりできます。

構文 1

```
condition1 logical-operator condition2
```

構文 2

```
NOT condition
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

構文 3

```
expression IS [ NOT ] { truth-value | NULL }
```

備考

探索条件は、SELECT 文の FROM 句で使用してテーブル内のローのサブセットを選択するか、IF や CASE などの式で使用して特定の値を選択できます。Ultra Light では、すべての条件が TRUE、FALSE、または UNKNOWN のいずれかに評価されます。これらを組み合わせた場合、3 値的論理と呼ばれます。比較される値のいずれかが NULL の場合、比較結果は UNKNOWN になります。探索条件は、条件の結果が TRUE の場合にのみ満たされます

AND

両方の条件が TRUE の場合、結合した条件は TRUE になります。条件のいずれかが FALSE の場合は FALSE、それ以外の場合は UNKNOWN になります。

```
condition1 AND condition2
```

OR

条件のいずれかが TRUE の場合、結合した条件は TRUE になります。両方の条件が FALSE の場合は FALSE、それ以外の場合は UNKNOWN になります。

NOT

`condition` が FALSE の場合、NOT 条件は TRUE です。`condition` が TRUE の場合は FALSE、`condition` が UNKNOWN の場合は UNKNOWN になります。

IS

`expression` が指定の `truth-value` (TRUE、FALSE、または UNKNOWN のいずれか) と評価されれば条件は TRUE になります。それ以外の場合、値は FALSE です。

例

IS NULL 条件は、カラムに NULL 値が含まれる場合に満たされます。IS NOT NULL 演算子を使用すると、この条件は、カラムに NULL でない値が含まれる場合に満たされます。たとえば、WHERE paid_date IS NULL は IS NULL 条件です。

関連情報

比較演算子 - Ultra Light [265 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.13.1.10.3 ALL 検索条件 - Ultra Light

ALL 条件を比較演算子と組み合せて使用すると、1 つの値を、サブクエリが生成するデータ値と比較できます。

構文

```
expression compare   [ NOT ] ALL ( subquery )
```

パラメータ

```
compare:
= | > | < | >= | <= | <> | != | !< | !>
```

備考

Ultra Light は指定された比較演算子を使用して、テスト値を結果セットのデータ値と比較します。すべての比較の結果が TRUE になる場合、ALL テストは TRUE を返します。

例

注文番号 2001 のすべての製品が出荷された後に受けた注文の注文 ID と顧客 ID を検索します。

```
SELECT ID, CustomerID
FROM SalesOrders
WHERE OrderDate > ALL (
 SELECT ShipDate
 FROM SalesOrderItems
 WHERE ID=2001)
```

関連情報

比較演算子 - Ultra Light [265 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.13.1.10.4 ANY 検索条件 - Ultra Light

ANY 条件を比較演算子と組み合せて使用すると、1 つの値を、サブクエリが生成するデータ値のカラムと比較できます。

構文 1

```
expression compare  [ NOT ] ANY ( subquery )
```

構文 2

```
expression = ANY ( subquery )
```

パラメータ

```
compare:
= | > | < | >= | <= | <> | != | !< | !>
```

備考

Ultra Light は指定された比較演算子を使用して、テスト値をカラムのデータ値と比較します。いずれかの比較の結果が TRUE になる場合、ANY テストは TRUE を返します。

構文 1

`expression` がサブクエリ結果のいずれかの値と等しい場合は TRUE、expression が NULL ではなく、サブクエリから返されたいずれの値とも等しくない場合は FALSE です。`expression` が NULL 値の場合、サブクエリ結果にローがあれば ANY 条件は UNKNOWN です。サブクエリ結果にローがなければ、条件は必ず FALSE になります。

例

注文番号 2005 の最初の製品が出荷された後に受けた注文の注文 ID と顧客 ID を検索します。

```
SELECT ID, CustomerID
FROM SalesOrders
WHERE OrderDate > ANY (
 SELECT ShipDate
 FROM SalesOrderItems
 WHERE ID=2005)
```


© 2016 SAP SE or an SAP affiliate company. All rights reserved.

関連情報

比較演算子 - Ultra Light [265 ページ]

1.13.1.10.5 BETWEEN 検索条件 - Ultra Light

包括的範囲を指定します。包括的範囲には、その範囲の間の値だけではなく、上限値と下限値も含まれます。

構文

```
expression [ NOT ] BETWEEN start-expression AND end-expression
```

備考

BETWEEN 条件は TRUE、FALSE、または UNKNOWN として評価できます。NOT キーワードがない場合は、`expression` が `start-expression` と `end-expression` との間にあれば、条件は TRUE と評価されます。NOT キーワードを使用すると条件の意味が逆になりますが、UNKNOWN は変わりません。

BETWEEN 条件は、次の 2 つの不等式の組み合わせと等価です。

```
[ NOT ] ( expression >= start-expression
                AND expression <= end-expression )
```

例

$10 未満か、$15 を超える製品をすべてリストします。

```
SELECT Name, UnitPrice
FROM Products
WHERE UnitPrice NOT BETWEEN 10 AND 15
```

1.13.1.10.6 EXISTS 検索条件 - Ultra Light

サブクエリがクエリ結果のローを生成するかどうかを調べます。

構文

```
[ NOT ] EXISTS ( subquery )
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

備考

EXISTS 条件は、サブクエリ結果にローが少なくとも 1 つあれば TRUE で、ローがなければ FALSE です。EXISTS 条件は、UNKNOWN にはなりません。

EXISTS 条件の論理は、NOT EXISTS というフォームで否定できます。この場合、テストはサブクエリがローを返さない場合に TRUE を、ローを返す場合に FALSE を返します。

例

2001 年 7 月 13 日より後に注文した顧客をリストします。

```
SELECT GivenName, Surname
FROM Customers
WHERE EXISTS (
 SELECT *
 FROM SalesOrders
 WHERE (OrderDate > '2001-07-13') AND
       (Customers.ID = SalesOrders.CustomerID))
```

1.13.1.10.7 IN 検索条件 - Ultra Light

メインクエリの値からサブクエリの別の値を探索することでメンバーシップをチェックします。

構文

```
expression [ NOT ] IN
{ ( subquery ) | ( value-expr, ... ) }
```

パラメータ

`value-expr` は、単一値をとる式です。これには、文字列、数字、日付、または他の任意の SQL データ型などがあります。

備考

NOT キーワードがない場合、IN 条件は次の規則に従って評価されます。

- `expression` が NULL でなく、少なくとも 1 つの値と等しい場合、TRUE です。
- `expression` が NULL で、値リストが空でない場合、または少なくとも 1 つの値が NULL で、`expression` が他の値のいずれとも等しくない場合、UNKNOWN です。
- `expression` が NULL で、`subquery` が値を返さない場合、または `expression` が NULL でなく、いずれの値も NULL でなく、`expression` がいずれの値とも等しくない場合、FALSE です。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

IN 条件の論理は、NOT IN という形式で否定できます。

検索条件 `expression` *IN* (`values`) は、検索条件 `expression` = *ANY* (`values`) と同じです。検索条件 `expression` *NOTIN* (`values`) は、検索条件 `expression` <> *ALL* (`values`) と同じです。

例

カナダのオンタリオ州、マニトバ州、ケベック州に在住する顧客の会社名と状態を選択します。

```
SELECT CompanyName , Province
FROM Customers
WHERE State IN( 'ON', 'MB', 'PQ')
```

1.13.1.10.8 LIKE 検索条件 - Ultra Light

式の中にパターンが見つかるかどうかを確認します。

構文

LIKE 探索条件の構文は次のとおりです。

```
expression [ NOT ] LIKE pattern
```

パラメータ

expression

検索される文字列。

pattern

`expression` を検索するためのパターン。

備考

LIKE 探索条件は、`expression` を `pattern` に一致させようとし、TRUE、FALSE、または UNKNOWN と評価されます。`expression` が `pattern` に一致すれば、探索条件は TRUE と評価されます (NOT が指定されない場合)。`expression` または `pattern` が NULL 値の場合、探索条件は UNKNOWN と評価されます。

NOT キーワードを使用すると探索条件の意味が逆になりますが、UNKNOWN は変わりません。

`expression` と `pattern` は、CHAR 文字列として解釈されます。`pattern` は、次の表のサポートされているワイルドカードをいくつでも含めることができます。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

ワイルドカード	一致するもの
_ (アンダースコア)	任意の 1 文字。たとえば、a_ は ab と ac に一致しますが、a には一致しません。
% (パーセント記号)	0 個以上の文字からなる任意の文字列。たとえば、bl% は bl と bla に一致します。
[]	指定範囲内、または一連の指定文字の任意の 1 文字。たとえば、T[oi]m は Tom または Tim に一致します。
[^]	指定範囲、または一連の指定文字に含まれない任意の 1 文字。たとえば、M[^c] は Mb と Md に一致しますが、Mc には一致しません。

LIKE 探索条件を使用するさまざまな方法

検索対象	例	その他の情報
一連の文字	LIKE 'sm[iy]th'	検索対象の一連の文字は、角カッコ内にリストして指定します。この例では、探索条件は *smith* と *smyth* に一致します。
範囲内の文字	LIKE '[a-r]ough'	検索対象の文字範囲は、角カッコ内に範囲を書いて指定します。範囲の始めと終わりの間にハイフンを書きます。この例では、探索条件は bough と rough に一致しますが、tough には一致しません。 文字の範囲 [a-z] は "a 以上 z 以下" と解釈され、データベースの照合では大なり演算と小なり演算が実行されます。 範囲は、まず小さい方の値、次に大きい方の値を指定してください。たとえば、[z-a] は何も一致しません。これは、[z-a] の範囲にいずれの文字も一致しないためです。
文字範囲と一連の文字の結合	... LIKE '[a-rt]ough'	角カッコ内で、文字範囲と文字セットを組み合わせて指定できます。この例では、... LIKE '[a-rt]ough' は bough、rough、tough に一致します。 パターン [a-rt] は、a ～ r の範囲に含まれる文字または t と一致する 1 文字と解釈されます。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

検索対象	例	その他の情報
範囲外の 1 文字	`... LIKE '[^a-r]ough'`	脱字記号 (^) は、検索から除外する文字範囲を指定します。この例では、`LIKE '[^a-r]ough'` は文字列 tough に一致しますが、文字列 rough や bough には一致しません。 脱字記号は、この記号以外のカッコ内の内容を否定します。たとえば、角カッコ [^a-rt] は、a ～ r の範囲に含まれない文字かつ t ではない 1 文字と解釈されます。
後続ブランクのある探索パターン	`'90 '`、`'90[ ]'`、および `'90_'`	探索パターンに後続ブランクが含まれる場合、データベースサーバではブランクが含まれる値に対してのみパターンが一致します。文字列へのブランクの埋め込みは行われません。たとえば、検索されている値が幅 3 文字以上の CHAR 型または VARCHAR 型カラムにあっても、'90 '、'90[ ]'、'90_' のパターンは、式 '90 ' には一致しますが、式 '90' には一致しません。

文字範囲と一連の文字の特殊なケース

角カッコ内の任意の 1 文字は、その文字を指しています。たとえば、`[a]` は、文字 a に一致します。`[^]` は脱字記号、`[%]` はパーセント記号 (パーセント記号は、このコンテキストではワイルドカードになりません)、`[_]` はアンダースコア文字に一致します。また、`[[]` は文字 [に一致します。

- パターン `[a-]` は、文字 a または - に一致します。
- パターン `[]` は、一致する文字がないのでローを返すことはありません。
- パターン `[` または `[abp-q` は、閉じカッコが欠けているため構文エラーを返します。
- 角カッコ内ではワイルドカードは使用できません。パターン `[a%b]` は a、%、または b のいずれかを検出します。
- 脱字記号を使用しても、カッコ内の先頭になければ範囲を否定できません。パターン `[a^b]` は a、^、または b のいずれかを検出します。

大文字と小文字の区別と、比較の実行

データベース照合が大文字と小文字を区別する場合は、探索条件でも大文字と小文字を区別します。大文字と小文字を区別する照合で大文字と小文字を区別しない検索を実行するには、大文字と小文字を両方指定します。たとえば、次の探索条件では、文字列 Bough、rough、TOUGH が true と評価されます。

```
LIKE '[a-zA-Z][oO][uU][gG][hH]'
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

例

次の探索条件では、文字 a で始まり、末尾から 2 つ目の文字が b のカラム名のローがすべて TRUE になります。

```
SELECT * FROM table-name WHERE column-name LIKE 'a%b_'
```

1.13.1.11 Ultra Light の演算子

演算子は値の計算に使用します。計算された値は、さらに上位の式でオペランドとして使用できます。

Ultra Light SQL では、次のタイプの演算子がサポートされています。

- 比較演算子は、1 つ (単項) または 2 つ (バイナリ) の比較オペランドを使用して評価し、結果を返します。比較の結果は、通常の 3 つの論理値、true、false、または unknown になります。
- 算術演算子は、浮動小数点、小数、整数の数値を評価し、結果セットを返します。
- 文字列演算子は、2 つの文字列を連結します。たとえば、"my" + "string" は文字列 "mystring" を返します。
- ビット処理演算子は、整数の内部表現内の特定のビットをオンまたはオフにします。
- 論理演算子は、探索条件を評価します。論理的評価の結果は、通常の 3 つの論理値、true、false、または unknown になります。

一般的な演算子の優先度が適用されます。

このセクションの内容:

算術演算子 - Ultra Light [276 ページ]
算術演算子を使用すると、計算を実行できます。

文字列演算子 - Ultra Light [276 ページ]
文字列演算子を使用すると、文字列を結合できます。ただし、LONGVARCHAR データ型と LONGBINARY データ型ではできません。

ビット処理演算子 - Ultra Light [277 ページ]
ビット処理演算子は、2 つの式間でビット操作を実行します。Ultra Light では、次の演算子を整数データ型に対して使用できます。

演算子の優先度 - Ultra Light [277 ページ]
式の中の演算子の順番は、部分式が評価される順番に影響するため、重要です。

関連情報

比較演算子 - Ultra Light [265 ページ]
論理演算子 - Ultra Light [266 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.13.1.11.1 算術演算子 - Ultra Light

算術演算子を使用すると、計算を実行できます。

expression + expression

加算。いずれかの式が NULL 値の場合、結果は NULL 値になります。

expression - expression

減算。いずれかの式が NULL 値の場合、結果は NULL 値になります。

-expression

反転。式が NULL 値の場合、結果は NULL 値になります。

expression * expression

乗算。いずれかの式が NULL 値の場合、結果は NULL 値になります。

expression / expression

除算。いずれかの式が NULL の場合、または 2 番目の式が 0 の場合、結果は NULL になります。

expression % expression

モジュロによる、2 つの整数での除算の余り (整数) の算出。たとえば、21 を 11 で割ると商は 1、余りは 10 なので、21 % 11 = 10 になります。いずれかの式が NULL 値の場合、結果は NULL 値になります。

1.13.1.11.2 文字列演算子 - Ultra Light

文字列演算子を使用すると、文字列を結合できます。ただし、LONGVARCHAR データ型と LONGBINARY データ型ではできません。

expression || expression

文字列連結 (2 つのパイプ記号)。いずれかの文字列が NULL 値の場合、連結には空文字列として扱われます。

expression + expression

代替の文字列連結。+ 連結演算子を使用する場合は、暗黙的データ変換を行わないで、必ずオペランドを文字データ型に明示設定してください。

たとえば、次のクエリは整数値 *579* を返します。

```
SELECT 123 + 456
```

これに対し、次のクエリは文字列 *123456* を返します。

```
SELECT '123' + '456'
```

CAST または CONVERT 関数を使用すると、データ型を明示的に変換できます。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.13.1.11.3 ビット処理演算子 - Ultra Light

ビット処理演算子は、2 つの式間でビット操作を実行します。Ultra Light では、次の演算子を整数データ型に対して使用できます。

演算子	説明
&	bitwise AND
|	bitwise OR
^	bitwise exclusive OR
~	bitwise NOT

ビット処理演算子 &、|、~ は、論理演算子 AND、OR、NOT で代用することはできません。ビット処理演算子は、値のビット表現を使用して整数値に対して作用します。

例

次の文は指定するビットが設定されているローを選択します。

```
SELECT *
FROM tableA
WHERE (options & 0x0101) <> 0
```

1.13.1.11.4 演算子の優先度 - Ultra Light

式の中の演算子の順番は、部分式が評価される順番に影響するため、重要です。

カッコ内の式が最初に評価され、続いて乗算と除算、最後に加算と減算が評価されます。その後、文字列の連結が行われます。リストの最上部にある演算子から順に評価されます。

注記

演算子の順序を明確に指定してください。たとえば、式の中で 1 つ以上の演算子を指定するときは、演算子の優先度に頼るのではなく、カッコを使用してください。

1. 名前、関数、定数、IF 式、CASE 式
2. ()
3. 単項演算子 (1 つのオペランドを必要とする演算子): +、-
4. ~
5. &、|、^
6. *, /, %
7. +, -
8. ||
9. 比較:>、<、<>、!=、<=、>=、[NOT] BETWEEN、[NOT] IN、[NOT] LIKE

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

10. 比較: IS [NOT] TRUE、FALSE、UNKNOWN
11. NOT
12. AND
13. OR

1.13.1.12 Ultra Light の変数

Ultra Light アプリケーションでは、SQL 変数 (グローバル変数を含む) は使用できません。

1.13.2 SQL データ型

Ultra Light は、SQL Anywhere で使用できるデータ型のサブセットをサポートします。

このセクションの内容:

文字データ型 [278 ページ]
文字データ型は、文字、数字、記号などの文字列を格納します。

数値データ型 [283 ページ]
数値データ型は数値データを格納します。

日付と時刻データ型 [293 ページ]
日付の値は 4 桁の西暦で出力でき、日付の内部記憶領域には、年の値の世紀にあたる部分が常に明示的に含まれています。

バイナリデータ型 [300 ページ]
バイナリデータ型は、データベースに認識されないイメージやその他の種類の情報を含めたバイナリデータを格納します。

1.13.2.1 文字データ型

文字データ型は、文字、数字、記号などの文字列を格納します。

何種類かの文字データ型と、それらのデータ型を使用して定義されたいくつかのドメインがあります。

CHAR、VARCHAR、LONG VARCHAR

シングルバイトまたはマルチバイトの文字セットに格納される文字データ。多くの場合、データベースに格納されるプライマリ言語に最も対応するデータ型が選択されます。

NCHAR、NVARCHAR、LONG NVARCHAR

UTF-8 Unicode のエンコーディングに格納される文字データ。データベースに格納されているプライマリ言語に関係なく、これらのデータ型を使用してすべての Unicode コードポイントを格納できます。

TEXT、UNIQUEIDENTIFIERSTR、XML

他の文字データ型に基づくドメイン。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

Ultra Light: Ultra Light は、シングルバイトまたはマルチバイトの文字セットに格納される CHAR、VARCHAR、LONG VARCHAR データ型をサポートしており、多くの場合、データベースに格納されるプライマリ言語に最も対応するデータ型が選択されます。

格納

すべての文字データ値は同じ方法で格納されます。デフォルトでは、1 つの値に最大で 128 バイトを格納できます。値が 128 バイトを超える場合、4 バイトのプレフィクスがデータベースページに格納され、全体の値は他の 1 つまたは複数のデータベースページに格納されます。このデフォルトサイズを制御するには、CREATE TABLE 文の INLINE 句と PREFIX 句を使用します。

Ultra Light: 固定文字型 (VARCHAR など) はローに埋め込まれ、LONG 文字型 (LONG VARCHAR など) は個別に格納されます。大きな固定型のカラム多数で構成されるテーブルを作成するときは、ページサイズを考慮します。1 行全体が 1 ページに収まる必要があります。また、固定文字カラム型は 1 行で格納します。たとえば、ページサイズを 1000 にして作成されたデータベースでは、1000 よりも大きな文字値はページに収まらないため保持できません。

このセクションの内容:

CHAR データ型 [279 ページ]
CHAR データ型は、32767 バイトまでの文字データを格納します。

LONG VARCHAR データ型 [281 ページ]
LONG VARCHAR データ型は、任意の長さの文字データを格納します。

VARCHAR データ型 [282 ページ]
VARCHAR データ型は、32767 バイトまでの文字データを格納します。

関連情報

CREATE TABLE 文 [Ultra Light] [495 ページ]

1.13.2.1.1 CHAR データ型

CHAR データ型は、32767 バイトまでの文字データを格納します。

構文

```
CHAR [ ( max-length [ CHAR | CHARACTER ] ) ]
```

Ultra Light:

```
CHAR [ ( max-length) ]
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

パラメータ

max-length

文字列の最大長。バイト長のセマンティックを使用する場合 (長さの部分に CHAR または CHARACTER を指定しない場合)、長さはバイト単位になります。また、範囲は 1 ～ 32767 にします。長さを指定しない場合、値は 1 になります。

文字長のセマンティックを使用する場合 (長さの部分に CHAR または CHARACTER を指定する場合)、長さは文字単位になります。また、`max-length` を指定します。`max-length` は最大 32767 文字までです。

備考

マルチバイト文字は CHAR 型に格納できますが、文字長のセマンティックを使用していなければ、宣言される長さは文字数ではなくバイト数です。

CHAR は CHARACTER と指定することもできます。どの構文を使用する場合でも、データ型は CHAR と記述されます。

セマンティック上、CHAR は VARCHAR と同じですが、型は異なります。CHAR は可変長型です。他のリレーショナルデータベース管理システムでは、CHAR は固定長型であり、データには `max-length` バイトまでブランクが埋め込まれて格納されます。SQL Anywhere では格納される文字データにブランクを埋め込みません。

カラムがどのように記述されるかは、クライアントインタフェース、使用される文字セット、および文字長セマンティックが使用されるかどうかによって異なります。たとえば、Embedded SQL では、記述される長さはクライアント文字セットの最大バイト数となります。記述される長さが 32767 バイトを超える場合、カラムは DT_LONGVARCHAR 型として記述されます。次の表は、Embedded SQL の例と、DESCRIBE を実行した場合に返される結果を示します。

記述される型	データベースの文字セット	クライアントの文字セット	DESCRIBE の結果
CHAR(10)	Windows-1252	Windows-1252	DT_FIXCHAR 長さ 10
CHAR(10)	UTF-8	UTF-8	DT_FIXCHAR 長さ 10
CHAR(10)	Windows-1252	UTF-8	DT_FIXCHAR 長さ 30
CHAR(20000)	Windows-31J	UTF-8	DT_LONGVARCHAR
CHAR(10 CHAR)	Windows-1252	Windows-1252	DT_FIXCHAR 長さ 10
CHAR(10 CHAR)	UTF-8	UTF-8	DT_FIXCHAR 長さ 40

ODBC では、odbc_distinguish_char_and_varchar オプションに応じて、CHAR は SQL_CHAR または SQL_VARCHAR として記述されます。

Ultra Light の備考

CHAR はドメインであり、VARCHAR として実装されます。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

関連情報

VARCHAR データ型 [282 ページ]

1.13.2.1.2 LONG VARCHAR データ型

LONG VARCHAR データ型は、任意の長さの文字データを格納します。

構文

LONG VARCHAR

備考

最大バイトサイズは 2 GB から 1 バイトを差し引いた値です (2^{31} - 1)。

LONG VARCHAR データに、または LONG VARCHAR データから文字列をキャストできます。LONG VARCHAR データを連結することはできません。

SQL 文の条件 (WHERE 句の条件など) は、LONG VARCHAR カラムでは実行できません。LONG VARCHAR カラムでは、INSERT、UPDATE、DELETE の各操作が許可されます。LONG VARCHAR カラムは、SELECT クエリの結果セットに含めることもできます。

マルチバイト文字は LONG VARCHAR として格納できますが、長さは文字数ではなくバイト数です。

Ultra Light: LONG VARCHAR 型に対してインデックスを作成することはできません。LONG VARCHAR 型は LENGTH 関数と CAST 関数でのみ使用できます。

関連情報

CHAR データ型 [279 ページ]
VARCHAR データ型 [282 ページ]
LENGTH 関数 [文字列] [390 ページ]
CAST 関数 [データ型変換] [339 ページ]

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

1.13.2.1.3 VARCHAR データ型

VARCHAR データ型は、32767 バイトまでの文字データを格納します。

構文

```
VARCHAR [ ( max-length [ CHAR | CHARACTER ] ) ]
```

Ultra Light:

```
VARCHAR [ ( max-length) ]
```

パラメータ

max-length

文字列の最大長。このデフォルト値は 1 です。

バイト長のセマンティックを使用する場合 (長さの部分に CHAR または CHARACTER を指定しない場合)、長さはバイト単位になります。また、範囲は 1 ～ 32767 にします。文字長のセマンティックを使用する場合 (長さの部分に CHAR または CHARACTER を指定する場合)、長さは文字単位になります。また、`max-length` を指定します。`max-length` は最大 32767 文字までです。

Ultra Light: Ultra Light データベースはバイト長のセマンティックのみをサポートしています。英語以外の文字は最大 3 バイトの記憶領域が必要になることがあります。

備考

マルチバイト文字は VARCHAR として格納できますが、宣言される長さは文字数ではなくバイト数です。

Ultra Light: Ultra Light はデータを可能な限りコンパクトにします。VARCHAR 値が `max-length` で指定されたバイト数を必要としない場合は、値を格納するために必要なバイト数のみが使用されます。式を評価するときのテンポラリ文字値の最大長は 2048 バイトです。

警告

Ultra Light:

`max-length` がページサイズを超えている VARCHAR カラムを使用してテーブルを作成することはできますが、そのページサイズを超える長さの値を挿入するとエラーが発生します。

ODBC では、VARCHAR は SQL_VARCHAR と記述されます。VARCHAR は CHAR VARYING または CHARACTER VARYING と指定することもできます。どの構文を使用する場合でも、データ型は VARCHAR と記述されます。セマンティック上、VARCHAR は CHAR と同じですが、型は異なります。SQL Anywhere では、VARCHAR は可変長型です。他のリレーショナルデータベース管理システムでは、VARCHAR は固定長型であり、データには `max-length` バイトまでブランクが埋め

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

込まれて格納されます。SQL Anywhere では格納される文字データにブランクを埋め込みません。VARCHAR カラムがどのように記述されるかは、クライアントインタフェース、使用される文字セット、および文字長セマンティックが使用されるかどうかによって異なります。たとえば、Embedded SQL では、記述される長さはクライアント文字セットの最大バイト数となります。記述される長さが 32767 バイトを超える場合、カラムは DT_LONGVARCHAR 型として記述されます。次の表は、Embedded SQL の例と、DESCRIBE を実行した場合に返される結果を示します。

記述される型	データベースの文字セット	クライアントの文字セット	DESCRIBE の結果
VARCHAR(10)	Windows-1252	Windows-1252	DT_VARCHAR 長さ 10
VARCHAR(10)	UTF-8	UTF-8	DT_VARCHAR 長さ 10
VARCHAR(10)	Windows-1252	UTF-8	DT_VARCHAR 長さ 30
VARCHAR(20000)	Windows-31J	UTF-8	DT_LONGVARCHAR
VARCHAR(10 CHAR)	Windows-1252	Windows-1252	DT_VARCHAR 長さ 10
VARCHAR(10 CHAR)	UTF-8	UTF-8	DT_VARCHAR 長さ 40

関連情報

CHAR データ型 [279 ページ]
LONG VARCHAR データ型 [281 ページ]

1.13.2.2 数値データ型

数値データ型は数値データを格納します。

NUMERIC データ型、DECIMAL データ型、さまざまな INTEGER データ型は真数値データ型と呼ぶこともあります。これと対照的なのが、概数値データ型である FLOAT、DOUBLE、REAL です。

真数値データ型は、精度と小数点以下の桁数の値を指定できる型です。一方、概数値データ型は事前に定義された方法で格納されます。真数値データだけが、算術演算後に指定した最小有効桁数に対して正確性が保証されます。

1 より小さいデータ型の長さと精度は使用できません。

互換性

NUMERIC と DECIMAL のデータ型にデフォルトの精度と位取りの設定を使用する場合は注意が必要です。他のデータベースソリューションでは設定が異なる場合があるためです。デフォルトの精度は 30、デフォルトの位取りは 6 です。

FLOAT (p) データ型は、p の値によって、REAL か DOUBLE のどちらかの同義語になります。SQL Anywhere では、カットオフ値はプラットフォームによって異なりますが、どのプラットフォームでもカットオフ値は 16 以上です。

Transact-SQL の identity カラムでは、scale = 0 の NUMERIC データ型のみ使用できます。NUMERIC データ型と DECIMAL データ型では、デフォルトの精度と位取り設定を使用しないでください。これは、SQL Anywhere と Adaptive

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

Server Enterprise で設定内容が異なるためです。SQL Anywhere では、デフォルトの精度は 30、デフォルトの位取りは 6 です。Adaptive Server Enterprise では、デフォルトの精度は 18、デフォルトの位取りは 0 です。

このセクションの内容:

BIGINT データ型 [284 ページ]
BIGINT データ型は、8 バイトの記憶領域を必要とする整数である BIGINT を格納します。

BIT データ型 [285 ページ]
BIT データ型は、1 つのビット (0 または 1) を格納します。

DECIMAL データ型 [286 ページ]
DECIMAL データ型は、総桁数の `precision` と小数点以下の桁数の `scale` を持つ 10 進数です。

DOUBLE データ型 [287 ページ]
DOUBLE データ型は、倍精度の浮動小数点数を格納します。

FLOAT データ型 [288 ページ]
FLOAT データ型は、単精度または倍精度の浮動小数点数を格納します。

INTEGER データ型 [289 ページ]
INTEGER データ型は、4 バイトの記憶領域を必要とする整数を格納します。

NUMERIC データ型 [290 ページ]
NUMERIC データ型は、総桁数の `precision` と小数点以下の桁数の `scale` を持つ 10 進数を格納します。

REAL データ型 [291 ページ]
REAL データ型は、4 バイトで格納される単精度の浮動小数点数を格納します。

SMALLINT データ型 [292 ページ]
SMALLINT データ型は、2 バイトの記憶領域を必要とする整数を格納します。

TINYINT データ型 [292 ページ]
TINYINT データ型は、1 バイトの記憶領域を必要とする符号なし整数を格納します。

1.13.2.2.1 BIGINT データ型

BIGINT データ型は、8 バイトの記憶領域を必要とする整数である BIGINT を格納します。

構文

```
[ UNSIGNED ] BIGINT
```

備考

BIGINT データ型は真数値データ型です。精度は算術演算の後で保存されます。

BIGINT 値には 8 バイトの記憶領域が必要です。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

BIGINT 値の範囲は、-2^{63} ～ 2^{63} - 1、または -9223372036854775808 ～ 9223372036854775807 です。

UNSIGNED BIGINT 値の範囲は、0 ～ 2^{64} - 1、または 0 ～ 18446744073709551615 です。

デフォルトでは、このデータ型は符号付きです。

文字列を BIGINT に変換すると、前後のスペースは削除されます。先行文字が + の場合は無視されます。先行文字が - の場合は、残りの数字は負の数として解釈されます。先行の 0 文字はスキップされ、残りの文字は整数値に変換されます。変換しようとしたデータ型の有効な範囲外の値である場合、文字列に不正な文字が含まれる場合、または文字列を整数値として復号化できない場合にエラーが返されます。

関連情報

数値関数 [315 ページ]
集合関数 [308 ページ]
BIT データ型 [285 ページ]
VARCHAR データ型 [282 ページ]
SMALLINT データ型 [292 ページ]
TINYINT データ型 [292 ページ]

1.13.2.2.2 BIT データ型

BIT データ型は、1 つのビット (0 または 1) を格納します。

構文

```
BIT
```

備考

BIT は、0 または 1 の値を格納できる整数型です。

デフォルトでは、BIT データ型には NULL を入力できません。

文字列を BIT に変換すると、前後のスペースは削除されます。先行文字が + の場合は無視されます。先行文字が - の場合は、残りの数字は負の数として解釈されます。先行の 0 文字はスキップされ、残りの文字は整数値に変換されます。値が 0 または 1 でない場合はエラーが返されます。

BIT 値には 1 バイトの記憶領域が必要です。

Ultra Light: BIT 値には 1 ビットの記憶領域が必要です。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

関連情報

数値関数 [315 ページ]
集合関数 [308 ページ]
BIGINT データ型 [284 ページ]
VARCHAR データ型 [282 ページ]
SMALLINT データ型 [292 ページ]
TINYINT データ型 [292 ページ]

1.13.2.2.3 DECIMAL データ型

DECIMAL データ型は、総桁数の `precision` と小数点以下の桁数の `scale` を持つ 10 進数です。

構文

```
DECIMAL [ ( precision [ , scale ] ) ]
```

パラメータ

precision

式の桁数を指定する整数式。1 ～ 127 の範囲。デフォルト設定値は 30 です。

scale

小数点以下の桁数を指定する整数式。0 ～ 127 の範囲。scale 値は precision 値以下にする必要があります。デフォルト設定値は 6 です。

デフォルト値はデータベースオプションの設定によって変更します。

Ultra Light: デフォルト値は該当する作成パラメータによって変更します。

備考

DECIMAL データ型は真数値データ型です。精度は、算術演算の後、最小の有効桁数まで保存されます。

10 進数を格納するために必要なバイト数は、次のように計算できます。

```
2 + INT(((precision - scale) + 1) / 2) + INT((scale + 1) / 2);
```

INT 関数は引数の整数部分です。記憶領域の計算の基準は、カラムの中で使われている最大精度と小数点以下の桁数ではなく、格納されている値を使います。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

使用する精度が 20 以下で小数点以下の桁数が 0 の場合、代わりに整数データ型のいずれか (BIGINT、INTEGER、SMALLINT、または TINYINT) を使用できることがあります。整数値では、有効桁数が同じくらいの NUMERIC 値や DECIMAL 値よりも少ない記憶領域が使用されます。整数値の操作 (フェッチ、挿入など) や算術演算子は、通常、NUMERIC 値や DECIMAL 値の操作よりも速く実行されます。

注記

DECIMAL データ型のカラムまたは変数を作成するとき、これらのカラムまたは変数の精度または位取りがデータベースの精度および位取りの設定を超過する場合は、値はデータベースの設定に合わせてトランケートされます。このため、DECIMAL として定義されているカラムまたは変数の値がトランケートされる場合は、精度と位取りがデータベースのオプション設定を超過していないかどうかを確認します。

DECIMAL は DEC と指定することもできます。どの構文を使用する場合でも、データ型は DECIMAL と記述されます。セマンティック上、DECIMAL は NUMERIC と同じです。

関連情報

数値関数 [315 ページ]
集合関数 [308 ページ]
FLOAT データ型 [288 ページ]
REAL データ型 [291 ページ]
DOUBLE データ型 [287 ページ]
NUMERIC データ型 [290 ページ]
Ultra Light precision 作成オプション [162 ページ]
Ultra Light scale 作成オプション [164 ページ]

1.13.2.2.4 DOUBLE データ型

DOUBLE データ型は、倍精度の浮動小数点数を格納します。

構文

```
DOUBLE
```

備考

DOUBLE データ型は概数値データ型であり、算術演算後に丸め誤差が出ます。概数値という DOUBLE 値の性質により、通常、DOUBLE 値を比較するときは等号を使用するクエリを避けてください。

DOUBLE 値には 8 バイトの記憶領域が必要です。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

値の範囲は -1.79769313486231e+308 ～ 1.79769313486231e+308 です。2.22507385850721e-308 がほぼ 0 です。DOUBLE として保持される値の有効桁数は厳密には 15 桁です。15 桁を超えると丸め誤差が出る可能性があります。

関連情報

数値関数 [315 ページ]
集合関数 [308 ページ]
FLOAT データ型 [288 ページ]
REAL データ型 [291 ページ]
DECIMAL データ型 [286 ページ]
NUMERIC データ型 [290 ページ]

1.13.2.2.5 FLOAT データ型

FLOAT データ型は、単精度または倍精度の浮動小数点数を格納します。

構文

```
FLOAT [ ( precision ) ]
```

パラメータ

precision

仮数部 (常用対数の小数部) の桁数を指定する整数式。たとえば、5.63428 という数値の仮数は 0.63428 です。IEEE 標準 754 による浮動小数点精度は、次のとおりです。

提供されている精度値	10 進数精度	等価の SQL データ型	記憶サイズ
1～24	7 桁	REAL	4 バイト
25-53	15 桁	DOUBLE	8 バイト

備考

FLOAT (`precision`) データ型を使用してカラムを作成した場合、すべてのプラットフォーム上でカラムが少なくとも指定の最小精度の値を持つことが保証されます。REAL と DOUBLE の場合、プラットフォームに依存しない最小精度は保証されません。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

precision を指定しない場合、FLOAT データ型は単精度の浮動小数点数です。これは REAL データ型と等価で、4 バイトの記憶領域を必要とします。

precision を指定する場合、FLOAT データ型は指定した精度の値によって、単精度か倍精度のいずれかになります。REAL になるか DOUBLE になるかは、プラットフォームによって異なります。単精度の FLOAT 値は、4 バイトの記憶領域を必要とし、倍精度の FLOAT 値は、8 バイトを必要とします。

FLOAT データ型は概数値データ型です。算術演算後の丸め誤差がでます。概数値という FLOAT 値の性質により、FLOAT 値を比較するときは等号を使用するクエリを避けてください。

1.13.2.2.6 INTEGER データ型

INTEGER データ型は、4 バイトの記憶領域を必要とする整数を格納します。

構文

```
[ UNSIGNED ] INTEGER
```

備考

INTEGER データ型は真数値データ型で、精度は算術演算の後で保存されます。

UNSIGNED を指定した場合、整数に負の数を割り当てることはできません。デフォルトでは、このデータ型は符号付きです。

INTEGER 値の範囲は、-2^{31} ～ $2^{31} - 1$、または -2147483648 ～ 2147483647 です。

UNSIGNED INTEGER 値の範囲は、0 ～ $2^{32} - 1$、または 0 ～ 4294967295 です。

文字列を INTEGER に変換すると、前後のスペースは削除されます。先行文字が + の場合は無視されます。先行文字が - の場合は、残りの数字は負の数として解釈されます。先行の 0 文字はスキップされ、残りの文字は整数値に変換されます。変換しようとしたデータ型の有効な範囲外の値である場合、文字列に不正な文字が含まれる場合、または文字列を整数値として復号化できない場合にエラーが返されます。

関連情報

数値関数 [315 ページ]
集合関数 [308 ページ]
BIGINT データ型 [284 ページ]
BIT データ型 [285 ページ]
SMALLINT データ型 [292 ページ]
TINYINT データ型 [292 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.13.2.2.7 NUMERIC データ型

NUMERIC データ型は、総桁数の `precision` と小数点以下の桁数の `scale` を持つ 10 進数を格納します。

構文

```
NUMERIC [ ( precision [ , scale ] ) ]
```

パラメータ

precision

式の桁数を指定する整数式。1 ～ 127 の範囲。デフォルト設定値は 30 です。

scale

小数点以下の桁数を指定する整数式。0 ～ 127 の範囲。scale 値は precision 値以下にする必要があります。デフォルト設定値は 6 です。

備考

NUMERIC データ型は真数値データ型です。精度は、算術演算の後、最小の有効桁数まで保存されます。

Ultra Light: NUMERIC はドメインであり、DECIMAL として実装されます。

10 進数を格納するために必要なバイト数は、次のように計算できます。

```
2 + INT( (BEFORE+1)/2 ) + INT( (AFTER+1)/2 );
```

INT 関数は引数の整数部分であり、BEFORE と AFTER は小数点の前後の有効桁数です。記憶領域の計算の基準は、カラムの中で使われている最大精度と小数点以下の桁数ではなく、格納されている値を使います。

使用する精度が 20 以下で小数点以下の桁数が 0 の場合、代わりに整数データ型のいずれか (BIGINT、INTEGER、SMALLINT、または TINYINT) を使用できることがあります。整数値では、有効桁数が同じくらいの NUMERIC 値や DECIMAL 値よりも少ない記憶領域が使用されます。整数値の操作 (フェッチ、挿入など) や算術演算子は、通常、NUMERIC 値や DECIMAL 値の操作よりも速く実行されます。

セマンティック上、NUMERIC は DECIMAL と同じです。

注記

NUMERIC データ型のカラムまたは変数を作成するとき、これらのカラムまたは変数の精度または位取りがデータベースの精度および位取りの設定を超過する場合は、値はデータベースの設定に合わせてトランケートされます。このため、NUMERIC として定義されているカラムまたは変数の値がトランケートされる場合は、精度と位取りがデータベースのオプション設定を超過していないかどうかを確認します。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

関連情報

Ultra Light scale 作成オプション [164 ページ]
DECIMAL データ型 [286 ページ]

1.13.2.2.8 REAL データ型

REAL データ型は、4 バイトで格納される単精度の浮動小数点数を格納します。

構文

REAL

備考

REAL データ型は概数値データ型であり、算術演算後に丸め誤差が出ます。概数値という REAL 値の性質により、通常、REAL 値を比較するときは等号を使用するクエリを避けてください。

REAL 値には 4 バイトの記憶領域が必要です。

値の範囲は -3.402823e+38 ～ 3.402823e+38 です。1.175494351e-38 がほぼ 0 です。REAL として保持される値は、厳密には 7 有効桁数ですが、7 桁以上になると丸め誤差が出ます。

関連情報

数値関数 [315 ページ]
集合関数 [308 ページ]
DOUBLE データ型 [287 ページ]
DOUBLE データ型 [287 ページ]
DECIMAL データ型 [286 ページ]
NUMERIC データ型 [290 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.13.2.2.9 SMALLINT データ型

SMALLINT データ型は、2 バイトの記憶領域を必要とする整数を格納します。

構文

```
[ UNSIGNED ] SMALLINT
```

備考

SMALLINT データ型は真数値データ型です。精度は算術演算の後で保存されます。2 バイトの記憶領域を必要とします。

SMALLINT 値の範囲は、-2^{15} ～ $2^{15} - 1$、または -32768 ～ 32767 です。

UNSIGNED SMALLINT 値の範囲は、0 ～ $2^{16} - 1$、または 0 ～ 65535 です。

文字列を SMALLINT に変換すると、前後のスペースは削除されます。先行文字が + の場合は無視されます。先行文字が - の場合は、残りの数字は負の数として解釈されます。先行の 0 文字はスキップされ、残りの文字は整数値に変換されます。変換しようとしたデータ型の有効な範囲外の値である場合、文字列に不正な文字が含まれる場合、または文字列を整数値として復号化できない場合にエラーが返されます。

関連情報

数値関数 [315 ページ]
集合関数 [308 ページ]
BIGINT データ型 [284 ページ]
BIT データ型 [285 ページ]
INTEGER データ型 [289 ページ]
TINYINT データ型 [292 ページ]

1.13.2.2.10 TINYINT データ型

TINYINT データ型は、1 バイトの記憶領域を必要とする符号なし整数を格納します。

構文

```
TINYINT
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

備考

TINYINT データ型は真数値データ型です。精度は算術演算の後で保存されます。

TINYINT 値の範囲は、0 ～ 2^8 - 1、または 0 ～ 255 です。

文字列を TINYINT に変換すると、前後のスペースは削除されます。先行文字が + の場合は無視されます。先行文字が - の場合は、残りの数字は負の数として解釈されます。先行の 0 文字はスキップされ、残りの文字は整数値に変換されます。変換しようとしたデータ型の有効な範囲外の値である場合、文字列に不正な文字が含まれる場合、または文字列を整数値として復号化できない場合にエラーが返されます。

Embedded SQL では、TINYINT カラムを CHAR または UNSIGNED CHAR と定義された変数にフェッチしないでください。そうした場合、カラムの値を文字列に変換し、最初のバイトをプログラムの変数に割り当てようとすることになるからです。TINYINT カラムは 2 バイトまたは 4 バイト整数にフェッチしてください。TINYINT 値を C で書かれたアプリケーションからデータベースに送るには、C 変数の型に INTEGER を指定してください。

Ultra Light: Embedded SQL では、TINYINT カラムを CHAR と定義された変数にフェッチしないでください。そうした場合、カラムの値を文字列に変換し、最初のバイトをプログラムの変数に割り当てようとすることになるからです。TINYINT カラムは 2 バイトまたは 4 バイト整数にフェッチしてください。TINYINT 値を C で書かれたアプリケーションからデータベースに送るには、C 変数の型に INTEGER を指定してください。

関連情報

数値関数 [315 ページ]
集合関数 [308 ページ]
BIGINT データ型 [284 ページ]
BIT データ型 [285 ページ]
VARCHAR データ型 [282 ページ]
SMALLINT データ型 [292 ページ]

1.13.2.3 日付と時刻データ型

日付の値は 4 桁の西暦で出力でき、日付の内部記憶領域には、年の値の世紀にあたる部分が常に明示的に含まれています。

算出した値が別の世紀にかかるかどうかにかかわらず、日付に関する有効な算術演算や論理演算に対して常に正しい値を返します。

このセクションの内容:

DATE データ型 [294 ページ]
DATE データ型は、年、月、日などの暦日を格納します。
DATETIME データ型 [295 ページ]
DATETIME は、日付と時刻の情報を格納します。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

TIME データ型 [296 ページ]
TIME データ型は、時、分、秒、秒以下で構成される時刻を格納します。

TIMESTAMP データ型 [297 ページ]
TIMESTAMP データ型は、年、月、日、時、分、秒、秒以下 (小数第 6 位まで) で構成される時刻を格納します。

TIMESTAMP WITH TIME ZONE データ型 [298 ページ]
TIMESTAMP WITH TIME ZONE データ型は、タイムゾーンオフセット付きの時刻を格納します。

1.13.2.3.1 DATE データ型

DATE データ型は、年、月、日などの暦日を格納します。

構文

DATE

備考

DATE 値には 4 バイトの記憶領域が必要です。

アプリケーションが文字列として DATE 値を取得するときのフォーマットは、date_format オプションの設定によって決まります。たとえば、2010 年 7 月 19 日を表す DATE 値は、date_format オプション設定に従い、アプリケーションに 2010/07/19 または Jul 19, 2010 として返されます。

Ultra Light: アプリケーションが文字列として DATE 値を取得するときのフォーマットは、date_format 作成パラメータによって決まります。たとえば、2010 年 7 月 19 日を表す DATE 値は、date_format 作成パラメータ設定に従い、アプリケーションに 2010/07/19 または Jul 19, 2010 として返されます。

関連情報

CURRENT TIME 特別値 - Ultra Light [252 ページ]
CURRENT TIMESTAMP 特別値 - Ultra Light [253 ページ]
日付および時刻関数 [311 ページ]
CURRENT TIMESTAMP 特別値 - Ultra Light [253 ページ]
DATE 関数 [日付と時刻] [355 ページ]
Ultra Light の date_format 作成オプション [145 ページ]
Ultra Light の date_order 作成オプション [147 ページ]
DATETIME データ型 [295 ページ]
DATETIME 関数 [日付と時刻] [363 ページ]
ISDATE 関数 [データ型変換] [386 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

Ultra Light nearest_century 作成オプション [157 ページ]
NOW 関数 [日付と時刻] [415 ページ]
TIME データ型 [296 ページ]
TIMESTAMP データ型 [297 ページ]
TIMESTAMP WITH TIME ZONE データ型 [298 ページ]

1.13.2.3.2 DATETIME データ型

DATETIME は、日付と時刻の情報を格納します。

構文

DATETIME

備考

DATETIME は Transact-SQL の型です。

Ultra Light: DATETIME はドメインであり、TIMESTAMP として実装されます。

アプリケーションが文字列として DATETIME 値を取得するときの形式は、timestamp_format オプションの設定によって決まります。たとえば、DATETIME 値 2010/04/01T23:59:59.999999 は、timestamp_format オプション設定に従い、アプリケーションに 2010/04/01 23:59:59 または April 1, 2010 23:59:59.999999 として返されます。

DATETIME 値には 8 バイトの記憶領域が必要です。

DATETIME データ型では、DATE 型と同じく 0001 ～ 9999 年の日付を格納でき、さらに 1600-02-28 23:59:59 ～ 7911-01-01 00:00:00 も格納できます。DATETIME 値では、この範囲より前または後の時間および分の部分は保持されません。

注記

DATETIME 値の精度が下がると、分または秒に関連する組み込み関数が無効な結果を生成します。

DATETIME 値を DATETIMEOFFSET に変換する場合、接続の time_zone_adjustment 設定が結果のタイムゾーンオフセットに使用されます。つまり、値はその接続に "ローカル" なものであると見なされます。DATETIMEOFFSET 値を DATETIME に変換すると、オフセットは廃棄されます。

関連情報

TIMESTAMP データ型 [297 ページ]

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

1.13.2.3.3 TIME データ型

TIME データ型は、時、分、秒、秒以下で構成される時刻を格納します。

構文

TIME

備考

TIME 値には 8 バイトの記憶領域が必要です

ODBC を使用するときに、(ODBC TIME_STRUCT 構造体を使用して) バイナリ値として送信または取得される TIME 値は、時、分、秒の精度に制限されます。秒の小数部分は構造体には含まれません。そのため、精度を高めることを希望する場合は、TIME 値を文字列として送信または取得してください。アプリケーションが文字列として TIME 値を取得するときのフォーマットは、time_format オプションの設定によって決まります。たとえば、TIME 値 23:59:59.999999 は、time_format オプションの設定に従い、23:59:59、23:59:59.999、または 23:59:59.999999 としてアプリケーションに返されます。

Ultra Light: アプリケーションが文字列として TIME 値を取得するときの形式は、time_format 作成パラメータによって決まります。たとえば、TIME 値 23:59:59.999999 は、time_format 作成パラメータの設定に従い、23:59:59、23:59:59.999、または 23:59:59.999999 としてアプリケーションに返されます。

関連情報

CURRENT TIME 特別値 - Ultra Light [252 ページ]
CURRENT TIMESTAMP 特別値 - Ultra Light [253 ページ]
CURRENT UTC TIMESTAMP 特別値 - Ultra Light [254 ページ]
日付および時刻関数 [311 ページ]
DATE データ型 [294 ページ]
DATETIME データ型 [295 ページ]
DATE 関数 [日付と時刻] [355 ページ]
DATETIME 関数 [日付と時刻] [363 ページ]
Ultra Light の式 [256 ページ]
GETDATE 関数 [日付と時刻] [377 ページ]
ISDATE 関数 [データ型変換] [386 ページ]
NOW 関数 [日付と時刻] [415 ページ]
TIMESTAMP データ型 [297 ページ]
TIMESTAMP WITH TIME ZONE データ型 [298 ページ]
Ultra Light timestamp_format 作成オプション [167 ページ]
Ultra Light timestamp_with_time_zone_format 作成オプション [171 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.13.2.3.4 TIMESTAMP データ型

TIMESTAMP データ型は、年、月、日、時、分、秒、秒以下 (小数第 6 位まで) で構成される時刻を格納します。

構文

TIMESTAMP

備考

TIMESTAMP 値には 8 バイトの記憶領域が必要です。

アプリケーションが文字列として TIMESTAMP 値を取得するときのフォーマットは、timestamp_format オプションの設定によって決まります。たとえば、TIMESTAMP 値 2010/04/01T23:59:59.999999 は、timestamp_format オプション設定に従い、アプリケーションに 2010/04/01 23:59:59 または April 1, 2010 23:59:59.999999 として返されます。

Ultra Light: アプリケーションが文字列として TIMESTAMP 値を取得するときのフォーマットは、timestamp_format 作成パラメータによって決まります。たとえば、TIMESTAMP 値 2010/04/01T23:59:59.999999 は、timestamp_format 作成パラメータに従い、アプリケーションに 2010/04/01 23:59:59 または April 1, 2010 23:59:59.999999 として返されます。

TIMESTAMP データ型では、DATE 型と同じく 0001 ～ 9999 年の日付を格納できますが、実質的に有効な範囲は 1600-02-28 23:59:59 ～ 7911-01-01 00:00:00 です。TIMESTAMP 値では、この範囲より前または後の時間および分の部分は保持されません。

注記

TIMESTAMP 値の精度が下がると、分または秒に関連する組み込み関数が無効な結果を生成します。

TIMESTAMP 値を TIMESTAMP WITH TIME ZONE に変換する場合、その接続の time_zone_adjustment 設定が、結果のタイムゾーンオフセットに使用されます。つまり、値はその接続に "ローカル" なものであると見なされます。TIMESTAMP WITH TIME ZONE 値を TIMESTAMP に変換すると、オフセットは廃棄されます。

Ultra Light: TIMESTAMP 値を TIMESTAMP WITH TIME ZONE に変換する場合、システムのローカルタイムゾーンオフセットが最終結果に使用されます。

関連情報

CURRENT TIME 特別値 - Ultra Light [252 ページ]
CURRENT TIMESTAMP 特別値 - Ultra Light [253 ページ]
CURRENT UTC TIMESTAMP 特別値 - Ultra Light [254 ページ]
日付および時刻関数 [311 ページ]
DATE データ型 [294 ページ]
DATETIME データ型 [295 ページ]
DATE 関数 [日付と時刻] [355 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

DATETIME 関数 [日付と時刻] [363 ページ]
Ultra Light の date_order 作成オプション [147 ページ]
Ultra Light の式 [256 ページ]
GETDATE 関数 [日付と時刻] [377 ページ]
ISDATE 関数 [データ型変換] [386 ページ]
Ultra Light nearest_century 作成オプション [157 ページ]
NOW 関数 [日付と時刻] [415 ページ]
TIME データ型 [296 ページ]
TIMESTAMP WITH TIME ZONE データ型 [298 ページ]
Ultra Light timestamp_format 作成オプション [167 ページ]
Ultra Light timestamp_with_time_zone_format 作成オプション [171 ページ]

1.13.2.3.5 TIMESTAMP WITH TIME ZONE データ型

TIMESTAMP WITH TIME ZONE データ型は、タイムゾーンオフセット付きの時刻を格納します。

構文

```
TIMESTAMP WITH TIME ZONE
```

備考

TIMESTAMP WITH TIME ZONE 値は、年、月、日、時、分、秒、秒の小数位、および協定世界時 (UTC: Coordinated Universal Time) との差異を示す分数を含んでいます。秒以下は 6 桁まで格納されます。

注記

デフォルトでは、TIMESTAMP WITH TIME ZONE 値はデータベースの現在のタイムゾーンです。

アプリケーションが文字列として TIMESTAMP WITH TIME ZONE 値を取得するときのフォーマットは、timestamp_with_time_zone_format オプションの設定によって決まります。たとえば、TIMESTAMP WITH TIME ZONE 値 2010/04/01T23:59:59.999999-6:00 は、timestamp_with_time_zone_format オプションの設定に従い、アプリケーションに 2010/04/01 23:59:59 -06:00 または April 1, 2010 23:59:59.999999 -06:00 として返されます。

Ultra Light: アプリケーションが文字列として TIMESTAMP WITH TIME ZONE 値を取得するときのフォーマットは、timestamp_with_time_zone_format 作成パラメータの設定によって決まります。たとえば、TIMESTAMP WITH TIME ZONE 値 2010/04/01T23:59:59.999999-6:00 は、timestamp_with_time_zone_format 作成パラメータに従い、アプリケーションに 2010/04/01 23:59:59 -06:00 または April 1, 2010 23:59:59.999999 -06:00 として返されます。

TIMESTAMP WITH TIME ZONE 値には 10 バイトの記憶領域が必要です。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

TIMESTAMP WITH TIME ZONE データ型では、DATE 型と同じく 0001 ～ 9999 年の日付を格納でき、さらに 1600-02-28 23:59:59 ～ 7911-01-01 00:00:00 も格納できます。TIMESTAMP WITH TIME ZONE 値では、この範囲より前または後の時間および分の部分は保持されません。

計算カラムまたはマテリアライズドビューに TIMESTAMP WITH TIME ZONE を使用しないでください。時刻を制御する time_zone_adjustment オプションの値は、場所と日付に基づいて、接続によって異なるためです。

2 つの TIMESTAMP WITH TIME ZONE 値が UTC において同じ時刻を表す場合、適用されている TIME ZONE オフセットに関係なく同じ時刻と見なされます。たとえば、次の文では、結果が同じと見なされるため、Yes が返されます。

```
IF CAST('2009-07-15 08:00:00 -08:00' AS TIMESTAMP WITH TIME ZONE) =
    CAST('2009-07-15 11:00:00 -05:00' AS TIMESTAMP WITH TIME ZONE) THEN
  SELECT  'Yes'
ELSE
  SELECT 'No'
END IF;
```

Ultra Light:

```
SELECT IF CAST('2009-07-15 08:00:00 -08:00' AS TIMESTAMP WITH TIME ZONE) =
    CAST('2009-07-15 11:00:00 -05:00' AS TIMESTAMP WITH TIME ZONE)
    THEN 'Yes'
    ELSE 'No'
END IF;
```

TIMESTAMP WITH TIME ZONE 値からタイムゾーンオフセットを省略すると、タイムスタンプが日付と時刻を標準時間または夏時間のいずれで表していても、クライアントの現在の UTC オフセットがデフォルト設定されます。たとえば、クライアントが東部標準タイムゾーンに属しており、夏時間が有効なときに次の文を実行した場合、大西洋標準タイムゾーン (UTC から -4 時間) に該当するタイムゾーンでタイムスタンプが返されます。

```
SELECT CAST('2009/01/30 12:34:55' AS TIMESTAMP WITH TIME ZONE)
```

TIMESTAMP WITH TIME ZONE とその他のデータ型の比較

TIMESTAMP WITH TIME ZONE 値をタイムゾーン付きではないタイムスタンプと比較することはおすすめしません。これは、クライアントのデフォルトのタイムゾーンオフセットは、クライアントの地理的な場所および日付によって異なるためです。

クライアントの現在のタイムゾーンオフセットを分単位で判別するには、次の文を実行します。

```
SELECT CONNECTION_PROPERTY( 'TimeZoneAdjustment' );
```

TIMESTAMP WITH TIME ZONE とその他のデータ型の間の変換

TIMESTAMP 値を TIMESTAMP WITH TIME ZONE に変換する場合、その接続の time_zone_adjustment 設定が、結果のタイムゾーンオフセットに使用されます。つまり、値はその接続に "ローカル" なものであると見なされます。
TIMESTAMP WITH TIME ZONE 値を TIMESTAMP に変換すると、オフセットは廃棄されます。文字列型、日付型、または時刻型以外の型との間の変換はサポートされていません。

Ultra Light: TIMESTAMP 値を TIMESTAMP WITH TIME ZONE に変換する場合、クライアントのタイムゾーンが結果のタイムゾーンオフセットに使用されます。つまり、値はその接続に "ローカル" なものであると見なされます。
TIMESTAMP WITH TIME ZONE 値を TIMESTAMP に変換すると、オフセットは廃棄されます。文字列型、日付型、または時刻型以外の型との間の変換はサポートされていません。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

関連情報

CURRENT TIME 特別値 - Ultra Light [252 ページ]
CURRENT TIMESTAMP 特別値 - Ultra Light [253 ページ]
CURRENT UTC TIMESTAMP 特別値 - Ultra Light [254 ページ]
日付および時刻関数 [311 ページ]
DATE データ型 [294 ページ]
DATETIME データ型 [295 ページ]
DATE 関数 [日付と時刻] [355 ページ]
DATETIME 関数 [日付と時刻] [363 ページ]
Ultra Light の date_order 作成オプション [147 ページ]
Ultra Light の式 [256 ページ]
GETDATE 関数 [日付と時刻] [377 ページ]
ISDATE 関数 [データ型変換] [386 ページ]
Ultra Light nearest_century 作成オプション [157 ページ]
NOW 関数 [日付と時刻] [415 ページ]
TIME データ型 [296 ページ]
TIMESTAMP データ型 [297 ページ]
Ultra Light timestamp_format 作成オプション [167 ページ]
Ultra Light timestamp_with_time_zone_format 作成オプション [171 ページ]

1.13.2.4 バイナリデータ型

バイナリデータ型は、データベースに認識されないイメージやその他の種類の情報を含めたバイナリデータを格納します。

このセクションの内容:

BINARY データ型 [301 ページ]
BINARY データ型は、指定された最大長 (バイト単位) のバイナリデータを格納します。

LONG BINARY データ型 [302 ページ]
LONG BINARY データ型は、任意の長さのバイナリデータを格納します。

UNIQUEIDENTIFIER データ型 [302 ページ]
UNIQUEIDENTIFIER データ型は、UUID (GUID とも呼ばれる) の値を格納します。

VARBINARY データ型 [303 ページ]
VARBINARY データ型は、指定された最大長 (バイト単位) のバイナリデータを格納します。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.13.2.4.1 BINARY データ型

BINARY データ型は、指定された最大長 (バイト単位) のバイナリデータを格納します。

構文

```
BINARY [ ( max-length ) ]
```

パラメータ

max-length

値のバイト単位での最大長。長さを指定しない場合、値は 1 になります。

長さは 1 ～ 32767 の範囲内で指定する必要があります。

備考

比較時に、BINARY 値はバイトごとに正確に比較されます。この比較方法は CHAR データ型とは異なります。CHAR の場合、データベースの照合順を使用して値が比較されます。

1 つのバイナリ文字列が他の文字列のプレフィクスである場合、短い文字列は長い文字列よりも文字数が少ないと見なされます。

CHAR 値とは異なり、文字セットの変換時に BINARY 値は変換されません。

セマンティック上、BINARY は VARBINARY と同じです。可変長型です。他のデータベース管理システムでは、BINARY は固定長型です。

Ultra Light: BINARY はドメインであり、VARBINARY として実装されます。

関連情報

VARBINARY データ型 [303 ページ]

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

1.13.2.4.2 LONG BINARY データ型

LONG BINARY データ型は、任意の長さのバイナリデータを格納します。

構文

LONG BINARY

備考

最大サイズは 2 GB から 1 バイトを差し引いた値です (2^{31} - 1)。

LONG BINARY データに、または LONG BINARY データから文字列をキャストできます。LONG BINARY データを連結することはできません。

SQL 文の条件 (WHERE 句の条件など) は、LONG BINARY カラムでは実行できません。LONG BINARY カラムでは、INSERT、UPDATE、DELETE の各操作が許可されます。LONG BINARY カラムは、SELECT クエリの結果セットに含めることもできます。

Ultra Light: LONG BINARY 型に対してインデックスを作成することはできません。LONG BINARY 型は LENGTH 関数と CAST 関数でのみ使用できます。

関連情報

BINARY データ型 [301 ページ]
VARBINARY データ型 [303 ページ]

1.13.2.4.3 UNIQUEIDENTIFIER データ型

UNIQUEIDENTIFIER データ型は、UUID (GUID とも呼ばれる) の値を格納します。

構文

UNIQUEIDENTIFIER

備考

UNIQUEIDENTIFIER データ型は、通常は、ローをユニークに識別する UUID (ユニバーサルユニーク識別子) 値を保持するために、プライマリキーまたはその他のユニークカラムに使用されます。NEWID 関数では、あるコンピュータで生成される

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

UUID 値が他のコンピュータで生成される UUID と一致しないように UUID 値が生成されます。したがって、NEWID を使用して生成された UNIQUEIDENTIFIER 値は、同期環境でキーとして使用できます。

例:

```
CREATE TABLE T1 (
    pk UNIQUEIDENTIFIER PRIMARY KEY DEFAULT NEWID(),
    c1 INT );
```

Ultra Light の場合:

```
CREATE TABLE T1 (
    pk UNIQUEIDENTIFIER PRIMARY KEY DEFAULT NEWID(),
    c1 INT );
```

UUID 値は、GUID (グローバルユニーク識別子) 値とも呼ばれます。UUID 値にはハイフンが含まれます。これは他の RDBMS と互換性を持たせるためです。

このデフォルト設定を変更するには、uuid_has_hyphens オプションを Off にします。

Ultra Light: このデフォルト設定を変更するには UUIDTOSTR 関数と STRTOUUID 関数を使用します。

UNIQUEIDENTIFIER 値は、必要に応じて文字列値とバイナリ値の間で自動的に変換されます。

UNIQUEIDENTIFIER 値は、BINARY(16) として格納されますが、クライアントアプリケーションには BINARY(36) として示されます。このため、クライアントが値を文字列としてフェッチした場合に、結果に対して十分な領域が割り付けられることが保証されます。

SQL Anywhere の場合、ODBC クライアントアプリケーションでは、uniqueidentifier 値が SQL_GUID 型として示されます。

関連情報

NEWID 関数 [その他] [414 ページ]
UUIDTOSTR 関数 [文字列] [469 ページ]
STRTOUUID 関数 [文字列] [454 ページ]
STRING 関数 [文字列] [453 ページ]

1.13.2.4.4 VARBINARY データ型

VARBINARY データ型は、指定された最大長 (バイト単位) のバイナリデータを格納します。

構文

```
VARBINARY [ ( max-length ) ]
```


© 2016 SAP SE or an SAP affiliate company. All rights reserved.

パラメータ

max-length

値のバイト単位での最大長。長さを指定しない場合、値は 1 になります。

長さは 1 ～ 32767 の範囲内で指定する必要があります。

備考

比較時に、VARBINARY 値はバイトごとに正確に比較されます。この比較方法は CHAR データ型とは異なります。CHAR の場合、データベースの照合順を使用して値が比較されます。

VARBINARY 値は、文字セットの変換時に変換されません。

VARBINARY は BINARY VARYING と指定することもできます。どの構文を使用する場合でも、データ型は VARBINARY と記述されます。1 つのバイナリ文字列が他の文字列のプレフィクスである場合、短い文字列は長い文字列よりも文字数が少ないと見なされます。

Ultra Light: 1 つのバイナリ文字列が他の文字列のプレフィクスである場合、他の文字列と比較するときに他よりも短い文字列はゼロ (0) が埋め込まれていると見なされます。式を評価するときのテンポラリ文字値の最大長は 2048 バイトです。

関連情報

ビット処理演算子 - Ultra Light [277 ページ]
BINARY データ型 [301 ページ]
STRING 関数 [文字列] [453 ページ]

1.13.3 空間データ型

空間データは、定義された空間内のオブジェクトの位置、形、および方向を記述するデータです。Ultra Light には、ポイント形式の空間データ用の記憶領域およびデータ管理機能が用意されており、これらの機能を使用して、地理的ロケーションに関する情報やルート指定情報などを格納できます。

ポイントは、空間型の ST_GEOMETRY を使用して定義します。空間データに対するアクセスと操作は、関数とコンストラクタを使用して行います。また、Ultra Light には、他の製品との互換性を確保するように設計された一連の SQL 空間関数が用意されています。

ポイントは空間内の単一の場所を定義します。ポイントジオメトリには長さも面積もありません。ポイントには必ず X 座標と Y 座標があります。

GIS データでは、ポイントは、通常、住所などの場所、または山などの地理的特性を表すために使用されます。

このセクションの内容:

 © 2016 SAP SE or an SAP affiliate company.All rights reserved.

ST_GEOMETRY データ型 - Ultra Light [305 ページ]
ST_GEOMETRY 型は、ポイント形式の空間データを格納するために使用されます。

1.13.3.1 ST_GEOMETRY データ型 - Ultra Light

ST_GEOMETRY 型は、ポイント形式の空間データを格納するために使用されます。

備考

ST_GEOMETRY 型は、ジオメトリ型階層の最大スーパータイプです。ST_GEOMETRY 型では、任意の空間値に適用できるメソッドがサポートされています。ST_GEOMETRY 型はインスタンス化できません。代わりに、サブタイプをインスタンス化してください。元のフォーマット (WKT または WKB) を使用する場合は、ST_PointFromText/ST_PointFromWKB などのメソッドを使用して、値を元のフォーマットで表す具体的な型をインスタンス化できます。

ST_SRID メソッドを使用すると、値に関連付けられている空間参照系を取り出すことができます。

ST_GEOMETRY 型またはそのサブタイプのカラムには、プライマリキー、ユニークなインデックス、または一意性制約を入れることはできません。

カラムとオブジェクトの定義

Ultra Light では 3 種類の固定の参照系を使用でき、カラムを作成するとき、これらの参照系を属性としてカラムに割り当てることができます。個々のジオメトリオブジェクトは、未定義の参照系を除き、どの SRID 値にも関連付けることができ、一致する SRID 値または未定義の参照系に関連付けられているカラムにのみ格納できます。

事前に定義された参照系は次のとおりです。

未定義 (null) 参照系

SRID 値が指定されていない場合、これがデフォルトの参照系になります。この参照系には、任意の有効な参照系を使用するジオメトリ値を含めることができます。また、この参照系では、ジオメトリオブジェクト間での参照系の一貫性を強制しない "キャッチオール" カラムが許可されます。

デフォルトの平面参照系

カラムの作成時に SRID 値 0 を指定することで定義されます。このカラムには、この参照系に関連付けられているジオメトリ値のみを格納できます。値は、2D の平面空間にあるものとして処理されます。

WGS 84 測地参照系

カラムの作成時に SRID 値 4326 を指定することで定義されます。このカラムには、この参照系に関連付けられているジオメトリ値のみを格納できます。値は、地球の表面上にあるものとして処理され、それに応じて操作が適用されます。

> **注記**
>
> SRID 4326 内のポイントは、WGS 84 参照系または未定義の参照系を使用するカラムには格納できますが、デフォルトの平面参照系を使用するカラムには格納できません。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

参照系間での変換はサポートされていません。

例

次の例は、デフォルトの平面参照系に関連付けられているカラムと未定義の参照系に関連付けられているカラムを 1 つずつ持つテーブルの作成方法を示しています。

```
CREATE TABLE T1 (
    V1 INTEGER PRIMARY KEY,
    V2 ST_GEOMETRY(SRID=0),
    V3 ST_GEOMETRY
)
```

次の SQL 文は、前の例の T1 テーブルへのデータの挿入方法を示しています。

```
INSERT INTO T1(V1, V2, V3)
    VALUES (1, ST_POINTFROMTEXT('POINT(10 20)', 0), ST_POINT(5, 6, 2163))
```

関連情報

ST_AsBinary 関数 [空間] - Ultra Light [441 ページ]
ST_AsText 関数 [空間] - Ultra Light [442 ページ]
ST_Distance 関数 [空間] - Ultra Light [443 ページ]
ST_Equals 関数 [空間] - Ultra Light [444 ページ]
ST_IntersectsRect 関数 [空間] - Ultra Light [445 ページ]
ST_Point 関数 [空間] - Ultra Light [446 ページ]
ST_PointFromExtText 関数 [空間] - Ultra Light [447 ページ]
ST_PointFromText 関数 [空間] - Ultra Light [447 ページ]
ST_PointFromWKB 関数 [空間] - Ultra Light [448 ページ]
ST_SRID 関数 [空間] - Ultra Light [449 ページ]
ST_X 関数 [空間] - Ultra Light [450 ページ]
ST_Y 関数 [空間] - Ultra Light [451 ページ]

1.13.4 ユーザ定義データ型とそれに相当する型

SQL Anywhere データベースとは異なり、Ultra Light ではユーザ定義データ型はサポートされていません。

SQL Anywhere の組み込みエイリアスに相当する Ultra Light データ型を次の表に示します。

SQL Anywhere のデータ型	Ultra Light で相当する型
MONEY	DECIMAL(19,4)
SMALLMONEY	DECIMAL(10,4)

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

SQL Anywhere のデータ型	Ultra Light で相当する型
TEXT	LONG VARCHAR
XML	LONG VARCHAR

関連情報

LONG VARCHAR データ型 [281 ページ]
DECIMAL データ型 [286 ページ]

1.13.5 SQL 関数

関数は、データベースの情報を返すために使用します。関数は、式を使用できるところではどこからでも呼び出すことができます。

マニュアルに特に指定がないかぎり、関数の引数のいずれかが NULL である場合は、NULL が返されます。

関数には、SQL 文と同じ構文の表記規則を使用します。

SQL Anywhere では、引数を省略できる場合、引数として DEFAULT を指定できます。

このセクションの内容:

関数のタイプ [307 ページ]
関数は、実行されるデータのタイプ、または使用されるコンテキストに従ってグループ化できます。

関数 [322 ページ]
関数を 1 つずつリストし、その右側に関数のタイプ (数値、文字など) を示します。

1.13.5.1 関数のタイプ

関数は、実行されるデータのタイプ、または使用されるコンテキストに従ってグループ化できます。

> **注記**
>
> 特に明記しないかぎり、NULL をパラメータとして受け取る SQL Anywhere 関数は、NULL を返します。

Ultra Light: Ultra Light では、SQL Anywhere 用に記載されている関数のサブセットをサポートしていますが、一部に違いがあります。

このセクションの内容:

集合関数 [308 ページ]

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

集合関数は、データベースから選択されたローグループのデータを要約します。グループは、SELECT 文の GROUP BY 句を使用して形成されます。集合関数は、SELECT リストと、SELECT 文の HAVING 句と ORDER BY 句でのみ使用できます。

データ型変換関数 [310 ページ]
データ型変換関数を使用して、引数のデータ型を他のデータ型に変換したり、変換が可能かどうかをテストしたりします。

日付および時刻関数 [311 ページ]
日付と時刻の関数は、DATE、TIME、TIMESTAMP、および TIMESTAMP WITH TIME ZONE の各データ型に対する操作を実行します。

その他の関数 [314 ページ]
その他の関数は、算術式、文字列式、日付/時刻式、他の関数の戻り値に対して操作を実行します。

数値関数 [315 ページ]
数値関数は、数値データ型の算術演算を実行したり、数値情報を返したりします。

空間関数 - Ultra Light [317 ページ]
空間データは、定義された空間内のオブジェクトの位置、形、および方向を記述するデータです。Ultra Light には、ポイント形式の空間データ用の記憶領域およびデータ管理機能が用意されており、これらの機能を使用して、地理的ロケーションに関する情報やルート指定情報などを格納できます。

文字列関数 [317 ページ]
文字列関数は、文字列に対して変換、抽出、操作の演算を実行したり、文字列に関する情報を返したりします。

システム関数 [320 ページ]
システム関数は、システム情報を返します。

関連情報

ST_GEOMETRY データ型 - Ultra Light [305 ページ]

1.13.5.1.1 集合関数

集合関数は、データベースから選択されたローグループのデータを要約します。グループは、SELECT 文の GROUP BY 句を使用して形成されます。集合関数は、SELECT リストと、SELECT 文の HAVING 句と ORDER BY 句でのみ使用できます。

SQL Anywhere 関数のリスト

次の集合関数を使用できます。

ARRAY_AGG 関数 [集合]
AVG 関数 [集合]
BIT_AND 関数 [集合]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

BIT_OR 関数 [集合]
BIT_XOR 関数 [集合]
COVAR_POP 関数 [集合]
COVAR_SAMP 関数 [集合]
COUNT 関数 [集合]
COUNT_BIG 関数 [集合]
CORR 関数 [集合]
FIRST_VALUE 関数 [集合]
GROUPING 関数 [集合]
LAST_VALUE 関数 [集合]
LIST 関数 [集合]
MAX 関数 [集合]
MEDIAN 関数 [集合]
MIN 関数 [集合]
REGR_AVGX 関数 [集合]
REGR_AVGY 関数 [集合]
REGR_COUNT 関数 [集合]
REGR_INTERCEPT 関数 [集合]
REGR_R2 関数 [集合]
REGR_SLOPE 関数 [集合]
REGR_SXX 関数 [集合]
REGR_SXY 関数 [集合]
REGR_SYY 関数 [集合]
SET_BITS 関数 [集合]
STDDEV 関数 [集合]
STDDEV_POP 関数 [集合]
STDDEV_SAMP 関数 [集合]
SUM 関数 [集合]
VAR_POP 関数 [集合]
VAR_SAMP 関数 [集合]
VARIANCE 関数 [集合]
XMLAGG 関数 [集合]

Ultra Light 関数のリスト

次の集合関数を使用できます。

AVG 関数 [集合] - Ultra Light
COUNT 関数 [集合] - Ultra Light
COUNT_UPLOAD_ROWS 関数 [集合] - Ultra Light
LIST 関数 [集合] - Ultra Light

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

MAX 関数 [集合] - Ultra Light
MIN 関数 [集合] - Ultra Light
SUM 関数 [集合] - Ultra Light

1.13.5.1.2 データ型変換関数

データ型変換関数を使用して、引数のデータ型を他のデータ型に変換したり、変換が可能かどうかをテストしたりします。

SQL Anywhere 関数のリスト

次のデータ型変換関数を使用できます。

BINTOHEX 関数 [データ型変換]
CAST 関数 [データ型変換]
CONVERT 関数 [データ型変換]
HEXTOBIN 関数 [データ型変換]
HEXTOINT 関数 [データ型変換]
INTTOHEX 関数 [データ型変換]
ISDATE 関数 [データ型変換]
ISNUMERIC 関数 [その他]
TREAT 関数 [データ型変換]

Ultra Light 関数のリスト

次のデータ型変換関数を使用できます。

CAST 関数 [データ型変換] - Ultra Light
CONVERT 関数 [データ型変換] - Ultra Light
HEXTOINT 関数 [データ型変換] - Ultra Light
INTTOHEX 関数 [データ型変換] - Ultra Light
ISDATE 関数 [データ型変換] - Ultra Light

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.13.5.1.3 日付および時刻関数

日付と時刻の関数は、DATE、TIME、TIMESTAMP、および TIMESTAMP WITH TIME ZONE の各データ型に対する操作を実行します。

SQL Anywhere には、Transact-SQL の日付型と時間型に対する互換性サポート (DATETIME と SMALLDATETIME を含む) があります。これらの Transact-SQL データ型は、ネイティブな TIMESTAMP データ型を介したドメインとして実装されます。

SQL Anywhere では、次の日付と時刻関数を使用できます。

- DATE 関数 [日付と時刻]
- DATEADD 関数 [日付と時刻]
- DATEDIFF 関数 [日付と時刻]
- DATEFORMAT 関数 [日付と時刻]
- DATENAME 関数 [日付と時刻]
- DATEPART 関数 [日付と時刻]
- DATETIME 関数 [日付と時刻]
- DAY 関数 [日付と時刻]
- DAYNAME 関数 [日付と時刻]
- DAYS 関数 [日付と時刻]
- DOW 関数 [日付と時刻]
- GETDATE 関数 [日付と時刻]
- HOUR 関数 [日付と時刻]
- HOURS 関数 [日付と時刻]
- MINUTE 関数 [日付と時刻]
- MINUTES 関数 [日付と時刻]
- MONTH 関数 [日付と時刻]
- MONTHNAME 関数 [日付と時刻]
- MONTHS 関数 [日付と時刻]
- NOW 関数 [日付と時刻]
- QUARTER 関数 [日付と時刻]
- SECOND 関数 [日付と時刻]
- SECONDS 関数 [日付と時刻]
- SWITCHOFFSET 関数 [日付と時刻]
- SYSDATETIMEOFFSET 関数 [日付と時刻]
- TODAY 関数 [日付と時刻]
- TODATETIMEOFFSET 関数 [日付と時刻]
- WEEKS 関数 [日付と時刻]
- YEAR 関数 [日付と時刻]
- YEARS 関数 [日付と時刻]
- YMD 関数 [日付と時刻]

Ultra Light では、次の日付と時刻関数を使用できます。

- DATE 関数 [日付と時刻] - Ultra Light

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

DATEADD 関数 [日付と時刻] - Ultra Light
DATEDIFF 関数 [日付と時刻] - Ultra Light
DATEFORMAT 関数 [日付と時刻] - Ultra Light
DATENAME 関数 [日付と時刻] - Ultra Light
DATEPART 関数 [日付と時刻] - Ultra Light
DATETIME 関数 [日付と時刻] - Ultra Light
DAY 関数 [日付と時刻] - Ultra Light
DAYNAME 関数 [日付と時刻] - Ultra Light
DAYS 関数 [日付と時刻] - Ultra Light
DOW 関数 [日付と時刻] - Ultra Light
GETDATE 関数 [日付と時刻] - Ultra Light
HOUR 関数 [日付と時刻] - Ultra Light
HOURS 関数 [日付と時刻] - Ultra Light
MINUTE 関数 [日付と時刻] - Ultra Light
MINUTES 関数 [日付と時刻] - Ultra Light
MONTH 関数 [日付と時刻] - Ultra Light
MONTHNAME 関数 [日付と時刻] - Ultra Light
MONTHS 関数 [日付と時刻] - Ultra Light
NOW 関数 [日付と時刻] - Ultra Light
QUARTER 関数 [日付と時刻] - Ultra Light
SECOND 関数 [日付と時刻] - Ultra Light
SECONDS 関数 [日付と時刻] - Ultra Light
SWITCHOFFSET 関数 [日付と時刻] - Ultra Light
TODAY 関数 [日付と時刻] - Ultra Light
TODATETIMEOFFSET 関数 [日付と時刻] - Ultra Light
WEEKS 関数 [日付と時刻] - Ultra Light
YEAR 関数 [日付と時刻] - Ultra Light
YEARS 関数 [日付と時刻] - Ultra Light
YMD 関数 [日付と時刻] - Ultra Light

このセクションの内容:

日付の単位の指定 [312 ページ]
日付関数の多くは、日付の単位で構成される日付を使用します。次の表は、使用できる日付の単位の値を示します。

1.13.5.1.3.1 日付の単位の指定

日付関数の多くは、日付の単位で構成される日付を使用します。次の表は、使用できる日付の単位の値を示します。

日付と時刻の関数を使用する場合、日付または時刻の減算にマイナス記号を指定できます。たとえば、31 日前のタイムスタンプを取得するには、次の文を実行します。

```
SELECT DATEADD(day, -31, NOW());
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

日付の単位	省略形	値
Year	YY	1 ～ 9999
Quarter	QQ	1 ～ 4
Month	MM	1 ～ 12
Week	WK	1 ～ 54 日曜日を週の最初の日とします。54 週ある年は、土曜日から始まる閏年で発生します。
Day	DD	1 ～ 31
Dayofyear	DY	1 ～ 366
Weekday	DW	1 ～ 7 (日曜日 = 1、...、土曜日 = 7)
Hour	HH	0 ～ 23
Minute	MI	0 ～ 59
Second	SS	0 ～ 59
Millisecond	MS	0 ～ 999
Microsecond	MCS または US	0 ～ 999999
Calyearofweek	CYR	1 ～ 9999。その週が何年に開始したかを示します。週に年の最初の数日が含まれている場合は、その年の最初の曜日に応じて、週の最初の日が前年になる場合があります。年の最初の曜日が月曜日～木曜日の場合は、前年に属する日がその年に含まれることはありませんが、年の最初の曜日が金曜日～日曜日の場合、年の最初の週はその年の最初の月曜日から開始します。
Calweekofyear	CWK	1 ～ 53。指定した日付がその年の第何週であるかを示します。 ISO の週のシステムと ISO 8601 の日付と時刻の標準の詳細については、ISO week date を参照してください。
Caldayofweek	CDW	1 ～ 7(月曜日 = 1、...、日曜日 = 7)
TZOffset	TZ	-840 ～ 840

関連情報

SQL データ型 [278 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.13.5.1.4 その他の関数

その他の関数は、算術式、文字列式、日付/時刻式、他の関数の戻り値に対して操作を実行します。

SQL Anywhere 関数のリスト

次の各種関数を使用できます。

- ARGN 関数 [その他]
- COALESCE 関数 [その他]
- CONFLICT 関数 [その他]
- ERRORMSG 関数 [その他]
- ESTIMATE 関数 [その他]
- ESTIMATE_SOURCE 関数 [その他]
- EXPERIENCE_ESTIMATE 関数 [その他]
- EXPLANATION 関数 [その他]
- EXPRTYPE 関数 [その他]
- GET_IDENTITY 関数 [その他]
- GRAPHICAL_PLAN 関数 [その他]
- GREATER 関数 [その他]
- IDENTITY 関数 [その他]
- IFNULL 関数 [その他]
- INDEX_ESTIMATE 関数 [その他]
- ISNULL 関数 [その他]
- LESSER 関数 [その他]
- NEWID 関数 [その他]
- NULLIF 関数 [その他]
- NUMBER 関数 [その他]
- PLAN 関数 [その他]
- REWRITE 関数 [その他]
- ROW_NUMBER 関数 [その他]
- SQLDIALECT 関数 [その他]
- SQLFLAGGER 関数 [その他]
- ERROR_LINE 関数 [その他]
- TRACEBACK 関数 [その他]
- TRANSACTSQL 関数 [その他]
- VAREXISTS 関数 [その他]
- WATCOMSQL 関数 [その他]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

Ultra Light 関数のリスト

次の各種関数を使用できます。

ARGN 関数 [その他]
COALESCE 関数 [その他]
EXPLANATION 関数 [その他]
GREATER 関数 [その他]
IFNULL 関数 [その他]
ISNULL 関数 [その他]
LESSER 関数 [その他]
NEWID 関数 [その他]
NULLIF 関数 [その他]

1.13.5.1.5 数値関数

数値関数は、数値データ型の算術演算を実行したり、数値情報を返したりします。

SQL Anywhere 関数のリスト

次の数値関数を使用できます。

ABS 関数 [数値]
ACOS 関数 [数値]
ASIN 関数 [数値]
ATAN 関数 [数値]
ATAN2 関数 [数値]
CEILING 関数 [数値]
COS 関数 [数値]
COT 関数 [数値]
DEGREES 関数 [数値]
EXP 関数 [数値]
FLOOR 関数 [数値]
LOG 関数 [数値]
LOG10 関数 [数値]
MOD 関数 [数値]
PI 関数 [数値]
POWER 関数 [数値]
RADIANS 関数 [数値]
RAND 関数 [数値]

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

REMAINDER 関数 [数値]
ROUND 関数 [数値]
SIGN 関数 [数値]
SIN 関数 [数値]
SQRT 関数 [数値]
TAN 関数 [数値]
TRUNCNUM 関数 [数値]

Ultra Light 関数のリスト

次の数値関数を使用できます。

ABS 関数 [数値]
ACOS 関数 [数値]
ASIN 関数 [数値]
ATAN 関数 [数値]
ATAN2 関数 [数値]
CEILING 関数 [数値]
COS 関数 [数値]
COT 関数 [数値]
DEGREES 関数 [数値]
EXP 関数 [数値]
FLOOR 関数 [数値]
LOG 関数 [数値]
LOG10 関数 [数値]
MOD 関数 [数値]
PI 関数 [数値]
POWER 関数 [数値]
RADIANS 関数 [数値]
REMAINDER 関数 [数値]
ROUND 関数 [数値]
SIGN 関数 [数値]
SIN 関数 [数値]
SQRT 関数 [数値]
TAN 関数 [数値]
TRUNCNUM 関数 [数値]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.13.5.1.6 空間関数 - Ultra Light

空間データは、定義された空間内のオブジェクトの位置、形、および方向を記述するデータです。Ultra Light には、ポイント形式の空間データ用の記憶領域およびデータ管理機能が用意されており、これらの機能を使用して、地理的ロケーションに関する情報やルート指定情報などを格納できます。

Ultra Light には、他の製品との互換性を確保するように設計された一連の SQL 空間関数が用意されています。空間データに対するアクセスと操作は、これらの関数とコンストラクタを使用して行います。

関数のリスト

次の空間関数を使用できます。

- ST_AsBinary 関数 [空間] - Ultra Light
- ST_AsExtText 関数 [空間] - Ultra Light
- ST_AsText 関数 [空間] - Ultra Light
- ST_Distance 関数 [空間] - Ultra Light
- ST_Equals 関数 [空間] - Ultra Light
- ST_IntersectsRect 関数 [空間] - Ultra Light
- ST_Point 関数 [空間] - Ultra Light
- ST_PointFromExtText 関数 [空間] - Ultra Light
- ST_PointFromText 関数 [空間] - Ultra Light
- ST_PointFromWKB 関数 [空間] - Ultra Light
- ST_SRID 関数 [空間] - Ultra Light
- ST_X 関数 [空間] - Ultra Light
- ST_Y 関数 [空間] - Ultra Light

1.13.5.1.7 文字列関数

文字列関数は、文字列に対して変換、抽出、操作の演算を実行したり、文字列に関する情報を返したりします。

マルチバイト文字セットを操作する場合は、使用する関数が文字とバイトのどちらの情報を返すかを十分に確認してください。

SQL Anywhere 関数のリスト

次の文字列関数を使用できます。

ASCII 関数 [文字列]
BASE64_DECODE 関数 [文字列]
BASE64_ENCODE 関数 [文字列]

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

BYTE_LENGTH 関数 [文字列]
BYTE_SUBSTR 関数 [文字列]
CHAR 関数 [文字列]
CHARINDEX 関数 [文字列]
CHAR_LENGTH 関数 [文字列]
COMPARE 関数 [文字列]
COMPRESS 関数 [文字列]
CSCONVERT 関数 [文字列]
DECOMPRESS 関数 [文字列]
DECRYPT 関数 [文字列]
DIFFERENCE 関数 [文字列]
ENCRYPT 関数 [文字列]
HASH 関数 [文字列]
INSERTSTR 関数 [文字列]
LCASE 関数 [文字列]
LEFT 関数 [文字列]
LENGTH 関数 [文字列]
LOCATE 関数 [文字列]
LOWER 関数 [文字列]
LTRIM 関数 [文字列]
NCHAR 関数 [文字列]
PATINDEX 関数 [文字列]
READ_CLIENT_FILE 関数 [文字列]
READ_SERVER_FILE 関数 [文字列]
REGEXP_SUBSTR 関数 [文字列]
REPEAT 関数 [文字列]
REPLACE 関数 [文字列]
REPLICATE 関数 [文字列]
REVERSE 関数 [文字列]
RIGHT 関数 [文字列]
RTRIM 関数 [文字列]
SIMILAR 関数 [文字列]
SORTKEY 関数 [文字列]
SOUNDEX 関数 [文字列]
SPACE 関数 [文字列]
STR 関数 [文字列]
STRING 関数 [文字列]
STRTOUUID 関数 [文字列]
STUFF 関数 [文字列]
SUBSTRING 関数 [文字列]
TO_CHAR 関数 [文字列]
TO_NCHAR 関数 [文字列]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

TRIM 関数 [文字列]
UCASE 関数 [文字列]
UNICODE 関数 [文字列]
UNISTR 関数 [文字列]
UPPER 関数 [文字列]
UUIDTOSTR 関数 [文字列]
XMLCONCAT 関数 [文字列]
XMLELEMENT 関数 [文字列]
XMLFOREST 関数 [文字列]
XMLGEN 関数 [文字列]

Ultra Light 関数のリスト

次の文字列関数を使用できます。

ASCII 関数 [文字列]
BYTE_LENGTH 関数 [文字列]
BYTE_SUBSTR 関数 [文字列]
CHAR 関数 [文字列]
CHARINDEX 関数 [文字列]
CHAR_LENGTH 関数 [文字列]
DIFFERENCE 関数 [文字列]
INSERTSTR 関数 [文字列]
LCASE 関数 [文字列]
LEFT 関数 [文字列]
LENGTH 関数 [文字列]
LOCATE 関数 [文字列]
LOWER 関数 [文字列]
LTRIM 関数 [文字列]
PATINDEX 関数 [文字列]
REPEAT 関数 [文字列]
REPLACE 関数 [文字列]
REPLICATE 関数 [文字列]
RIGHT 関数 [文字列]
RTRIM 関数 [文字列]
SIMILAR 関数 [文字列]
SOUNDEX 関数 [文字列]
SPACE 関数 [文字列]
STR 関数 [文字列]
STRING 関数 [文字列]
STRTOUUID 関数 [文字列]

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

STUFF 関数 [文字列]
SUBSTRING 関数 [文字列]
TRIM 関数 [文字列]
UCASE 関数 [文字列]
UPPER 関数 [文字列]
UUIDTOSTR 関数 [文字列]

1.13.5.1.8 システム関数

システム関数は、システム情報を返します。

関数のリスト

次のシステム関数を使用できます。

CONNECTION_EXTENDED_PROPERTY 関数 [文字列]
CONNECTION_PROPERTY 関数 [システム]
DATALENGTH 関数 [システム]
DB_ID 関数 [システム]
DB_NAME 関数 [システム]
DB_EXTENDED_PROPERTY 関数 [システム]
DB_PROPERTY 関数 [システム]
EVENT_CONDITION 関数 [システム]
EVENT_CONDITION_NAME 関数 [システム]
EVENT_PARAMETER 関数 [システム]
NEXT_CONNECTION 関数 [システム]
NEXT_DATABASE 関数 [システム]
PROPERTY 関数 [システム]
PROPERTY_DESCRIPTION 関数 [システム]
PROPERTY_NAME 関数 [システム]
PROPERTY_NUMBER 関数 [システム]
SUSER_ID 関数 [システム]
SUSER_NAME 関数 [システム]
TSEQUAL 関数 [システム] (旧式)
USER_ID 関数 [システム]
USER_NAME 関数 [システム]
DB_PROPERTY 関数 [システム]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

Ultra Light 関数

DB_PROPERTY 関数 [システム]
ML_GET_SERVER_NOTIFICATION 関数 [システム]
SYNC_PROFILE_OPTION_VALUE 関数 [システム]

SQL Anywhere の注意

- db_id、db_name、datalength は、組み込み関数として実装されます。
- 一部のシステム関数は、ストアドプロシージャとして実装されます。

他の場所で説明されていないシステム関数は次の表に注記してあります。これらの関数は、ストアドプロシージャとして実装されます。

構文: COL_LENGTH

```
COL_LENGTH( @object_name, @column_name )
```

指定されたカラムについて INTEGER によって定義された長さを返します。`@object_name` には、所有者を含めることができます ('GROUPO.Customers' など)。

構文: COL_TERM

```
COL_NAME( @object_id, @column_id [, @database_id ] )
```

CHAR(128) カラムの名前を返します。

構文: INDEX_COL

```
INDEX_COL ( @table_name, @index_id, @key_# [, @user_id ] )
```

インデックスカラムの CHAR(128) の名前を返します。`@table_name` には、所有者を含めることができます ('GROUPO.Customers' など)。

構文: OBJECT_ID

```
OBJECT_ID( @object_name )
```

INTEGER オブジェクト ID を返します。`@object_name` には、所有者を含めることができます ('GROUPO.Customers' など)。

構文: OBJECT_NAME

```
OBJECT_NAME ( @object_id [, @database_id ] )
```

CHAR(128) オブジェクトの名前を返します。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

1.13.5.2 関数

関数を 1 つずつリストし、その右側に関数のタイプ (数値、文字など) を示します。

このセクションの内容:

ABS 関数 [数値] [328 ページ]
数値式の絶対値を返します。

ACOS 関数 [数値] [329 ページ]
数値式のアークコサインをラジアン単位で返します。

ARGN 関数 [その他] [330 ページ]
引数リストから選択された引数を返します。

ASCII 関数 [文字列] [331 ページ]
文字列式の最初のバイトの ASCII 値を整数で返します。

ASIN 関数 [数値] [332 ページ]
数値のアークサインをラジアン単位で返します。

ATAN 関数 [数値] [333 ページ]
数値のアークタンジェントをラジアン単位で返します。

ATAN2 関数 [数値] [334 ページ]
2 つの数の比率のアークタンジェントをラジアン単位で返します。

AVG 関数 [集合] [335 ページ]
対象となるローセットの、数値式の平均値またはユニークな値からなるセットの平均値を計算します。

BYTE_LENGTH 関数 [文字列] [337 ページ]
文字列のバイト数を返します。

BYTE_SUBSTR 関数 [文字列] [338 ページ]
文字列の部分文字列を返します。部分文字列は、文字ではなくバイトを使用して計算されます。

CAST 関数 [データ型変換] [339 ページ]
指定されたデータ型に変換した式の値を返します。

CEILING 関数 [数値] [342 ページ]
指定した値以上になる最初の整数を返します。正の数値の場合、この処理は丸めとも呼ばれています。

CHAR 関数 [文字列] [343 ページ]
数値の ASCII 値を持つ文字を返します。

CHAR_LENGTH 関数 [文字列] [344 ページ]
文字列の文字数を返します。

CHARINDEX 関数 [文字列] [345 ページ]
ある文字列内で 1 つの文字列の位置を返します。

COALESCE 関数 [その他] [346 ページ]
リストの中から NULL でない最初の式を返します。この関数は ISNULL 関数と同じです。

CONVERT 関数 [データ型変換] [347 ページ]
指定されたデータ型に変換した式を返します。

COS 関数 [数値] [350 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

引数で与えられた角度のコサインをラジアン単位で返します。

COT 関数 [数値] [351 ページ]
引数で与えられた角度のコタンジェントをラジアン単位で返します。

COUNT 関数 [集合] [352 ページ]
指定されたパラメータに従って、グループのロー数をカウントします。

COUNT_UPLOAD_ROWS 関数 [集合] [353 ページ]
次回の同期でアップロードされるローの数を返します。

DATALENGTH 関数 [システム] [354 ページ]
式の結果に必要な基本となる記憶領域の長さ (バイト) を返します。

DATE 関数 [日付と時刻] [355 ページ]
式を日付に変換し、時間、分、秒を削除します。

DATEADD 関数 [日付と時刻] [356 ページ]
日付の単位をその引数に追加することによって求められる TIMESTAMP または TIMESTAMP WITH TIME ZONE の値を返します。

DATEDIFF 関数 [日付と時刻] [358 ページ]
2 つの日付間の期間を返します。

DATEFORMAT 関数 [日付と時刻] [360 ページ]
日付式を表す文字列を、指定した形式で返します。

DATENAME 関数 [日付と時刻] [361 ページ]
TIMESTAMP または TIMESTAMP WITH TIME ZONE の値の特定部分の名前 (月の名前 "June" など) を文字列で返します。

DATEPART 関数 [日付と時刻] [362 ページ]
TIMESTAMP または TIMESTAMP WITH TIME ZONE の値の一部を返します。

DATETIME 関数 [日付と時刻] [363 ページ]
式を TIMESTAMP 値に変換します。

DAY 関数 [日付と時刻] [364 ページ]
引数の日付を 1 ～ 31 の間の整数で返します。

DAYNAME 関数 [日付と時刻] [365 ページ]
日付から曜日の名前を返します。

DAYS 関数 [日付と時刻] [366 ページ]
TIMESTAMP を操作するか、2 つの TIMESTAMP 値の間の日数を返します。

DB_PROPERTY 関数 [システム] [368 ページ]
指定されたデータベースのプロパティの値を返します。

DEGREES 関数 [数値] [369 ページ]
数値をラジアンから度数に変換します。

DIFFERENCE 関数 [文字列] [370 ページ]
2 つの文字列式の SOUNDEX 値の差を返します。

DOW 関数 [日付と時刻] [371 ページ]
指定した日付の曜日を表す 1 ～ 7 の数を返します (日曜日 = 1、月曜日 = 2、以下同様)。

EXP 関数 [数値] [372 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

自然対数の底 e を、与えられた引数でべき乗した結果を返します。

EXPLANATION 関数 [その他] [373 ページ]
SQL 文の最適化方法をプレーンテキスト文字列で返します。

EXTRACT 関数 [日付と時刻] [375 ページ]
TIMESTAMP 式から日付要素を返します。

FLOOR 関数 [数値] [376 ページ]
指定された値以下の、最大の整数値を返します。

GETDATE 関数 [日付と時刻] [377 ページ]
現在の年、月、日、時、分、秒、秒以下を返します。

GREATER 関数 [その他] [378 ページ]
2 つのパラメータ値のうち、より大きい値を返します。

HEXTOINT 関数 [データ型変換] [379 ページ]
16 進文字列と同等の 10 進整数を返します。

HOUR 関数 [日付と時刻] [380 ページ]
TIMESTAMP 値の時間部分を返します。

HOURS 関数 [日付と時刻] [381 ページ]
TIMESTAMP を操作するか、2 つの TIMESTAMP 値の間の時間数を返します。

IFNULL 関数 [その他] [383 ページ]
ある式が NULL かどうかと、値を返すかどうかを評価します。

INSERTSTR 関数 [文字列] [384 ページ]
別の文字列の指定された位置に文字列を挿入します。

INTTOHEX 関数 [データ型変換] [385 ページ]
整数に対応する 16 進値の文字列を返します。

ISDATE 関数 [データ型変換] [386 ページ]
文字列引数を日付に変換できるかどうかをテストします。

ISNULL 関数 [その他] [387 ページ]
リストの中から NULL でない最初の式を返します。この関数は COALESCE 関数と同じです。

LCASE 関数 [文字列] [388 ページ]
文字列中のすべての文字を小文字に変換します。

LEFT 関数 [文字列] [389 ページ]
文字列の先頭からいくつかの文字を返します。

LENGTH 関数 [文字列] [390 ページ]
指定した文字列の文字数を返します。

LESSER 関数 [その他] [391 ページ]
2 つのパラメータ値のうち、より小さい値を返します。

LIST 関数 [集合] [392 ページ]
グループ内のローごとに値のデリミタ付きリストを返します。

LOCATE 関数 [文字列] [395 ページ]
異なる文字列内のある文字列の位置を返します。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

LOG 関数 [数値] [397 ページ]
数の自然対数を返します。

LOG10 関数 [数値] [398 ページ]
数値の対数 (基数 10) を返します。

LOWER 関数 [文字列] [399 ページ]
文字列中のすべての文字を小文字に変換します。

LTRIM 関数 [文字列] [400 ページ]
文字列の先行ブランクを削除します。

MAX 関数 [集合] [401 ページ]
各ローグループで見つかった式の最大値を返します。

MICROSECOND 関数 [日付と時刻] [402 ページ]
TIMESTAMP 式のマイクロ秒の部分を返します。

MILLISECOND 関数 [日付と時刻] [403 ページ]
TIMESTAMP 式のミリ秒の部分を返します。

MIN 関数 [集合] [404 ページ]
各ローグループで見つかった式の最小値を返します。

MINUTE 関数 [日付と時刻] [405 ページ]
TIMESTAMP 値の分部分を返します。

MINUTES 関数 [日付と時刻] [406 ページ]
TIMESTAMP を操作するか、2 つの TIMESTAMP 値の間の分境界の数を返します。

ML_GET_SERVER_NOTIFICATION 関数 [システム] [408 ページ]
この関数を使用すると、Ultra Light ユーザはライトウェイトポーリングを使用して、サーバ起動同期要求について Mobile Link サーバ上の Notifier にクエリできます。

MOD 関数 [数値] [409 ページ]
整数を整数で割ったときの余りを返します。

MONTH 関数 [日付と時刻] [410 ページ]
指定した日付の月を返します。

MONTHNAME 関数 [日付と時刻] [411 ページ]
日付から月の名前を返します。

MONTHS 関数 [日付と時刻] [412 ページ]
TIMESTAMP を操作するか、2 つの TIMESTAMP 値の間の月境界の数を返します。

NEWID 関数 [その他] [414 ページ]
UUID (ユニバーサルユニーク識別子) 値を生成します。UUID は、GUID (グローバルユニーク識別子) と同じです。

NOW 関数 [日付と時刻] [415 ページ]
現在の日付と時刻を TIMESTAMP 値で返します。精度はシステムクロックの精度によって制限されます。

NULLIF 関数 [その他] [416 ページ]
式を比較して、CASE 式の省略形を提供します。

PATINDEX 関数 [文字列] [417 ページ]
文字列のパターンが最初に出現した開始位置を表す整数を返します。

PI 関数 [数値] [420 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

PI の数値を返します。

POWER 関数 [数値] [421 ページ]
数のべき乗を表す数を計算します。

QUARTER 関数 [日付と時刻] [421 ページ]
指定した TIMESTAMP 式から、四半期を示す数を返します。

RADIANS 関数 [数値] [422 ページ]
度数をラジアンに変換します。

REMAINDER 関数 [数値] [423 ページ]
整数を整数で割ったときの余りを返します。

REPEAT 関数 [文字列] [424 ページ]
文字列を指定された回数だけ連結します。

REPLACE 関数 [文字列] [425 ページ]
文字列を別の文字列で置換し、新しい結果を返します。

REPLICATE 関数 [文字列] [427 ページ]
文字列を指定された回数だけ連結します。

RIGHT 関数 [文字列] [428 ページ]
文字列の一番右側にある文字を返します。

ROUND 関数 [数値] [429 ページ]
`numeric-expression` を `integer-expression` で指定した小数点以下の桁数に丸めます。

RTRIM 関数 [文字列] [430 ページ]
文字列の後続ブランクを削除します。

SECOND 関数 [日付と時刻] [431 ページ]
TIMESTAMP 引数の秒の値を返します。

SECONDS 関数 [日付と時刻] [432 ページ]
TIMESTAMP を操作するか、2 つの TIMESTAMP 値の間の秒境界の数を返します。

SHORT_PLAN 関数 [その他] [434 ページ]
SQL 文の Ultra Light プランの最適化方法を短い記述の文字列で返します。この記述は、EXPLANATION 関数が返す記述と同じです。

SIGN 関数 [数値] [435 ページ]
指定された数値の符号 (正または負) を返します。

SIMILAR 関数 [文字列] [436 ページ]
2 つの文字列の類似性を示す数を返します。

SIN 関数 [数値] [437 ページ]
数のサインを返します。

SOUNDEX 関数 [文字列] [438 ページ]
文字列の発音を表す数を返します。

SPACE 関数 [文字列] [439 ページ]
指定した数のスペースを返します。

SQRT 関数 [数値] [440 ページ]
数の平方根を返します。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

ST_AsBinary 関数 [空間] - Ultra Light [441 ページ]
指定されたジオメトリを表すバイナリ文字列を返します。

ST_AsExtText 関数 [空間] - Ultra Light [441 ページ]
指定されたジオメトリを表すバイナリ文字列を返します。

ST_AsText 関数 [空間] - Ultra Light [442 ページ]
指定されたジオメトリを表すバイナリ文字列を返します。

ST_Distance 関数 [空間] - Ultra Light [443 ページ]
指定した 2 つのジオメトリ値間の最短距離を返します。

ST_Equals 関数 [空間] - Ultra Light [444 ページ]
ST_Geometry 値が別の ST_Geometry 値と空間的に等しいかどうかをテストします。2 つのジオメトリ値が同じ x 座標と y 座標を使用し、同じ参照系にある場合、値は等しいと見なされます。

ST_IntersectsRect 関数 [空間] - Ultra Light [445 ページ]
2 つのポイント (最小と最大) によって指定されたボックス内にポイントがあるかどうかをテストします。

ST_Point 関数 [空間] - Ultra Light [446 ページ]
x 座標と y 座標に基づくポイントを作成します。

ST_PointFromExtText 関数 [空間] - Ultra Light [447 ページ]
ST_Geometry の EWKT 表現を含む VARCHAR 値から変換される ST_Geometry 値を返します。

ST_PointFromText 関数 [空間] - Ultra Light [447 ページ]
ST_Geometry の WKT 表現を含む VARCHAR 値から変換される ST_Geometry 値を返します。

ST_PointFromWKB 関数 [空間] - Ultra Light [448 ページ]
ST_Geometry の WKB 表現を含む BINARY 値から変換される ST_Geometry 値を返します。

ST_SRID 関数 [空間] - Ultra Light [449 ページ]
ジオメトリ値に関連付けられている空間参照系 (SRID) を取得します。

ST_X 関数 [空間] - Ultra Light [450 ページ]
ST_Geometry 値の x 座標を返します。

ST_Y 関数 [空間] - Ultra Light [451 ページ]
ST_Geometry 値の y 座標を返します。

STR 関数 [文字列] [452 ページ]
指定した数に相当する文字列を返します。

STRING 関数 [文字列] [453 ページ]
1 つ以上の文字列を連結して 1 つの長い文字列にします。

STRTOUUID 関数 [文字列] [454 ページ]
文字列の値をユニークな識別子 (UUID または GUID) の値に変換します。

STUFF 関数 [文字列] [455 ページ]
1 つの文字列から複数の文字を削除して、別の文字列に置き換えます。

SUBSTRING 関数 [文字列] [457 ページ]
文字列の部分文字列を返します。

SUM 関数 [集合] [459 ページ]
ローグループごとに、指定された式の合計を返します。

SWITCHOFFSET 関数 [日付と時刻] [460 ページ]

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

元のタイムゾーンオフセットから指定のタイムゾーンオフセットに変換される TIMESTAMP WITH TIME ZONE 値を返します。

SYNC_PROFILE_OPTION_VALUE 関数 [システム] - Ultra Light [461 ページ]
指定されたオプション名に対応するオプションの値を返します。

TAN 関数 [数値] [462 ページ]
数のタンジェントを返します。

TODATETIMEOFFSET 関数 [日付と時刻] [463 ページ]
指定のタイムゾーンオフセットを使用して、TIMESTAMP 値を TIME STAMP WITH TIME ZONE 値に変換します。

TODAY 関数 [日付と時刻] [464 ページ]
現在の日付を DATE 値で返します。

TRIM 関数 [文字列] [465 ページ]
先行ブランクと後続ブランクを文字列から削除します。

TRUNCNUM 関数 [数値] [466 ページ]
指定した桁数で小数点以下を切り捨てます。

UCASE 関数 [文字列] [467 ページ]
文字列中のすべての文字を大文字に変換します。

UPPER 関数 [文字列] [468 ページ]
文字列中のすべての文字を大文字に変換します。

UUIDTOSTR 関数 [文字列] [469 ページ]
ユニークな識別子の値 (UUID または GUID) を文字列の値に変換します。

WEEKS 関数 [日付と時刻] [470 ページ]
TIMESTAMP を操作するか、2 つの TIMESTAMP 値の間の週の数を返します。

YEAR 関数 [日付と時刻] [472 ページ]
TIMESTAMP 引数の年部分を返します。

YEARS 関数 [日付と時刻] [473 ページ]
TIMESTAMP を操作するか、2 つの TIMESTAMP 値の間の年数を返します。

YMD 関数 [日付と時刻] [474 ページ]
指定した年、月、日に相当する日付値を返します。引数は -32768 ～ 32767 の INTEGER です。

1.13.5.2.1 ABS 関数 [数値]

数値式の絶対値を返します。

構文

```
ABS( numeric-expression )
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

パラメータ

numeric-expression

絶対値が返される数値。

戻り値

数値式の絶対値。

数値式データ型	戻り値
INT	INT
FLOAT	FLOAT
DOUBLE	DOUBLE
NUMERIC	NUMERIC

例

次の文は、値 66 を返します。

```
SELECT ABS( -66 );
```

1.13.5.2.2 ACOS 関数 [数値]

数値式のアークコサインをラジアン単位で返します。

構文

```
ACOS( numeric-expression )
```

パラメータ

numeric-expression

角度のコサイン。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

戻り値

DOUBLE

備考

この関数は、引数を DOUBLE に変換し、倍精度浮動小数点で計算を行います。

例

次の文は、0.52 のアークコサイン値を返します。

```
SELECT ACOS( 0.52 );
```

関連情報

ASIN 関数 [数値] [332 ページ]
ATAN 関数 [数値] [333 ページ]
ATAN2 関数 [数値] [334 ページ]
COS 関数 [数値] [350 ページ]

1.13.5.2.3 ARGN 関数 [その他]

引数リストから選択された引数を返します。

構文

```
ARGN( integer-expression, expression [ ,...] )
```

パラメータ

integer-expression

式リスト内での引数の位置。

expression

関数に渡される任意のデータ型の式。すべて同じデータ型の式を指定してください。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

戻り値

integer-expression の値を n とした場合、引数リストの n 番目の引数 (1 から開始) を返します。

備考

式のデータ型は任意ですが、すべて同じデータ型にしてください。integer-expression は、1 からリスト内の式の数までの範囲内で指定してください。範囲外の値を指定すると、NULL が返されます。複数の式は、カンマで区切って指定します。

例

次の文は、値 6 を返します。

```
SELECT ARGN( 6, 1,2,3,4,5,6 );
```

1.13.5.2.4 ASCII 関数 [文字列]

文字列式の最初のバイトの ASCII 値を整数で返します。

構文

```
ASCII( string-expression )
```

パラメータ

string-expression

文字列。

戻り値

SMALLINT

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

備考

文字列が空の場合は、0 を返します。リテラル文字列は、引用符で囲んで指定します。データベース文字セットがマルチバイトであり、パラメータ文字列の最初の文字が複数バイトから構成される場合、結果は NULL です。

例

次の文は、値 90 を返します。

```
SELECT ASCII( 'Z' );
```

関連情報

CHAR 関数 [文字列] [343 ページ]
STRING 関数 [文字列] [453 ページ]

1.13.5.2.5 ASIN 関数 [数値]

数値のアークサインをラジアン単位で返します。

構文

```
ASIN( numeric-expression )
```

パラメータ

numeric-expression

角度のサイン。

戻り値

DOUBLE

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

備考

SIN 関数と ASIN 関数は逆変換の演算です。

この関数は、引数を DOUBLE に変換し、倍精度浮動小数点で計算を行います。

例

次の文は、0.52 のアークサイン値を返します。

```
SELECT ASIN( 0.52 );
```

関連情報

ACOS 関数 [数値] [329 ページ]
ATAN 関数 [数値] [333 ページ]
ATAN2 関数 [数値] [334 ページ]
SIN 関数 [数値] [437 ページ]

1.13.5.2.6 ATAN 関数 [数値]

数値のアークタンジェントをラジアン単位で返します。

構文

```
ATAN( numeric-expression )
```

パラメータ

numeric-expression

角度のタンジェント。

戻り値

DOUBLE

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

備考

この関数は、引数を DOUBLE に変換し、倍精度浮動小数点で計算を行います。

ATAN 関数と TAN 関数は逆変換の演算です。

例

次の文は、0.52 のアークタンジェント値を返します。

```
SELECT ATAN( 0.52 );
```

関連情報

ACOS 関数 [数値] [329 ページ]
ASIN 関数 [数値] [332 ページ]
ATAN2 関数 [数値] [334 ページ]
TAN 関数 [数値] [462 ページ]

1.13.5.2.7 ATAN2 関数 [数値]

2 つの数の比率のアークタンジェントをラジアン単位で返します。

構文

```
ATAN2 ( numeric-expression-1, numeric-expression-2 )
```

パラメータ

numeric-expression-1

アークタンジェントが計算される比率の分子。

numeric-expression-2

アークタンジェントが計算される比率の分母。

戻り値

DOUBLE

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

備考

この関数は、引数を DOUBLE に変換し、倍精度浮動小数点で計算を行います。

例

次の文は、比率 0.52 ～ 0.60 のアークタンジェント値を返します。

```
SELECT ATAN2( 0.52, 0.60 );
```

関連情報

ACOS 関数 [数値] [329 ページ]
ASIN 関数 [数値] [332 ページ]
ATAN 関数 [数値] [333 ページ]
TAN 関数 [数値] [462 ページ]

1.13.5.2.8 AVG 関数 [集合]

対象となるローセットの、数値式の平均値またはユニークな値からなるセットの平均値を計算します。

構文

数値式

```
AVG( [ ALL | DISTINCT ] numeric-expression )
```

Window 関数

```
AVG( [ ALL ] numeric-expression) OVER ( window-spec )
```

```
window-spec: 以下の備考部分を参照してください。
```

Ultra Light 数値式

```
AVG( [ DISTINCT ] numeric-expression )
```

パラメータ

[ALL] numeric-expression

各グループのローで平均値が計算される式。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

DISTINCT 句

グループごとにユニークな数値の平均を計算します。

戻り値

グループにローが含まれていない場合は、NULL 値を返します。

引数が DOUBLE の場合は DOUBLE を、それ以外の場合は NUMERIC を返します。

備考

この平均には、`numeric-expression` が NULL 値であるローは含まれません。

この関数ではオーバーフローエラーが発生することがあり、これにより、エラーが返される場合があります。オーバーフローエラーを回避するために、`numeric-expression` で CAST 関数を使用できます。

`window-spec` を使用してこの関数を指定した場合、SELECT 文で Window 関数として使用することを意味します。そのように、`window-spec` の要素は、関数構文の中 (インライン) に指定するか、または SELECT 文の WINDOW 句と組み合わせて指定できます。

例

次の文は、SQL Anywhere 17 Demo に接続されると値 49988.623200 を返します。

```
SELECT AVG( Salary ) FROM Employees;
```

次の文は、SQL Anywhere 17 Demo データベースに接続されると Products テーブルの製品価格の平均を返します。

```
SELECT AVG( DISTINCT UnitPrice ) FROM Products;
```

次の文は、サブクエリからの限定式と外部 SELECT ブロックからの外部参照 (p.Quantity) の両方が AVG の引数に含まれているため、SQL Anywhere 17 Demo に接続されると、SQLSTATE 42W68 エラーを返します。

```
SELECT * from GROUPO.Products as p
WHERE p.Quantity > ( SELECT AVG( 0.5 * p.Quantity + 0.5 * s.Quantity )
                     from GROUPO.SalesOrderItems as s
                     WHERE s.ProductID = p.ProductID )
```

関連情報

SUM 関数 [集合] [459 ページ]
COUNT 関数 [集合] [352 ページ]

 © 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.13.5.2.9 BYTE_LENGTH 関数 [文字列]

文字列のバイト数を返します。

構文

```
BYTE_LENGTH( string-expression )
```

パラメータ

string-expression

長さが計算される文字列。

戻り値

INT

備考

`string-expression` の後続空白スペースは、返される長さに含まれます。

NULL 文字列の戻り値は NULL です。

マルチバイト文字セットの文字列の場合、BYTE_LENGTH の値は、CHAR_LENGTH で返される文字数と異なることがあります。

この関数は NCHAR の入力または出力をサポートしています。

Ultra Light: Ultra Light では NCHAR の入力または出力がサポートされていません。

例

次の文は、値 12 を返します。

```
SELECT BYTE_LENGTH( 'Test Message' );
```

関連情報

CHAR_LENGTH 関数 [文字列] [344 ページ]

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

DATALENGTH 関数 [システム] [354 ページ]
LENGTH 関数 [文字列] [390 ページ]
STRING 関数 [文字列] [453 ページ]

1.13.5.2.10 BYTE_SUBSTR 関数 [文字列]

文字列の部分文字列を返します。部分文字列は、文字ではなくバイトを使用して計算されます。

構文

```
BYTE_SUBSTR( source-string, start-position [, length ] )
```

パラメータ

source-string

部分文字列が取得される文字列。`source-string` の長さに制限はありません。

start-position

部分文字列の開始位置を示す整数式。正の整数は、文字列の先頭から開始し、最初の文字は位置 1 です。負の整数は、文字列の末尾から始まる部分文字列を指定し、最後の文字の位置は -1 です。

length

部分文字列の長さを示す整数式。正の `length` は、取得するバイト数を開始位置から開始することを指定します。負の `length` は、開始位置から最大 `length` バイト左側の文字を返します。

戻り値

入力に応じて BINARY または LONG BINARY。

備考

`length` を指定すると、部分文字列は指定したバイト数に制限されます。`start-position` と `length` には、正または負の値を指定できます。負の数と正の数を適切に組み合わせて使用して、文字列の先頭または末尾のどちらかから部分文字列を取得します。

`start-position` が 0 で length が負でない場合は、1 の `start-position` 値が使用されます。`start-position` が 0 で `length` が負の場合は、-1 の start 値が使用されます。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

引数 source-string は、バイナリデータ型に変換可能などのデータ型でもかまいません。バイナリ文字列として扱われます。部分文字列は、文字ではなくバイトを使用して計算されます。

例

次の文は、値 Test を返します。

```
SELECT BYTE_SUBSTR( 'Test Message', 1, 4 );
```

関連情報

SUBSTRING 関数 [文字列] [457 ページ]
STRING 関数 [文字列] [453 ページ]

1.13.5.2.11 CAST 関数 [データ型変換]

指定されたデータ型に変換した式の値を返します。

構文

```
CAST( expression AS datatype )
```

パラメータ

expression

変換される式。

datatype

キャスト後の式のデータ型。データ型を明示的にセットするか、%TYPE 属性を指定して、データ型をテーブルまたはビュー内のカラムのデータ型、または変数のデータ型に設定します。

戻り値

要求されたデータ型によって異なります。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

備考

CAST 関数を使用して文字列をトランケートする場合、string_rtruncation データベースオプションは OFF に設定する必要があります。OFF に設定されていない場合はエラーになります。文字列をトランケートするには、LEFT 関数を使用します。

文字列型の長さを指定しない場合は、適切な長さが選択されます。DECIMAL 変換で精度も位取りも指定しない場合は、データベースサーバによって適切な値が選択されます。

Ultra Light: CAST 関数で精度と位取りを明示的に示すことをおすすめします。変換できるかどうかは、変換で使用する値によります。変換エラーが発生しないように、元のデータ型の値は新しいデータ型と互換性がある必要があります。次のチャートを使用して、変換がサポートされているかどうかを判断します。

変換前: \ 変換後:	BIT	TINYINT	UNSIGNED SMALLINT	SMALLINT	UNSIGNED INTEGER	INTEGER	UNSIGNED BIGINT	BIGINT	FLOAT	REAL	DOUBLE	NUMERIC または DECIMAL	DATE	TIME	DATETIME または TIMESTAMP	TIMESTAMP WITH TIME ZONE	UNIQUEIDENTIFIER	BINARY または VARBINARY	LONG BINARY	CHAR または VARCHAR	LONG VARCHAR	ST_GEOMETRY
BIT		✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✗	✗	✗	✗	✗	✓	✓	✓	✗	✗
TINYINT	⚠		✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✗	✗	✗	✗	✗	✓	✗	✓	✗	✗
UNSIGNED SMALLINT	⚠	⚠		✓	✓	✓	✓	✓	✓	✓	✓	✓	✗	✗	✗	✗	✗	✓	✗	✓	✗	✗
SMALLINT	⚠	⚠	⚠		✓	✓	✓	✓	✓	✓	✓	✓	✗	✗	✗	✗	✗	✓	✗	✓	✗	✗
UNSIGNED INTEGER	⚠	⚠	⚠	⚠		✓	✓	✓	✓	✓	✓	✓	✗	✗	✗	✗	✗	✓	✗	✓	✗	✗
INTEGER	⚠	⚠	⚠	⚠	⚠		✓	✓	✓	✓	✓	✓	✗	✗	✗	✗	✗	✓	✗	✓	✗	✗
UNSIGNED BIGINT	⚠	⚠	⚠	⚠	⚠	⚠		✓	✓	✓	✓	✓	✗	✗	✗	✗	✗	✓	✗	✓	✗	✗
BIGINT	⚠	⚠	⚠	⚠	⚠	⚠	⚠		✓	✓	✓	✓	✗	✗	✗	✗	✗	✓	✗	✓	✗	✗
FLOAT	⚠	⚠	⚠	⚠	⚠	⚠	⚠	⚠		⚠	✓	✓	✗	✗	✗	✗	✗	⚠	✗	✓	✗	✗
REAL	⚠	⚠	⚠	⚠	⚠	⚠	⚠	⚠	✓		✓	✓	✗	✗	✗	✗	✗	✗	✗	✓	✗	✗
DOUBLE	⚠	⚠	⚠	⚠	⚠	⚠	⚠	⚠	✓	⚠		✓	✗	✗	✗	✗	✗	⚠	✗	✓	✗	✗
NUMERIC または DECIMAL	⚠	⚠	⚠	⚠	⚠	⚠	⚠	⚠	✓	✓	✓		✗	✗	✗	✗	✗	⚠	✗	✓	✗	✗
DATE	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗		✓	✓	✓	✗	✗	✗	✓	✗	✗
TIME	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✓		✓	✓	✗	✗	✓	✓	✗	✗
DATETIME または TIMESTAMP	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✓	✓		✓	✗	✗	✗	✓	✗	✗
TIMESTAMP WITH TIME ZONE	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓		✗	✗	✗	✓	✗	✗
UNIQUEIDENTIFIER	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗		⚠	✗	✓	✗	✗
BINARY または VARBINARY	✓	✓	✓	✓	✓	✓	✓	✓	✗	✗	✗	⚠	✗	✗	✗	✗	⚠		✓	✓	✗	⚠
LONG BINARY	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✓		✗	✗	✗
CHAR または VARCHAR	⚠	⚠	⚠	⚠	⚠	⚠	⚠	⚠	⚠	⚠	⚠	⚠	⚠	⚠	⚠	⚠	⚠	✓	✗		✓	⚠
LONG VARCHAR	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✓		✗
ST_GEOMETRY	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	⚠	✗	✓	✗	

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

シンボル	互換性
✓	常に変換
✗	変換しない
⚠	値依存

注記

Ultra Light の場合:

- VARBINARY と UNIQUEIDENTIFIER 間で変換を行うには、VARBINARY 値が 16 バイト長である必要があります。
- NUMERIC と VARBINARY 間で変換を行うには、NUMERIC ソースが BIGINT としてもキャスト可能な値を持っている必要があります。
- VARCHAR から ST_GEOMETRY に変換するには、VARCHAR ソースが WKT フォーマットまたは EWKT フォーマットのいずれかで有効なジオメトリを表している必要があります。
- VARBINARY から ST_GEOMETRY に変換するには、VARBINARY ソースが WKB フォーマットで有効なジオメトリを表している必要があります。
- WKB または WKT でフォーマットされているソースから ST_GEOMETRY にキャストするには、0 の SRID が ST_GEOMETRY 値に割り当てられている必要があります。ST_GEOMETRY からキャストする場合、VARCHAR 値は EWKT でフォーマットされ、VARBINARY 値は WKB でフォーマットされます。

HEXTOINT および INTTOHEX 関数を使用すると、16 進値変換を行うことができます。

例

次の関数は、文字列を日付として使用することを保証します。

```
SELECT CAST( '2000-10-31' AS DATE );
```

式 `1 + 2` の値を計算し、その結果を 1 文字の文字列にキャストします。

```
SELECT CAST( 1 + 2 AS CHAR );
```

VARCHAR と ST_GEOMETRY 間のキャストは通常暗黙的に行われます。たとえば、次の文は、ST_POINT 関数と VARCHAR を使用して ST_GEOMETRY カラムに値を追加します。それぞれの値はテーブルカラム全体で ST_GEOMETRY データ型に暗黙的にキャストされますが、その結果は依然として VARCHAR として示されます。

```
INSERT INTO T1 VALUES (2, ST_POINT(1,2,0), 'SRID=2163;Point(1 2)');
```

次の文は、Employees テーブルの BirthDate カラム (DATE データ型) に定義されているデータ型に値をキャストします。

```
SELECT CAST ( '1966-10-30' AS Employees.BirthDate%TYPE );
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

Ultra Light: 次のように CAST 関数を使用して、文字列を短縮できます。

```
SELECT CAST ( 'Surname' AS CHAR(5) );
```

関連情報

CONVERT 関数 [データ型変換] [347 ページ]

1.13.5.2.12 CEILING 関数 [数値]

指定した値以上になる最初の整数を返します。正の数値の場合、この処理は丸めとも呼ばれています。

構文

```
{ CEILING | CEIL } ( numeric-expression )
```

パラメータ

numeric-expression

切り上げ値を計算する対象の数値。

戻り値

DOUBLE

備考

この関数は、引数を DOUBLE に変換し、倍精度浮動小数点で計算を行います。

例

次の文は、値 60 を返します。

```
SELECT CEILING( 59.84567 );
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

関連情報

FLOOR 関数 [数値] [376 ページ]

1.13.5.2.13 CHAR 関数 [文字列]

数値の ASCII 値を持つ文字を返します。

構文

```
CHAR( integer-expression )
```

パラメータ

integer-expression

ASCII 文字に変換される数値。0 ～ 255 の範囲内の数を指定します。

戻り値

VARCHAR

備考

返される文字は、バイナリソート順に従って、現在のデータベース側文字セットの指定した数値式に対応しています。

整数式の値が 255 より大きいか、0 より小さい場合、CHAR は NULL を返します。

例

次の文を実行すると、値 Y が返ります。

```
SELECT CHAR( 89 );
```

関連情報

STRING 関数 [文字列] [453 ページ]

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

1.13.5.2.14 CHAR_LENGTH 関数 [文字列]

文字列の文字数を返します。

構文

```
CHAR_LENGTH ( string-expression )
```

パラメータ

string-expression

長さが計算される文字列。

戻り値

INT

備考

後続空白スペースは、返される長さに含まれます。

NULL 文字列の戻り値は NULL です。

マルチバイト文字セットの文字列の場合、CHAR_LENGTH 関数で返される値は、BYTE_LENGTH 関数で返されるバイト数と異なることがあります。

データ型が CHAR、VARCHAR、LONG VARCHAR、NCHAR の場合、CHAR_LENGTH 関数と LENGTH 関数を使用すると同じ結果が得られます。ただし、BINARY データ型とビット配列データ型には LENGTH 関数を使用します。この関数は NCHAR の入力または出力をサポートしています。

Ultra Light: データ型が CHAR、VARCHAR、LONG VARCHAR の場合、CHAR_LENGTH 関数と LENGTH 関数を使用すると同じ結果が得られます。ただし、BINARY データ型とビット配列データ型には LENGTH 関数を使用します。

例

次の文は、値 8 を返します。

```
SELECT CHAR_LENGTH( 'Chemical' );
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

関連情報

BYTE_LENGTH 関数 [文字列] [337 ページ]
STRING 関数 [文字列] [453 ページ]

1.13.5.2.15 CHARINDEX 関数 [文字列]

ある文字列内で 1 つの文字列の位置を返します。

構文

```
CHARINDEX( string-expression-1, string-expression-2 )
```

パラメータ

string-expression-1

検索する文字列。256 バイト未満の値にしてください。

string-expression-2

検索される文字列。

戻り値

INT

備考

`string-expression-1` の先頭文字の位置を 1 とします。検索される文字列内に、検索する文字列のインスタンスが複数存在する場合、CHARINDEX 関数は最初のインスタンスの位置を返します。

検索される文字列内に、検索する文字列が存在しない場合、CHARINDEX 関数は 0 を返します。

いずれかの引数が NULL の場合、結果も NULL になります。

この関数は NCHAR の入力または出力をサポートしています。

Ultra Light では NCHAR の入力または出力がサポートされていません。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

例

次の文は、姓が K という文字で始まる場合にのみ、Employees テーブルの Surname カラムと GivenName カラムから姓名を返します。

```
SELECT Surname, GivenName
FROM GROUPO.Employees
WHERE CHARINDEX( 'K', Surname ) = 1;
```

次の結果が返されます。

Surname	GivenName
Klobucher	James
Kuo	Felicia
Kelly	Moira

関連情報

SUBSTRING 関数 [文字列] [457 ページ]
REPLACE 関数 [文字列] [425 ページ]
LOCATE 関数 [文字列] [395 ページ]
STRING 関数 [文字列] [453 ページ]

1.13.5.2.16 COALESCE 関数 [その他]

リストの中から NULL でない最初の式を返します。この関数は ISNULL 関数と同じです。

構文

```
COALESCE( expression, expression [ ,...] )
```

パラメータ

expression

任意の式。

2 つ以上の式を関数に渡します。すべての式は比較可能であることが必要です。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

戻り値

この関数の戻り値は、指定した式によって異なります。具体的には、データベースサーバが関数を評価するとき、まず、式の比較が可能なデータ型を検索します。該当するデータ型が見つかると、データベースサーバは式を比較し、リストから NULL でない最初の式を返します。データベースサーバは、比較が可能な共通のデータ型を見つけることができないと、エラーを返します。

備考

結果が NULL になるのはすべての引数が NULL の場合のみです。

このパラメータにはスカラ型を指定できますが、同じ型を指定する必要はありません。

例

次の文は、値 34 を返します。

```
SELECT COALESCE( NULL, 34, 13, 0 );
```

関連情報

ISNULL 関数 [その他] [387 ページ]

1.13.5.2.17 CONVERT 関数 [データ型変換]

指定されたデータ型に変換した式を返します。

構文

```
CONVERT( datatype, expression [ , format-style ] )
```

パラメータ

datatype

変換後の式のデータ型です。データ型を明示的にセットするか、%TYPE 属性を指定して、データ型をテーブルまたはビュー内のカラムのデータ型、または変数のデータ型に設定します。

expression

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

変換される式。

format-style

出力値に適用されるスタイルコードです。このパラメータを使用するのは、文字列を日付または時刻のデータ型に変換するとき、またはその反対方向に変換するときです。次の表は、サポートされるスタイルコードと、そのスタイルコードで作成される出力形式の表現です。スタイルコードは世紀を出力形式に含めるかどうかによって 2 つのカラムに分けられます (たとえば、06 と 2006 など)。

スタイルコード 0 は、引数が指定されない場合に使用されます。

100 以上の位なし (yy) のスタイルコード	100 以上の位あり (yyyy) のスタイルコード	出力形式
-	0 または 100	Mmm dd yyyy hh:nnAA
1	101	mm/dd/yy[yy]
2	102	[yy]yy.mm.dd
3	103	dd/mm/yy[yy]
4	104	dd.mm.yy[yy]
5	105	dd-mm-yy[yy]
6	106	dd Mmm yy[yy]
7	107	Mmm dd, yy[yy]
8	108	hh:nn:ss
-	9 または 109	Mmm dd yyyy hh:nn:ss:sssAA
10	110	mm-dd-yy[yy]
11	111	[yy]yy/mm/dd
12	112	[yy]yymmdd
-	13 または 113	dd Mmm yyyy hh:nn:ss:sss (24 時間表記、ヨーロッパのデフォルト、ミリ秒、4 桁の年)
-	14 または 114	hh:nn:ss:sss (24 時間表記)
-	20 または 120	yyyy-mm-dd hh:nn:ss (24 時間表記、ODBC 標準、4 桁の年)
-	21 または 121	yyyy-mm-dd hh:nn:ss.sss (24 時間表記、ODBC 標準、ミリ秒、4 桁の年)

戻り値

指定されたデータ型によって異なります。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

備考

CONVERT 関数を使用して、文字列の解析時にあいまいさがない場合、文字列を DATE、TIME、または TIMESTAMP データ型に変換できます。format-style が指定されている場合、データベースサーバはこれを文字列の解析方法のヒントとして使用します。データベースサーバは、文字列を明確に解析できない場合にエラーを返します。

Ultra Light: この関数は CAST 関数と同様ですが、形式スタイルを指定して、日付と時刻のデータ型を変換するのに使用できます。

例

次の文は、形式スタイルの使用方法を示します。

```
SELECT CONVERT( CHAR( 20 ), OrderDate, 104 ) FROM GROUPO.SalesOrders;
```

OrderDate
16.03.2000
20.03.2000
23.03.2000
25.03.2000
...

```
SELECT CONVERT( CHAR( 20 ), OrderDate, 7 ) FROM GROUPO.SalesOrders;
```

OrderDate
Mar 16, 00
Mar 20, 00
Mar 23, 00
Mar 25, 00
...

次の文は、整数への変換を示し、値 5 を返します。

```
SELECT CONVERT( integer, 5.2 );
```

次の文は、Employees テーブルの BirthDate カラム (DATE データ型) に定義されているデータ型に値を変換します。

```
SELECT CONVERT ( Employees.BirthDate%TYPE, '1966-10-30' );
```

関連情報

Ultra Light の date_format 作成オプション [145 ページ]
CAST 関数 [データ型変換] [339 ページ]

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

1.13.5.2.18 COS 関数 [数値]

引数で与えられた角度のコサインをラジアン単位で返します。

構文

```
COS( numeric-expression )
```

パラメータ

numeric-expression

角度 (ラジアン)。

戻り値

この関数は、引数を DOUBLE に変換し、計算を倍精度浮動小数点で行い、結果を DOUBLE で返します。パラメータが NULL 値の場合、結果は NULL 値になります。

例

次の文は、角度 0.52 ラジアンのコサイン値を返します。

```
SELECT COS( 0.52 );
```

関連情報

ACOS 関数 [数値] [329 ページ]
COT 関数 [数値] [351 ページ]
SIN 関数 [数値] [437 ページ]
TAN 関数 [数値] [462 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.13.5.2.19 COT 関数 [数値]

引数で与えられた角度のコタンジェントをラジアン単位で返します。

構文

```
COT( numeric-expression )
```

パラメータ

numeric-expression

角度 (ラジアン)。

戻り値

この関数は、引数を DOUBLE に変換し、計算を倍精度浮動小数点で行い、結果を DOUBLE で返します。パラメータが NULL 値の場合、結果は NULL 値になります。

例

次の文は、0.52 のコタンジェント値を返します。

```
SELECT COT( 0.52 );
```

関連情報

COS 関数 [数値] [350 ページ]
SIN 関数 [数値] [437 ページ]
TAN 関数 [数値] [462 ページ]

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

1.13.5.2.20 COUNT 関数 [集合]

指定されたパラメータに従って、グループのロー数をカウントします。

構文

式

```
COUNT( [ * | [ ALL | DISTINCT ] expression ] )
```

Window 関数

```
COUNT( [ * | [ ALL ]expression ]) OVER ( window-spec )
```

```
window-spec: 以下の備考部分を参照してください。
```

Ultra Light 式

```
COUNT( [ * | [ DISTINCT ] expression ] )
```

パラメータ

*****

各グループの中のロー数を返します。セマンティック上、COUNT(*) と COUNT() は同義です。

[ALL] expression

`expression` の値が NULL でないロー数をグループごとに返します。

Ultra Light expression

`expression` の値が NULL でないロー数をグループごとに返します。

p

DISTINCT expression

`expression` の値が NULL でないすべてのローについて、重複しない `expression` 値の数をグループごとに返します。

戻り値

COUNT 関数は、INT データ型の値を返します。

COUNT は、NULL 値を返しません。グループにローがないか、またはグループにおいて `expression` の値がすべて NULL である場合、COUNT は 0 を返します。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

備考

SQL Anywhere では、COUNT 関数は、最大値 2147483647 を返します。大規模な結果セットをカウントする場合、結果に含まれるローの数が多くなる可能性がある場合、またはオーバーフローの可能性がある場合には、COUNT_BIG 関数を使用してください。`window-spec` を使用してこの関数を指定した場合、SELECT 文で Window 関数として使用することを意味します。そのように、`window-spec` の要素は、関数構文の中 (インライン) に指定するか、または SELECT 文の WINDOW 句と組み合わせて指定できます。

例

次の文は、ユニークな各都市と、その都市で働く従業員数を返します。

```
SELECT City, COUNT( * ) FROM GROUPO.Employees GROUP BY City;
```

関連情報

AVG 関数 [集合] [335 ページ]
SUM 関数 [集合] [459 ページ]

1.13.5.2.21 COUNT_UPLOAD_ROWS 関数 [集合]

次回の同期でアップロードされるローの数を返します。

構文

```
COUNT_UPLOAD_ROWS( pubs,threshold)
```

パラメータ

pubs

ローをチェックするパブリケーションのカンマ区切りのリスト。

threshold

カウントするローの最大数 (値 0 は最大限度を示します)。

戻り値

INT

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

例

次の文は、mypub1 と mypub2 にアップロードするローの総数を返します。

```
SELECT COUNT_UPLOAD_ROWS('mypub1,mypub2', 0);
```

1.13.5.2.22 DATALENGTH 関数 [システム]

式の結果に必要な基本となる記憶領域の長さ (バイト) を返します。

構文

```
DATALENGTH( expression )
```

パラメータ

expression

通常は、カラム名です。expression が文字列定数の場合は、引用符で囲みます。

戻り値

UNSIGNED INT

備考

DATALENGTH 関数の戻り値は次のとおりです。

データ型	DATALENGTH
BIT	1
TINYINT	1
SMALLINT	2
INTEGER	4
BIGINT	8
REAL	4

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

データ型	DATALENGTH
DOUBLE	8
TIME	8
DATE	4
TIMESTAMP	8
DATETIME	8
TIMESTAMP WITH TIME ZONE	29
UNIQUEIDENTIFIER	16
CHAR	データの長さ
VARCHAR	データの長さ
BINARY	データの長さ
VARBINARY	データの長さ
NCHAR	データの長さ
NVARCHAR	データの長さ
TEXT	データの長さ
NTEXT	データの長さ
IMAGE	データの長さ
XML	データの長さ

SQL Anywhere の場合、この関数は NCHAR の入力と出力をサポートしています。

例

次の文は、CompanyName カラムで最も長い文字列の長さ 27 を返します。

```
SELECT MAX( DATALENGTH( CompanyName ) )
FROM GROUPO.Customers;
```

次の文は、文字列 '8sdofinsv8s7a7s7gehe4h' の長さを返します。

```
SELECT DATALENGTH( '8sdofinsv8s7a7s7gehe4h' );
```

1.13.5.2.23 DATE 関数 [日付と時刻]

式を日付に変換し、時間、分、秒を削除します。

構文

```
DATE( expression )
```


© 2016 SAP SE or an SAP affiliate company. All rights reserved.

パラメータ

expression

日付形式に変換される値。通常、文字列。

戻り値

DATE

例

次の文は、値 1999-01-02 を日付として返します。

```
SELECT DATE( '1999-01-02 21:20:53' );
```

次の文は、SYSOBJECT システムビューにリストされるすべてのオブジェクトについて、作成日を返します。

```
SELECT DATE( creation_time ) FROM SYSOBJECT;
```

関連情報

Ultra Light の date_order 作成オプション [147 ページ]

1.13.5.2.24 DATEADD 関数 [日付と時刻]

日付の単位をその引数に追加することによって求められる TIMESTAMP または TIMESTAMP WITH TIME ZONE の値を返します。

構文

```
DATEADD( date-part, integer-expression, timestamp-expression )
```

```
date-part :
year
| quarter
| month
| week
| day
| dayofyear
| hour
| minute
| second
| millisecond
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

| *microsecond*

パラメータ

date-part

integer-expression があらわす日付要素。

integer-expression

timestamp-expression に追加する date-part 値の数。integer-expression には任意の数値型を指定できますが、値は INTEGER にトランケートされます。

timestamp-expression

変更される TIMESTAMP または TIMESTAMP WITH TIME ZONE の値。

戻り値

timestamp-expression が TIMESTAMP WITH TIME ZONE の場合は TIMESTAMP WITH TIME ZONE を返し、それ以外の場合は TIMESTAMP を返します。

例

次の文は TIMESTAMP 値 2016/05/02 0:00:00.000 を返します。

```
SELECT DATEADD( month, 12, '2015/05/02' );
```

次の文は TIMESTAMP 値 2015/05/02 04:00:00.000 を返します。

```
SELECT DATEADD( hour, 4, '2015/05/02' );
```

次の文は、TIMESTAMP WITH TIME ZONE 値 2015/05/06 11:33:00.000+04:00 を返します。

```
SELECT DATEADD( day, 4, CAST( '2015/05/02 11:33:00.000000+04:00' as TIMESTAMP
WITH TIME ZONE ));
```

関連情報

日付の単位の指定 [312 ページ]

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

1.13.5.2.25 DATEDIFF 関数 [日付と時刻]

2 つの日付間の期間を返します。

構文

```
DATEDIFF( date-part, date-expression-1, date-expression-2 )
```

```
date-part :
year
| quarter
| month
| week
| day
| dayofyear
| hour
| minute
| second
| millisecond
| microsecond
```

パラメータ

date-part

期間を測定する日付の単位を指定します。

date-expression-1

期間の開始日。この値を `date-expression-2` から引いて、2 つの引数の間の `date-parts` の数を返す。

date-expression-2

期間の終了日。この値から `Date-expression-1` を引いて、2 つの引数の間の `date-parts` の数を返す。

戻り値

年、四半期、月、週、日、通し日数を示す INT。時、分、秒、ミリ秒、マイクロ秒を表す BIGINT。

備考

この関数は、指定した 2 つの日付間に存在する日付の単位の数を計算します。結果は、日付の単位の数を表す (date-expression-2 - date-expression-1) と同値の符号付き整数です。

DATEDIFF 関数の結果が日付の単位の整数倍でない場合、結果は丸められずに切り捨てになります。

day を日付の単位として使用する場合、DATEDIFF 関数は指定された 2 つの時刻の間の午前 0 時の回数を返します。このとき、2 番目の日付は計算に含まれますが、最初の日付は含まれません。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

month を日付の単位として使用する場合、DATEDIFF 関数は 2 つの日付の間に存在する月の初日の数を返します。このとき、2 番目の日付は計算に含まれますが、最初の日付は含まれません。

week を日付の単位として使用する場合、DATEDIFF 関数は 2 つの日付の間に存在する日曜日の数を返します。このとき、2 番目の日付は計算に含まれますが、最初の日付は含まれません。

例

次の文は、1 を返します。

```
SELECT DATEDIFF( hour, '4:00AM', '5:50AM' );
```

次の文は、102 を返します。

```
SELECT DATEDIFF( month, '1987/05/02', '1995/11/15' );
```

次の文は、0 を返します。

```
SELECT DATEDIFF( day, '00:00', '23:59' );
```

次の文は、4 を返します。

```
SELECT DATEDIFF( day,
   '1999/07/19 00:00',
   '1999/07/23 23:59' );
```

次の文は、0 を返します。

```
SELECT DATEDIFF( month, '1999/07/19', '1999/07/23' );
```

次の文は、1 を返します。

```
SELECT DATEDIFF( month, '1999/07/19', '1999/08/23' );
```

次の例は、DATEDIFF 関数を使用し、GROUPO.Customers テーブルで 3 段階のジョインを行うミリ秒数を返す方法を示しています。出力は、データベースサーバウィンドウに送信されます。

```
BEGIN
    DECLARE startTime, endTime TIMESTAMP;
    DECLARE rowCount INT;

    SET startTime = CURRENT TIMESTAMP;
    SELECT count(*) INTO rowCount FROM GROUPO.Customers AS T1, GROUPO.Customers
AS T2, GROUPO.Customers AS T3;
    SET endTime = CURRENT TIMESTAMP;

    MESSAGE 'Time to count rows: ' || DATEDIFF( MILLISECOND, startTime, endTime )
|| ' ms';
END
```

関連情報

日付の単位の指定 [312 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.13.5.2.26 DATEFORMAT 関数 [日付と時刻]

日付式を表す文字列を、指定した形式で返します。

構文

```
DATEFORMAT( datetime-expression, string-expression )
```

パラメータ

datetime-expression

変換される日時。

string-expression

変換後の日付の形式。

この関数は NCHAR の入力または出力をサポートしています。

Ultra Light では NCHAR の入力または出力がサポートされていません。

戻り値

VARCHAR

備考

string-expression には、許容される任意の日付形式を使用できます。

例

次の文は、値 Jan 01, 1989 を返します。

```
SELECT DATEFORMAT( '1989-01-01', 'Mmm dd, yyyy' );
```

関連情報

Ultra Light の date_format 作成オプション [145 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.13.5.2.27 DATENAME 関数 [日付と時刻]

TIMESTAMP または TIMESTAMP WITH TIME ZONE の値の特定部分の名前 (月の名前 "June" など) を文字列で返します。

構文

```
DATENAME( date-part, timestamp-expression )
```

パラメータ

date-part

名前が返される日付の単位。

timestamp-expression

日付の単位名が返される TIMESTAMP または TIMESTAMP WITH TIME ZONE の値。意味のある結果を取得するには、`timestamp-expression` に必要な `date-part` を含めてください。

戻り値

VARCHAR

備考

結果が数値 (23 日など) の場合でも、DATENAME 関数は文字列を返します。

SQL Anywhere では、英語のロケールでは英語の名前が返され、英語以外のロケールでは英語以外の名前が返されます。たとえば、Language (LANG) 接続パラメータを使用すれば他の言語を指定できます。

日付の単位 TZOffset (TZ) が指定されていると、DATENAME はオフセットを{ + | - }hh:nn 形式の文字列で返します。

例

次の文は、英語のロケールの場合、`May` という値を返します。

```
SELECT DATENAME( month, '1987/05/02' );
```

SQL Anywhere では、ドイツ語のロケールの場合、`Mai` という値を返します。スペイン語のロケールの場合、`Mayo` という値を返します。複数のロケールがサポートされています。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

次の文は、値 -05:00 を返します。

```
SELECT DATENAME( TZ, CAST('2016/02/03 12:02:00-5:00' AS TIMESTAMP WITH TIME
ZONE) ) AS TZOffset;
```

関連情報

日付の単位の指定 [312 ページ]
DATEPART 関数 [日付と時刻] [362 ページ]

1.13.5.2.28 DATEPART 関数 [日付と時刻]

TIMESTAMP または TIMESTAMP WITH TIME ZONE の値の一部を返します。

構文

```
DATEPART( date-part, timestamp-expression )
```

パラメータ

date-part

名前が返される日付の単位。

timestamp-expression

一部分が返される TIMESTAMP または TIMESTAMP WITH TIME ZONE の値。

戻り値

INT

備考

意味のある結果を取得するには、timestamp-expression に必要な date-part 部分を含めてください。

曜日に対応する数値は、first_day_of_week データベースオプションの設定によって変わります。デフォルトは日曜日 = 7 です。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

例

次の文は、値 5 を返します。

```
SELECT DATEPART( month , '1987/05/02' );
```

次の例では、TableStatistics テーブルを作成し、SalesOrders テーブルに格納されている年ごとの注文の総数を挿入します。

```
CREATE TABLE TableStatistics (
   ID INTEGER NOT NULL DEFAULT AUTOINCREMENT,
   Year INT,
   NumberOrders INT );
INSERT INTO TableStatistics ( Year, NumberOrders )
   SELECT DATEPART( Year, OrderDate ), COUNT(*)
   FROM GROUPO.SalesOrders
   GROUP BY DATEPART( Year, OrderDate );
```

関連情報

日付の単位の指定 [312 ページ]

1.13.5.2.29 DATETIME 関数 [日付と時刻]

式を TIMESTAMP 値に変換します。

構文

```
DATETIME( expression )
```

パラメータ

expression

変換される式。通常は文字列です。

戻り値

TIMESTAMP

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

備考

数値を変換しようとするとエラーが返ります。

例

次の文は、値 1998-09-09 12:12:12.000 のタイムスタンプを返します。

```
SELECT DATETIME( '1998-09-09 12:12:12.000' );
```

関連情報

CAST 関数 [データ型変換] [339 ページ]

1.13.5.2.30 DAY 関数 [日付と時刻]

引数の日付を 1 ～ 31 の間の整数で返します。

構文

```
DAY( date-expression )
```

パラメータ

date-expression

DATE データ型の日付。

戻り値

SMALLINT

備考

DAY 関数は、引数の日付に対応する 1 ～ 31 の間の整数を返します。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

例

次の文は、値 12 を返します。

```
SELECT DAY( '2001-09-12' );
```

1.13.5.2.31 DAYNAME 関数 [日付と時刻]

日付から曜日の名前を返します。

構文

```
DAYNAME( date-expression )
```

パラメータ

date-expression

日付。

戻り値

VARCHAR

備考

返される曜日名は、日曜日、月曜日、火曜日、水曜日、木曜日、金曜日、土曜日です。

SQL Anywhere は、英語のロケールでは英語の名前を、英語以外のロケールでは英語以外の名前を返します。たとえば、Language (LANG) 接続パラメータを使用すれば他の言語を指定できます。

例

次の文は、英語のロケールの場合、`Saturday` という値を返します。

```
SELECT DAYNAME( '1987/05/02' );
```

ドイツ語のロケールの場合、`Samstag` という値を返します。イタリア語のロケールの場合、`sabato` という値を返します。複数のロケールがサポートされています。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

1.13.5.2.32 DAYS 関数 [日付と時刻]

TIMESTAMP を操作するか、2 つの TIMESTAMP 値の間の日数を返します。

構文

0000-02-29 と TIMESTAMP 値の間の日数を返します。

```
DAYS( timestamp-expression )
```

2 つの TIMESTAMP 値の間の日数を返します。

```
DAYS( timestamp-expression, timestamp-expression )
```

TIMESTAMP に時刻を追加します。

```
DAYS( timestamp-expression, integer-expression )
```

パラメータ

timestamp-expression

TIMESTAMP 値。

integer-expression

`timestamp-expression` に追加する日数。`integer-expression` が負の場合、適切な日数が `timestamp-expression` から減算されます。integer-expression を指定する場合は、`timestamp-expression` を TIME、DATE または TIMESTAMP として明示的にキャストしてください。`timestamp-expression` が TIME 値の場合、現在の日付が使用されます。

戻り値

タイムスタンプに時刻を追加する場合は TIMESTAMP。それ以外の場合は INTEGER。

備考

DAYS 関数の結果は、その引数によって異なります。DAYS 関数では、引数内の時間、分、および秒は無視されます。

0000-02-29 以降の日数を返します。

1 つの `timestamp-expression` を DAYS 関数に渡すと、0000-02-29 から `timestamp-expression` までの日数を INTEGER として返します。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

注記

0000-02-29 は実際の日付を指すための値ではありません。DAYS 関数で使用されるデフォルトの日付です。

2 つの TIMESTAMP 値の間の日数を返します。

2 つの TIMESTAMP 値を DAYS 関数に渡すと、その間の日数を整数で返します。

DATEDIFF 関数を使用して 2 つの日付間の間隔を取得することもできます。

TIMESTAMP に時間を追加します。

TIMESTAMP 値と整数を DAYS 関数に渡すと、整数値を `timestamp-expression` 引数に加算した TIMESTAMP 結果を返します。

DATEADD 関数を使用して、TIMESTAMP に日付要素を追加することもできます。

例

次の文は、整数 729889 を返します。

```
SELECT DAYS( '1998-07-13 06:07:12' );
```

次の文は、整数値 -366 を返します。このことは、2 番目の DATE 値が最初の日付より 366 日前の日付であることを示します。2 つ目の例 (DATEDIFF) の使用をお奨めします。

```
SELECT DAYS( '1998-07-13 06:07:12',
             '1997-07-12 10:07:12' );
```

```
SELECT DATEDIFF( day,
   '1998-07-13 06:07:12',
   '1997-07-12 10:07:12' );
```

次の文は、TIMESTAMP 値 1999/07/14 0:00:00.000 を返します。2 つ目の例 (DATEADD) の使用をお奨めします。

```
SELECT DAYS( CAST('1998-07-13' AS DATE ), 366 );
```

```
SELECT DATEADD( day, 366, '1998-07-13' );
```

関連情報

DATEDIFF 関数 [日付と時刻] [358 ページ]
DATEADD 関数 [日付と時刻] [356 ページ]
CAST 関数 [データ型変換] [339 ページ]

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

1.13.5.2.33 DB_PROPERTY 関数 [システム]

指定されたデータベースのプロパティの値を返します。

構文

```
DB_PROPERTY(
{  property-id |  property-name }
[, database-id | database-name ]
)
```

Ultra Light:

```
DB_PROPERTY( property-name )
```

パラメータ

property-id

データベースプロパティ ID。

property-name

データベースプロパティ名。

database-id

DB_ID 関数が返すデータベース ID 番号。通常、データベース名が使用されます。

database-name

DB_NAME 関数から返されるデータベース名。

戻り値

VARCHAR、LONG VARCHAR

備考

文字列を返します。

2 番目の引数を省略すると、現在のデータベースが使用されます。

Ultra Light: Ultra Light でオプションを設定するには、SET OPTION 文、またはコンポーネントの API に固有のデータベースオプション設定メソッドを使用します。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

権限

現在のデータベースに対してこの関数を実行する場合、権限は必要ありません。この関数を他のデータベースに対して実行するには、SERVER OPERATOR または MONITOR のシステム権限を持っていることが必要です。

無効なパラメータ値を指定した場合、または必要とされるシステム権限のどれかがない場合は、NULL が戻ります。

Ultra Light: これらの権限は Ultra Light には適用されません。

例

次の文は、現在のデータベースのページサイズをバイト単位で返します。

```
SELECT DB_PROPERTY( 'PageSize' );
```

Ultra Light: 次の文は、現在のデータベースのページサイズをバイト単位で返します。

```
SELECT DB_PROPERTY( 'page_size');
```

関連情報

Ultra Light データベースプロパティ [197 ページ]
SET OPTION 文 [Ultra Light] [519 ページ]

1.13.5.2.34 DEGREES 関数 [数値]

数値をラジアンから度数に変換します。

構文

```
DEGREES( numeric-expression )
```

パラメータ

numeric-expression

角度 (ラジアン)。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

戻り値

DOUBLE

備考

この関数は、引数を DOUBLE に変換し、計算を倍精度浮動小数点で行い、`numeric-expression` で指定される角度を返します。パラメータが NULL 値の場合、結果は NULL 値になります。

例

次の文は、値 29.79380534680281 を返します。

```
SELECT DEGREES( 0.52 );
```

1.13.5.2.35 DIFFERENCE 関数 [文字列]

2 つの文字列式の SOUNDEX 値の差を返します。

構文

```
DIFFERENCE ( string-expression-1, string-expression-2 )
```

パラメータ

string-expression-1

最初の SOUNDEX 引数。

string-expression-2

2 番目の SOUNDEX 引数。

戻り値

SMALLINT

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

備考

DIFFERENCE 関数は 2 つの文字列の SOUNDEX 値を比較し、値の類似性を評価し、0 ～ 4 の値を返します。最も一致する場合は 4 です。

この関数は常に何らかの値を返します。結果が NULL になるのは引数の 1 つが NULL の場合のみです。

例

次の文は、test と chest という単語の類似点を返します。

```
SELECT DIFFERENCE( 'test', 'chest' );
```

関連情報

SOUNDEX 関数 [文字列] [438 ページ]
STRING 関数 [文字列] [453 ページ]

1.13.5.2.36 DOW 関数 [日付と時刻]

指定した日付の曜日を表す 1 ～ 7 の数を返します (日曜日 = 1、月曜日 = 2、以下同様)。

構文

```
DOW( date-expression )
```

パラメータ

date-expression

評価される値 (DATE データ型)。

戻り値

SMALLINT

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

備考

DOW 関数は、first_day_of_week データベースオプションで指定された値の影響を受けません。たとえば、first_day_of_week が月曜日に設定されている場合でも、DOW 関数は月曜日に 2 を返します。

例

次の文は、値 5 を返します。

```
SELECT DOW( '1998-07-09' );
```

次の文は、値 1 を返します。

```
SELECT DOW( CAST( '2010/05/30 11:33:00.000000+04:00' as TIMESTAMP WITH TIME
ZONE ));
```

次の文は、Employees テーブルに問い合わせ、週の曜日を表す数として従業員の StartDate を返します。

```
SELECT DOW( StartDate ) FROM GROUPO.Employees;
```

1.13.5.2.37 EXP 関数 [数値]

自然対数の底 e を、与えられた引数でべき乗した結果を返します。

構文

```
EXP( numeric-expression )
```

パラメータ

numeric-expression

指数。

戻り値

DOUBLE

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

備考

EXP 関数は、自然対数の底 e を `numeric-expression` で指定された値でべき乗した結果を返します。

この関数は、引数を DOUBLE に変換し、計算を倍精度浮動小数点で行い、結果を DOUBLE で返します。パラメータが NULL 値の場合、結果は NULL 値になります。

例

次の文は、値 3269017.3724721107 を返します。

```
SELECT EXP( 15 );
```

1.13.5.2.38 EXPLANATION 関数 [その他]

SQL 文の最適化方法をプレーンテキスト文字列で返します。

構文

```
EXPLANATION(
string-expression
[ , cursor-type ]
[, update-status ]
)
```

Ultra Light:

```
EXPLANATION( string-expression )
```

パラメータ

string-expression

SQL 文。通常は SELECT 文ですが、UPDATE 文、MERGE 文、または DELETE 文も指定できます。

cursor-type

カーソルタイプ。文字列として表現されます。使用できる値は、asensitive、insensitive、sensitive、または keyset-driven です。`cursor-type` が指定されない場合、デフォルトで asensitive が使用されます。

update-status

次のいずれかの値を受け入れる文字列パラメータ。これらの値は、指定されたカーソルをオプティマイザがどのように処理するかを示します。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

値	説明
READ-ONLY	このカーソルは読み込み専用です。
READ-WRITE (デフォルト)	このカーソルは読み込みや書き込みが可能です。
FOR UPDATE	このカーソルは読み込みや書き込みが可能です。READ-WRITE と同じです。

戻り値

LONG VARCHAR

備考

文字列として返されるクエリの実行プランです。

GRAPHICAL_PLAN 関数では、文の最適化方法に影響を及ぼすシステムプロパティなど、アクセスプランに関するさまざまな情報が提供されます。

この情報は、追加するインデックスの決定や、パフォーマンスを向上するためのデータベース構造の決定に役立ちます。

例

次の文は、SELECT 文を文字列パラメータとして渡し、クエリを実行するためのプランを返します。

```
SELECT EXPLANATION( 'SELECT * FROM Departments WHERE DepartmentID > 100' );
```

次の文は、'SELECT * FROM Departments WHERE....' クエリに対する INSENSITIVE カーソルのプランを短いテキスト形式で表した文字列を返します。

```
SELECT EXPLANATION( 'SELECT * FROM GROUPO.Departments WHERE DepartmentID > 100',
   'insensitive', 'read-only' );
```

関連情報

Ultra Light の実行プラン [534 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.13.5.2.39 EXTRACT 関数 [日付と時刻]

TIMESTAMP 式から日付要素を返します。

構文

```
EXTRACT( date-part FROM timestamp-expression )
```

パラメータ

date-part

名前が返される日付の単位。有効な値は、YEAR、MONTH、DAY、HOUR、MINUTE、SECOND、TIMEZONE_HOUR、および TIMEZONE_MINUTE です。

timestamp-expression

TIMESTAMP または TIMESTAMP WITH TIME ZONE の値。

戻り値

`date-part` が SECOND の場合、戻り値は NUMERIC 値となり、1 秒に満たない時間が含まれます (マイクロ秒までの精度)。`date-part` がこれ以外の値の場合、関数の戻り値は INTEGER です。

備考

EXTRACT 関数は DATEPART 関数と似ていますが、まったく同じではありません。EXTRACT 関数は、日付要素の一部しか受け付けません。また、`date-part` が SECOND の場合、2 つの関数の戻り値は異なります。

例

次の文は、56.789000 を返します。

```
SELECT EXTRACT( SECOND FROM '2015-07-01 12:34:56.789000' );
```

次の文は、30 を返します。

```
SELECT EXTRACT( TIMEZONE_MINUTE FROM '2015-07-01 12:34:56.789000 +05:30' );
```

次の文は、5 を返します。

```
SELECT EXTRACT( TIMEZONE_HOUR FROM '2015-07-01 12:34:56.789000 +05:30' );
```


© 2016 SAP SE or an SAP affiliate company. All rights reserved.

1.13.5.2.40 FLOOR 関数 [数値]

指定された値以下の、最大の整数値を返します。

構文

```
FLOOR( numeric-expression )
```

パラメータ

numeric-expression

トランケートされる値。通常は、位取りがゼロ以外の固定数値型または適切な数値型 (DOUBLE、REAL、または FLOAT) です。

戻り値

DOUBLE

備考

この関数は、引数を DOUBLE に変換し、倍精度浮動小数点で計算を行います。

例

次の文は、123 の Floor 値を返します。

```
SELECT FLOOR (123);
```

次の文は、123 の Floor 値を返します。

```
SELECT FLOOR (123.45);
```

次の文は、-124 の Floor 値を返します。

```
SELECT FLOOR (-123.45);
```

関連情報

CEILING 関数 [数値] [342 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.13.5.2.41 GETDATE 関数 [日付と時刻]

現在の年、月、日、時、分、秒、秒以下を返します。

構文

```
GETDATE( )
```

戻り値

TIMESTAMP

備考

精度はシステムクロックの精度によって制限されます。

GETDATE 関数が返す情報は、NOW 関数と CURRENT TIMESTAMP 特別値が返す情報と同じです。

注記

データベースでシミュレートされたタイムゾーンが使用されている場合、シミュレートされたタイムゾーンを使用してこの関数の結果が計算されます。

例

次の文は、システムの日付と時刻を返します。

```
SELECT GETDATE( );
```

関連情報

NOW 関数 [日付と時刻] [415 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.13.5.2.42 GREATER 関数 [その他]

2 つのパラメータ値のうち、より大きい値を返します。

構文

```
GREATER( expression-1, expression-2 )
```

パラメータ

expression-1

比較される最初のパラメータ値。

expression-2

比較される 2 番目のパラメータ値。

戻り値

この関数の戻り値は、指定した式によって異なります。具体的には、データベースサーバが関数を評価するとき、まず、式の比較が可能なデータ型を検索します。該当するデータ型が見つかると、データベースサーバは式を比較し、比較に使用したデータ型で結果を返します。データベースサーバは、一般に比較が可能なデータ型を見つけることができないと、エラーを返します。

備考

パラメータが等しい場合は、最初のパラメータを返します。

TABLE REF 型として定義された値は、この関数ではサポートされません。

例

次の文は、値 10 を返します。

```
SELECT GREATER( 10, 5 ) FROM SYS.DUMMY;
```

関連情報

LESSER 関数 [その他] [391 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.13.5.2.43 HEXTOINT 関数 [データ型変換]

16 進文字列と同等の 10 進整数を返します。

構文

```
HEXTOINT( hexadecimal-string )
```

パラメータ

hexadecimal-string

整数に変換される文字列。

戻り値

CAST、CONVERT、HEXTOINT、INTTOHEX 関数を使用すると、16 進値変換を行うことができます。

HEXTOINT 関数は、プラットフォームに依存しない SQL INTEGER に相当する 16 進文字列を INT として返します。右から 8 桁目が数値 8 ～ 9 か、大文字または小文字の A ～ F のいずれかであり、その前の桁がすべて大文字または小文字の F の場合は、16 進値が負の整数値になります。次の HEXTOINT は無効な使用例です。これは、引数が符号付き 32 ビット整数で表現できない正の整数値を示しているためです。

```
SELECT HEXTOINT( '0x0080000001' );
```

備考

HEXTOINT 関数は、数値、大文字または小文字の A ～ F のみで構成される文字列キーコードまたは変数を受け入れます (0x プレフィクスが付いている場合、付いていない場合の両方)。次に HEXTOINT の有効な使用例をすべて示します。

```
SELECT HEXTOINT( '0xFFFFFFFF' );
SELECT HEXTOINT( '0x00000100' );
SELECT HEXTOINT( '100' );
SELECT HEXTOINT( '0xffffffff80000001' );
```

HEXTOINT 関数は 0x プレフィクスがあれば削除します。データが 8 桁を超える場合、符号付き 32 ビット整数値として表現できる値を示す必要があります。

この関数は NCHAR の入力または出力をサポートしています。

Ultra Light では NCHAR の入力または出力がサポートされていません。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

例

次の文は、値 420 を返します。

```
SELECT HEXTOINT( '1A4' );
```

関連情報

INTTOHEX 関数 [データ型変換] [385 ページ]

1.13.5.2.44 HOUR 関数 [日付と時刻]

TIMESTAMP 値の時間部分を返します。

構文

```
HOUR( timestamp-expression )
```

パラメータ

timestamp-expression

TIMESTAMP 値。

戻り値

SMALLINT

備考

TIMESTAMP 式の時間部分である、0 ～ 23 の間の SMALLINT 値が返されます。

例

次の文は、値 21 を返します。

```
SELECT HOUR( '1998-07-09 21:12:13' );
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.13.5.2.45 HOURS 関数 [日付と時刻]

TIMESTAMP を操作するか、2 つの TIMESTAMP 値の間の時間数を返します。

構文

0000-02-29 の真夜中と TIMESTAMP 値の間の時間数を返します。

```
HOURS ( timestamp-expression )
```

2 つの TIMESTAMP 値の間の時間数を返します。

```
HOURS ( timestamp-expression, timestamp-expression )
```

TIMESTAMP に時間を追加します。

```
HOURS ( time-or-timestamp-expression, integer-expression )
```

パラメータ

time-or-timestamp-expression

TIME または TIMESTAMP データ型の値。

timestamp-expression

TIMESTAMP データ型の値。

integer-expression

`time-or-timestamp-expression` に追加する時間数。`integer-expression` が負の場合、適切な時間数が `time-or-timestamp-expression` から減算されます。

戻り値

2 つの `time-or-timestamp-expression` 値の間の時間数を返す場合は INTEGER。

時間を `time-or-timestamp-expression` に追加する場合は TIME または TIMESTAMP。

備考

HOURS 関数の結果は、その引数によって異なります。

0000-02-29 の真夜中以降の時間数を返します。

1 つの `timestamp-expression` を HOURS 関数に渡すと、0000-02-29 の真夜中から `timestamp-expression` までの時間数を INTEGER として返します。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

> **注記**
>
> 0000-02-29 は実際の日付を指すための値ではありません。HOURS 関数で使用されるデフォルトの TIMESTAMP 値です。

2 つの TIMESTAMP 値の間の時間数を返します。

2 つの TIMESTAMP 値を HOURS 関数に渡すと、その間の時間数を整数で返します。

TIMESTAMP に時間を追加します。

TIMESTAMP 値と INTEGER 値を HOURS 関数に渡すと、整数の時間数を `time-or-timestamp-expression` 引数に加算した TIMESTAMP 結果を返します。同様に、TIME 値を最初の引数として渡すと、TIME 値を結果として返します。この構文では、最初の引数の暗黙的変換をサポートしていません。最初の引数を DATE、TIME または TIMESTAMP の値に明示的にキャストしてください。最初の引数が DATE の場合、時刻部分には真夜中が使用されます。

これらの計算に DATEDIFF 関数と DATEADD 関数を使用することもできます。

> **例**
>
> 次の文は、値 4 を返します。これは、2 番目の TIMESTAMP 値が、最初のタイムスタンプの 4 時間後であることを示します。2 つ目の例 (DATEDIFF) の使用をお奨めします。

```
SELECT HOURS( '1999-07-13 06:07:12', '1999-07-13 10:07:12' );
SELECT DATEDIFF( hour, '1999-07-13 06:07:12', '1999-07-13 10:07:12' );
```

次の文は、値 17517342 を返します。

```
SELECT HOURS( '1998-07-13 06:07:12' );
```

次の文は、日時 1999/05/13 2:05:07.000 を返します。2 つ目の例 (DATEADD) の使用をお奨めします。

```
SELECT HOURS( CAST( '1999-05-12 21:05:07' AS DATETIME ), 5 );
SELECT DATEADD( hour, 5, '1999-05-12 21:05:07' );
```

関連情報

DATEDIFF 関数 [日付と時刻] [358 ページ]
DATEADD 関数 [日付と時刻] [356 ページ]
CAST 関数 [データ型変換] [339 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.13.5.2.46 IFNULL 関数 [その他]

ある式が NULL かどうかと、値を返すかどうかを評価します。

構文

```
IFNULL( expression-1, expression-2 [ , expression-3 ] )
```

パラメータ

expression-1

評価される式。この値によって、expression-2 と expression-3 のどちらを返すかが決まります。

expression-2

expression-1 が NULL の場合の戻り値。

expression-3

expression-1 が NULL でない場合の戻り値。

戻り値

返されるデータ型は、expression-2 と expression-3 のデータ型によって異なります。

備考

最初の式が NULL 値の場合は、2 番目の式の値を返します。最初の式が NULL でない場合は、3 番目の式の値を返します。最初の式が NULL ではなく、3 番目の式が存在しない場合は、NULL を返します。

例

次の文は、値 -66 を返します。

```
SELECT IFNULL( NULL, -66 );
```

次の文は、最初の式が NULL ではなく、3 番目の式が存在しないため、を返します。

```
SELECT IFNULL( -66, -66 );
```


© 2016 SAP SE or an SAP affiliate company. All rights reserved.

1.13.5.2.47 INSERTSTR 関数 [文字列]

別の文字列の指定された位置に文字列を挿入します。

構文

```
INSERTSTR( integer-expression, string-expression-1, string-expression-2 )
```

パラメータ

integer-expression

文字列の挿入位置。文字列の先頭に挿入する場合は、0 を使用します。

string-expression-1

別の文字列が挿入される文字列。

string-expression-2

挿入する文字列。

戻り値

入力式のデータ型に応じて、LONG BINARY、LONG VARCHAR、または LONG NVARCHAR。

備考

この関数は NCHAR の入力または出力をサポートしています。

Ultra Light では NCHAR の入力または出力がサポートされていません。

例

次の文は、値 `backoffice` を返します。

```
SELECT INSERTSTR( 0, 'office ', 'back' );
```

関連情報

STUFF 関数 [文字列] [455 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

STRING 関数 [文字列] [453 ページ]

1.13.5.2.48 INTTOHEX 関数 [データ型変換]

整数に対応する 16 進値の文字列を返します。

構文

```
INTTOHEX( integer-expression )
```

パラメータ

integer-expression

16 進に変換される整数。

戻り値

VARCHAR

備考

CAST、CONVERT、HEXTOINT、INTTOHEX 関数を使用すると、16 進値変換を行うことができます。

例

次の文は、値 0000009c を返します。

```
SELECT INTTOHEX( 156 );
```

関連情報

HEXTOINT 関数 [データ型変換] [379 ページ]

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

1.13.5.2.49 ISDATE 関数 [データ型変換]

文字列引数を日付に変換できるかどうかをテストします。

構文

```
ISDATE( string )
```

パラメータ

string

文字列が有効な日付を表すかどうかを判断するために分析される文字列。

戻り値

INT

備考

変換できる場合は 1 を返し、できない場合は 0 を返します。引数が NULL の場合は 0 を返します。

この関数は NCHAR の入力または出力をサポートしています。

Ultra Light では NCHAR の入力または出力がサポートされていません。

例

次の例は、外部ファイルからサンプルデータベースにデータをインポートし、無効な値があるローをエクスポートし、残りのローを永久テーブルにコピーします。

```
CREATE GLOBAL TEMPORARY TABLE MyData(
      person VARCHAR(100),
      birth_date VARCHAR(30),
      height_in_cms VARCHAR(10)
   ) ON COMMIT PRESERVE ROWS;
   LOAD TABLE MyData FROM 'exported.dat';
   UNLOAD
      SELECT * FROM MyData
      WHERE ISDATE( birth_date ) = 0
  OR ISNUMERIC( height_in_cms ) = 0
   TO 'badrows.dat';
   INSERT INTO PermData
      SELECT person, birth_date, height_in_cms
      FROM MyData
      WHERE ISDATE( birth_date ) = 1
  AND ISNUMERIC( height_in_cms ) = 1;
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

```
    COMMIT;
    DROP TABLE MyData;
```

1.13.5.2.50 ISNULL 関数 [その他]

リストの中から NULL でない最初の式を返します。この関数は COALESCE 関数と同じです。

構文

```
ISNULL( expression, expression [, ...] )
```

パラメータ

expression

NULL かどうかがテストされる式。

2 つ以上の式を関数に渡します。すべての式は比較可能であることが必要です。

戻り値

この関数の戻り値は、指定した式によって異なります。具体的には、データベースサーバが関数を評価するとき、まず、式の比較が可能なデータ型を検索します。該当するデータ型が見つかると、データベースサーバは式を比較し、リストから NULL でない最初の式を返します。データベースサーバは、比較が可能な共通のデータ型を見つけることができないと、エラーを返します。

例

次の文は、値 -66 を返します。

```
SELECT ISNULL( NULL ,-66, 55, 45, NULL, 16 );
```

関連情報

COALESCE 関数 [その他] [346 ページ]

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

1.13.5.2.51 LCASE 関数 [文字列]

文字列中のすべての文字を小文字に変換します。

構文

```
LCASE( string-expression )
```

パラメータ

string-expression

小文字に変換される文字列。

戻り値

データベース照合が UCA の場合、NCHAR データに対して使用されるときは LONG NVARCHAR、CHAR データに対して使用されるときは LONG VARCHAR が返されます。それ以外の場合、データ型は入力データ型と同じになります。

Ultra Light: 返されるデータ型は入力データ型と同じになります。

備考

LCASE 関数は LOWER 関数と同じです。

例

次の文は、値 `chocolate` を返します。

```
SELECT LCASE( 'ChoCOlatE' );
```

関連情報

LOWER 関数 [文字列] [399 ページ]
UCASE 関数 [文字列] [467 ページ]
UPPER 関数 [文字列] [468 ページ]
STRING 関数 [文字列] [453 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.13.5.2.52 LEFT 関数 [文字列]

文字列の先頭からいくつかの文字を返します。

構文

```
LEFT( string-expression, integer-expression )
```

パラメータ

string-expression

文字列。

integer-expression

返す文字数。

戻り値

LONG VARCHAR または LONG NVARCHAR

Ultra Light: LONG VARCHAR

備考

文字列にマルチバイト文字があり、適切な照合が使用されている場合、返されるバイト数が指定した文字数よりも大きい場合があります。

引数 string-expression の値より大きな `integer-expression` を指定できます。この場合、全体の値が返されます。

入力文字が文字長のセマンティックを使用している場合、可能であれば、戻り値が文字長のセマンティックで記述されます。

この関数は NCHAR の入力または出力をサポートしています。

Ultra Light では NCHAR の入力または出力がサポートされていません。

例

次の文は、Customers テーブルに含まれる各 Surname 値の最初の 5 文字を返します。

```
SELECT LEFT( Surname, 5) FROM GROUPO.Customers;
```


© 2016 SAP SE or an SAP affiliate company. All rights reserved.

関連情報

RIGHT 関数 [文字列] [428 ページ]
STRING 関数 [文字列] [453 ページ]

1.13.5.2.53 LENGTH 関数 [文字列]

指定した文字列の文字数を返します。

構文

```
{ LENGTH | LEN }( string-expression )
```

Ultra Light:

```
LENGTH( string-expression )
```

パラメータ

string-expression

文字列。

戻り値

INT

備考

この関数を使用すると、文字列の長さがわかります。たとえば、`string-expression` にカラム名を指定すると、カラムの値の長さがわかります。

文字列にマルチバイト文字があり、適切な照合が使用されている場合、LENGTH はバイト数ではなく、文字数を返します。文字列が BINARY データ型の場合、LENGTH 関数は BYTE_LENGTH 関数のように動作します。

データ型が CHAR、VARCHAR、LONG VARCHAR、NCHAR の場合、LENGTH 関数と CHAR_LENGTH 関数を使用すると同じ結果が得られます。ただし、LENGTH 関数は BINARY 型およびビット配列データ型に使用します。この関数は NCHAR の入力または出力をサポートしています。

Ultra Light では NCHAR の入力または出力がサポートされていません。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

例

次の文は、値 9 を返します。

```
SELECT LENGTH( 'chocolate' );
```

関連情報

BYTE_LENGTH 関数 [文字列] [337 ページ]
STRING 関数 [文字列] [453 ページ]

1.13.5.2.54 LESSER 関数 [その他]

2 つのパラメータ値のうち、より小さい値を返します。

構文

```
LESSER( expression-1, expression-2 )
```

パラメータ

expression-1

比較される最初のパラメータ値。

expression-2

比較される 2 番目のパラメータ値。

戻り値

この関数の戻り値は、指定した式によって異なります。具体的には、データベースサーバが関数を評価するとき、まず、式の比較が可能なデータ型を検索します。該当するデータ型が見つかると、データベースサーバは式を比較し、比較に使用したデータ型で結果を返します。データベースサーバは、一般に比較が可能なデータ型を見つけることができないと、エラーを返します。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

備考

パラメータが等しい場合は、最初の値を返します。

TABLE REF 型として定義された値は、この関数ではサポートされません。

例

次の文は、値 5 を返します。

```
SELECT LESSER( 10, 5 ) FROM SYS.DUMMY;
```

関連情報

GREATER 関数 [その他] [378 ページ]

1.13.5.2.55 LIST 関数 [集合]

グループ内のローごとに値のデリミタ付きリストを返します。

構文

```
LIST( [ALL | DISTINCT ] string-expression [, delimiter-string ] [ ORDER BY order-by-expression [ ASC | DESC ], ... ] )
```

Ultra Light:

```
LIST(
[ DISTINCT ] string-expression
[   , delimiter-string ] )
```

パラメータ

string-expression

文字列式。通常はカラム名です。ALL が指定されていると (デフォルト)、グループ内のローごとに `string-expression` の値が結果文字列に追加されます。各値は、`delimiter-string` で区切られます。DISTINCT を指定すると、ユニークな `string-expression` 値のみが追加されます。

Ultra Light: グループ内のローごとに `string-expression` の値が結果文字列に追加されます。各値は、`delimiter-string` で区切られます。

delimiter-string

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

リスト項目のデリミタ文字列。デフォルト設定はカンマです。NULL 値または空の文字列を指定した場合は、デリミタはありません。`delimiter-string` は定数でなければなりません。

order-by-expression

関数によって返された項目を並べ替えます。この引数の前にカンマは必要ありません。このため、`delimiter-string` を指定しない場合、使用が簡単になります。

`order-by-expression` には整数リテラルを指定できません。ただし、整数リテラルを含む変数を指定できます。

ORDER BY 句に定数が含まれている場合、それらの定数はオプティマイザによって解釈され、同義の ORDER BY 句に置き換えられます。たとえば、オプティマイザは ORDER BY 'a' を ORDER BY 式として解釈します。

クエリブロックに、有効な ORDER BY 句が指定された複数の集合関数が含まれているとき、それらの ORDER BY 句を単一の ORDER BY 句に論理的に結合できる場合は、そのクエリブロックを実行できます。たとえば、次の ORDER BY 句の場合は、

```
ORDER BY expression1, 'a', expression2
```

```
ORDER BY expression1, 'b', expression2, 'c', expression3
```

次の ORDER BY 句として結合されます。

```
ORDER BY expression1, expression2, expression3
```

戻り値

LONG VARCHAR
LONG NVARCHAR

> **注記**
>
> Ultra Light は LONG NVARCHAR を返しません。

備考

LIST 関数は、グループの各ローの、NULL 以外のすべての X の値を (デリミタ付きで) 連結して返します。グループ内に明確な X 値を持ったローが 1 つ以上存在しない場合、LIST(X) は空の文字列を返します。

LIST 関数では、NULL 値と空の文字列は無視されます。

LIST 関数は Window 関数として使用できませんが、Window 関数の入力には使用できます。

この関数は NCHAR の入力または出力をサポートしています。

Ultra Light では NCHAR の入力または出力がサポートされていません。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

例

次の文は、値 `487 Kennedy Court,547 School Street` を返します。

```
SELECT LIST( Street ) FROM GROUPO.Employees
WHERE GivenName = 'Thomas';
```

次の文は、従業員 ID をリストします。結果セットの各ローには、部門ごとの従業員 ID のカンマで区切られたリストが入っています。

```
SELECT LIST( EmployeeID )
FROM GROUPO.Employees
GROUP BY DepartmentID;
```

LIST(EmployeeID)
102,105,160,243,247,249,266,278,...
129,195,299,467,641,667,690,856,...
148,390,586,757,879,1293,1336,...
184,207,318,409,591,888,992,1062,...
191,703,750,868,921,1013,1570,...

次の文は、従業員の姓を基準に従業員 ID をソートします。

```
SELECT LIST( EmployeeID ORDER BY Surname ) AS "Sorted IDs"
FROM GROUPO.Employees
GROUP BY DepartmentID;
```

Sorted IDs
1013,191,750,921,868,1658,...
1751,591,1062,1191,992,888,318,...
1336,879,586,390,757,148,1483,...
1039,129,1142,195,667,1162,902,...
160,105,1250,247,266,249,445,...

次の文は、セミコロンで区切られたリストを返します。ORDER BY 句とリストセパレータの位置に注意してください。

```
SELECT LIST( EmployeeID, ';' ORDER BY Surname ) AS "Sorted IDs"
FROM GROUPO.Employees
GROUP BY DepartmentID;
```

Sorted IDs
1013;191;750;921;868;1658;703;...
1751;591;1062;1191;992;888;318;...
1336;879;586;390;757;148;1483;...
1039;129;1142;195;667;1162;902; ...
160;105;1250;247;266;249;445;...

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

前の文と次の文は区別してください。次の文では、(Surname, ';') を複合ソートキーに使用してソートした従業員 ID のカンマ区切りのリストが返ります。

```
SELECT LIST( EmployeeID ORDER BY Surname, ';' ) AS "Sorted IDs"
FROM GROUPO.Employees
GROUP BY DepartmentID;
```

Ultra Light: 次の文は、Employees テーブルのすべての住所を返します。

```
SELECT LIST( Street ) FROM GROUPO.Employees;
```

1.13.5.2.56 LOCATE 関数 [文字列]

異なる文字列内のある文字列の位置を返します。

構文

```
LOCATE( string-expression-1, string-expression-2 [, integer-expression ] )
```

パラメータ

string-expression-1

検索される文字列。

string-expression-2

検索する文字列。

この文字列は 255 バイトに制限されています。

integer-expression

文字列内の、検索を開始する位置。最初の文字の位置は 1 です。開始オフセットが負の場合は、最初に一致した文字列ではなく、最後に一致した文字列のオフセットを返します。負のオフセットは、文字列の末尾のどれだけの部分が検索から除外されるかを示します。除外されるバイト数は、(-1 * offset) - 1 で計算されます。

戻り値

INT

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

備考

integer-expression を指定すると、文字列のオフセットから検索が開始します。

最初の文字列には長い文字列 (255 バイト以上) を指定できますが、2 番目の文字列は 255 バイトに制限されます。長い文字列を 2 番目の引数として指定すると、LOCATE 関数は NULL 値を返します。文字列が見つからない場合は、0 を返します。長さが 0 の文字列を検索すると、1 が返されます。いずれかの引数が NULL の場合、結果も NULL になります。

マルチバイト文字が使用され、適切な照合が使用されている場合は、開始位置と返される値はバイトで計算した位置と異なる場合があります。

引数 string-expression-1 および string-expression-2 がバイナリデータ型の場合、LOCATE 関数の動作は BYTE_LOCATE 関数と同じになります。

例

次の文は、値 8 を返します。

```
SELECT LOCATE(
   'office party this week - rsvp as soon as possible',
   'party',
   2 );
```

次の文は、

```
BEGIN
   DECLARE STR LONG VARCHAR;
   DECLARE POS INT;
   SET str = 'c:¥test¥functions¥locate.sql';
   SET pos = LOCATE( str, '¥', -1 );
   select str, pos,
      SUBSTR( str, 1, pos -1 ) AS path,
      SUBSTR( str, pos +1 ) AS filename;
END;
```

次の出力を返します。

str	pos	path	filename
c:¥test¥functions¥locate.sql	18	c:¥test¥functions	locate.sql

関連情報

CHARINDEX 関数 [文字列] [345 ページ]
STRING 関数 [文字列] [453 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.13.5.2.57 LOG 関数 [数値]

数の自然対数を返します。

構文

```
LOG( numeric-expression )
```

パラメータ

numeric-expression

数値。

戻り値

この関数は、引数を DOUBLE に変換し、計算を倍精度浮動小数点で行い、結果を DOUBLE で返します。パラメータが NULL 値の場合、結果は NULL 値になります。

備考

引数は、組み込みの数値データ型の値を返す式です。

例

次の文は、50 の自然対数を返します。

```
SELECT LOG( 50 );
```

関連情報

LOG10 関数 [数値] [398 ページ]

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

1.13.5.2.58 LOG10 関数 [数値]

数値の対数 (基数 10) を返します。

構文

```
LOG10( numeric-expression )
```

パラメータ

numeric-expression

数値。

戻り値

この関数は、引数を DOUBLE に変換し、倍精度浮動小数点で計算を行います。パラメータが NULL 値の場合、結果は NULL 値になります。

備考

引数は、組み込みの数値データ型の値を返す式です。

例

次の文は、50 の底が 10 の対数を返します。

```
SELECT LOG10( 50 );
```

関連情報

LOG 関数 [数値] [397 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.13.5.2.59 LOWER 関数 [文字列]

文字列中のすべての文字を小文字に変換します。

構文

```
LOWER( string-expression )
```

パラメータ

string-expression

小文字に変換される文字列。

戻り値

NCHAR データで使用された場合は LONG NVARCHAR
データベース照合が UCA の場合、CHAR データで使用されるときは LONG VARCHAR
それ以外の場合、データ型は入力データ型と同じになります。
Ultra Light では、入力データ型と同じデータ型が返されます。

備考

LCASE 関数は LOWER 関数と同じです。

例

次の文は、値 `chocolate` を返します。

```
SELECT LOWER( 'chOCOLate' );
```

関連情報

LCASE 関数 [文字列] [388 ページ]
UCASE 関数 [文字列] [467 ページ]
UPPER 関数 [文字列] [468 ページ]
STRING 関数 [文字列] [453 ページ]

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

1.13.5.2.60 LTRIM 関数 [文字列]

文字列の先行ブランクを削除します。

構文

```
LTRIM( string-expression )
```

パラメータ

string-expression

削除する文字列。

戻り値

VARCHAR
NVARCHAR
LONG VARCHAR
LONG NVARCHAR
Ultra Light の場合、VARCHAR または LONG VARCHAR だけが返されます。

備考

結果の実際の長さは、式の長さから、削除する文字数を引いた値です。すべての文字を削除すると、結果は空の文字列になります。

パラメータに NULL を指定できる場合、結果は NULL になります。

パラメータが NULL の場合、結果は NULL 値になります。

この関数は NCHAR の入力または出力をサポートしています。

Ultra Light では NCHAR の入力または出力がサポートされていません。

例

次の文は、すべての先行ブランクを削除して、値 `Test Message` を返します。

```
SELECT LTRIM( '     Test Message' );
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

関連情報

RTRIM 関数 [文字列] [430 ページ]
TRIM 関数 [文字列] [465 ページ]
STRING 関数 [文字列] [453 ページ]

1.13.5.2.61 MAX 関数 [集合]

各ローグループで見つかった式の最大値を返します。

構文

式:

```
MAX( [ ALL | DISTINCT ] expression )
```

Window 関数:

```
MAX( [ ALL ] expression ) OVER ( window-spec )
```

`window-spec`: 以下の備考部分を参照してください。

Ultra Light 式

```
MAX( [ DISTINCT ] expression )
```

パラメータ

expression

最大値が計算される式。通常はカラム名です。
DISTINCT expression

MAX(`expression`) の場合と同じ値を返します。万全を期すために含まれています。

戻り値

引数と同じデータ型。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

備考

`expression` が NULL になるローは無視されます。グループにローが含まれていない場合は、NULL を返します。

TABLE REF 型として定義された値は、この関数ではサポートされません。

また、2 つの式を単純に比較する場合は、GREATER 関数を使用することができます。

`window-spec` を使用してこの関数を指定した場合、SELECT 文で Window 関数として使用することを意味します。そのように、`window-spec` の要素は、関数構文の中 (インライン) に指定するか、または SELECT 文の WINDOW 句と組み合わせて指定できます。この関数は NCHAR の入力または出力をサポートしています。

例

次の文は、Employees テーブル内の給与の最大値 138948.000 を返します。

```
SELECT MAX( Salary )
FROM GROUPO.Employees;
```

関連情報

MIN 関数 [集合] [404 ページ]

1.13.5.2.62 MICROSECOND 関数 [日付と時刻]

TIMESTAMP 式のマイクロ秒の部分を返します。

構文

```
MICROSECOND( timestamp-expression )
```

パラメータ

timestamp-expression

TIMESTAMP 値。

戻り値

INTEGER

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

備考

この関数は、1 秒に満たない時間 (マイクロ秒ともいいます) を表す 0 から 999999 までの間の値を返します。この関数は、`DATEPART( MICROSECOND, timestamp-expression )` と同等です。

例

次の文を実行すると、マイクロ秒の値 789012 が返されます。

```
SELECT MICROSECOND( '12:34:56.789012' );
```

1.13.5.2.63 MILLISECOND 関数 [日付と時刻]

TIMESTAMP 式のミリ秒の部分を返します。

構文

```
MILLISECOND( timestamp-expression )
```

パラメータ

timestamp-expression

TIMESTAMP 値。

戻り値

INTEGER

備考

この関数は、1 秒に満たない時間をミリ秒単位で表す 0 から 999 までの間の値を返します。タイムスタンプに 1 ミリ秒より小さい時間が含まれている場合は、ミリ秒単位になるよう切り捨てられます。

この関数は、`DATEPART( MILLISECOND, timestamp-expression )` と同等です。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

例

次の文を実行すると、ミリ秒の値 789 が返されます。

```
SELECT MILLISECOND( '12:34:56.78901' );
```

1.13.5.2.64 MIN 関数 [集合]

各ローグループで見つかった式の最小値を返します。

構文

式

```
MIN( [ DISTINCT ] expression )
```

Window 関数

```
MIN( [ ALL ] expression ) OVER ( window-spec )
```

```
window-spec: 以下の備考部分を参照してください。
```

Ultra Light 式

```
MIN( [ ALL | DISTINCT ] expression )
```

パラメータ

expression

最小値が計算される式。通常はカラム名です。

DISTINCT expression

MIN(`expression`) の場合と同じ値を返します。万全を期すために含まれています。

戻り値

引数と同じデータ型。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

備考

`expression` が NULL になるローは無視されます。グループにローが含まれていない場合は、NULL を返します。

TABLE REF 型として定義された値は、この関数ではサポートされません。

この関数は NCHAR の入力または出力をサポートしています。

Ultra Light: Ultra Light では NCHAR の入力または出力がサポートされていません。

`window-spec` を使用してこの関数を指定した場合、SELECT 文で Window 関数として使用することを意味します。そのように、`window-spec` の要素は、関数構文の中 (インライン) に指定するか、または SELECT 文の WINDOW 句と組み合わせて指定できます。

例

次の文は、Employees テーブル内の給与の最小値 24903.000 を返します。

```
SELECT MIN( Salary )
FROM GROUPO.Employees;
```

関連情報

MAX 関数 [集合] [401 ページ]

1.13.5.2.65 MINUTE 関数 [日付と時刻]

TIMESTAMP 値の分部分を返します。

構文

```
MINUTE( timestamp-expression )
```

パラメータ

timestamp-expression

TIMESTAMP 値。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

戻り値

SMALLINT

備考

TIMESTAMP 式の分部分である、0 ～ 59 の間の SMALLINT 値が返されます。

例

次の文は、値 22 を返します。

```
SELECT MINUTE( '1998-07-13 12:22:34' );
```

1.13.5.2.66 MINUTES 関数 [日付と時刻]

TIMESTAMP を操作するか、2 つの TIMESTAMP 値の間の分境界の数を返します。

構文

0000-02-29 の真夜中と TIMESTAMP 値の間の分数を返します。

```
MINUTES( timestamp-expression )
```

2 つの TIMESTAMP 値の間の分数を返します。

```
MINUTES( timestamp-expression, timestamp-expression )
```

分を TIMESTAMP 値に追加します。

```
MINUTES( timestamp-or-time-expression, integer-expression )
```

パラメータ

timestamp-expression

TIMESTAMP データ型の式。

timestamp-or-time-expression

TIME または TIMESTAMP データ型の式。

integer-expression

`timestamp-or-time-expression` に追加する分数。`integer-expression` が負の場合、適切な分数が `timestamp-or-time-expression` から減算されます。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

戻り値

使用方法に応じて INTEGER、TIME、または TIMESTAMP。

備考

MINUTES 関数の結果は、その引数によって異なります。

0000-02-29 の真夜中以降の分数を返します。

1 つの `timestamp-expression` を MINUTES 関数に渡すと、0000-02-29 の真夜中から `timestamp-expression` までの秒数を INTEGER として返します。

> **注記**
>
> 0000-02-29 は実際の日付を指すための値ではありません。MINUTES 関数で使用されるデフォルトの日付です。

2 つの TIMESTAMP 値の間の分数を返します。

2 つの TIMESTAMP 値を MINUTES 関数に渡すと、その間の分境界の数を整数で返します。

分を TIMESTAMP 値に追加します。

TIMESTAMP 値と INTEGER 値を MINUTES 関数に渡すと、整数の分数を `timestamp-expression` 引数に加算した TIMESTAMP 結果を返します。同様に、MINUTES の最初の引数が TIME 値の場合は、結果も TIME 値になります。この構文では、最初の引数の暗黙的変換をサポートしていません。最初の引数を DATE、TIME または TIMESTAMP の値に明示的にキャストしてください。最初の引数が DATE データ型の場合、時刻部分には真夜中が使用されます。

MINUTES は整数を返すので、4083-03-23 02:08:00 以上の TIMESTAMP 値で使用すると、オーバーフローが起こる場合があります。

計算の一部に DATEDIFF 関数と DATEADD 関数を使用することもできます。

> **例**
>
> 次の文は、同一の値 240 を返します。これは、2 番目の TIMESTAMP 値が、最初のタイムスタンプの 240 分後であることを示します。DATEDIFF を使用することをおすすめします。

```
SELECT
  MINUTES( '1999-07-13 06:07:12',
           '1999-07-13 10:07:12' ),
  DATEDIFF( minute,
           '1999-07-13 06:07:12',
           '1999-07-13 10:07:12' );
```

次の文は、値 1051040527 を返します。

```
SELECT MINUTES( '1998-07-13 06:07:12' );
```

次の文は、TIMESTAMP 値 1999-05-12 21:10:07.000 を返します。最初の文では、リテラル文字列パラメータの明示的なキャストが必要です。2 つ目の例 (DATEADD) の使用をおすすめします。

```
SELECT MINUTES( CAST( '1999-05-12 21:05:07' AS TIMESTAMP ), 5);
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

```
SELECT DATEADD( minute, 5, '1999-05-12 21:05:07' );
```

関連情報

CAST 関数 [データ型変換] [339 ページ]

1.13.5.2.67 ML_GET_SERVER_NOTIFICATION 関数 [システム]

この関数を使用すると、Ultra Light ユーザはライトウェイトポーリングを使用して、サーバ起動同期要求について Mobile Link サーバ上の Notifier にクエリできます。

構文

```
ML_GET_SERVER_NOTIFICATION( notifier, address, key)
```

パラメータ

notifier

ポーリング対象の Mobile Link サーバ上の Notifier 名。

address

ストリームパラメータ。たとえば、

```
tcpip{host=pc1;port=1234}
```

key

省略可能です。通知要求キーです。

戻り値

指定された要求キーに対する通知要求の件名と内容を返します。

備考

指定された要求キーに対する要求がない場合、または指定された Notifier 名が Mobile Link サーバ上に存在しない場合、結果は NULL になります。要求キーに NULL が指定された場合、そのユーザのリモート ID が要求キーとして使用されます。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

要求が存在する場合、結果メッセージは次の形式で返されます。[subject]content (たとえば [sync]profile1 など)。

この関数は、Mobile Link サーバからの応答を取得する際にネットワーク経由の通信を行います。そのため、ネットワーク遅延により実行に時間がかかることがあります。実行中、この関数をバックグラウンドで実行できる期間がある場合があります。その場合、他の接続上のランタイム内で作業を行うことができます。ただし、このような期間の存在が保証されているわけではなく、SQL の複雑さに依存します。この関数で使用する Mobile Link アドレスを取得する場合は、sync_profile_option_value 関数で既存の同期プロファイルを使用して、Stream プロファイルオプションの値を取得することをおすすめします。この関数が返す値は、そのまま Mobile Link アドレスパラメータとして使用できます。

例

次の文は、要求キー MyKey に対する通知要求の件名と内容を返します。

```
SELECT ML_GET_SERVER_NOTIFICATION('Notifier1', 'tcpip{host=sap;port=1234}',
'MyKey');
```

関連情報

SYNC_PROFILE_OPTION_VALUE 関数 [システム] - Ultra Light [461 ページ]

1.13.5.2.68 MOD 関数 [数値]

整数を整数で割ったときの余りを返します。

構文

```
MOD( dividend, divisor)
```

パラメータ

dividend

被除数 (分数の分子)。

divisor

除数 (分数の分母)。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

戻り値

- SMALLINT
- INT
- NUMERIC

備考

被除数が負の場合、除算の結果は負または 0 になります。除数の符号は計算結果に影響を与えません。

例

次の文は、値 2 を返します。

```
SELECT MOD( 5, 3 );
```

関連情報

REMAINDER 関数 [数値] [423 ページ]

1.13.5.2.69 MONTH 関数 [日付と時刻]

指定した日付の月を返します。

構文

```
MONTH( date-expression )
```

パラメータ

date-expression

DATE データ型の値。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

戻り値

SMALLINT

備考

戻り値は、指定の日付の月に対応する 1 ～ 12 の間の数です。

例

次の文は、値 7 を返します。

```
SELECT MONTH( '1998-07-13' );
```

1.13.5.2.70 MONTHNAME 関数 [日付と時刻]

日付から月の名前を返します。

構文

```
MONTHNAME( date-expression )
```

パラメータ

timestamp-expression

TIMESTAMP 値。

戻り値

VARCHAR

備考

結果が数値 (2 月を示す 2 など) の場合でも、MONTHNAME 関数は文字列を返します。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

SQL Anywhere では、英語のロケールでは英語の名前が返され、英語以外のロケールでは英語以外の名前が返されます。たとえば、Language (LANG) 接続パラメータを使用すれば他の言語を指定できます。

例

次の文は、英語のロケールの場合、September という値を返します。

```
SELECT MONTHNAME( '1998-09-05' );
```

SQL Anywhere では、フランス語のロケールの場合、septembre という値を返します。スペイン語のロケールの場合、Septiembre という値を返します。複数のロケールがサポートされています。

関連情報

DATEPART 関数 [日付と時刻] [362 ページ]

1.13.5.2.71 MONTHS 関数 [日付と時刻]

TIMESTAMP を操作するか、2 つの TIMESTAMP 値の間の月境界の数を返します。

構文

0000-02 と TIMESTAMP 値の間の月数を返します。

```
MONTHS( timestamp-expression )
```

2 つの TIMESTAMP 値の間の月数を返します。

```
MONTHS( timestamp-expression, timestamp-expression )
```

月を TIMESTAMP 値に追加します。

```
MONTHS( timestamp-expression, integer-expression )
```

パラメータ

timestamp-expression

TIMESTAMP データ型の日付と時刻。

integer-expression

timestamp-expression に加算される整数の月数 (SMALLINT データ型)。integer-expression が負の場合、適切な月数が timestamp-expression から減算されます。integer-expression を指定する場合は、

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

timestamp-expression を TIME、DATE または TIMESTAMP データ型として明示的にキャストしてください。timestamp-expression が TIME 値の場合、現在の月が使用されます。

戻り値

使用方法に応じて INTEGER または TIMESTAMP。

備考

MONTHS 関数の結果は、その引数によって異なります。MONTHS 関数では、引数内の時間、分、および秒は無視されます。

0000-02 以降の月数を返します。

1 つの timestamp-expression を MONTHS 関数に渡すと、0000-02 から timestamp-expression までの月数を INTEGER として返します。

> **注記**
>
> 0000-02-29 は実際の日付を指すための値ではありません。MONTHS 関数で使用されるデフォルトの日付です。

2 つの TIMESTAMP 値の間の月数を返します。

2 つの TIMESTAMP 値を MONTHS 関数に渡すと、その間の月境界を整数で返します。

月を TIMESTAMP 値に追加します。

TIMESTAMP 値と SMALLINT 値を MONTHS 関数に渡すと、整数の月数を timestamp-expression に加算した TIMESTAMP 結果を返します。

DATEDIFF 関数と DATEADD 関数を使用すると、これらの計算の一部を実行できます。

MONTHS の値を求めるには、2 つの日付の間に月の最初の日がいくつあるかを計算します。

> **例**
>
> 次の文は、値 2 を返します。これは、2 番目の日付が、最初の日付の 2 か月後であることを示します。2 つ目の例 (DATEDIFF) の使用をお奨めします。

```
SELECT MONTHS( '1999-07-13 06:07:12', '1999-09-13 10:07:12' );
```

```
SELECT DATEDIFF( month,
   '1999-07-13 06:07:12',
   '1999-09-13 10:07:12' );
```

次の文は、値 23981 を返します。

```
SELECT MONTHS( '1998-07-13 06:07:12' );
```


© 2016 SAP SE or an SAP affiliate company. All rights reserved.

次の文は、TIMESTAMP 値 1999/10/12 21:05:07.000 を返します。2 つ目の例 (DATEADD) の使用をお奨めします。

```
SELECT MONTHS( CAST( '1999-05-12 21:05:07' AS DATETIME ), 5);
```

```
SELECT DATEADD( month, 5, '1999-05-12 21:05:07' );
```

関連情報

DATEDIFF 関数 [日付と時刻] [358 ページ]
DATEADD 関数 [日付と時刻] [356 ページ]
CAST 関数 [データ型変換] [339 ページ]

1.13.5.2.72 NEWID 関数 [その他]

UUID (ユニバーサルユニーク識別子) 値を生成します。UUID は、GUID (グローバルユニーク識別子) と同じです。

構文

NEWID( )

パラメータ

NEWID 関数に関連付けられているパラメータはありません。

戻り値

UNIQUEIDENTIFIER

備考

NEWID 関数は、カラムの DEFAULT 句で使用できます。

UUID を使用して、テーブルのローをユニークに識別できます。コンピュータが異なると生成される値も異なるので、値は同期環境やレプリケーション環境でキーとして使用できます。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

他の RDBMS と互換性を保つために、UUID にはハイフンが含まれます。このデフォルト設定は、uuid_has_hyphens option を Off にして変更することができます。

Ultra Light の UUID にはハイフンは含まれません。

NEWID 関数は、非決定的関数です。NEWID 関数を連続して呼び出すと、毎回異なる値が返されます。クエリオプティマイザは、NEWID 関数の結果をキャッシュしません。

例

次の文は、2 つのカラムを持つテーブル mytab を作成します。カラム pk は uniqueidentifier データ型とし、NEWID 関数をデフォルト値として割り当てます。カラム c1 は integer データ型です。

```
CREATE TABLE mytab(
   pk UNIQUEIDENTIFIER PRIMARY KEY DEFAULT NEWID(),
   c1 INT );
```

次の文は、ユニーク識別子を文字列として返します。

```
SELECT UUIDTOSTR( NEWID() );
```

たとえば、戻り値が 96603324-6FF6-49DE-BF7D-F44C1C7E6856 の場合もあります。

関連情報

STRTOUUID 関数 [文字列] [454 ページ]
UUIDTOSTR 関数 [文字列] [469 ページ]

1.13.5.2.73 NOW 関数 [日付と時刻]

現在の日付と時刻を TIMESTAMP 値で返します。精度はシステムクロックの精度によって制限されます。

構文

```
NOW( [ * ] )
```

戻り値

TIMESTAMP

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

備考

NOW は、GETDATE 関数および CURRENT TIMESTAMP 特別値と同義です。NOW(*) と NOW() の構成は同義です。

要求に含まれる NOW 関数の各インスタンスは、多くても 1 回のみ評価されます。同じ要求に含まれる NOW の複数のインスタンスは、同じ TIMESTAMP 値を共有する場合もあれば、そうでない場合もあります。

注記

データベースでシミュレートされたタイムゾーンが使用されている場合、シミュレートされたタイムゾーンを使用してこの関数の結果が計算されます。

例

次の文は、現在の日付と時刻を返します。

```
SELECT NOW( * );
```

関連情報

CURRENT TIMESTAMP 特別値 - Ultra Light [253 ページ]
GETDATE 関数 [日付と時刻] [377 ページ]

1.13.5.2.74 NULLIF 関数 [その他]

式を比較して、CASE 式の省略形を提供します。

構文

```
NULLIF( expression-1, expression-2 )
```

パラメータ

expression-1

比較される式。

expression-2

比較される式。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

戻り値

最初の引数のデータ型。

備考

NULLIF は 2 つの式の値を比較します。

1 番目の式と 2 番目の式が等しい場合、NULLIF は NULL を返します。

1 番目の式と 2 番目の式が異なる場合、または 2 番目の式が NULL の場合、NULLIF は 1 番目の式を返します。

NULLIF 関数は、いくつかの CASE 式の簡単な作成方法を提供します。

例

次の文は、値 a を返します。

```
SELECT NULLIF( 'a', 'b' );
```

次の文は、NULL を返します。

```
SELECT NULLIF( 'a', 'a' );
```

関連情報

CASE 式 - Ultra Light [259 ページ]

1.13.5.2.75 PATINDEX 関数 [文字列]

文字列のパターンが最初に出現した開始位置を表す整数を返します。

構文

```
PATINDEX( '%pattern%', string-expression )
```

パラメータ

pattern

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

検索するパターン。先頭の % ワイルドカードを省略すると、PATINDEX 関数は、パターンが文字列の先頭に出現する場合は 1 を返し、それ以外の場合は 0 を返します。

パターンは、LIKE 比較と同じワイルドカードを使用します。これらのワイルドカードを次の表に示します。

Ultra Light のパターンは、次の表のワイルドカードを使用します。

ワイルドカード	一致するもの
_ (アンダースコア)	任意の 1 文字
% (パーセント記号)	0 個以上の文字からなる任意の文字列
[]	指定範囲内、または一連の指定文字の任意の 1 文字
[^]	指定範囲外、または一連の指定文字以外の任意の 1 文字

string-expression

パターンを検索する文字列。

戻り値

INT

備考

PATINDEX 関数は、パターンが最初に出現した開始位置を返します。パターンが見つからない場合は、0 を返します。

例

次の文は、値 2 を返します。

```
SELECT PATINDEX( '%hoco%', 'chocolate' );
```

次の文は、値 11 を返します。

```
SELECT PATINDEX( '%4_5_', '0a1A 2a3A 4a5A' );
```

次の文は、14 を返します。これは、文字列式の最初の英数字以外の文字です。データベースで大文字と小文字が区別されない場合は、`'%[^a-zA-Z0-9]%'` の代わりにパターン `'%[^a-z0-9]%'` を使用できます。

```
SELECT PATINDEX( '%[^a-zA-Z0-9]%', 'SQLAnywhere17 has many new features' );
```

次の文を使用して、文字列の最初の英数字以外の文字までのすべての文字を取得することができます。

```
SELECT LEFT( @string, PATINDEX( '%[^a-zA-Z0-9]%', @string ) );
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

次の文は、myTable というテーブルを作成し、英数字文字、スペース (ブランク)、英数字以外の文字が含まれる各種の文字列を入力します。続いて、SELECT 文とその後の結果により、PATINDEX を使用して文字列内のスペースと英数字以外の文字の開始位置を検索する方法がわかります。

```
CREATE TABLE myTable( col1 LONG VARCHAR );
INSERT INTO myTable (col1) VALUES( 'the quick brown fox jumped over the lazy
dog' ),
( 'the quick brown fox $$$$ jumped over the lazy dog' ),
( 'the quick brown fox 0999 jumped over the lazy dog' ),
( 'the quick brown fox ** jumped over the lazy dog' ),
( 'thequickbrownfoxjumpedoverthelazydog' ),
( 'thequickbrownfoxjum999pedoverthelazydog' ),
( 'thequick$$$$brownfox' ),
( 'the quick brown fox$$ jumped over the lazy dog' );
SELECT col1,
   //position of first non-alphanumeric character or space:
   PATINDEX( '%[^a-z0-9]%', col1) AS blank_posn,
   //position of first non-alphanumeric char that isn't a space:
   PATINDEX( '%[^ a-z0-9]%', col1) AS non_alpha_char,
   //everything up to and including first non-alphanumeric char that isn't a
space:
   LEFT ( col1, PATINDEX( '%[^ a-zA-Z0-9]%', col1) ) AS left_str,
   //first non-alphanumeric char that isn't a space, and everything to the right:
   SUBSTRING ( col1, PATINDEX( '%[^ a-zA-Z0-9]%', col1) ) AS sub_str
FROM myTable;
```

col1	blank_posn	non_alpha_char	left_str	sub_str
the quick brown fox jumped over the lazy dog	4	0		the quick brown fox jumped over the lazy dog
the quick brown fox $$$$ jumped over the lazy dog	4	21	the quick brown fox $	$$$$ jumped over the lazy dog
the quick brown fox 0999 jumped over the lazy dog	4	0		the quick brown fox 0999 jumped over the lazy dog
the quick brown fox ** jumped over the lazy dog	4	21	the quick brown fox *	** jumped over the lazy dog
thequickbrownfoxjumpedoverthelazydog	0	0		thequickbrownfoxjumpedoverthelazydog
thequickbrownfoxjum999pedoverthelazydog	0	0		thequickbrownfoxjum999pedoverthelazydog
thequick$$$$brownfox	9	9	thequick$	$$$$brownfox

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

col1	blank_posn	non_alpha_char	left_str	sub_str
the quick brown fox$$ jumped over the lazy dog	4	20	the quick brown fox$	$$ jumped over the lazy dog

関連情報

LOCATE 関数 [文字列] [395 ページ]
STRING 関数 [文字列] [453 ページ]

1.13.5.2.76 PI 関数 [数値]

PI の数値を返します。

構文

```
PI( [ * ] )
```

戻り値

DOUBLE

備考

この関数は DOUBLE 値を返します。

セマンティック上、PI(*) と PI() は同義です。

例

次の文は、値 3.141592653(...) を返します。

```
SELECT PI( * );
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.13.5.2.77 POWER 関数 [数値]

数のべき乗を表す数を計算します。

構文

```
POWER( numeric-expression-1, numeric-expression-2 )
```

パラメータ

numeric-expression-1

べき乗の底。

numeric-expression-2

指数。

戻り値

DOUBLE

備考

この関数は、引数を DOUBLE に変換し、倍精度浮動小数点で計算を行います。引数のいずれかが NULL の場合、結果は NULL 値になります。

例

次の文は、値 64 を返します。

```
SELECT POWER( 2, 6 );
```

1.13.5.2.78 QUARTER 関数 [日付と時刻]

指定した TIMESTAMP 式から、四半期を示す数を返します。

構文

```
QUARTER( timestamp-expression )
```


© 2016 SAP SE or an SAP affiliate company. All rights reserved.

パラメータ

timestamp-expression

四半期にする日付。

戻り値

INTEGER

備考

各四半期は次のように定めます。

四半期	期間 (開始日と終了日を含む)
1	1 月 1 日～ 3 月 31 日
2	4 月 1 日～ 6 月 30 日
3	7 月 1 日～ 9 月 30 日
4	10 月 1 日～ 12 月 31 日

例

次の文は、値 2 を返します。

```
SELECT QUARTER( '1987/05/02' );
```

1.13.5.2.79 RADIANS 関数 [数値]

度数をラジアンに変換します。

構文

```
RADIANS( numeric-expression )
```

パラメータ

numeric-expression

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

度数。この角度をラジアンに変換します。

戻り値

DOUBLE

備考

この関数は、引数を DOUBLE に変換し、倍精度浮動小数点で計算を行います。

例

次の文は、約 0.5236 の値を返します。

```
SELECT RADIANS( 30 );
```

1.13.5.2.80 REMAINDER 関数 [数値]

整数を整数で割ったときの余りを返します。

構文

```
REMAINDER( dividend, divisor )
```

パラメータ

dividend

被除数 (分数の分子)。

divisor

除数 (分数の分母)。

戻り値

- INTEGER

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

- NUMERIC

備考

MOD 関数を使用して、余りを返すこともできます。

例

次の文は、値 2 を返します。

```
SELECT REMAINDER( 5, 3 );
```

関連情報

MOD 関数 [数値] [409 ページ]

1.13.5.2.81 REPEAT 関数 [文字列]

文字列を指定された回数だけ連結します。

構文

```
REPEAT( string-expression, integer-expression )
```

パラメータ

string-expression

繰り返される文字列。

integer-expression

文字列を繰り返す回数。`integer-expression` が正数でない場合は、空の文字列を返します。

戻り値

LONG VARCHAR

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

LONG NVARCHAR
Ultra Light の場合、LONG VARCHAR が返されます。

備考

実際の結果文字列の長さが戻り値の最大長を超えると、エラーが発生します。この場合、結果は文字列に使用できる最大サイズまでトランケートされます。

この関数の動作は、REPLICATE 関数と同じです。

この関数は NCHAR の入力または出力をサポートしています。

Ultra Light では NCHAR の入力または出力がサポートされていません。

例

次の文は、値 `repeatrepeatrepeat` を返します。

```
SELECT REPEAT( 'repeat', 3 );
```

関連情報

REPLICATE 関数 [文字列] [427 ページ]
STRING 関数 [文字列] [453 ページ]

1.13.5.2.82 REPLACE 関数 [文字列]

文字列を別の文字列で置換し、新しい結果を返します。

構文

```
REPLACE( original-string, search-string, replace-string )
```

パラメータ

いずれかの引数が NULL の場合、NULL を返します。

original-string

検索される文字列。任意の長さの文字列を指定できます。
search-string

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

検索して replace-string に置き換えられる文字列。この文字列は 255 バイトに制限されています。search-string が空の文字列の場合は、元の文字列がそのまま返されます。

replace-string

置換文字列。search-string を置き換えます。任意の長さの文字列を指定できます。replace-string が空の文字列の場合は、検索されたすべての search-string が削除されます。

戻り値

LONG BINARY
LONG VARCHAR
LONG NVARCHAR
Ultra Light は NVARCHAR を返しません。

備考

この関数は、すべての search-string を replace-string に置き換えます。

すべての引数がバイナリデータ型の場合、REPLACE 関数の動作は BYTE_REPLACE と同じになります。

大文字と小文字を区別するデータベースでの比較では、大文字と小文字が区別されます。

この関数は NCHAR の入力または出力をサポートしています。

Ultra Light では NCHAR の入力または出力がサポートされていません。

例

次の文は、値 xx.def.xx.ghi を返します。

```
SELECT REPLACE( 'abc.def.abc.ghi', 'abc', 'xx' );
```

次の文は、ALTER PROCEDURE 文を含む結果セットを生成します。ALTER PROCEDURE を実行すると、名前が変更されたテーブルを参照する格納済みプロシージャが修復されます (テーブル名を一意にすることをおすすめします)。

```
SELECT REPLACE(
   REPLACE( proc_defn, 'OldTableName', 'NewTableName' ),
   'CREATE PROCEDURE',
   'ALTER PROCEDURE')
FROM SYS.SYSPROCEDURE
WHERE proc_defn LIKE '%OldTableName%';
```

関連情報

SUBSTRING 関数 [文字列] [457 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

CHARINDEX 関数 [文字列] [345 ページ]
STRING 関数 [文字列] [453 ページ]

1.13.5.2.83 REPLICATE 関数 [文字列]

文字列を指定された回数だけ連結します。

構文

```
REPLICATE( string-expression, integer-expression )
```

パラメータ

string-expression

繰り返される文字列。

integer-expression

文字列を繰り返す回数。

戻り値

LONG VARCHAR
LONG NVARCHAR
Ultra Light は NVARCHAR を返しません。

備考

実際の結果文字列の長さが戻り値の最大長を超えると、エラーが発生します。この場合、結果は文字列に使用できる最大サイズまでトランケートされます。

この関数の動作は、REPEAT 関数と同じです。

この関数は NCHAR の入力または出力をサポートしています。

Ultra Light では NCHAR の入力または出力がサポートされていません。

例

次の文は、値 `repeatrepeatrepeat` を返します。

```
SELECT REPLICATE( 'repeat', 3 );
```


© 2016 SAP SE or an SAP affiliate company. All rights reserved.

関連情報

REPEAT 関数 [文字列] [424 ページ]
STRING 関数 [文字列] [453 ページ]

1.13.5.2.84 RIGHT 関数 [文字列]

文字列の一番右側にある文字を返します。

構文

```
RIGHT( string-expression, integer-expression )
```

パラメータ

string-expression

一番右側にある文字を返す文字列。

integer-expression

返される文字列の末尾の文字数。

戻り値

LONG VARCHAR
LONG NVARCHAR
Ultra Light は NVARCHAR を返しません。

備考

文字列にマルチバイト文字が含まれている場合は、返されるバイト数が指定した文字数よりも大きい場合があります。

カラムの値より大きな `integer-expression` を指定できます。この場合、全体の値が返されます。

この関数は NCHAR の入力または出力をサポートしています。入力文字が文字長のセマンティックを使用している場合、可能であれば、戻り値が文字長のセマンティックで記述されます。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

例

次の文は、Customers テーブルに含まれる各 Surname 値の最後の 5 文字を返します。

```
SELECT RIGHT( Surname, 5 ) FROM GROUPO.Customers;
```

関連情報

LEFT 関数 [文字列] [389 ページ]
STRING 関数 [文字列] [453 ページ]

1.13.5.2.85 ROUND 関数 [数値]

numeric-expression を integer-expression で指定した小数点以下の桁数に丸めます。

構文

```
ROUND( numeric-expression, integer-expression )
```

パラメータ

numeric-expression

関数に渡される、丸めの対象となる数。

integer-expression

正の整数は、丸めを行う小数点の右側の有効桁数を指定します。負の式は、丸めを行う小数点の左側の有効桁数を指定します。

戻り値

NUMERIC

備考

この関数の結果は numeric または double です。数値の結果があり、整数 integer-expression が負の値の場合、精度は 1 ずつ増えます。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

例

次の文は、値 123.200 を返します。

```
SELECT ROUND( 123.234, 1 );
```

関連情報

TRUNCNUM 関数 [数値] [466 ページ]

1.13.5.2.86 RTRIM 関数 [文字列]

文字列の後続ブランクを削除します。

構文

```
RTRIM( string-expression )
```

パラメータ

string-expression

削除する文字列。

戻り値

VARCHAR
NVARCHAR
LONG VARCHAR
LONG NVARCHAR
Ultra Light の場合、VARCHAR および LONG VARCHAR

備考

結果の実際の長さは、式の長さから、削除する文字数を引いた値です。すべての文字を削除すると、結果は空の文字列になります。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

引数が NULL の場合、結果は NULL 値になります。

この関数は NCHAR の入力または出力をサポートしています。

Ultra Light では NCHAR の入力または出力がサポートされていません。

例

次の文は、すべての後続ブランクが削除された文字列 `Test Message` を返します。

```
SELECT RTRIM( 'Test Message     ' );
```

関連情報

TRIM 関数 [文字列] [465 ページ]
LTRIM 関数 [文字列] [400 ページ]
STRING 関数 [文字列] [453 ページ]

1.13.5.2.87 SECOND 関数 [日付と時刻]

TIMESTAMP 引数の秒の値を返します。

構文

```
SECOND( timestamp-expression )
```

パラメータ

timestamp-expression

TIMESTAMP 値。

戻り値

SMALLINT

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

備考

指定した TIMESTAMP 引数値の秒に相当する 0 ～ 59 の数を返します。

例

次の文は、値 25 を返します。

```
SELECT SECOND( '1998-07-13 21:21:25' );
```

1.13.5.2.88 SECONDS 関数 [日付と時刻]

TIMESTAMP を操作するか、2 つの TIMESTAMP 値の間の秒境界の数を返します。

構文

0000-02-29 の真夜中と TIMESTAMP 値の間の月数を返します。

```
SECONDS( timestamp-expression )
```

2 つの TIMESTAMP 値の間の秒数を返します。

```
SECONDS( timestamp-expression, timestamp-expression )
```

秒を TIMESTAMP 値に追加します。

```
SECONDS( time-or-timestamp-expression, integer-expression )
```

パラメータ

timestamp-expression

TIMESTAMP 値。

time-or-timestamp-expression

TIME または TIMESTAMP データ型の値。

integer-expression

`time-or-timestamp-expression` に追加する秒数。`integer-expression` が負の場合、適切な秒数が `time-or-timestamp-expression` から減算されます。integer-expression を指定する場合は、`time-or-timestamp-expression` を TIME、DATE または TIMESTAMP データ型として明示的にキャストしてください。`time-or-timestamp-expression` が DATE データ型の場合、時刻部分には真夜中が使用されます。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

戻り値

0000-02-29 の真夜中と TIMESTAMP 値の間の月数を返す場合は UNSIGNED BIGINT

2 つの TIMESTAMP 値の間の秒数を返す場合は SIGNED BIGINT

TIMESTAMP 値に秒を追加する場合は TIME または TIMESTAMP

備考

SECONDS 関数の結果は、その引数によって異なります。

0000-02-29 の真夜中と TIMESTAMP 値の間の月数を返します。

1 つの `timestamp-expression` を SECONDS 関数に渡すと、0000-02-29 の真夜中から `timestamp-expression` までの秒数を UNSIGNED BIGINT として返します。

> **注記**
>
> 0000-02 は実際の日付を指すための値ではありません。SECONDS 関数で使用されるデフォルトの日付です。

2 つの TIMESTAMP 値の間の秒数を返します。

2 つの TIMESTAMP 値を SECONDS 関数に渡すと、その間の秒境界の数を整数の SIGNED BIGINT 値で返します。

秒を TIMESTAMP 値に追加します。

TIMESTAMP 値と INTEGER 値を SECONDS 関数に渡すと、整数の秒数を `time-or-timestamp-expression` に加算した TIMESTAMP 結果を返します。同様に、TIME 値を SECONDS 関数に渡すと、TIME データ型の値を返します。

DATEDIFF 関数と DATEADD 関数を使用すると、これらの計算の一部を実行できます。

> **例**
>
> 次の文は、同一の値 14400 を返します。これは、2 番目の TIMESTAMP 値が、最初のタイムスタンプの 14400 秒後であることを示します。

```
SELECT
  SECONDS( '1999-07-13 06:07:12',
           '1999-07-13 10:07:12' ),
  DATEDIFF( second,
           '1999-07-13 06:07:12',
           '1999-07-13 10:07:12' );
```

次の文は、値 63062431632 を返します。

```
SELECT SECONDS( '1998-07-13 06:07:12' );
```

次の文は TIMESTAMP 値 1999/05/12 21:05:12.000 を返します。

```
SELECT SECONDS( CAST( '1999-05-12 21:05:07' AS TIMESTAMP ), 5);
SELECT DATEADD( second, 5, '1999-05-12 21:05:07' );
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

関連情報

CAST 関数 [データ型変換] [339 ページ]
DATEADD 関数 [日付と時刻] [356 ページ]
DATEDIFF 関数 [日付と時刻] [358 ページ]

1.13.5.2.89 SHORT_PLAN 関数 [その他]

SQL 文の Ultra Light プランの最適化方法を短い記述の文字列で返します。この記述は、EXPLANATION 関数が返す記述と同じです。

構文

```
SHORT_PLAN( string-expression )
```

パラメータ

string-expression

SQL 文。通常は SELECT 文ですが、UPDATE 文または DELETE 文も指定できます。

戻り値

LONG VARCHAR

備考

クエリによっては、SQL Anywhere に対して選択したプランと Ultra Light の実行プランが異なる場合があります。

例

次の文は、SELECT 文を文字列パラメータとして渡し、クエリを実行するためのプランを返します。

```
SELECT SHORT_PLAN(
   'SELECT * FROM WHERE DepartmentID > 100' );
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

関連情報

EXPLANATION 関数 [その他] [373 ページ]

1.13.5.2.90 SIGN 関数 [数値]

指定された数値の符号 (正または負) を返します。

構文

```
SIGN( numeric-expression )
```

パラメータ

numeric-expression

符号を返す数値。`numeric-expression` の型は、INTEGER、DOUBLE、または NUMERIC です。

戻り値

SMALLINT

備考

負の数を指定すると、SIGN 関数は -1 を返します。

0 を指定すると、SIGN 関数は 0 を返します。

正の数を指定すると、SIGN 関数は 1 を返します。

例

次の文は、値 -1 を返します。

```
SELECT SIGN( -550 );
```


© 2016 SAP SE or an SAP affiliate company. All rights reserved.

1.13.5.2.91 SIMILAR 関数 [文字列]

2 つの文字列の類似性を示す数を返します。

構文

```
SIMILAR( string-expression-1, string-expression-2 )
```

パラメータ

string-expression-1

比較する最初の文字列。

string-expression-2

比較する 2 番目の文字列。

戻り値

SMALLINT

備考

SIMILAR 関数は、2 つの文字列の類似性を表す 0 ～ 100 の整数を返します。結果は、2 つの文字列の文字が一致している割合として解釈できます。値が 100 の場合は、2 つの文字列は同じです。

この関数を使用して、名前 (顧客名など) のリストを修正できます。顧客がわずかに異なる名前で重複して登録されている場合があります。SIMILAR 関数を使用して、類似する顧客名を見つけることができます。このことを行うには、customer テーブルをそのテーブル自体にジョインし、類似性が 90 パーセントより大きく、かつ 100 パーセント未満のすべての顧客のレポートを作成します。

SIMILAR 関数で実行される計算は、単に文字数が一致する場合よりも複雑になります。

例

次の文は値 75 を返します。2 つの値が 75 % 前後であることを示します。

```
SELECT SIMILAR( 'toast', 'coast' );
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

関連情報

STRING 関数 [文字列] [453 ページ]

1.13.5.2.92 SIN 関数 [数値]

数のサインを返します。

構文

```
SIN( numeric-expression )
```

パラメータ

numeric-expression

角度 (ラジアン)。

戻り値

DOUBLE

備考

SIN 関数は、引数のサインを返します。この引数はラジアン単位で表現される角度です。SIN 関数と ASIN 関数は逆変換の演算です。

この関数は、引数を DOUBLE に変換し、計算を倍精度浮動小数点で行い、結果を DOUBLE で返します。

例

次の文は、0.52 の SIN 値を返します。

```
SELECT SIN( 0.52 );
```


© 2016 SAP SE or an SAP affiliate company. All rights reserved.

関連情報

ASIN 関数 [数値] [332 ページ]
COS 関数 [数値] [350 ページ]
COT 関数 [数値] [351 ページ]
TAN 関数 [数値] [462 ページ]

1.13.5.2.93 SOUNDEX 関数 [文字列]

文字列の発音を表す数を返します。

構文

```
SOUNDEX( string-expression )
```

パラメータ

string-expression

評価される文字列。

戻り値

SMALLINT

備考

文字列の SOUNDEX 関数の値は、先頭の文字とそれに続く H、Y、W 以外の 3 つの子音がベースになっています。文字列の最初の文字である場合を除き、`string-expression` の母音は無視されます。同じ文字が 2 つ続く場合は 1 文字としてカウントされます。たとえば、"apples" という単語は、文字 A、P、L、S がベースになります。

マルチバイト文字は、SOUNDEX 関数では無視されます。

完全とは言えませんが、SOUNDEX 関数は通常、類似発音で同じ文字で始まる語句に対して同じ数を返します。

SOUNDEX 関数は、英単語の処理に最も優れています。他の言語の処理は英語の場合よりも劣ります。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

例

次の文は、それぞれの名前の発音を表す 2 つの同じ数 3827 を返します。

```
SELECT SOUNDEX( 'Smith' ), SOUNDEX( 'Smythe' );
```

関連情報

STRING 関数 [文字列] [453 ページ]

1.13.5.2.94 SPACE 関数 [文字列]

指定した数のスペースを返します。

構文

```
SPACE( integer-expression )
```

パラメータ

integer-expression

返すスペースの数。

戻り値

LONG VARCHAR

備考

`integer-expression` が正数でない場合は、NULL 文字列を返します。

例

次の文は、10 個のスペースを含む文字列を返します。

```
SELECT SPACE( 10 );
```


© 2016 SAP SE or an SAP affiliate company. All rights reserved.

関連情報

STRING 関数 [文字列] [453 ページ]

1.13.5.2.95 SQRT 関数 [数値]

数の平方根を返します。

構文

```
SQRT( numeric-expression )
```

パラメータ

numeric-expression

平方根が計算される数。

戻り値

DOUBLE

備考

この関数は、引数を DOUBLE に変換し、計算を倍精度浮動小数点で行い、結果を DOUBLE で返します。

例

次の文は、値 3 を返します。

```
SELECT SQRT( 9 );
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.13.5.2.96 ST_AsBinary 関数 [空間] - Ultra Light

指定されたジオメトリを表すバイナリ文字列を返します。

構文

```
ST_AsBinary(geometry-expression)
```

戻り値

BINARY

`geometry-expression` の WKB 表現を返します。

備考

出力フォーマットは、OGC SFS 1.1 で定義されている WKB です。

このフォーマットには、Z 値と M 値は含まれません。

例

次の文は、0x0101000000000000000000f03f0000000000000040 を返します。

```
SELECT ST_AsBinary(ST_Point(1.0, 2.0, 4326))
```

ST_AsBinary 関数は、サーバでジオメトリを BINARY に変換するときに暗黙的に呼び出されます。たとえば、次の文は結果 0x0101000000000000000000f03f0000000000000040 を返します。

```
SELECT CAST(ST_Point(1.0, 2.0, 4326) AS BINARY(50))
```

1.13.5.2.97 ST_AsExtText 関数 [空間] - Ultra Light

指定されたジオメトリを表すバイナリ文字列を返します。

構文

```
ST_AsExtText(geometry-expression)
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

戻り値

VARCHAR

`geometry-expression` の EWKT 表現を返します。

備考

出力フォーマットは EWKT です。

例

次の文は、SRID をプレフィクスとして含む結果 SRID=4326;Point (1 2) を返します。

```
SELECT ST_AsExtText(ST_Point(1.0, 2.0, 4326))
```

ST_AsExtText() 関数は、ジオメトリを VARCHAR 型に変換するときに暗黙的に呼び出されます。たとえば、次の文は結果 SRID=4326;Point(1 2) を返します。

```
SELECT CAST(ST_Point(1.0, 2.0, 4326) AS VARCHAR(25))
```

1.13.5.2.98 ST_AsText 関数 [空間] - Ultra Light

指定されたジオメトリを表すバイナリ文字列を返します。

構文

```
ST_AsText(geometry-expression)
```

戻り値

VARCHAR

`geometry-expression` の WKT 表現を返します。

備考

出力フォーマットは、OGC SFS 1.1 で定義されている WKT です。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

例

次の文は、Point (1 2) を返します。

```
SELECT ST_AsText(ST_Point(1.0, 2.0, 4326))
```

1.13.5.2.99 ST_Distance 関数 [空間] - Ultra Light

指定した 2 つのジオメトリ値間の最短距離を返します。

構文

```
ST_Distance( geo1,geo2 )
```

パラメータ

名前	タイプ	説明
geo1	ST_Geometry	2 つのジオメトリ値間の距離を計算するために使用する最初のジオメトリ値
geo2	ST_Geometry	2 つのジオメトリ値間の距離を計算するために使用する 2 番目のジオメトリ値

戻り値

DOUBLE

指定したジオメトリ値間の最短距離を返します。

備考

ポイントが SRID 4326 内にある場合、単位はメートルです。

例

次の文は、3367142.4632130372 を返します。

```
SELECT ST_Distance(ST_Point(-79.38,43.65,4326),ST_Point(-123.1,49.28,4326))
```


© 2016 SAP SE or an SAP affiliate company. All rights reserved.

1.13.5.2.100 ST_Equals 関数 [空間] - Ultra Light

ST_Geometry 値が別の ST_Geometry 値と空間的に等しいかどうかをテストします。2 つのジオメトリ値が同じ x 座標と y 座標を使用し、同じ参照系にある場合、値は等しいと見なされます。

構文

```
ST_Equals( geo1,geo2 )
```

パラメータ

名前	タイプ	説明
geo1	ST_Geometry	比較する最初のジオメトリ値
geo2	ST_Geometry	比較する 2 番目のジオメトリ値

戻り値

BIT

2 つのジオメトリ値が空間的に等しい場合は 1 を返し、それ以外の場合は 0 を返します。

備考

空間参照系の精度またはデータの精度によってテストが制限される場合があります。

ST_Equals 関数は、比較述部 (= と <>)、IN リスト述部、DISTINCT、GROUP BY に使用されるセマンティックを定義します。

例

次の文は、1 を返します。

```
SELECT ST_Equals(ST_Point(1,1,4326),ST_Point(1,1,4326))
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.13.5.2.101 ST_IntersectsRect 関数 [空間] - Ultra Light

2 つのポイント (最小と最大) によって指定されたボックス内にポイントがあるかどうかをテストします。

構文

```
ST_IntersectsRect(location,min,max)
```

パラメータ

名前	タイプ	説明
location	ST_Geometry	テストするポイント
min	ST_Geometry	ボックスの定義に使用する最小ポイント値
max	ST_Geometry	ボックスの定義に使用する最大ポイント値

戻り値

BIT

location が指定したボックスと交差する場合は 1 を返し、それ以外の場合は 0 を返します。

備考

なし。

例

次の文は、1 を返します。

```
SELECT ST_IntersectsRect(ST_Point(1,1,4326),ST_Point(0,0,4326),
ST_Point(3,3,4326))
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.13.5.2.102 ST_Point 関数 [空間] - Ultra Light

x 座標と y 座標に基づくポイントを作成します。

構文

```
ST_Point (x,y,SRID)
```

パラメータ

名前	タイプ	説明
x	DOUBLE	ポイントの作成に使用する x 座標
y	DOUBLE	ポイントの作成に使用する y 座標
SRID	INTEGER	ポイントに関連付けられる SRID 値

戻り値

ST_Point

入力文字列から作成された ST_Geometry 値を返します。

備考

なし。

例

次の文は、2163 参照系にポイント (10.0,20.0) を作成します。

```
SELECT ST_Point(10.0,20.0,2163)
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.13.5.2.103 ST_PointFromExtText 関数 [空間] - Ultra Light

ST_Geometry の EWKT 表現を含む VARCHAR 値から変換される ST_Geometry 値を返します。

構文

```
ST_PointFromExtText(ewkt)
```

パラメータ

名前	タイプ	説明
ewkt	VARCHAR	EWKT 表現

戻り値

ST_Geometry

入力文字列から作成された ST_Geometry 値を返します。

備考

なし。

例

次の文は、結果 SRID=4326;Point(10 20) を返して、4326 参照系にポイント (10,20) が作成されたことを示します。

```
SELECT ST_PointFromExtText('SRID=4326;Point(10 20)')
```

1.13.5.2.104 ST_PointFromText 関数 [空間] - Ultra Light

ST_Geometry の WKT 表現を含む VARCHAR 値から変換される ST_Geometry 値を返します。

構文

```
ST_PointFromText(wkt, srid)
```


© 2016 SAP SE or an SAP affiliate company. All rights reserved.

パラメータ

名前	タイプ	説明
wkt	VARCHAR	WKT 表現
srid	INT	結果の空間参照系識別子は、SRID パラメータで示したものです。

戻り値

ST_Geometry

入力文字列から作成された ST_Geometry 値を返します。

結果の空間参照系識別子は、パラメータ `srid` で指定します。

備考

なし。

例

次の文は、結果 SRID=4326;Point(10 20) を返して、4326 参照系にポイント (10, 20) が作成されたことを示します。

```
SELECT ST_PointFromText('Point(10 20)',4326)
```

1.13.5.2.105 ST_PointFromWKB 関数 [空間] - Ultra Light

ST_Geometry の WKB 表現を含む BINARY 値から変換される ST_Geometry 値を返します。

構文

```
ST_PointFromWKB(wkb, srid)
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

パラメータ

名前	タイプ	説明
wkb	BINARY	WKB 表現。
srid	INTEGER	ポイントに関連付けられる SRID 値

戻り値

ST_Geometry

入力文字列から作成された ST_Geometry 値を返します。

備考

なし。

例

次の文は、結果 (1.0, 2.0, 4326) を返します。

```
SELECT ST_PointFromWKB(0x0101000000000000000000f03f0000000000000040,4326)
```

1.13.5.2.106 ST_SRID 関数 [空間] - Ultra Light

ジオメトリ値に関連付けられている空間参照系 (SRID) を取得します。

構文

```
ST_SRID( geo1, srid )
```

パラメータ

名前	タイプ	説明
geo1	ST_Geometry	ポイント値

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

名前	タイプ	説明
srid	INTEGER	ポイントに関連付けられる SRID 値

戻り値

INT

ジオメトリの SRID を返します。

備考

なし。

例

次の文は、4326 を返します。

```
SELECT ST_SRID( ST_Point ( 10, 20, 4326 ) );
```

1.13.5.2.107 ST_X 関数 [空間] - Ultra Light

ST_Geometry 値の x 座標を返します。

構文

```
ST_X(geo1)
```

パラメータ

名前	タイプ	説明
geo1	ST_Geometry	x 座標を決定する ST_Geometry 値。

戻り値

DOUBLE

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

ST_Geometry 値の x 座標を返します。

備考

なし。

例

次の文は、10.0 を返します。

```
SELECT ST_X(ST_Point(10.0,20.0,4326))
```

1.13.5.2.108 ST_Y 関数 [空間] - Ultra Light

ST_Geometry 値の y 座標を返します。

構文

```
.ST_Y(geo1)
```

パラメータ

名前	タイプ	説明
geo1	ST_Geometry	y 座標を決定する ST_Geometry 値。

戻り値

DOUBLE

ST_Geometry 値の y 座標を返します。

備考

なし。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

例

次の例は、結果 20.0 を返します。

```
SELECT ST_Y(ST_Point(10.0,20.0,4326))
```

1.13.5.2.109 STR 関数 [文字列]

指定した数に相当する文字列を返します。

構文

```
STR( numeric-expression [, length [, decimal ] ] )
```

パラメータ

numeric-expression

-1E126 と 1E127 との間の任意の概数 (浮動小数点、実数、または倍精度)。

length

返される文字数 (小数点、小数点の左右のすべての桁、ブランクを含む)。デフォルトは 10 です。

decimal

返される小数点以下の桁数。デフォルトは 0 です。

戻り値

VARCHAR

備考

数の整数部分が指定した長さに合わない場合は、指定した長さの文字列がすべてアスタリスクで埋められて返されます。たとえば、次の文は *** を返します。

```
SELECT STR( 1234.56, 3 );
```

注記

サポートされる最大長は 128 です。1 ～ 128 の範囲にない長さでは結果が NULL になります。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

例

次の文は、6 つのスペースの後に 1235 が続く、合計で 10 の文字からなる文字列を返します。

```
SELECT STR( 1234.56 );
```

次の文は、1234.6 を返します。

```
SELECT STR( 1234.56, 6, 1 );
```

関連情報

STRING 関数 [文字列] [453 ページ]

1.13.5.2.110 STRING 関数 [文字列]

1 つ以上の文字列を連結して 1 つの長い文字列にします。

構文

```
STRING( string-expression [, ... ] )
```

パラメータ

string-expression

評価される文字列。

引数を 1 つだけ指定する場合は、1 つの式に変換されます。複数の引数を指定する場合は、連結されて 1 つの文字列になります。

戻り値

LONG VARCHAR
LONG BINARY

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

備考

数値または日付をパラメータとして指定した場合は、文字列に変換されてから連結されます。また、1 つの式を唯一のパラメータとして指定すると、STRING 関数を使用して、その式を文字列に変換できます。

すべてのパラメータが NULL の場合、STRING は NULL を返します。NULL でないパラメータが存在すると、NULL パラメータはすべて空の文字列として処理されます。

例

次の文は、値 `testing123` を返します。

```
SELECT STRING( 'testing', NULL, 123 );
```

関連情報

STR 関数 [文字列] [452 ページ]

1.13.5.2.111 STRTOUUID 関数 [文字列]

文字列の値をユニークな識別子 (UUID または GUID) の値に変換します。

構文

```
STRTOUUID( string-expression )
```

パラメータ

string-expression

`xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx` 形式の文字列。

戻り値

UNIQUEIDENTIFIER

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

備考

xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx の形式の文字列 (x は 16 進の桁) を、一意な識別子の値に変換します。

この関数は、UUID 値をデータベースに挿入するときに役立ちます。

文字列が有効な UUID 文字列でない場合は、conversion_error オプションが OFF に設定されていないかぎり変換エラーが返されます。このオプションが OFF の場合は NULL が返されます。この関数は NCHAR の入力または出力をサポートしています。波括弧は、string-expression の最初と最後の文字として使用できます。バージョン 9.0.2 より前に作成されたデータベースでは、UNIQUEIDENTIFIER データ型はユーザ定義データ型として定義されており、UUID 値のバイナリ表現と文字列表現の間の変換には STRTOUUID 関数と UUIDTOSTR 関数が必要でした。バージョン 9.0.2 以降を使用して作成されたデータベースでは、UNIQUEIDENTIFIER データ型がネイティブデータ型に変更されています。データ型の変換は、データベースサーバが必要に応じて実行します。これらのバージョンでは STRTOUUID 関数と UUIDTOSTR 関数を使用する必要はありません。

Ultra Light: バージョン 9.0.2 より前に作成されたデータベースでは、UUID 値のバイナリ表現と文字列表現を変換するために STRTOUUID 関数と UUIDTOSTR 関数が必要でした。バージョン 9.0.2 以降を使用して作成されたデータベースでは、UNIQUEIDENTIFIER データ型がネイティブデータ型に変更されています。これらのバージョンでは STRTOUUID 関数と UUIDTOSTR 関数を使用する必要はありません。

例

次の文は同等であり、いずれも結果 0x6c2b64a93c6f47dc901536b9ed49fec2 を返します。

```
SELECT STRTOUUID( '6c2b64a9-3c6f-47dc-9015-36b9ed49fec2' );
SELECT STRTOUUID( '{6c2b64a9-3c6f-47dc-9015-36b9ed49fec2}' );
```

関連情報

SQL データ型 [278 ページ]
UUIDTOSTR 関数 [文字列] [469 ページ]
NEWID 関数 [その他] [414 ページ]
STRING 関数 [文字列] [453 ページ]

1.13.5.2.112 STUFF 関数 [文字列]

1 つの文字列から複数の文字を削除して、別の文字列に置き換えます。

構文

```
STUFF( string-expression-1, start, length, string-expression-2 )
```


© 2016 SAP SE or an SAP affiliate company. All rights reserved.

パラメータ

string-expression-1

STUFF 関数によって変更される文字列。

start

削除を開始する文字の位置。文字列の先頭文字の位置を 1 とします。

length

削除する文字数。

string-expression-2

挿入する文字列。STUFF 関数を使用して文字列の一部を削除するには、NULL の置換文字列を使用します。

戻り値

入力式のデータ型に応じて、LONG BINARY、LONG VARCHAR、または LONG NVARCHAR。

備考

この関数は NCHAR の入力または出力をサポートしています。

Ultra Light では NCHAR の入力または出力がサポートされていません。

例

次の文は、値 chocolate pie を返します。

```
SELECT STUFF( 'chocolate cake', 11, 4, 'pie' );
```

関連情報

INSERTSTR 関数 [文字列] [384 ページ]
STRING 関数 [文字列] [453 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.13.5.2.113 SUBSTRING 関数 [文字列]

文字列の部分文字列を返します。

構文

```
{ SUBSTRING | SUBSTR }( string-expression, start
[, length ] )
```

パラメータ

string-expression

部分文字列が返される文字列。

start

文字単位で指定した、返される部分文字列の開始位置。

length

文字単位で指定した、返される部分文字列の長さ。`length` を指定すると、指定した長さの部分文字列に制限されます。

戻り値

LONG BINARY
LONG VARCHAR
LONG NVARCHAR
Ultra Light では LONG BINARY および LONG VARCHAR が返されます。

備考

文字列の末尾の文字を取得するには、RIGHT 関数を使用します。

`string-expression` が binary データ型の場合、SUBSTRING 関数は BYTE_SUBSTR のように動作します。

この関数は NCHAR の入力または出力をサポートしています。入力文字が文字長のセマンティックを使用している場合、可能であれば、戻り値が文字長のセマンティックで記述されます。この関数の動作は、ansi_substring データベースオプションの設定によって変わります。ansi_substring オプションを On (デフォルト) に設定した場合、SUBSTRING 関数は ANSI/ISO SQL 標準と同じ動作をします。動作を次に示します。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

ansi_substring オプション設定	start 値	length 値
On	文字列の先頭文字の位置が 1 になります。負またはゼロの開始オフセットは、文字列の左側が文字以外で埋められたように扱われます。	正の `length` は、部分文字列が開始位置の右側から `length` 文字で終わることを示します。 負の `length` はエラーを返します。
Off	文字列の先頭文字の位置が 1 になります。負の開始位置を指定する場合は、文字列の最初からの文字数ではなく、文字列の最後からの文字数を指定します。 `start` が 0 で length が負でない場合は、1 の start 値が使用されます。start が 0 で `length` が負の場合は、-1 の start 値が使用されます。	正の `length` は、部分文字列が開始位置の右側から `length` 文字で終わることを示します。 負の `length` は、開始位置から最大 `length` 文字左側の文字を返します。

Ultra Light: 入力文字が文字長のセマンティックを使用している場合、可能であれば、戻り値が文字長のセマンティックで記述されます。Ultra Light では、データベースに ansi_substring オプションがありませんが、デフォルトでは、SUBSTR 関数は ansi_substring が ON に設定されているように動作します。関数の動作は ANSI/ISO SQL 標準の動作に対応しています。

Start 値

文字列の先頭文字の位置が 1 になります。負またはゼロの開始オフセットは、文字列の左側が文字以外で埋められたように扱われます。

Length 値

正の `length` は、部分文字列が開始位置の右側から `length` 文字で終わることを示します。

負の `length` はエラーを返します。

0 の `length` は、空の文字列を返します。

例

次の表は、SUBSTRING 関数から返される値を示します。

Ultra Light: 次の表は、SUBSTRING 関数から返される値を示します。

例	結果
SUBSTRING('front yard', 1, 4)	fron
SUBSTRING('back yard', 6, 4)	yard
SUBSTR('abcdefgh', 0, -2)	エラーを返します
SUBSTR('abcdefgh', -2, 2)	空の文字列を返します

関連情報

BYTE_SUBSTR 関数 [文字列] [338 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

LEFT 関数 [文字列] [389 ページ]
RIGHT 関数 [文字列] [428 ページ]
CHARINDEX 関数 [文字列] [345 ページ]
STRING 関数 [文字列] [453 ページ]

1.13.5.2.114 SUM 関数 [集合]

ローグループごとに、指定された式の合計を返します。

構文

式

```
SUM( [ ALL | DISTINCT ] expression )
```

Window 関数

```
SUM( [ ALL ] expression ) OVER ( window-spec )
```

```
window-spec: 以下の備考部分を参照してください。
```

Ultra Light 構文:式

```
SUM( [ DISTINCT ] expression )
```

パラメータ

expression

合計される式の名前。通常はカラム名です。
[ALL] expression

合計される式の名前。通常はカラム名です。
DISTINCT expression

各グループの `expression` で一意の値の合計を計算します。

戻り値

- INTEGER
- DOUBLE
- NUMERIC

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

備考

指定された式が NULL のローは含まれません。

グループにローが含まれていない場合は、NULL を返します。

この関数ではオーバーフローエラーが発生することがあり、これにより、エラーが返される場合があります。オーバーフローエラーを回避するために、`numeric-expression` で CAST 関数を使用できます。

`window-spec` を使用してこの関数を指定した場合、SELECT 文で Window 関数として使用することを意味します。そのように、`window-spec` の要素は、関数構文の中 (インライン) に指定するか、または SELECT 文の WINDOW 句と組み合わせて指定できます。

例

次の文は、値 3749146.740 を返します。

```
SELECT SUM( Salary )
FROM GROUPO.Employees;
```

関連情報

COUNT 関数 [集合] [352 ページ]
AVG 関数 [集合] [335 ページ]

1.13.5.2.115 SWITCHOFFSET 関数 [日付と時刻]

元のタイムゾーンオフセットから指定のタイムゾーンオフセットに変換される TIMESTAMP WITH TIME ZONE 値を返します。

構文

```
SWITCHOFFSET( tmz-expression, time-zone-offset )
```

パラメータ

tmz-expression

変換される TIMESTAMP WITH TIME ZONE 値。

time-zone-offset

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

結果のタイムゾーンオフセット。協定世界時 (UTC: Coordinated Universal Time) の前または後の分数を表す整数、{ + | - } hh:nn 形式の文字列、またはズールータイムゾーンを表す Z のいずれかをこの値として使用できます。ズールータイムゾーンは、UTC と同じタイムゾーンです。

戻り値

TIMESTAMP WITH TIME ZONE

例

次の例は、タイムゾーンオフセット値を -04:00 時から -07:00 時に変更します。戻り値は 2009-04-03 11:45:12.123-07:00 です。

```
SELECT CAST ( '2009-04-03 14:45:12.123-04:00' AS datetimeoffset ) AS EDT,
SWITCHOFFSET( EDT,'-07:00' ) AS PDT;
```

1.13.5.2.116 SYNC_PROFILE_OPTION_VALUE 関数 [システム] - Ultra Light

指定されたオプション名に対応するオプションの値を返します。

構文

```
SYNC_PROFILE_OPTION_VALUE(profile_name, option_name)
```

パラメータ

profile_name

調べる同期プロファイルの名前。

option_name

値の取得対象であるオプションの名前。

戻り値

指定されたオプション名に対応するオプションの値を返します。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

備考

ピリオドを含むオプション名を指定すると、ピリオドの前の部分を基本オプション名、ピリオドの後ろの部分をサブリストオプション名として、サブリストから値を取得します。

例

次の文は、Example という名前の同期プロファイルに対する MobiLinkUid オプションの値を返します。

```
SELECT SYNC_PROFILE_OPTION_VALUE('Example', 'MobiLinkUid')
```

関連情報

ML_GET_SERVER_NOTIFICATION 関数 [システム] [408 ページ]

1.13.5.2.117 TAN 関数 [数値]

数のタンジェントを返します。

構文

```
TAN( numeric-expression )
```

パラメータ

numeric-expression

角度 (ラジアン)。

戻り値

DOUBLE

備考

ATAN 関数と TAN 関数は逆変換の演算です。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

この関数は、引数を DOUBLE に変換し、計算を倍精度浮動小数点で行い、結果を DOUBLE で返します。

例

次の文は、0.52 の tan 値を返します。

```
SELECT TAN( 0.52 );
```

関連情報

COS 関数 [数値] [350 ページ]
SIN 関数 [数値] [437 ページ]

1.13.5.2.118 TODATETIMEOFFSET 関数 [日付と時刻]

指定のタイムゾーンオフセットを使用して、TIMESTAMP 値を TIME STAMP WITH TIME ZONE 値に変換します。

構文

```
TODATETIMEOFFSET( timestamp-expression, time-zone-offset )
```

パラメータ

timestamp-expression

変換される TIMESTAMP 式。

time-zone-offset

タイムゾーンオフセット。UTC の前または後の分数を表す INTEGER、{ + | - } hh:nn 形式の VARCHAR、またはズールータイムゾーンを表す文字列 "Z" のいずれかをこの値として使用できます。ズールータイムゾーンは、UTC と同じタイムゾーンです。

戻り値

TIMESTAMP WITH TIME ZONE

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

例

次の例は、TIMESTAMP 値を TIMESTAMP WITH TIME ZONE 値に変換します。

```
SELECT CAST('2009-04-03 14:45:12.123' AS TIMESTAMP) AS orig,
        TODATETIMEOFFSET (orig,'+11:00');
```

1.13.5.2.119 TODAY 関数 [日付と時刻]

現在の日付を DATE 値で返します。

構文

```
TODAY( [ * ] )
```

戻り値

DATE

備考

セマンティック上、TODAY(*) と TODAY() は同義です。TODAY は CURRENT DATE 特別値と同義です。

要求に含まれる TODAY 関数の各インスタンスは、多くても 1 回のみ評価されます。同じ要求に含まれる TODAY の複数のインスタンスは、同じ DATE 値を共有する場合もあれば、そうでない場合もあります。

例

次の文は、システムクロックによる現在の日付を返します。

```
SELECT TODAY( * );
SELECT CURRENT DATE;
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.13.5.2.120 TRIM 関数 [文字列]

先行ブランクと後続ブランクを文字列から削除します。

構文

```
TRIM( string-expression )
```

パラメータ

string-expression

削除する文字列。

戻り値

VARCHAR
NVARCHAR
LONG VARCHAR
LONG NVARCHAR
Ultra Light の場合、VARCHAR または LONG VARCHAR が返されます。

備考

この関数は NCHAR の入力または出力をサポートしています。

Ultra Light では NCHAR の入力または出力がサポートされていません。

例

次の文は、先行ブランクや後続ブランクがすべて削除された値 chocolate を返します。

```
SELECT TRIM( '   chocolate   ' );
```

関連情報

LTRIM 関数 [文字列] [400 ページ]
RTRIM 関数 [文字列] [430 ページ]

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

STRING 関数 [文字列] [453 ページ]

1.13.5.2.121 TRUNCNUM 関数 [数値]

指定した桁数で小数点以下を切り捨てます。

構文

```
{ TRUNCNUM | TRUNCATE }( numeric-expression, integer-expression )
```

パラメータ

numeric-expression

トランケートされる数。この引数には、NUMERIC または DOUBLE データ型を使用できます。

integer-expression

正の整数は、丸めを行う小数点の右側の有効桁数を指定します。負の値は、丸めを行う小数点の左側の有効桁数を指定します。

戻り値

NUMERIC または DOUBLE

備考

数字をトランケートする場合、TRUNCATE 関数ではなく TRUNCNUM を使用します。

TRUNCATE 関数の使用はおすすめしません。truncate という単語はキーワードなので、quoted_identifier オプションを OFF に設定するか、単語 TRUNCATE を引用符で囲む必要があるためです。

Ultra Light: いずれかのパラメータが NULL 値の場合、結果は NULL 値になります。

例

次の文は、値 600 を返します。

```
SELECT TRUNCNUM( 655, -2 );
```

次の文は、値 655.340 を返します。

```
SELECT TRUNCNUM( 655.348, 2 );
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

関連情報

ROUND 関数 [数値] [429 ページ]

1.13.5.2.122 UCASE 関数 [文字列]

文字列中のすべての文字を大文字に変換します。

構文

```
UCASE( string-expression )
```

パラメータ

string-expression

大文字に変換される文字列。

戻り値

NCHAR データで使用された場合は LONG NVARCHAR
データベース照合が UCA の場合、CHAR データで使用されるときは LONG VARCHAR
それ以外の場合、データ型は入力データ型と同じになります。
Ultra Light では、入力データ型と同じデータ型が返されます。

備考

この関数は UPPER 関数と同じです。

例

次の文を実行すると、値 CHOCOLATE が返ります。

```
SELECT UCASE( 'ChocoLate' );
```


© 2016 SAP SE or an SAP affiliate company. All rights reserved.

関連情報

UPPER 関数 [文字列] [468 ページ]
LCASE 関数 [文字列] [388 ページ]
STRING 関数 [文字列] [453 ページ]

1.13.5.2.123 UPPER 関数 [文字列]

文字列中のすべての文字を大文字に変換します。

構文

```
UPPER( string-expression )
```

パラメータ

string-expression

大文字に変換される文字列。

戻り値

NCHAR データで使用された場合は LONG NVARCHAR
データベース照合が UCA の場合、CHAR データで使用されるときは LONG VARCHAR
それ以外の場合、データ型は入力データ型と同じになります。
Ultra Light では、入力データ型と同じデータ型が返されます。

備考

この関数は UCASE 関数と同じです。

例

次の文を実行すると、値 CHOCOLATE が返ります。

```
SELECT UPPER( 'ChocoLate' );
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

関連情報

UCASE 関数 [文字列] [467 ページ]
LCASE 関数 [文字列] [388 ページ]
LOWER 関数 [文字列] [399 ページ]
STRING 関数 [文字列] [453 ページ]

1.13.5.2.124 UUIDTOSTR 関数 [文字列]

ユニークな識別子の値 (UUID または GUID) を文字列の値に変換します。

構文

```
UUIDTOSTR( uuid-expression )
```

パラメータ

uuid-expression

ユニークな識別子の値。

戻り値

VARCHAR

備考

一意な識別子を、`xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx` の形式の文字列値に変換します (x は 16 進の桁)。バイナリ値が有効な uniqueidentifier でない場合は、NULL を返します。

この関数は、UUID 値を表示するのに便利です。

注記

バージョン 9.0.2 より前に作成されたデータベースでは、UNIQUEIDENTIFIER データ型はユーザ定義データ型として定義されており、UUID 値のバイナリ表現と文字列表現の間の変換には STRTOUUID 関数と UUIDTOSTR 関数が必要でした。バージョン 9.0.2 以降を使用して作成されたデータベースでは、UNIQUEIDENTIFIER データ型がネイティブデータ型に変更されています。データ型の変換は、データベースサーバが必要に応じて実行します。これらのバージョンでは STRTOUUID 関数と UUIDTOSTR 関数を使用する必要はありません。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

Ultra Light: バージョン 9.0.2 より前に作成されたデータベースでは、UUID 値のバイナリ表現と文字列表現の間を変換するための STRTOUUID 関数と UUIDTOSTR 関数が必要です。バージョン 9.0.2 以降を使用して作成されたデータベースでは、UNIQUEIDENTIFIER データ型がネイティブデータ型に変更されています。これらのバージョンでは STRTOUUID 関数と UUIDTOSTR 関数を使用する必要はありません。

例

次の文は、2 つのカラムを持つテーブル mytab を作成します。カラム pk は uniqueidentifier データ型で、カラム c1 は integer データ型です。それぞれに値 1 と値 2 を持つ 2 つのローをカラム c1 に挿入します。

```
CREATE TABLE mytab(
    pk UNIQUEIDENTIFIER PRIMARY KEY DEFAULT NEWID(),
    c1 INT );
INSERT INTO mytab( c1 ) values ( 1 );
INSERT INTO mytab( c1 ) values ( 2 );
```

次の SELECT 文を実行すると、新規に作成したテーブルのすべてのデータを返します。

```
SELECT * FROM mytab;
```

2 つのカラムと 2 つのローを持ったテーブルが表示されます。カラム pk に表示される値は、バイナリ値です。

ユニークな識別子の値を読みやすい形式に変換するには、次の文を実行します。

```
SELECT UUIDTOSTR(pk), c1 FROM mytab;
```

UUIDTOSTR 関数は、バージョン 9.0.2 以降で作成されたデータベースでは不要です。

関連情報

SQL データ型 [278 ページ]
NEWID 関数 [その他] [414 ページ]
STRTOUUID 関数 [文字列] [454 ページ]
STRING 関数 [文字列] [453 ページ]

1.13.5.2.125 WEEKS 関数 [日付と時刻]

TIMESTAMP を操作するか、2 つの TIMESTAMP 値の間の週の数を返します。

構文

0000-02-29 と TIMESTAMP 値の間の週数を返します。

```
WEEKS( timestamp-expression )
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

2 つの TIMESTAMP 値の間の週数を返します。

```
WEEKS( timestamp-expression, timestamp-expression )
```

週を TIMESTAMP 値に追加します。

```
WEEKS( timestamp-expression, integer-expression )
```

パラメータ

timestamp-expression

TIMESTAMP データ型の値の日付と時刻。

integer-expression

`timestamp-expression` に追加する週数。`integer-expression` が負の場合、適切な週数が `timestamp-expression` から減算されます。`integer-expression` を指定する場合は、`timestamp-expression` を DATE または TIMESTAMP として明示的にキャストしてください。

戻り値

2 つの TIMESTAMP 値を比較するときは INTEGER。

TIMESTAMP 値に週を追加する場合は TIMESTAMP。

備考

1 つの日付を指定すると、WEEKS 関数は、0000-02-29 からの週数を返します。

2 つの日付を指定すると、WEEKS 関数は、2 つの日付の間の週数を返します。WEEKS 関数は DATEDIFF 関数に似ていますが、2 つの日付間の週数を計算する場合に使用される方法はこれと同じではなく、返される結果が異なる場合があります。WEEKS の戻り値は、2 つの日付の間の日数を 7 で割り、余りを切り捨てて求められます。ただし、DATEDIFF では週の境目の数が使用されます。このため、2 つの関数から返される値が異なる場合があります。たとえば、最初の日付が金曜日で次の日付が翌月曜日の場合、WEEKS 関数は週数として 0 を返しますが、DATEDIFF 関数は 1 を返します。どちらかが優れているという問題ではないので、WEEKS と DATEDIFF との動作の違いをふまえて使用してください。

1 つの日付と整数を指定すると、WEEKS 関数は、`timestamp-expression` に、指定された整数の週数を加算します。この構文では、`timestamp-expression` を TIME、DATE、または TIMESTAMP データ型として明示的にキャストしてください。`timestamp-expression` が TIME 値の場合、現在の日付が使用されます。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

例

次の文は、値 8 を返します。これは、2008-09-13 10:07:12 が、2008-07-13 06:07:12 の 8 週間後であることを示します。

```
SELECT WEEKS( '2008-07-13 06:07:12', '2008-09-13 10:07:12' );
```

次の文は、値 104792 を返します。これは、指定された日付が、0000-02-29 の 104792 週間後であることを示します。

```
SELECT WEEKS( '2008-07-13 06:07:12' );
```

次の文は、TIMESTAMP 値 2008-06-16 21:05:07.0 を返します。これは、返された日時が 2008-05-12 21:05:07 の 5 週間後であることを示します。

```
SELECT WEEKS( CAST( '2008-05-12 21:05:07' AS TIMESTAMP ), 5);
```

関連情報

DATEDIFF 関数 [日付と時刻] [358 ページ]
DATEADD 関数 [日付と時刻] [356 ページ]
CAST 関数 [データ型変換] [339 ページ]

1.13.5.2.126 YEAR 関数 [日付と時刻]

TIMESTAMP 引数の年部分を返します。

構文

```
YEAR( timestamp-expression)
```

パラメータ

timestamp-expression

TIMESTAMP 値。

戻り値

SMALLINT

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

備考

指定された TIMESTAMP 値の年部分が SMALLINT として返されます。

例

次の例は、値 2001 を返します。

```
SELECT YEAR( '2001-09-12' );
```

1.13.5.2.127 YEARS 関数 [日付と時刻]

TIMESTAMP を操作するか、2 つの TIMESTAMP 値の間の年数を返します。

構文

0000 年と TIMESTAMP 値の間の年数を返します。

```
YEARS( timestamp-expression )
```

2 つの TIMESTAMP 値の間の年数を返します。

```
YEARS( timestamp-expression, timestamp-expression )
```

年を TIMESTAMP 値に追加します。

```
YEARS( timestamp-expression, integer-expression )
```

パラメータ

timestamp-expression

TIMESTAMP データ型の値の日付と時刻。

integer-expression

`timestamp-expression` に (SMALLINT 値として) 追加される年数。`integer-expression` が負の場合、適切な年数が `timestamp-expression` から減算されます。`integer-expression` を指定する場合は、`timestamp-expression` を DATE、TIME、または TIMESTAMP 値として明示的にキャストしてください。`timestamp-expression` が TIME の場合、現在の年が使用されます。

戻り値

2 つの TIMESTAMP 値を比較するときは SMALLINT。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

TIMESTAMP 値に年を追加する場合は TIMESTAMP。

備考

YEARS の値は、2 つの日付の間に年の最初の日がいくつあるかをカウントして求められます。

例

次の文はどちらも -4 を返します。

```
SELECT YEARS( '1998-07-13 06:07:12',
              '1994-03-13 08:07:13' );
```

```
SELECT DATEDIFF( year,
   '1998-07-13 06:07:12',
   '1994-03-13 08:07:13' );
```

次の文は、1998 を返します。

```
SELECT YEARS( '1998-07-13 06:07:12' )
SELECT DATEPART( year, '1998-07-13 06:07:12' );
```

次の文は、指定した日付の 300 年後を返します。

```
SELECT YEARS( CAST( '1998-07-13 06:07:12' AS TIMESTAMP ), 300 )
```

```
SELECT DATEADD( year, 300, '1998-07-13 06:07:12' );
```

関連情報

DATEDIFF 関数 [日付と時刻] [358 ページ]
DATEADD 関数 [日付と時刻] [356 ページ]
CAST 関数 [データ型変換] [339 ページ]

1.13.5.2.128 YMD 関数 [日付と時刻]

指定した年、月、日に相当する日付値を返します。引数は -32768 ～ 32767 の INTEGER です。

構文

```
YMD( smallint-expression1, smallint-expression2, smallint-expression3 )
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

パラメータ

smallint-expression1

年

smallint-expression2

月の数。月が 1 ～ 12 の範囲外である場合は、年が相応に調整されます。

smallint-expression3

日の数。任意の整数を指定でき、それに応じて日付が調整されます。

戻り値

DATE

例

次の文を実行すると、値 1998/06/12 が返ります。

```
SELECT YMD( 1998, 06, 12 );
```

値が通常の範囲から外れている場合、それに応じて日付が調整されます。たとえば、次の文は DATE 値 2000-03-01 を返します。

```
SELECT YMD( 1999, 15, 1 );
```

1.13.6 Ultra Light SQL 文

Ultra Light SQL でサポートされている SQL 文は、SQL Anywhere データベースでサポートされている文のサブセットです。

始める前に

- Ultra Light のテーブルでは、所有者の概念がサポートされていません。既存の SQL と、プログラムで生成される SQL のために、Ultra Light では `owner.table-name` の構文が許可されます。ただし、テーブル所有者は Ultra Light ではサポートされていないので、所有者はチェックされません。
- Ultra Light SQL 文のマニュアルは、SQL Anywhere の文で使用されている構文と同じ表記規則に従っています。この表記規則について、またこの表記規則を使用して SQL 構文が表されている方法を理解している必要があります。
- Ultra Light SQL を使用するとトランザクションが作成されます。トランザクションは、最後の ROLLBACK または COMMIT 文以降のすべての変更内容 (INSERT、UPDATE、DELETE 文) で構成されます。
  変更内容は、COMMIT を実行することによって確定できます。ROLLBACK 文を実行すると、変更内容が削除されます。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

このセクションの内容:

Ultra Light の文のカテゴリ [477 ページ]
SQL 文は、文の最初のキーワードで分類されています。この語はほとんど常に動詞です。このアクション指向型の構文では、言語の性質が、データベースに対する一連の命令文 (コマンド) になります。

ALTER DATABASE SCHEMA FROM FILE 文 [Ultra Light] [479 ページ]
SQL スクリプトを使用して既存の Ultra Light データベースのスキーマ定義を変更します。

ALTER PUBLICATION 文 [Ultra Light] [480 ページ]
パブリケーションを変更します。

ALTER SYNCHRONIZATION PROFILE 文 [Ultra Light] [481 ページ]
Ultra Light の同期プロファイルを変更します。

ALTER TABLE 文 [Ultra Light] [483 ページ]
テーブル定義を変更します。

ALTER USER 文 [Ultra Light] [487 ページ]
ユーザ設定を変更します。

CHECKPOINT 文 [Ultra Light] [488 ページ]
データベースにチェックポイントを設定します。

COMMIT 文 [Ultra Light] [489 ページ]
データベースへの変更を永続的なものします。

CREATE INDEX 文 [Ultra Light] [490 ページ]
指定したテーブルにインデックスを作成します。

CREATE PUBLICATION 文 [Ultra Light] [492 ページ]
パブリケーションを作成します。

CREATE SYNCHRONIZATION PROFILE 文 [Ultra Light] [493 ページ]
Ultra Light の同期プロファイルを作成または置換します。

CREATE TABLE 文 [Ultra Light] [495 ページ]
テーブルを作成します。

CREATE USER 文 [Ultra Light] [502 ページ]
データベースユーザまたはグループを作成します。

DELETE 文 [Ultra Light] [503 ページ]
データベースのテーブルからローを削除します。

DROP INDEX 文 [Ultra Light] [504 ページ]
インデックスを削除します。

DROP PUBLICATION 文 [Ultra Light] [505 ページ]
パブリケーションを削除します。

DROP SYNCHRONIZATION PROFILE 文 [Ultra Light] [506 ページ]
同期プロファイルを削除します。

DROP TABLE 文 [Ultra Light] [507 ページ]
テーブルとそのすべてのデータをデータベースから削除します。

DROP USER 文 [Ultra Light] [507 ページ]
ユーザを削除します。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

FROM 句 [Ultra Light] [509 ページ]
この句は、SELECT 文に必要なテーブルまたはビューを指定するために使用します。

INSERT 文 [Ultra Light] [511 ページ]
テーブルにローを挿入します。

LOAD TABLE 文 [Ultra Light] [512 ページ]
外部ファイルからデータベーステーブルにバルクデータをインポートします。

ROLLBACK 文 [Ultra Light] [516 ページ]
トランザクションを終了し、COMMIT 文または ROLLBACK 文が最後に実行されて以降の変更を元に戻します。

SELECT 文 [Ultra Light] [516 ページ]
データベースから情報を取得します。

SET OPTION 文 [Ultra Light] [519 ページ]
データベースオプションの値を変更します。

START SYNCHRONIZATION DELETE 文 [Ultra Light] [520 ページ]
Mobile Link 同期で削除されたローのロギングを再起動します。

STOP SYNCHRONIZATION DELETE 文 [Ultra Light] [521 ページ]
Mobile Link 同期で削除されたローのロギングを停止します。

SYNCHRONIZE 文 [Ultra Light] [522 ページ]
Mobile Link サーバを通じて Ultra Light データベースを同期させます。

TRUNCATE TABLE 文 [Ultra Light] [523 ページ]
テーブルを削除しないで、そのテーブルのすべてのローを削除します。

UNION 文 [Ultra Light] [525 ページ]
2 つ以上の SELECT 文の結果を結合します。

UPDATE 文 [Ultra Light] [526 ページ]
テーブルのローを変更します。

関連情報

Ultra Light でのトランザクション処理 [543 ページ]

1.13.6.1 Ultra Light の文のカテゴリ

SQL 文は、文の最初のキーワードで分類されています。この語はほとんど常に動詞です。このアクション指向型の構文では、言語の性質が、データベースに対する一連の命令文 (コマンド) になります。

Ultra Light でサポートされている SQL 文は次のように分類できます。

データ検索文

クエリとも呼ばれます。これらの文は、テーブルからデータ式のローを選択できます。データ検索は SELECT 文を使用して行います。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

データ操作文

データベースの内容を変更できます。データ操作は、次の文を使用して実行します。

- INSERT
- UPDATE
- DELETE

データ定義文

データベースの構造またはスキーマを定義できます。スキーマの変更には次の文を使用します。

- ALTER DATABASE SCHEMA FROM FILE
- CREATE INDEX
- CREATE TABLE
- DROP INDEX
- DROP TABLE
- ALTER TABLE
- TRUNCATE TABLE

トランザクション制御文

Ultra Light アプリケーション内でトランザクションを制御できます。トランザクション制御は、次の文を使用して実行します。

- CHECKPOINT
- COMMIT
- ROLLBACK

同期管理

Mobile Link サーバとの同期を一時的に制御できます。同期管理には次の文を使用します。

- START SYNCHRONIZATION DELETE
- STOP SYNCHRONIZATION DELETE
- CREATE PUBLICATION
- ALTER PUBLICATION
- DROP PUBLICATION

関連情報

Ultra Light の演算子 [275 ページ]
Ultra Light の式 [256 ページ]
SELECT 文 [Ultra Light] [516 ページ]
INSERT 文 [Ultra Light] [511 ページ]
UPDATE 文 [Ultra Light] [526 ページ]
DELETE 文 [Ultra Light] [503 ページ]
ALTER DATABASE SCHEMA FROM FILE 文 [Ultra Light] [479 ページ]
CREATE INDEX 文 [Ultra Light] [490 ページ]
CREATE TABLE 文 [Ultra Light] [495 ページ]
DROP INDEX 文 [Ultra Light] [504 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

DROP TABLE 文 [Ultra Light] [507 ページ]
ALTER TABLE 文 [Ultra Light] [483 ページ]
TRUNCATE TABLE 文 [Ultra Light] [523 ページ]
CHECKPOINT 文 [Ultra Light] [488 ページ]
COMMIT 文 [Ultra Light] [489 ページ]
ROLLBACK 文 [Ultra Light] [516 ページ]
START SYNCHRONIZATION DELETE 文 [Ultra Light] [520 ページ]
STOP SYNCHRONIZATION DELETE 文 [Ultra Light] [521 ページ]
CREATE PUBLICATION 文 [Ultra Light] [492 ページ]
ALTER PUBLICATION 文 [Ultra Light] [480 ページ]
DROP PUBLICATION 文 [Ultra Light] [505 ページ]

1.13.6.2 ALTER DATABASE SCHEMA FROM FILE 文 [Ultra Light]

SQL スクリプトを使用して既存の Ultra Light データベースのスキーマ定義を変更します。

構文

```
ALTER DATABASE SCHEMA FROM FILE filename
```

パラメータ

filename

既存の Ultra Light データベースのスキーマをアップグレードするときに使用する SQL スクリプトの名前とパスを定義します。

備考

スクリプトに必要な DDL 文を抽出するには、ulinit または ulunload のいずれかを使用します。これらのユーティリティを使用して、DDL 文が構文的に正しいことを確認してください。ulinit (-l `logfile` オプション) または ulunload (-n -s `output-file` オプション) を使用します。

この文は、データベースのバックアップを作成した後に実行します。

SQL スクリプトファイルの文字セットは、アップグレード対象のデータベースの文字セットに一致している必要があります。

この文の実行中は、デバイスをリセットしないでください。スキーマのアップグレード中にデバイスをリセットすると、Ultra Light データベースは使用できなくなります。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

スキーマに適合しないローはすべて削除されます (たとえば、一意性制約が追加され、複数のローに同じ値が入っている場合、1 つのローだけを残して他のローはすべて削除されます)。この場合、SQLE_ROW_DROPPED_DURING_SCHEMA_UPGRADE 警告が生成されます。この警告によってエラーを検出し、データベースをバックアップからリストアできます。

例

次の文は、SQL スクリプト `MySchema.sql` を使用して、データベースのスキーマを修正します。

```
ALTER DATABASE SCHEMA FROM FILE 'MySchema.sql'
```

関連情報

Ultra Light データベースのスキーマ [51 ページ]
Ultra Light データベーススキーマのアップグレードの配備 [125 ページ]
Ultra Light データベース初期化ユーティリティ (ulinit) [221 ページ]
Ultra Light データベースのアンロードユーティリティ (ulunload) [236 ページ]

1.13.6.3 ALTER PUBLICATION 文 [Ultra Light]

パブリケーションを変更します。

構文

```
ALTER PUBLICATION publication-name alterpub-clause
```

```
alterpub-clause :
  ADD TABLE table-name [ WHERE search-condition ]
| ALTER TABLE table-name  [ WHERE search-condition ]
| { DROP | DELETE } TABLE table-name
| RENAME publication-name
```

備考

パブリケーションは、同期対象のリモートデータベース内のデータを識別します。

関連する動作

オートコミット。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

例

次の ALTER PUBLICATION 文は、顧客テーブルを pub_contact パブリケーションに追加します。

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

関連情報

Ultra Light クライアントの同期設計 [77 ページ]
Ultra Light の探索条件 [263 ページ]
CREATE PUBLICATION 文 [Ultra Light] [492 ページ]
DROP PUBLICATION 文 [Ultra Light] [505 ページ]
START SYNCHRONIZATION DELETE 文 [Ultra Light] [520 ページ]
STOP SYNCHRONIZATION DELETE 文 [Ultra Light] [521 ページ]

1.13.6.4 ALTER SYNCHRONIZATION PROFILE 文 [Ultra Light]

Ultra Light の同期プロファイルを変更します。

構文

```
ALTER SYNCHRONIZATION PROFILE sync-profile-name
MERGE   sync-option [, ... ]
```

```
sync-option :
sync-option-name = sync-option-value
```

```
sync-option-name : string
```

```
sync-option-value : string
```

パラメータ

sync-profile-name

同期プロファイルの名前。

MERGE 句

この句を使用して、既存のオプションを変更したり、同期プロファイルに新しいオプションを追加したりします。

sync-option

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

1 つ以上の同期オプションの値ペアがセミコロンで区切られた文字列。たとえば、`'option1=value1;option2=value2'` のように記述します。

sync-option-name

同期プロファイルオプションの名前。

sync-option-value

同期プロファイルオプションの値。

備考

同期プロファイルによって、Ultra Light データベースが Mobile Link サーバと同期する方法を定義します。

MERGE 句を使用して、既存の同期プロファイルを変更できます。この句を使用すると、MERGE 句で指定した同期オプションだけが変更されます。同期プロファイルから同期オプションを削除するには、`'option1=;'` のような sync-option 文字列を作成します (オプションの値を空に設定します)。

同期プロファイルオプション STREAM は、その値にサブリストが含まれているため他のオプションとは異なります。次に例を示します。`'STREAM=TCPIP{host=192.168.1.1;port=1234}'` と同等です。この例では、`'host=192.168.1.1;port=1234'` がサブリストです。サブリストの値を追加または削除するには、STREAM の sync-option-name と sub-option-name の間にピリオドを使用します。たとえば、`MERGE 'stream.port=5678;stream.host=;compression=zlib'` のようにすると、同期プロファイルは `stream=TCPIP{port=5678;compression=zlib}` になります。ストリームを新しい値に設定しようとすると、ストリーム全体の値が置換されます。次に例を示します。`MERGE 'stream=HTTPS'` のようにすると、同期プロファイルは stream=HTTPS{} になります。

関連する動作

なし。

例

次の例は、ALTER SYNCHRONIZATION PROFILE...REPLACE 文を示します。

```
CREATE SYNCHRONIZATION PROFILE myProfile1;
ALTER SYNCHRONIZATION PROFILE myProfile1
   REPLACE 'publications=p1;uploadonly=on';
```

次の例は、ALTER SYNCHRONIZATION PROFILE...MERGE 文を示します。

```
CREATE SYNCHRONIZATION PROFILE myProfile2 'publications=p1;
ALTER SYNCHRONIZATION PROFILE myProfile2
   MERGE 'publications=p2;uploadonly=on';
```

次の例は、別のオプションで一連の ALTER SYNCHRONIZATION PROFILE コマンドを実行した後に発生する変更を示します。

`myProfile1='MobiLinkUID=mary;ScriptVersion=default'` とします。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

`ALTER SYNCHRONIZATION PROFILE myProfile1 REPLACE 'MobiLinkPwd=sql;ScriptVersion=1'` を実行した後、myProfile1 は `'MobiLinkPwd=sql;ScriptVersion=1'` になります。

`ALTER SYNCHRONIZATION PROFILE myProfile1 MERGE 'MobiLinkUID=mary;STREAM=tcpip'` を実行した後、myProfile1 は `'MobiLinkPwd=sql;ScriptVersion=1;MobiLinkUID=mary;STREAM=tcpip'` になります。

`ALTER SYNCHRONIZATION PROFILE myProfile1 MERGE 'MobiLinkUID=;STREAM.host=192.168.1.1;STREAM.port=1234;ScriptVersion=;'` を実行した後、myProfile1 は `'MobiLinkPwd=sql;STREAM=tcpip{192.168.1.1;port=1234}'` になります。

`ALTER SYNCHRONIZATION PROFILE myProfile1 MERGE 'MobiLinkPwd=;Ping=yes;STREAM =HTTP'` を実行した後、myProfile1 は `'Ping=yes;STREAM=HTTP'` になります。

`ALTER SYNCHRONIZATION PROFILE myProfile1 MERGE 'STREAM=HTTP{host=192.168.1.1}'` を実行した後、myProfile1 は `'Ping=yes;STREAM=HTTP{host=192.168.1.1}'` になります。

関連情報

Ultra Light データベース同期プロファイルオプション [234 ページ]
DROP SYNCHRONIZATION PROFILE 文 [Ultra Light] [506 ページ]
SYNCHRONIZE 文 [Ultra Light] [522 ページ]

1.13.6.5 ALTER TABLE 文 [Ultra Light]

テーブル定義を変更します。

構文

```
ALTER TABLE table-name {
add-clause
| modify-clause
| drop-clause
| rename-clause
}
```

```
add-clause :
 ADD { column-definition | table-constraint }
```

```
modify-clause :
ALTER column-definition | sync-constraint
```

```
drop-clause :
DROP { column-name | CONSTRAINT constraint-name }
```

```
rename-clause :
RENAME {
```


© 2016 SAP SE or an SAP affiliate company. All rights reserved.

```
    new-table-name
    | [ old-column-name TO ] new-column-name
    | CONSTRAINT old-constraint-name TO new-constraint-name }
```

```
column-definition :
column-name data-type
    [ [ NOT ] NULL ]
    [ DEFAULT column-default ]
    [ UNIQUE ]
```

```
column-default :
GLOBAL AUTOINCREMENT [ ( number ) ]
| AUTOINCREMENT
| CURRENT DATE
| CURRENT TIME
| CURRENT TIMESTAMP
| NULL
| NEWID( )
| constant-value
```

```
table-constraint :
[ CONSTRAINT constraint-name ]
{ fkey-constraint | unique-key-constraint }
[ WITH MAX HASH SIZE integer ]
```

```
fkey-constraint :
[ NOT NULL ] FOREIGN KEY [ role-name ] ( ordered-column-list )
    REFERENCES table-name ( column-name, ... )
    [ CHECK ON COMMIT ]
```

```
unique-key-constraint :
UNIQUE ( ordered-column-list )
```

```
 ordered-column-list :
( column-name [ ASC | DESC ], ... )
```

```
sync-constraint :SYNCHRONIZE {ON| OFF|ALL|DOWNLOAD}
```

パラメータ

add-clause

テーブルに新しいカラム制約またはテーブル制約を追加します。

テーブルに新しいカラム制約またはテーブル制約を追加します。

ADD column-definition clause

テーブルに新しいカラムを追加します。カラムにデフォルト値がある場合は、新しいカラムのすべてのローにそのデフォルト値が移植されます。

ADD table-constraint clause

テーブルに制約を追加します。オプションの制約名を使用すると、テーブル全体の制約を修正することなく、後で個々の制約を修正したり削除したりできます。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

新しいユニークな制約を追加する場合は、すべての制約カラムを非 NULL 可にする必要があります。ユニークな制約を追加するには、カラムを NOT NULL に変更してください。

> **注記**
>
> Ultra Light では、プライマリキーを追加することはできません。

modify-clause

1 つのカラム制約を変更します。`column-definition` が ALTER 文の一部の場合には、プライマリキーは使用できません。必要に応じて、変更されるカラムのデータを新しいデータ型に変換します。変換エラーが発生すると、操作は失敗となり、テーブルは変更されません。

drop-clause

カラム制約またはテーブル制約を削除します。

DROP column-name

テーブルからカラムを削除します。カラムがインデックス、一意性制約、外部キー、またはプライマリキーに含まれている場合は、それらのオブジェクトを削除しないと、Ultra Light がカラムを削除できません。

DROP CONSTRAINT table-constraint

テーブル定義から指定された制約を削除します。

> **注記**
>
> Ultra Light では、プライマリキーを削除することはできません。

rename-clause

テーブル名、カラム名、制約名を変更します。

RENAME new-table-name

テーブルの名前を `new-table-name` に変更します。古いテーブル名を使用しているアプリケーションは、修正される必要があります。古いテーブル名を自動的に割り当てられた外部キーは、名前を変更しません。

RENAME old-column-name TO new-column-name

カラムの名前を `new-column-name` に変更します。古いカラム名を使用しているアプリケーションは、修正する必要があります。

RENAME old-constraint-name TO new-constraint-name

制約の名前を `new-constraint-name` に変更します。古い定数名を使用しているアプリケーションは、修正する必要があります。

> **注記**
>
> Ultra Light では、プライマリキーの名前を変更することはできません。

column-constraint

カラム制約は、データベース内のデータの整合性を保つために、カラムに格納できる値を制限します。カラム制約は UNIQUE である必要があります。

UNIQUE

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

テーブル内の各ローをユニークに識別する 1 つまたは複数のカラムを識別します。テーブル内の異なるローが、指定されているすべてのカラムで同じ値を持つことはできません。1 つのテーブルに複数の一意性制約が存在することがあります。

sync-constraint 句

テーブルを同期可能にするかどうかおよびすべてのローをアップロードするか、テーブルへの変更だけをアップロードするか、テーブルへの変更をアップロードしないかを決定するための同期制約を指定します。

SYNCHRONIZE ON

デフォルト設定 - テーブルを同期することができ、テーブルへの変更のみがアップロードで送信されます。

SYNCHRONIZE OFF

テーブルを同期できず、テーブルをパブリケーションに含めるときにエラーが発生します。

SYNCHRONIZE ALL

テーブルを同期することができ、テーブル内のすべてのローがアップロードで送信されます。

SYNCHRONIZE DOWNLOAD

テーブルを統合データベースへの変更と同期できますが、ローカルの変更はアップロードされません。

備考

1 つの ALTER TABLE 文では、1 つの `table-constraint` または `column-constraint` だけを追加、変更、または削除できます。

役割名は外部キーの名前です。`role-name` の主な機能は、同じテーブルに対する 2 つの外部キーを区別することです。また、CONSTRAINT `constraint-name` を使用して外部キーに名前を付けることができます。ただし、両方のメソッドを使用して外部キーに名前を付けないでください。

テーブルとカラムの制約は MODIFY (変更) できません。制約を変更するには、古い制約を DELETE (削除) し、新しい制約を ADD (追加) します。

テーブルを対象とした文が、別の要求やクエリですでに参照されている場合、ALTER TABLE は実行できません。同様に、テーブルの変更中は、そのテーブルを参照する要求は処理されません。また、データベースにアクティブなクエリやコミットされていないトランザクションがある場合は ALTER TABLE を実行できません。

Ultra Light.NET の場合: すべてのデータオブジェクト (たとえば ULDataReader) に対して ULBulkCopy.Dispose メソッドも呼び出さないと、この文を実行できません。

スキーマの変更が同時に開始された場合、文は解放されません。

例

次の文は、MyEmployees という架空のテーブルから Street カラムを削除します。

```
ALTER TABLE MyEmployees
DROP Street
```

次の例は、MyCustomers という架空のテーブルの Street カラムを変更して、約 50 文字を格納するようにします。

```
ALTER TABLE MyCustomers
ALTER Street CHAR(50)
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

関連情報

Ultra Light データベースのスキーマ [51 ページ]
SQL データ型 [278 ページ]
分割サイズ [75 ページ]
最後に割り当てられた GLOBAL AUTOINCREMENT 値の調べ方 [74 ページ]
CREATE TABLE 文 [Ultra Light] [495 ページ]
DROP TABLE 文 [Ultra Light] [507 ページ]

1.13.6.6 ALTER USER 文 [Ultra Light]

ユーザ設定を変更します。

構文 1

```
ALTER USER user-name [ IDENTIFIED BY password ]
```

パラメータ

user-name

ユーザの名前。

IDENTIFIED BY 句

ユーザのパスワード。

備考

プロシージャの定義は SYSPROCEDURE システムビューに表示されるため、この文をプロシージャ内で使用する場合は、文字列リテラルとしてパスワード (IDENTIFIED BY 句) を指定しないでください。セキュリティ保護のため、プロシージャ定義の外部で宣言される変数を使用してパスワードを指定してください。

- ユーザ ID には、次に該当する値を指定できません。
  - 空白スペース、一重引用符、または二重引用符で始まる値
  - 空白スペースで終わる値
  - セミコロンを含む値
- パスワードは大文字と小文字が区別され、次に該当する値は指定できません。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

- 空白スペース、一重引用符、または二重引用符で始まる値
- 空白スペースで終わる値
- セミコロンを含む値
- 255 バイト長を超える値

関連する動作

なし。

例

次の文は、ユーザ SQLTester を変更します。パスワードは、"welcome" に設定されます。

```
ALTER USER SQLTester IDENTIFIED BY welcome
```

1.13.6.7 CHECKPOINT 文 [Ultra Light]

データベースにチェックポイントを設定します。

構文

```
CHECKPOINT
```

備考

CHECKPOINT 文をコミットフラッシュのトリガとして使用できます。コミットフラッシュが実行されると、コミットされていないトランザクションが記憶領域に書き込まれます。

Embedded SQL API を使用している場合は、ULCheckpoint メソッドを使用することもできます。C++ コンポーネントアプリケーションを作成している場合は、接続オブジェクトで Checkpoint メソッドを使用することもできます。その他すべての API では、この文を使用する必要があります。

関連する動作

この文によって、保留中のコミットされたトランザクションが記憶領域にすべてフラッシュされますが、現在のトランザクションはコミットまたはフラッシュされません。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

例

次の文は、データベースのチェックポイントを実行します。

```
CHECKPOINT
```

関連情報

単一のトランザクションまたはグループ化されたトランザクションのフラッシュ [544 ページ]
COMMIT 文 [Ultra Light] [489 ページ]
Ultra Light COMMIT_FLUSH 接続パラメータ [183 ページ]

1.13.6.8 COMMIT 文 [Ultra Light]

データベースへの変更を永続的なものします。

構文

```
COMMIT [ WORK ]
```

備考

Ultra Light SQL を使用するとトランザクションが作成されます。トランザクションは、最後の ROLLBACK または COMMIT 後のすべての変更内容 (INSERT、UPDATE、DELETE) から構成されます。COMMIT 文は、現在のトランザクションを終了し、このトランザクション中に行われたすべての変更内容をデータベースに永続化します。

ALTER、CREATE、DROP の各文を使用したデータベースオブジェクトの変更は、自動的にコミットされます。

例

次の文は、現在のトランザクションで行われた変更内容をデータベースに永続化します。

```
COMMIT
```

関連情報

CHECKPOINT 文 [Ultra Light] [488 ページ]
ROLLBACK 文 [Ultra Light] [516 ページ]

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

1.13.6.9 CREATE INDEX 文 [Ultra Light]

指定したテーブルにインデックスを作成します。

構文

```
CREATE [ UNIQUE ] INDEX [ IF NOT EXISTS ]  [ index-name ]
ON table-name ( ordered-column-list )
[ WITH MAX HASH SIZE integer ]
```

```
ordered-column-list :
( column-name [ ASC | DESC ], ... )
```

パラメータ

UNIQUE

UNIQUE 属性は、テーブル内の 2 つのローがインデックス内のすべてのカラムで同じ値にならないようにします。各インデックスキーはユニークであるか、少なくとも 1 つのカラムで NULL を持つ必要があります。

テーブル上の一意性制約とユニークインデックスの間には違いがあります。ユニークインデックスのカラムは NULL を使用できますが、一意性制約のカラムには使用できません。外部キーは、プライマリキーまたは一意性制約は参照できますが、ユニークインデックスは参照できません。ユニークインデックスは NULL の複数のインスタンスを含むことがあるからです。

一意性制約内のカラムが更新時に変更され、その一意性制約が外部キーによって参照されている場合、その一意性制約内のローを今後参照しないローは、すべてリモートから削除されます。

IF NOT EXISTS 句

IF NOT EXISTS 属性を指定し、指定したインデックスがすでに存在する場合、変更は行われず、エラーも返されません。

ordered-column-list

カラムの順序リストです。インデックスのカラム値は、昇順または降順にソートできません。

WITH MAX HASH SIZE

このインデックスのハッシュサイズをバイト単位で設定します。この値は、データベースに対して有効なデフォルトの MaxHashSize データベースプロパティを上書きします。

備考

Ultra Light では、プライマリキーと一意性制約のインデックスは自動的に作成されます。

インデックスによって、Ultra Light が特定のローを検索しやすくし、クエリのパフォーマンスを向上させることができます。ただし、インデックスを管理する必要があるので、同期や INSERT、DELETE、UPDATE の各文の処理は低速になる可能性があります。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

インデックスは、データベースに対して発行されたクエリのパフォーマンスを向上させたり、ORDER BY 句を使用してクエリをソートするときに自動的に使用されます。インデックスはいったん作成されると、DROP INDEX を使用して削除するときを除いて、SQL 文では二度と参照されません。

インデックスはデータベース内の領域を占有します。また、インデックスの管理に必要な追加作業は、データ修正操作のパフォーマンスに影響する場合があります。このため、クエリのパフォーマンスが向上しないインデックスの作成は避けてください。

CREATE INDEX 文の処理中は、そのインデックスを参照する要求やクエリは処理されません。また、データベースにアクティブなクエリやコミットされていないトランザクションがある場合は CREATE INDEX を実行できません。

実行プランを使用すると、クエリを最適化できます。

Ultra Light.NET を使用している場合、すべてのデータオブジェクト (たとえば ULDataReader) に対して ULBulkCopy.Dispose メソッドも呼び出さないと、この文を実行できません。

データベーススキーマの変更が同時に開始された場合、文は解放されません。

関連する動作

- オートコミット。

例

次の文は、従業員テーブルで 2 カラムのインデックスを作成します。

```
CREATE INDEX employee_name_index
ON Employees ( Surname, GivenName )
```

次の文は、ProductID カラム用に SalesOrderItems テーブル上でインデックスを作成します。

```
CREATE INDEX item_prod
ON SalesOrderItems ( ProductID )
```

次に、Ultra Light Java Edition データベースに対する MAX HASH SIZE の影響を示すシナリオを示します。従業員テーブルの Initials カラムは VARCHAR(3)、EmployeeID カラムは TINY です。

次の文は、ASCII7 文字のみが使用されると、すべての値を完全にハッシュします。

```
CREATE INDEX ascii_a ON Employees( Initials ) WITH MAX HASH SIZE 3
```

次の文は、使用される文字に関係なく、すべての値を完全にハッシュします。

```
CREATE INDEX unicode_a ON Employees( Initials ) WITH MAX HASH SIZE 9
```

次の文は、ASCII のみが使用されていても、Initials の値しかハッシュしません。Initials の最初の 9 バイトは予約されているためです。

```
CREATE INDEX compound_1 ON Employees( Initials, EmployeeID ) WITH MAX HASH SIZE 9
```

次の文は Initials と EmployeeID の値を両方とも完全にハッシュします。

```
CREATE INDEX compound_2 ON Employees( Initials, EmployeeID ) WITH MAX HASH SIZE 10
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

関連情報

Ultra Light のパフォーマンスに関するヒント [527 ページ]
Ultra Light インデックス [59 ページ]
Ultra Light データベースのスキーマ [51 ページ]
Ultra Light の実行プラン [534 ページ]
データベースプロパティの読み取り [43 ページ]
DROP INDEX 文 [Ultra Light] [504 ページ]
Ultra Light max_hash_size 作成オプション [155 ページ]

1.13.6.10 CREATE PUBLICATION 文 [Ultra Light]

パブリケーションを作成します。

構文

```
CREATE PUBLICATION [ IF NOT EXISTS ]publication-name
( TABLE table-name [ WHERE search-condition ], ... )
```

パラメータ

IF NOT EXISTS 句

IF NOT EXISTS 句を指定し、指定したパブリケーションがすでに存在する場合、変更は行われず、エラーも返されません。

TABLE 句

このテーブルは、パブリケーションに TABLE を含めるときに使用します。TABLE 句の数に制限はありません。

WHERE 句

WHERE 句を指定した場合、同期時に、`search-condition` を満たすローだけが、関連付けられたテーブルからのアップロード対象になります。

WHERE 句を指定しなかった場合は、最後の同期後にテーブル内で変更されたすべてのローがアップロード対象になります。

備考

パブリケーションは、Ultra Light リモートデータベース内の同期されたデータを示します。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

パブリケーションは、単一の同期操作で同期されるテーブルを設定し、Mobile Link サーバにアップロードされるデータを決定します。Mobile Link サーバのダウンロードセッション中には、これらのテーブルだけのローが送り返される可能性がありますが、ダウンロードされるローは、テーブルの WHERE 句を満たす必要はありません。

テーブル全体だけをパブリッシュできます。Ultra Light では、テーブルの特定のカラムはパブリッシュできません。

関連する動作

- オートコミット。

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 Contacts,
   TABLE Customers
)
```

次の文は、Customers テーブルで、State カラムに MN が含まれているローのみをパブリッシュします。

```
CREATE PUBLICATION pub_customer (
   TABLE Customers
   WHERE State = 'MN'
)
```

関連情報

Ultra Light クライアント [71 ページ]
Ultra Light の探索条件 [263 ページ]
DROP PUBLICATION 文 [Ultra Light] [505 ページ]
ALTER PUBLICATION 文 [Ultra Light] [480 ページ]
Ultra Light の探索条件 [263 ページ]

1.13.6.11 CREATE SYNCHRONIZATION PROFILE 文 [Ultra Light]

Ultra Light の同期プロファイルを作成または置換します。

構文

CREATE [OR REPLACE] SYNCHRONIZATION PROFILE `sync-profile-name sync-option [;...]`

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

```
sync-option :
sync-option-name = sync-option-value
```

```
sync-option-name : string
```

```
sync-option-value : string
```

パラメータ

OR REPLACE 句

指定の同期プロファイルがすでに存在する場合は、そのプロファイルが置換されます。プロファイルが存在しない場合は作成されます。

sync-profile-name

同期プロファイルの名前。

sync-option

1 つ以上の同期オプションの値ペアがセミコロンで区切られた文字列。たとえば、`'option1=value1;option2=value2'` のように記述します。

sync-option-name

同期プロファイルオプションの名前。

sync-option-value

同期プロファイルオプションの値。

備考

同期プロファイルによって、Ultra Light データベースが Mobile Link サーバと同期する方法を定義します。

REPLACE 句を使用して、既存の同期プロファイルを変更できます。この句により、同期プロファイルの内容が、新しい sync-option 文字列に含まれている内容に置換されます。この方法は、同期プロファイルを削除してから、新しい文字列を使用して同じ名前のプロファイルを作成する方法と同じです。したがって、同期プロファイルに完全な同期定義を含める必要はありません。同期の実行時にパラメータをマージまたは上書きできるからです。

同期プロファイルオプション STREAM は、その値にサブリストが含まれているため他のオプションとは異なります。次に例を示します。`'STREAM=TCPIP{host=192.168.1.1;port=1234}'`.この例では、`'host=192.168.1.1;port=1234'` がサブリストです。サブリストの値を追加または削除するには、STREAM の sync-option-name と sub-option-name の間にピリオドを使用します。たとえば、`MERGE 'stream.port=5678;stream.host=;compression=zlib'` のようにすると、同期プロファイルは `stream=TCPIP{port=5678;compression=zlib}` になります。ストリームを新しい値に設定しようとすると、ストリーム全体の値が置換されます。次に例を示します。`MERGE 'stream=HTTPS'` のようにすると、同期プロファイルは stream=HTTPS{} になります。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

関連する動作

なし。

例

次の文は、Test1 という同期プロファイルを作成します。

```
CREATE SYNCHRONIZATION PROFILE Test1
'MobiLinkUid=mary;Stream=TCPIP{host=192.168.1.1;port=1234}'
```

関連情報

Ultra Light データベース同期プロファイルオプション [234 ページ]
ALTER SYNCHRONIZATION PROFILE 文 [Ultra Light] [481 ページ]
DROP SYNCHRONIZATION PROFILE 文 [Ultra Light] [506 ページ]
SYNCHRONIZE 文 [Ultra Light] [522 ページ]

1.13.6.12 CREATE TABLE 文 [Ultra Light]

テーブルを作成します。

構文

```
CREATE TABLE [ IF NOT EXISTS ] table-name (
  { column-definition | table-constraint| sync-constraint }, ...
)
```

```
column-definition :
column-name  data-type
[ [ NOT ] NULL ]
[ DEFAULT column-default]
[STORE AS FILE (file-name-column) [CASCADE DELETE ]
[ column-constraint ]
```

```
column-default :
AUTOFILENAME(prefix,extension)
| GLOBAL AUTOINCREMENT [ ( number ) ]
| AUTOINCREMENT
| CURRENT DATE
| CURRENT TIME
| CURRENT TIMESTAMP
| CURRENT UTC TIMESTAMP
| NULL
| NEWID( )
| constant-value
```

```
file-name
```


© 2016 SAP SE or an SAP affiliate company. All rights reserved.

```
"filename"
```

```
column-constraint :
PRIMARY KEY
| UNIQUE
```

```
table-constraint :
{ [ CONSTRAINT constraint-name ]
   pkey-constraint
   | fkey-constraint
   | unique-key-constraint }
[ WITH MAX HASH SIZE integer ]
```

```
pkey-constraint :
PRIMARY KEY [ ordered-column-list ]
```

```
fkey-constraint :
[ NOT NULL ] FOREIGN KEY [ role-name ] ( ordered-column-list )
   REFERENCES table-name ( column-name, ... )
   [ CHECK ON COMMIT ]
```

```
unique-key-constraint :
UNIQUE ( ordered-column-list )
```

```
 ordered-column-list :
( column-name [ ASC | DESC ], ... )
```

```
sync-constraint :SYNCHRONIZE {ON| OFF| ALL| DOWNLOAD}
```

パラメータ

IF NOT EXISTS 句

この句は、テーブルを作成するために使用します。指定された名前のテーブルがすでに存在する場合、変更は行われず、エラーも返されません。

column-definition

テーブル内のカラムを定義します。この句には、次のパラメータを使用できます。

column-name

カラム名は識別子です。同じテーブル内の 2 つのカラムが同じ名前を持つことはできません。

data-type

カラムのデータ型。

[NOT] NULL

NOT NULL が指定されているか、カラムが PRIMARY KEY または UNIQUE 制約下にある場合、カラムはいずれのローでも NULL を持つことはできません。それ以外の場合は、NULL は許可されます。

column-default

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

カラムのデフォルト値を設定します。DEFAULT 値を指定する場合、カラムの値を指定しない INSERT 文のカラムの値としてこのデフォルト値が使用されます。INSERT 文はカラムの値を指定しません。DEFAULT を指定しない場合、DEFAULT NULL と等しくなります。デフォルトのオプションを以下に示します。

AUTOFILENAME

この句では、分割された Ultra Light Java Edition データベースへの外部 BLOB ファイルの格納がサポートされています。

データベースを分割するときに、ファイル名を格納するように指定されたカラムには、AUTOFILENAME(prefix,extensio) 句が必要です。この句では、ダウンロードされた BLOB 値用の新しいファイル名をどのように生成するかを指定します。prefix 値と extension 値は文字列リテラル定数です。

AUTOINCREMENT

AUTOINCREMENT を使用する場合、カラムは整数データ型の 1 つ、または真数値型にします。テーブルに挿入する場合、AUTOINCREMENT カラムの値を指定しないと、カラム内の任意の値より大きいユニーク値が生成されます。INSERT で、カラムの現在の最大値より大きい値を指定した場合、この値が後続の挿入処理の開始ポイントとして使用されます。

> **注記**
>
> Ultra Light では、テーブルが作成された時点でのオートインクリメントの初期値は 0 ではありません。カラムに符号付きデータ型が指定されている場合は、オートインクリメントによって負の値が生成されます。このため、オートインクリメントを適用するカラムを符号なし整数として宣言し、負の値が生成されないようにしてください。

GLOBAL AUTOINCREMENT

ドメインが分割されるという点を除いて、AUTOINCREMENT と同じです。各分割には同じ数の値が含まれます。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 では、データベースのデフォルト値は、そのデータベース番号でユニークに識別された分割からのみ設定されます。

> **注記**
>
> カラムの型が BIGINT または UNSIGNED BIGINT である場合、デフォルトの分割サイズは 2^32 = 4294967296 です。それ以外の型のカラムの場合、デフォルトの分割サイズは 2^16 = 65536 です。特に、カラムの型が INT または BIGINT ではない場合は、これらのデフォルト値が適切では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。

[NOT] NULL

カラムに NULL を格納できるかどうかを制御します。

NEWID()

ユニークな識別子の値を生成する関数です。

CURRENT TIMESTAMP

CURRENT DATE と CURRENT TIME を結合して、TIMESTAMP 値を形成します。年、月、日、時、分、秒、秒の小数位で構成されます。秒は小数第 3 位まで格納されます。精度はシステムクロックの精度によって制限されます。

CURRENT UTC TIMESTAMP

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

TIMESTAMP WITH TIME ZONE の値には、年、月、日、時、分、秒、秒の小数位、タイムゾーンで構成される協定世界時 (UTC: Coordinated Universal Time) が含まれます。秒は小数第 3 位まで格納されます。精度はシステムクロックの精度によって制限されます。

CURRENT DATE

現在の年、月、日を格納します。

CURRENT TIME

現在の時、分、秒 (小数位あり) で構成される時刻を格納します。

constant-value

カラムのデータ型の定数です。通常、この定数は数値または文字列です。

STORE AS FILE (file-name-column) [CASCADE DELETE]

Ultra Light Java Edition データベースでのみサポートされます。

外部保存される LONG BINARY カラム (データベース分割) を指定し、外部保存される BLOB 値のファイル名の保存に使用されるカラムを命名する `file-name-column` を指定します。この句を持つカラムは LONG BINARY 型で読み取り専用カラムとして動作する必要があります。

column-constraint 句

カラム制約を指定して、カラムに許容される値を制限します。カラム制約は次のいずれかです。

PRIMARY KEY

`column-constraint` の一部として設定するときに、PRIMARY KEY 句はテーブルのプライマリキーとしてカラムを設定します。プライマリキーは、テーブル内の各ローをユニークに識別します。デフォルトでは、プライマリキーに含まれるカラムには NULL を使用できません。

UNIQUE

テーブル内の各ローをユニークに識別する 1 つまたは複数のカラムを識別します。テーブル内の異なるローが、指定されているすべてのカラムで同じ値を持つことはできません。1 つのテーブルに複数の一意性制約が存在することがあります。NULL 値は使用できません。

table-constraint 句

テーブル制約を指定して、テーブル内の 1 つまたは複数のカラムに格納できる値を制限します。CONSTRAINT 句を使用して、テーブル制約の識別子を指定します。以下に定義するように、テーブル制約は、プライマリキー制約、外部キー制約、一意性制約のいずれかの形式にできます。

pkey-constraint 句

テーブルのプライマリキーとして、指定したカラムを設定します。プライマリキーは、テーブル内の各ローをユニークに識別します。プライマリキーに含まれるカラムには NULL を使用できません。

fkey-constraint 句

外部キー制約を指定して、1 つまたは複数のカラムの値を、別のテーブルのプライマリキー (または一意性制約) の値と一致するように制限します。

NOT NULL 句

NOT NULL を指定して、外部キーカラムを NULL 入力不可にします。外部キー内の NULL は、外部テーブルのこのローに該当するローがプライマリテーブル内にないことを意味します。複数カラムの外部キー内の少なくとも 1 つの値が NULL である場合、キーの他のカラムに保持できる値に関する制限はありません。

role-name 句

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

role-name を指定して、外部キーに名前を付けます。role-name は、同じテーブル内で外部キーを区別する場合に使用します。また、CONSTRAINT constraint-name を使用して外部キーに名前を付けることができます。ただし、両方のメソッドを使用して外部キーに名前を付けないでください。

REFERENCES 句

REFERENCES 句を指定し、プライマリテーブルで、外部キー制約として使用する 1 つまたは複数のカラムを定義します。REFERENCES カラム制約に column-name を指定する場合、一意性制約またはプライマリキーの制約を受ける、プライマリテーブルのカラム名を指定します。

CHECK ON COMMIT

Ultra Light Java Edition データベースではサポートされていません。データベースサーバで、COMMIT を待ってから外部キー制約を実行するように、CHECK ON COMMIT を指定します。デフォルトでは、挿入、更新、削除の各操作中に、外部キー制約がただちに実行されます。ただし、CHECK ON COMMIT を設定すると、データベースの変更が外部キー制約に違反している場合でも、次の COMMIT 前にデータの不整合が解決されればデータベースの変更を任意の順序で実行できます。

unique-key-constraint 句

一意性制約を指定して、テーブル内の各ローをユニークに識別する 1 つ以上のカラムを指定します。テーブル内の異なるローが、指定されているすべてのカラムで同じ値を持つことはできません。1 つのテーブルに複数の一意性制約が存在することがあります。

WITH MAX HASH SIZE

このインデックスのハッシュサイズをバイト単位で設定します。この値は、データベースに対して有効なデフォルトの MaxHashSize データベースプロパティを上書きします。

sync-constraint 句

テーブルを同期可能にするかどうかおよびすべてのローをアップロードするか、テーブルへの変更だけをアップロードするか、テーブルへの変更をアップロードしないかを決定するための同期制約を指定します。

SYNCHRONIZE ON

デフォルト設定 - テーブルを同期することができ、テーブルへの変更のみがアップロードで送信されます。

SYNCHRONIZE OFF

テーブルを同期できず、テーブルをパブリケーションに含めるときにエラーが発生します。

SYNCHRONIZE ALL

テーブルを同期することができ、テーブル内のすべてのローがアップロードで送信されます。この制約は、Ultra Light Java Edition データベースではサポートされていません。

SYNCHRONIZE DOWNLOAD

テーブルを統合データベースへの変更と同期できますが、ローカルの変更はアップロードされません。

備考

通常、カラム制約を使用するのは、制約がテーブル内で複数のカラムを参照しない場合です。複数のカラムを参照する場合は、テーブル制約を使用します。この文によって制約違反が発生した場合、文の実行は完了しません。エラーの検出前に文が行った変更が完了していない場合は、エラーがレポートされます。

テーブル内の各ローは、ユニークなプライマリキー値を持ちます。

役割名を指定しない場合、役割名は以下のように設定されます。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1. テーブル名と同じ役割名を含む外部キーが存在しない場合、テーブル名が役割名として割り当てられます。
2. テーブル名がすでに使用されている場合、役割名は、0 が埋め込まれた、テーブルに固有の 3 桁の数字と結合されたテーブル名になります。

   スキーマの変更

   データベーススキーマの変更が同時に開始された場合、文は解放されません。

   CREATE TABLE 文の処理中は、そのテーブルを参照する要求やクエリは処理されません。また、データベースにアクティブなクエリやコミットされていないトランザクションがある場合は CREATE TABLE を実行できません。

   Ultra Light.NET の場合: すべてのデータオブジェクト (たとえば ULDataReader) に対して ULBulkCopy.Dispose メソッドも呼び出さないと、この文を実行できません。

   外部 BLOB カラムの同期 (Ultra Light J のみ)

   統合データベースでは、ファイル名カラムは通常の CHAR カラムとして格納され、BLOB ファイルカラムは通常の BLOB (LONG BINARY) カラムとして格納されます。ダウンロード時には、ファイル名カラムは無視され、データベースオプション (Connection.OPTION_BLOB_FILE_BASE_DIR) と、DEFAULT AUTOFILENAME 句で指定したプレフィクスと拡張子の文字列を使用して新しいファイル名が生成されます。J2SE の場合、ファイル名カラムに保存された値は構文 <database_option_blob_file_base_dir><prefix><auto generated integer value>.<extension> に一致します。

   外部 BLOB カラムへのアクセス (Ultra Light J のみ)

   外部 BLOB 値を含むファイルは、クライアントアプリケーションがカラム値を読み取ろうとする場合にのみ開かれます。その時点で、STORE AS FILE 句で指定されたカラムに保存されたファイル名は無効なファイル名になります。Ultra Light でファイル名がプレフィクス "file://" で始まっていないと判断された場合、ファイル名の先頭に OPTION_BLOB_FILE_BASE_DIR オプションの値が付加されてから、そのファイルが開かれます。

   BLOB ファイルは、`file-name-column` ファイル名を指定することによって、データベースに挿入されます。挿入されるファイル名は有効なファイル名である必要があります。挿入時に、STORE AS FILE カラムの値が指定されない場合があります。

   ファイルがデータベースに挿入されると、データベースはファイルに対してフルコントロール権を持ち、外部変更が行われていないと見なします。

関連する動作

オートコミット。

例

次の文は、図書データベース用にテーブルを作成し、図書情報を保持します。

```
CREATE TABLE library_books (
   isbn CHAR(20)       PRIMARY KEY,
   copyright_date      DATE,
   title               CHAR(100),
   author              CHAR(50),
   location            CHAR(50),
   FOREIGN KEY location REFERENCES room
)
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

次の文は、図書データベース用にテーブルを作成して、貸し出し図書の情報を保持します。date_borrowed のデフォルト値は、エントリが作成される日に本が貸し出されることを示します。date_returned カラムは、本が返却されるまでは NULL です。

```
CREATE TABLE borrowed_book (
   loaner_name   CHAR(100) PRIMARY KEY,
   date_borrowed DATE NOT NULL DEFAULT CURRENT DATE,
   date_returned DATE,
   book CHAR(20),
   FOREIGN KEY (book) REFERENCES library_books (isbn)
)
```

次の文は、販売データベース用にテーブルを作成し、注文と注文項目情報を保持します。

```
CREATE TABLE Orders (
   order_num INTEGER NOT NULL PRIMARY KEY,
   date_ordered DATE,
   name CHAR(80)
);
CREATE TABLE Order_item (
   order_num        INTEGER NOT NULL,
   item_num         SMALLINT NOT NULL,
   PRIMARY KEY (order_num, item_num),
   FOREIGN KEY (order_num)
   REFERENCES Orders (order_num)
)
```

関連情報

Ultra Light の識別子 [248 ページ]
SQL データ型 [278 ページ]
Ultra Light の GLOBAL AUTOINCREMENT カラム [73 ページ]
CURRENT TIMESTAMP 特別値 - Ultra Light [253 ページ]
CURRENT UTC TIMESTAMP 特別値 - Ultra Light [254 ページ]
CURRENT DATE 特別値 - Ultra Light [252 ページ]
CURRENT TIME 特別値 - Ultra Light [252 ページ]
SQL データ型 [278 ページ]
分割サイズ [75 ページ]
Ultra Light データベースのスキーマ [51 ページ]
データベースプロパティの読み取り [43 ページ]
Ultra Light global_database_id オプション [205 ページ]
NEWID 関数 [その他] [414 ページ]
Ultra Light の式 [256 ページ]
DROP TABLE 文 [Ultra Light] [507 ページ]
Ultra Light max_hash_size 作成オプション [155 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.13.6.13 CREATE USER 文 [Ultra Light]

データベースユーザまたはグループを作成します。

構文

```
CREATE USER user-name IDENTIFIED BY password
```

パラメータ

user-name

作成するユーザの名前。

password

作成するユーザのパスワード。

備考

プロシージャの定義は SYSPROCEDURE システムビューに表示されるため、この文をプロシージャ内で使用する場合は、文字列リテラルとしてパスワードを指定しないでください。セキュリティ保護のため、プロシージャ定義の外部で宣言される変数を使用してパスワードを指定してください。

- ユーザ ID には、次に該当する値を指定できません。
  - 空白スペース、一重引用符、または二重引用符で始まる値
  - 空白スペースで終わる値
  - セミコロンを含む値
- パスワードは大文字と小文字が区別され、次に該当する値は指定できません。
  - 空白スペース、一重引用符、または二重引用符で始まる値
  - 空白スペースで終わる値
  - セミコロンを含む値
  - 255 バイト長を超える値

関連する動作

なし。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

例

次の例は、ユーザ SQLTester を作成し、パスワードを welcome に設定します。

```
CREATE USER SQLTester IDENTIFIED BY welcome
```

1.13.6.14 DELETE 文 [Ultra Light]

データベースのテーブルからローを削除します。

構文

```
DELETE [ FROM ] table-name[[AS] correlation-name]
[ WHERE search-condition ]
```

パラメータ

correlation-name

文の他の部分からテーブルを参照するときに使用する識別子。

WHERE 句

WHERE 句を指定すると、search-condition を満たすローだけが削除されます。

WHERE 句では、非確定の関数 (RAND など) や変数を使用できません。この句では、カラムも制限されません。カラムをサブクエリで使用する場合、別のテーブルを参照する必要がある場合があります。

備考

Ultra Light では、独自の方法でローのステータスがトレースされます。削除やローのステータスの意味を理解している必要があります。

例

次の文は、従業員テーブルから従業員 105 を削除します。

```
DELETE
FROM Employees
WHERE EmployeeID = 105
```

次の文は、FinancialData テーブルから、2000 年より前のデータをすべて削除します。

```
DELETE
FROM FinancialData
```


© 2016 SAP SE or an SAP affiliate company. All rights reserved.

```
WHERE Year < 2000
```

関連情報

Ultra Light データベースのローのステータス管理 [542 ページ]
Ultra Light の探索条件 [263 ページ]
START SYNCHRONIZATION DELETE 文 [Ultra Light] [520 ページ]
STOP SYNCHRONIZATION DELETE 文 [Ultra Light] [521 ページ]

1.13.6.15 DROP INDEX 文 [Ultra Light]

インデックスを削除します。

構文

```
DROP INDEX [ IF EXISTS ] [ table-name.]index-name
```

備考

テーブルのプライマリインデックスを削除することはできません。

DROP INDEX 文の処理中は、そのインデックスを参照する要求やクエリは処理されません。また、データベースにアクティブなクエリやコミットされていないトランザクションがある場合は DROP INDEX を実行できません。

IF EXISTS 句は、存在しないインデックスを DROP INDEX 文が削除しようとしたときにエラーを返さないようにする場合に使用します。

IF EXISTS 句を指定し、指定のテーブルが見つからない場合、エラーが返されます。

Ultra Light.NET の場合: すべてのデータオブジェクト (たとえば ULDataReader) に対して ULBulkCopy.Dispose メソッドも呼び出さないと、この文を実行できません。

データベーススキーマの変更が同時に開始された場合、文は解放されません。

例

次の文は、FinancialData テーブルで、架空のインデックス fin_codes_idx を削除します。

```
DROP INDEX FinancialData.fin_codes_idx
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

関連情報

Ultra Light インデックス [59 ページ]
Ultra Light データベースのスキーマ [51 ページ]
CREATE INDEX 文 [Ultra Light] [490 ページ]

1.13.6.16 DROP PUBLICATION 文 [Ultra Light]

パブリケーションを削除します。

構文

```
DROP PUBLICATION [ IF EXISTS ]  publication-name, ...
```

備考

IF EXISTS 句は、存在しないパブリケーションを DROP PUBLICATION 文が削除しようとしたときにエラーを返さないようにする場合に使用します。

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
```

関連情報

Ultra Light クライアントの同期設計 [77 ページ]
ALTER PUBLICATION 文 [Ultra Light] [480 ページ]
CREATE PUBLICATION 文 [Ultra Light] [492 ページ]

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

1.13.6.17 DROP SYNCHRONIZATION PROFILE 文 [Ultra Light]

同期プロファイルを削除します。

構文

```
DROP SYNCHRONIZATION PROFILE [ IF EXISTS ] sync-profile-name
```

パラメータ

sync-profile-name

同期プロファイルの名前。

備考

同期プロファイルによって、Ultra Light データベースが Mobile Link サーバと同期する方法を定義します。

IF EXISTS 句は、存在しない同期プロファイルを DROP SYNCHRONIZATION PROFILE 文が削除しようとしたときにエラーを返さないようにする場合に使用します。

関連する動作

なし。

例

次の例は、Test1 という同期プロファイルを削除するための構文を示します。

```
DROP SYNCHRONIZATION PROFILE Test1
```

関連情報

CREATE SYNCHRONIZATION PROFILE 文 [Ultra Light] [493 ページ]
ALTER SYNCHRONIZATION PROFILE 文 [Ultra Light] [481 ページ]
SYNCHRONIZE 文 [Ultra Light] [522 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.13.6.18 DROP TABLE 文 [Ultra Light]

テーブルとそのすべてのデータをデータベースから削除します。

構文

```
DROP TABLE [ IF EXISTS ]  table-name
```

備考

DROP TABLE 文は、データベースから指定したテーブルを削除します。テーブル内にあるすべてのデータ、およびすべてのインデックスとキーも削除されます。

DROP TABLE 文の処理中は、そのテーブルやテーブルのインデックスを参照する要求やクエリは処理されません。また、アクティブなクエリやコミットされていないトランザクションがある場合は DROP TABLE を実行できません。

IF EXISTS 句は、存在しないテーブルを DROP TABLE 文が削除しようとしたときにエラーを返さないようにする場合に使用します。

Ultra Light.NET の場合、すべてのデータオブジェクト (たとえば ULDataReader) に対して ULBulkCopy.Dispose メソッドも呼び出さないと、この文を実行できません。

スキーマの変更が同時に開始された場合、文は解放されません。

例

次の文は、データベースから架空のテーブル EmployeeBenefits を削除します。

```
DROP TABLE EmployeeBenefits
```

関連情報

Ultra Light データベースのスキーマ [51 ページ]
ALTER TABLE 文 [Ultra Light] [483 ページ]
CREATE TABLE 文 [Ultra Light] [495 ページ]

1.13.6.19 DROP USER 文 [Ultra Light]

ユーザを削除します。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

構文

```
DROP USER userid  IDENTIFIED BY password
```

パラメータ

userid

削除するユーザのユーザ ID。

password

ユーザのパスワード。

備考

プロシージャの定義は SYSPROCEDURE システムビューに表示されるため、この文をプロシージャ内で使用する場合は、文字列リテラルとしてパスワードを指定しないでください。セキュリティ保護のため、プロシージャ定義の外部で宣言される変数を使用してパスワードを指定してください。

関連する動作

なし。

例

次の例は、データベースからユーザ SQLTester を削除します。

```
DROP USER SQLTester
```

関連情報

ALTER USER 文 [Ultra Light] [487 ページ]
CREATE USER 文 [Ultra Light] [502 ページ]

 © 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.13.6.20 FROM 句 [Ultra Light]

この句は、SELECT 文に必要なテーブルまたはビューを指定するために使用します。

構文

```
FROM  table-expression, ...
```

```
table-expression :
table-name [ [ AS ] correlation-name ]
| ( select-list ) [ AS ] derived-table-name ( column-name, ... )
| ( table-expression )
| table-expression join-operator table-expression [ ON search-condition ] ...
```

```
join-operator :
,
| INNER JOIN
| CROSS JOIN
| LEFT OUTER JOIN
| JOIN
```

パラメータ

table-name

ベーステーブルまたはテンポラリテーブル。Ultra Light では、異なるユーザがテーブルを所有できません。ユーザ ID でテーブルを修飾した場合、ID は無視されます。

correlation-name

文の他の部分からテーブルを参照するときに使用する識別子。たとえば、次の文で、a は Contacts テーブルの相関名、b は Customers テーブルの相関名として定義されています。

```
SELECT *
FROM Contacts a, Customers b
WHERE a.CustomerID=b.ID
```

derived-table-name

派生テーブルは、FROM 句でネストされている SELECT 文です。

派生テーブルの SELECT リストの項目は、派生テーブル名 (オプション) に続いてピリオド (.) とカラム名を指定して参照します。カラム名があいまいではない場合は、カラム名だけを使用できます。

SELECT 文内から派生テーブルを参照することはできません。

join-operator

ジョインのタイプを指定します。カンマ (,) または CROSS JOIN を指定した場合は、ON サブ句を指定できません。JOIN を指定した場合は、ON サブ句を指定する必要があります。INNER JOIN と LEFT OUTER JOIN の場合は、ON サブ句はオプションです。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

備考

FROM 句を指定しない場合、SELECT 文の式は定数式である必要があります。

> **注記**
>
> この説明はテーブルについてのものですが、特に注意書きがなければ派生テーブルにも適用します。

FROM 句は、指定した全テーブルのすべてのカラムで構成される結果セットを作成します。最初に、指定したテーブルのすべてのローの組み合わせが結果セットの中に入ります。次に、JOIN 条件か WHERE 条件、またはその両方の分だけ、通常は組み合わせの数が減ります。

ジョインのタイプを指定しないで、カンマ区切りリストとしてテーブルをリストする場合、デフォルトでは CROSS JOIN が使用されます。

INNER ジョインの場合、ON 句または WHERE 句を使用してジョインの結果を制限すると、同じ結果が返されます。OUTER ジョインの場合、これらは同じではありません。

> **注記**
>
> Ultra Light では、KEY JOIN と NATURAL JOIN の各ジョインはサポートされていません。

> **例**
>
> 次は、有効な FROM 句です。

```
...
FROM Employees
...
```

```
...
FROM Customers
CROSS JOIN SalesOrders
CROSS JOIN SalesOrderItems
CROSS JOIN Products
...
```

次のクエリは、派生テーブルを使用して、SalesOrders テーブルに 4 件以上の注文がある Customers テーブルの顧客名を返します。

```
SELECT Surname, GivenName, number_of_orders
FROM Customers JOIN
     ( SELECT CustomerID, COUNT(*)
       FROM SalesOrders
       GROUP BY CustomerID )
     AS sales_order_counts( CustomerID, number_of_orders )
ON ( Customers.id = sales_order_counts.CustomerID )
WHERE number_of_orders > 3
```

関連情報

式のサブクエリ - Ultra Light [261 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

DELETE 文 [Ultra Light] [503 ページ]
SELECT 文 [Ultra Light] [516 ページ]
UPDATE 文 [Ultra Light] [526 ページ]

1.13.6.21 INSERT 文 [Ultra Light]

テーブルにローを挿入します。

構文

```
INSERT [ INTO ]
table-name [ ( column-name, ... ) ]
{ VALUES ( expression, ... ) | select-statement }
```

備考

INSERT 文を使用すると、1 つのローを挿入するか、クエリの結果セットから複数のローを挿入できます。

カラムを指定した場合は、指定したカラムに 1 対 1 で値が挿入されます。カラム名のリストを指定しないと、テーブルでの表示順 (SELECT * を使用して取り出すときと同じ順序) に値がテーブルカラムに挿入されます。ローは、任意の順序でテーブルに挿入されます。

テーブルに挿入した文字列は、データベースが大文字と小文字を区別するかどうかに関係なく、常に入力された大文字と小文字の状態のままで格納されます。

例

次の文は、データベースに Eastern Sales 部を追加します。

```
INSERT
INTO Departments ( DepartmentID, DepartmentName )
VALUES ( 230, 'Eastern Sales' )
```

次の文は、otertable の c の値が 10 を超えている場合、othertable から mytable に a と b の値を追加します。

```
INSERT INTO mytable( col1, col2 ) SELECT a, b FROM othertable WHERE c > 10
```

関連情報

SELECT 文 [Ultra Light] [516 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.13.6.22 LOAD TABLE 文 [Ultra Light]

外部ファイルからデータベーステーブルにバルクデータをインポートします。

構文

```
LOAD [ INTO ] TABLE [ owner.]table-name
( column-name, ... )
FROM stringfilename
[ load-option ... ]
```

```
load-option :
CHECK CONSTRAINTS { ON | OFF }
| COMPUTES { ON | OFF}
| DEFAULTS { ON | OFF }
| DELIMITED BY string
| ENCODING encoding
| ESCAPES { ON }
| FORMAT { ASCII | TEXT}
| ORDER { ON | OFF}
| QUOTES { ON | OFF }
| SKIP integer
| STRIP { ON | OFF | BOTH }
| WITH CHECKPOINT { ON | OFF }
```

```
comment-prefix : string
```

```
encoding : string
```

パラメータ

column-name

この句は、データのロード先となる 1 つ以上のカラムを指定するときに使用します。DEFAULTS が OFF に設定されている場合、カラムリストにないカラムは NULL になります。DEFAULTS オプションが ON で、カラムにデフォルト値が入っている場合は、その値が使用されます。DEFAULTS オプションが OFF で、NULL 入力不可のカラムがカラムリストから省かれている場合は、データベースサーバは、空の文字列をカラムの型に変換しようとします。

カラムリストが指定されていると、ファイルに存在すると思われるカラムと、想定されるファイル内でのカラムの順序がカラムリストにリストされます。カラム名を繰り返すことはできません。

FROM string-filename

この句は、データのロード元となるファイルを指定するために使用します。`string-filename` は、文字列としてサーバに渡されます。したがって、文字列は他の SQL 文字列と同じデータベースのフォーマット要件に従います。特に、次の点に注意してください。

- フォルダパスを示すには、円記号 (¥) を 2 つの円記号で表してください。たとえば、`c:¥temp¥input.dat` ファイルから従業員テーブルにデータをロードする `string-filename` は、`c:¥¥temp¥¥input.dat` です。
- パス名はデータベースサーバを基準にした相対パスを指定します。クライアントアプリケーションではありません。
- UNC パス名を使用すると、データベースサーバ以外のコンピュータ上のファイルからデータをロードできます。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

load-option 句

さまざまなロードオプションを使用して、データのロード方法を制御できます。次のリストは、サポートされているロードオプションを示します。

CHECK CONSTRAINTS 句

この句は、ロード中に制約をチェックするかどうかを制御します。CHECK CONSTRAINTS はデフォルトで ON に設定されますが、アンロードユーティリティ (ulunload) は CHECK CONSTRAINTS を OFF に設定して LOAD TABLE 文を書き出します。CHECK CONSTRAINTS を OFF に設定すると、データベースの再構築などの場合に役立つ検査制約が無効になります。

COMPUTES 句

このオプションは処理されますが、Ultra Light では無視されます。

DEFAULTS 句

デフォルトでは、DEFAULTS は OFF に設定されています。DEFAULTS が OFF の場合は、カラムリストにないカラムすべてに NULL が割り当てられます。DEFAULTS が OFF に設定されており、NULL 入力不可のカラムがカラムリストから省かれている場合は、データベースサーバは、空の文字列をカラムの型に変換しようとします。DEFAULTS が ON に設定されており、カラムにデフォルト値が入っている場合は、その値が使用されます。

DELIMITED BY 句

この句は、カラムデリミタ文字を指定するときに使用します。デフォルトのカラムデリミタ文字列はカンマです。ただし、最長で 255 バイトの文字列を指定できます。たとえば、`... DELIMITED BY '###' ...` などです。他の SQL 文字列と同じフォーマット要件が適用されます。タブで区切った値を指定する場合、タブ文字を表す 16 進のエスケープシーケンス (9) を使用して、`... DELIMITED BY '¥x09' ...` のように指定します。

ENCODING 句

この句は、データベースにロードするデータに使用する文字エンコードを指定します。Ultra Light は文字セットの変換を実行しません。データファイルのエンコードはデータベースと一致している必要があります。

ESCAPES 句

ESCAPES は常に ON であるため、データベースサーバによって円記号に続く文字が認識され、特殊文字として解釈されます。改行文字は ¥n という組み合わせとしてインクルードされ、他の文字はタブ文字の ¥x09 のような 16 進の ASCII のコードとしてデータにインクルードされます。2 つの円記号 (¥) は 1 つの円記号として解釈されます。円記号 (¥) の後に n、x、X、¥ 以外の文字がある場合、それらは別々の文字と解釈されます。たとえば、¥q であれば、円記号と q が挿入されます。

FORMAT 句

この句は、データのロード元となるデータソースのフォーマットを指定します。TEXT の場合、(ENCODING オプションで定義したとおり) 入力行は文字であり、1 行あたり 1 つのローで構成され、カラムデリミタ文字列によって値が区切られているものとみなされます。ASCII もサポートされています。

QUOTES 句

この句は、文字列を引用符で囲むかどうかを指定します。Ultra Light は ON のみをサポートしているため、LOAD TABLE 文は引用符文字で囲まれた文字列を探します。引用符文字はアポストロフィ (一重引用符) です。文字列の中で最初に出てくるこのような文字は、その文字列の引用符文字として扱われます。文字列の終わりには先頭にあるものと同じ引用符が必要です。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

カラムデリミタ文字列をカラム値の中に入れることができます。また、引用符文字は値の一部とは見なされません。したがって、次の行はアドレスにカンマがあるかどうかに関係なく、3 つではなく 2 つの値として扱われます。また、アドレスを囲む引用符は、データベースへは挿入されません。

```
'123 High Street, Anytown',(715)398-2354
```

値の中に引用符文字をインクルードするには、2 つの引用符を使用する必要があります。次の行は、第 3 カラムの中へ一重引用符文字である値を入れます。

```
'123 High Street, Anytown','(715)398-2354',''''
```

SKIP 句

この句は、ファイルの最初の数行を無視するかどうかを指定するときに使用します。`integer` 引数では、スキップする行数を指定します。たとえば、この句を使用して、カラム見出しを含む行をスキップできます。

STRIP 句

この句は処理されますが、無視されます。この句は、引用符がない値に、値を挿入する前に削除された先行ブランクまたは後続ブランクがあるかどうかを指定します。STRIP オプションは次のオプションを受け入れます。

STRIP ON

先行ブランクを削除します。

STRIP OFF

先行ブランクも後続ブランクも削除しません。

STRIP BOTH

先行ブランクと後続ブランクの両方を削除します。

WITH CHECKPOINT 句

この句は、チェックポイントを実行するかどうかを指定するときに使用します。デフォルト設定は OFF です。この句を ON に設定すると、文が正常に完了した後、チェックポイントが発行されます。

備考

またこの文は、SQL Anywhere の dbunload ユーティリティ (reload.sql ファイル) の出力処理もサポートします。LOAD TABLE は、Windows 上で DBISQL を使用してのみ使用できます。

Ultra Light データベースをアンロードまたは再ロードするための推奨メソッドは、ulunload および ulload ユーティリティです。ulinit ユーティリティはスキーマとデータを直接 SQL Anywhere データベースからロードできることにも注意してください。

LOAD TABLE を使うと、ファイルからデータベーステーブルの中へ効率よく大量の挿入を行えます。この文は主に、SQL Anywhere の dbunload ユーティリティ (reload.sql ファイル) の出力処理をサポートする手段です。

FORMAT TEXT の場合、値を指定しないことが、NULL 値を指定することになります。たとえば、`1,,'Fred',` という値を含むファイルに 3 つの値が予想される場合、挿入される値は 1、NULL、Fred です。ファイルに `1,2,` が含まれる場合、値 1、2、NULL が挿入されます。空白のみで構成される値も、NULL 値と見なされます。ファイルに `1, ,'Fred',` が含まれる場合、値 1、NULL、Fred が挿入されます。他の値はすべて NULL 以外と見なされます。たとえば、'' (一重引用符が 2 つ続く) は空の文字列です。'NULL' は 4 文字の文字列です。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

LOAD TABLE でロードされるカラムが NULL 値を許容しておらず、ファイル値が NULL の場合、数値カラムには値 0 (ゼロ)、文字カラムには空の文字列 ('') が指定されます。LOAD TABLE でロードされたカラムが NULL 値を許容し、ファイル値が NULL の場合、カラム値は (すべての型で) NULL になります。

テーブルに a、b、c の各列が含まれ、入力データに a、b、c が含まれているときに、LOAD 文がデータのロード先として a と b のみを指定した場合は、カラム c に次の値が挿入されます。

- DEFAULTS ON が指定されると、カラム c にデフォルト値が入っている場合は、デフォルト値が使用されます。
- カラム c にデフォルト値がなく、NULL が許容されている場合は、NULL が使用されます。
- カラム c にデフォルト値がなく、NULL が許容されていない場合は、カラムのデータ型に応じて、ゼロ (0) または空の文字列 ('') が使用されるかエラーが返されます。

関連する動作

オートコミット。

例

次に、LOAD TABLE の例を示します。まずテーブルを作成し、次に `input.txt` というファイルを使用してテーブルにデータをロードします。

```
CREATE TABLE t( a CHAR(100) primary key, let_me_default INT DEFAULT 1, c
CHAR(100) )
```

次に、ファイル `input.txt` の内容を示します。

```
'this_is_for_column_c', 'this_is_for_column_a', ignore_me
```

次の LOAD 文は、`input.txt` というファイルをロードします。

```
LOAD TABLE T ( c, a ) FROM 'input.txt' FORMAT TEXT DEFAULTS ON
```

コマンド `SELECT * FROM t` は、次の結果セットを返します。

a	let_me_default	c
this_is_for_column_a	1	this_is_for_column_c

関連情報

INSERT 文 [Ultra Light] [511 ページ]
Ultra Light データベースのアンロードユーティリティ (ulunload) [236 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.13.6.23 ROLLBACK 文 [Ultra Light]

トランザクションを終了し、COMMIT 文または ROLLBACK 文が最後に実行されて以降の変更を元に戻します。

構文

```
ROLLBACK [ WORK ]
```

備考

Ultra Light SQL を使用するとトランザクションが作成されます。トランザクションは、最後の ROLLBACK または COMMIT 後のすべての変更内容 (INSERT、UPDATE、DELETE) から構成されます。ROLLBACK 文は、現在のトランザクションを終了し、前回の COMMIT または ROLLBACK 以降にデータベースに対して行われたすべての変更を元に戻します。

例

次の文は、前回コミット時の状態にデータベースをロールバックします。

```
ROLLBACK
```

関連情報

COMMIT 文 [Ultra Light] [489 ページ]

1.13.6.24 SELECT 文 [Ultra Light]

データベースから情報を取得します。

構文

```
SELECT [ DISTINCT ] [ row-limitation ]
select-list
[ FROM table-expression, ... ]
[ WHERE search-condition ]
[ GROUP BY group-by-expression, ... ]
[ ORDER BY order-by-expression, ... ]
[ FOR { UPDATE | READ ONLY } ]
[ OPTION ( FORCE ORDER ) ]
```

```
row-limitation :
FIRST
| TOP n [ START AT m ]
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

```
select-list :
expression [ [ AS ] alias-name ], ...
```

```
order-by-expression :
{ integer | expression } [ ASC | DESC ]
```

パラメータ

DISTINCT 句

DISTINCT を指定して、重複するローを結果から削除します。DISTINCT を指定しない場合は、SELECT 文の句を満たすすべてのローを返します。ここには重複するローも含まれます。多くの場合、DISTINCT を指定すると、文の実行時間が非常に長くなります。したがって、DISTINCT を使用するのは、必要な場合だけにしてください。

row-limitation 句

ロー制限を使用して、結果のサブセットを返します。たとえば、結果セットの最初のローを取得するには、FIRST を指定します。結果の最初の n 個のローを返すには、TOPn を使用します。TOPn 個のローを取得するときの開始ローの位置を制御するには、START ATm を指定します。これらの句が意味のある結果を返すようにローの順序を指定するには、SELECT 文に ORDER BY 句を指定します。

select-list

式のリストであり、データベースから何を取り出すかを指定します。通常、SELECT リスト内の式はカラム名です。ただし、関数のように、他のタイプの式にすることもできます。アスタリスク (*) を使用すると、FROM 句にリストされた全テーブルのすべてのカラムを選択できます。オプションとして、`select-list` 内のそれぞれの式に対して、エイリアスを定義できます。エイリアスを使用すると、クエリ内の別の位置 (WHERE 句や ORDER BY 句など) から、`select-list` 式を参照できるようになります。

FROM 句

`table-expression` の中で指定されるテーブルとビューからローを取り出します。

WHERE 句

WHERE 句を指定すると、`search-condition` を満たすローだけが選択されます。

GROUP BY 句

GROUP BY 句を使用したクエリの結果には、GROUP BY 式の中の個別の値の各セットに対し 1 つのローが入ります。テーブルリストのローの各グループに対する結果にはローが 1 つずつ含まれるため、結果ローはグループとして頻繁に参照されます。これらのグループ内のローには、集合関数を適用できます。NULL があった場合、ユニークな値と見なされます。

ORDER BY 句

この句で指定した式に従って、クエリの結果がソートされます。ORDER BY 句の各式は、昇順 (ASC) または降順 (DESC) (デフォルト) でソートすることができます。式が整数 n である場合、クエリ結果は SELECT リストの n 番目の式でソートされます。

特定の順序でローが返されるようにする唯一の方法は ORDER BY を使用することです。ORDER BY 句がない場合は、Ultra Light が最も効率のよい順序でローを返します。

Ultra Light は、LONG VARCHAR 値や LONG BINARY 値の順序をサポートしていません。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

FOR 句

この句には、クエリの動作を制御する 2 つの形式があります。

FOR READ ONLY

この句は、クエリが更新に使用されないことを示します。クエリが更新に使用されないことがわかっている場合は、Ultra Light のパフォーマンスが向上することがあるため、この句はできるだけ指定してください。たとえば、読み込み専用アクセスが必要であることがわかると、Ultra Light はダイレクトテーブルスキャンを実行できます。FOR READ ONLY はデフォルトの動作です。

FOR UPDATE

この句によってクエリを更新に使用できるようになります。この句は、明示的に指定する必要があります。明示的に指定しないと、更新は許可されません (FOR READ ONLY がデフォルトの動作)。

OPTION (FORCE ORDER) 句

この句は一般的には使用しないことをおすすめします。この句は、Ultra Light で選択されたテーブルアクセス順序を上書きし、クエリに出現する順序でテーブルにアクセスするよう Ultra Light に要求します。この句は、クエリの順序が Ultra Light の順序よりも確実に効率的である場合にのみ使用してください。

Ultra Light では、実行プランを使用してクエリを最適化することもできます。

備考

クエリは必ず閉じてください。そうしないと、メモリが解放されず、存在し続けるテンポラリテーブルの数が不必要に増加することになります。

例

次の文は、従業員テーブルから従業員の数を選択します。

```
SELECT COUNT(*)
FROM Employees
```

次の文は、従業員テーブルで 40 番目から 49 番目までの 10 個のローを選択します。

```
SELECT TOP 10 START AT 40 * FROM Employees
```

関連情報

Ultra Light のパフォーマンスに関するヒント [527 ページ]
Ultra Light の実行プラン [534 ページ]
ダイレクトページスキャン [537 ページ]
Ultra Light の探索条件 [263 ページ]
FROM 句 [Ultra Light] [509 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.13.6.25 SET OPTION 文 [Ultra Light]

データベースオプションの値を変更します。

構文

```
SET OPTION option-name=[option-value]
```

```
option-name: identifier
```

```
option-value: string, identifier, or number
```

備考

この文を使用して、Ultra Light データベース上でオプションを設定できます。ほとんどの Ultra Light オプションはデータベースが最初に作成されたときに設定され、後で変更することはできません。

オプションが永続的かどうかの指定はできません。Ultra Light は永続的なオプションか一時的なオプションかを判別します。永続的なオプションはデータベースに格納されます。一時的なオプションが使用されるのは、接続またはデータベースの実行が停止するまでです。

Ultra Light は、global_database_id や ml_remote_id といった永続的なオプションが設定されているとコミット操作を実行します。Ultra Light は一時的なオプションや接続ベースのオプションでは、コミット操作を実行しません。

設定解除が可能なデータベースオプションは ml_remote_id だけです。次に例を示します。

```
SET OPTION ml_remote_id=
```

この結果、ID が NULL に設定されます。この場合、Ultra Light は次の同期時に自動的に新しい値を生成します。

例

次の文を実行すると、global_database_id オプションが 100 に設定されます。

```
SET OPTION global_database_id=100
```

関連情報

Ultra Light データベースオプション [201 ページ]
Ultra Light global_database_id オプション [205 ページ]
Ultra Light ml_remote_id オプション [206 ページ]
DB_PROPERTY 関数 [システム] [368 ページ]

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

1.13.6.26 START SYNCHRONIZATION DELETE 文 [Ultra Light]

Mobile Link 同期で削除されたローのロギングを再起動します。

構文

START SYNCHRONIZATION DELETE

備考

Ultra Light データベースでは、同期が必要なローへの変更のログを自動的に取ります。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。この文を使用すると、その前に STOP SYNCHRONIZATION DELETE 文で停止されていた、削除されたローの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。

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。

Ultra Light では、独自の方法でローのステータスがトレースされます。削除やローのステータスの意味を理解している必要があります。

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。

```
STOP SYNCHRONIZATION DELETE;
DELETE FROM PROPOSAL
  WHERE last_modified < months( CURRENT TIMESTAMP, -1 );
START SYNCHRONIZATION DELETE;
COMMIT;
```

関連情報

Ultra Light データベースのローのステータス管理 [542 ページ]
STOP SYNCHRONIZATION DELETE 文 [Ultra Light] [521 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.13.6.27 STOP SYNCHRONIZATION DELETE 文 [Ultra Light]

Mobile Link 同期で削除されたローのロギングを停止します。

構文

STOP SYNCHRONIZATION DELETE

備考

Ultra Light データベースでは、同期が必要なローへの変更のログを自動的に取ります。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。この文を使用すると、その前に START SYNCHRONIZATION DELETE 文で起動されていた、削除されたローのロギングを停止できます。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。削除のロギングを停止するのは、一時的のみにしてください。

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。

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。

Ultra Light では、独自の方法でローのステータスがトレースされます。削除やローのステータスの意味を理解している必要があります。

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。

```
STOP SYNCHRONIZATION DELETE;
DELETE FROM PROPOSAL
WHERE last_modified < months( CURRENT TIMESTAMP, -1 );
START SYNCHRONIZATION DELETE;
COMMIT;
```

関連情報

Ultra Light データベースのローのステータス管理 [542 ページ]
START SYNCHRONIZATION DELETE 文 [Ultra Light] [520 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.13.6.28 SYNCHRONIZE 文 [Ultra Light]

Mobile Link サーバを通じて Ultra Light データベースを同期させます。

構文

```
SYNCHRONIZE {
PROFILE sync-profile-name [ MERGE sync-option [ ;... ] ]
| USING sync-option  [ ;... ]
}
```

```
sync-option :
sync-option-name = sync-option-value
```

```
sync-option-name : string
```

```
sync-option-value : string
```

パラメータ

sync-profile-name

同期プロファイルの名前。

MERGE 句

この句は、同期プロファイルに指定されているオプションを追加または上書きするために使用します。

USING 句

この句は、同期プロファイルを参照しないで同期オプションを指定するために使用します。

sync-option

1 つ以上の同期オプションの値ペアがセミコロンで区切られた文字列。たとえば、`'option1=value1;option2=value2'` のように記述します。

sync-option-name

同期オプションの名前。

sync-option-value

同期オプションの値。

備考

同期は、同期プロファイルのパラメータに応じて設定します。または、文自体の中でパラメータを指定できます。

同期のオプションをマージできると、開発者は、(MobiLinkPwd などのように) 一部のオプションをデータベースに格納する作業を省略できます。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

同期コールバック関数が定義され、Ultra Light に登録されている場合は、SYNCHRONIZE 文を実行するたびに、その同期の進行状況情報がコールバック関数に渡されます。コールバック関数が登録されていない場合、進行状況情報は渡されません。

関連する動作

なし。

例

次の例は、Test1 という同期プロファイルを削除するための構文を示します。ここでは、MobiLinkPwd がプロファイルの一部として格納されていません。

```
SYNCHRONIZE PROFILE Test1 MERGE 'MobiLinkPwd=sql'
```

次の例は、Test1 という同期プロファイルに、publication オプションと uploadonly オプションを追加するための構文を示します。

```
SYNCHRONIZE PROFILE Test1
   MERGE 'publications=p2;uploadonly=on'
```

次の例は、USING の使用方法を示します。

```
SYNCHRONIZE USING
'MobiLinkUid=joe;MobiLinkPwd=sql;ScriptVersion=1;Stream=TCPIP{host=localhost}'
```

次の例は、publication オプションと uploadonly オプションを同期するための構文を示します。

```
SYNCHRONIZE
   USING 'publications=p2;uploadonly=on'
```

関連情報

CREATE SYNCHRONIZATION PROFILE 文 [Ultra Light] [493 ページ]
ALTER SYNCHRONIZATION PROFILE 文 [Ultra Light] [481 ページ]
DROP SYNCHRONIZATION PROFILE 文 [Ultra Light] [506 ページ]

1.13.6.29 TRUNCATE TABLE 文 [Ultra Light]

テーブルを削除しないで、そのテーブルのすべてのローを削除します。

構文

```
TRUNCATE TABLE table-name
```


© 2016 SAP SE or an SAP affiliate company. All rights reserved.

備考

TRUNCATE TABLE 文によってテーブル内のローがすべて削除され、この後の同期時に Mobile Link サーバにこの削除について通知されません。これは、次の文と同義です。

```
STOP SYNCHRONIZATION DELETE;
DELETE FROM TABLE;
START SYNCHRONIZATION DELETE;
```

注記

この文は、同期またはレプリケーション対象のデータベースに対して慎重に使用する必要があります。Mobile Link サーバに通知されないので、この削除によって整合性が失われ、その結果、同期またはレプリケーションに失敗する可能性があります。

TRUNCATE TABLE 文の実行後、テーブル構造体、すべてのインデックス、制約およびカラム定義は存在し続けます。削除されるのはデータのみです。

テーブルを対象とした文が、別の要求やクエリですでに参照されている場合、TRUNCATE TABLE は実行できません。同様に、テーブルの変更中は、そのテーブルを参照する要求は処理されません。また、データベースにアクティブなクエリやコミットされていないトランザクションがある場合は TRUNCATE TABLE を実行できません。

Ultra Light.NET の場合: すべてのデータオブジェクト (たとえば ULDataReader) に対して ULBulkCopy.Dispose メソッドも呼び出さないと、この文を実行できません。

スキーマの変更

スキーマの変更が同時に開始された場合、文は解放されません。

関連する動作

テーブルに DEFAULT AUTOINCREMENT または DEFAULT GLOBAL AUTOINCREMENT と定義されたカラムがある場合、TRUNCATE TABLE はそのカラムの次に使用可能な値をリセットします。

TRUNCATE TABLE でローが削除済みとマーク付けされると、この処理を実行したユーザは、ROLLBACK 文を発行しないかぎり、ローにアクセスできなくなります。ただし、他の接続からはアクセスできます。COMMIT を使用すると削除が確定し、すべての接続からデータにアクセスできなくなります。

トランケート対象のテーブルを同期すると、テーブルに適用されているすべての INSERT 文が、TRUNCATE TABLE 文より優先されます。

例

次の文は Departments テーブルからすべてのローを削除します。

```
TRUNCATE TABLE Departments
```

この例を実行する場合は、ROLLBACK 文を実行して、変更内容を元に戻すようにしてください。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

関連情報

Ultra Light データベースのスキーマ [51 ページ]
DELETE 文 [Ultra Light] [503 ページ]
START SYNCHRONIZATION DELETE 文 [Ultra Light] [520 ページ]
STOP SYNCHRONIZATION DELETE 文 [Ultra Light] [521 ページ]

1.13.6.30 UNION 文 [Ultra Light]

2 つ以上の SELECT 文の結果を結合します。

構文

```
select-statement-without-ordering
[ UNION [ ALL | DISTINCT ] select-statement-without-ordering ]...
[ ORDER BY [ number [ ASC | DESC ] , ... ]
```

備考

いくつかの SELECT 文の結果を、UNION を使ってより大きな結果へと結合することができます。各 SELECT 文には、それぞれの SELECT リストに同じ数の式を指定します。各文には ORDER BY 句を含めることはできません。

UNION ALL の結果は、統合された SELECT 文の結果を結合したものです。ローが重複しないようにして結果を取得するには、UNION または UNION DISTINCT を指定します。ただし、重複するローを削除すると、文の合計実行時間が増えます。ローの重複を許容するには、UNION ALL を指定します。

データ型の異なる対応する式を結合しようとすると、Ultra Light は、結合された値を表すデータ型を検索しようとします。この検索ができない場合、統合操作は失敗し、エラーが返されます (例: "値 'Surname' をデータ型数値に変換できません")。

結果に表示されるカラム名は、最初の SELECT 文に対して表示されるカラム名 (またはエイリアス) です。

UNION の ORDER BY は整数フォーマットに制限されています。ORDER BY 句では、整数を使用して順序が確立されます。ここで整数は結果をソートするクエリ式を示します。

例

次の例は、結合された従業員テーブルと顧客テーブルで見つかったそれぞれの姓すべてをリストします。

```
SELECT Surname FROM Employees
UNION
SELECT Surname FROM Customers
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

関連情報

SELECT 文 [Ultra Light] [516 ページ]

1.13.6.31 UPDATE 文 [Ultra Light]

テーブルのローを変更します。

構文

```
UPDATE table-name[[AS] correlation-name]
SET column-name  = expression, ...
[ WHERE search-condition ]
```

パラメータ

table-name

table-name は、更新するテーブルの名前を指定します。使用できるのは単一テーブルのみです。

correlation-name

文の他の部分からテーブルを参照するときに使用する識別子。

SET 句

指定した各カラムに、等号の右側の式の値を設定します。式には制限がありません。式が column-name である場合は、古い値が使われます。

SET 句で指定されたカラムの値のみ変更されます。特に、UPDATE を使用して、カラムの値をデフォルトに設定することはできません。

WHERE 句

WHERE 句を指定すると、search-condition を満たすローだけが更新されます。

備考

UPDATE 文は、テーブル内の値を修正します。

テーブルに挿入した文字列は、データベースが大文字と小文字を区別するかどうかに関係なく、常に入力された大文字と小文字の状態のままで格納されます。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

例

次の文は、従業員 Philip Chin (従業員 129) を販売部からマーケティング部 (部門 400) に移動します。

```
UPDATE Employees
SET DepartmentID = 400
WHERE EmployeeID = 129
```

correlation-name を使用した例は次のとおりです。

```
UPDATE Employee E
SET salary = salary * 1.05
WHERE EXISTS( SELECT 1 FROM Sales S HAVING E.Sales > Avg( S.sales)
GROUP BY S.dept_no )
```

関連情報

INSERT 文 [Ultra Light] [511 ページ]
DELETE 文 [Ultra Light] [503 ページ]
Ultra Light の探索条件 [263 ページ]

1.14 Ultra Light のパフォーマンスに関するヒント

Ultra Light データベースのパフォーマンス向上に役立ついくつかのトピックについて説明します。

このセクションの内容:

Ultra Light データベースのキャッシュサイズの調整 [528 ページ]
キャッシュサイズを調整する必要はありませんが、モバイルデバイス上のオペレーティングシステムによって、Ultra Light データベースアプリケーションのメモリ使用量を減らすように要求された場合は、キャッシュサイズを調整してください。

クエリのパフォーマンスに関するヒント [528 ページ]
Ultra Light データベースのクエリのパフォーマンス向上に役立ついくつかのトピックについて説明します。

挿入と更新のパフォーマンスのヒント [540 ページ]
Ultra Light データベースの挿入と更新のパフォーマンス向上に役立ついくつかのトピックについて説明します。

Ultra Light のベンチマークに関するヒント [545 ページ]
ベンチマークテストは、一般的に、プリケーション開発サイクルの運用段階に入る前に行われます。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

1.14.1 Ultra Light データベースのキャッシュサイズの調整

キャッシュサイズを調整する必要はありませんが、モバイルデバイス上のオペレーティングシステムによって、Ultra Light データベースアプリケーションのメモリ使用量を減らすように要求された場合は、キャッシュサイズを調整してください。

Ultra Light データベースのキャッシュサイズは、使用可能なデバイスのメモリが指定したパラメータの範囲内であれば、データ操作に応じて動的に増加されます。通常は、パラメータを指定する必要はありません。データベースが大きい (400 MB など) 場合は、CACHE_MAX_SIZE パラメータを指定して、最大許容サイズをデフォルト値よりも大きくすることができます。Ultra Light は、最大キャッシュサイズに基づいて、いくつかのデータ構造を割り当てます。そのため、デフォルト値はさほど大きく設定されていません。メモリのオーバーヘッドを追加するには、より大きな最大値を明示的に要求する必要があります。最大キャッシュサイズに、実際のデータベースファイルの最大サイズより大きい値を指定しても、利点はありません。

Ultra Light のキャッシュサイズは自動的には縮小されません。データベースのキャッシュサイズを制御するには、アプリケーションの cache_allocation データベースオプションを使用する必要があります。オペレーティングシステムによって発行されたメモリ容量低下イベントへの対応として、データベースへの接続後、cache_allocation データベースオプションを調整します。

例

次の Ultra Light C++ コードサンプルは、接続文字列の更新によって最大キャッシュサイズを 100 MB に設定する方法を示しています。

```
static ul_char const * ConnectionParms =
    "UID=DBA;PWD=sql;DBF=sample.udb;CACHE_MAX_SIZE=100m";
```

次の Ultra Light C++ コードサンプルは、キャッシュ割り当てを半分に減らしてキャッシュサイズを変更する方法を示しています。

```
ULConnection * conn = ULDatabaseManager::OpenConnection(ConnectionParms);
ul_u_long percent;
percent = conn->GetDatabasePropertyInt( "cache_allocation" );
conn->SetDatabaseOptionInt( "cache_allocation", percent / 2 );
```

関連情報

Ultra Light cache_allocation オプション [202 ページ]
Ultra Light CACHE_SIZE 接続パラメータ [181 ページ]
Ultra Light CACHE_MIN_SIZE 接続パラメータ [180 ページ]
Ultra Light CACHE_MAX_SIZE 接続パラメータ [179 ページ]

1.14.2 クエリのパフォーマンスに関するヒント

Ultra Light データベースのクエリのパフォーマンス向上に役立ついくつかのトピックについて説明します。

このセクションの内容:

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

インデックススキャンの作成と保持 [529 ページ]
1 つ以上のインデックスを作成することで、クエリのパフォーマンスを向上させたり、ローの一意性を確保したりできます。

インデックスのハッシュ [530 ページ]
ハッシュの上限として特定のサイズを選択することでクエリのパフォーマンスをチューニングできます。

最適なハッシュサイズ制限 [531 ページ]
Ultra Light のデフォルトの最大ハッシュサイズである 4 バイトは、ほとんどの場合に適しています。

Ultra Light の実行プラン [534 ページ]
Ultra Light の実行プランは、クエリ実行時のテーブルとインデックスへのアクセス方法を示します。

1.14.2.1 インデックススキャンの作成と保持

1 つ以上のインデックスを作成することで、クエリのパフォーマンスを向上させたり、ローの一意性を確保したりできます。

インデックスは、一部またはすべてのカラムの値を基に、テーブルのローに順序を付けます。インデックスを作成する場合、インデックス付けするカラムを指定する順序が、インデックスでカラムが実際に出現する順序になります。インデックスを戦略的に使用すると、インデック付けしたカラムの検索パフォーマンスを大幅に向上できます。

クエリのパフォーマンスを向上するには、次の推奨する手順に従ってください。

- 次のようなカラムにインデックスを作成してください。
  - 定期的に検索する値
  - クエリでテーブルのジョインに使用するカラム
  - ORDER BY 句、GROUP BY 句、または WHERE 句で頻繁に使用するカラム
- 複合インデックスを作成し、インデックスの最初のカラムが、クエリの作成時に述部で最も頻繁に使用されるようにします。
- インデックスによって生じる更新管理オーバヘッドが、デバイスのメモリに負荷をかけすぎていないか確認します。
- 不必要に、インデックスを作成したり保持したりしないでください。インデックスは、カラムのデータが変更されたときに更新する必要があります。したがって、挿入、更新、削除の各操作はインデックスにも実行されます。
- 大規模なテーブルでインデックスを作成してください。
- 冗長なインデックスは作成しないでください。たとえば、カラム (x、y) に対するインデックスをテーブル T に作成した場合に、カラム (x、y、z) に対する別の既存インデックスが T にあると、冗長的になります。

関連情報

Ultra Light インデックス [59 ページ]
テンポラリテーブルの管理 [536 ページ]
ダイレクトページスキャン [537 ページ]
実行プランの表示 [538 ページ]
EXPLANATION 関数 [その他] [373 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.14.2.2 インデックスのハッシュ

ハッシュの上限として特定のサイズを選択することでクエリのパフォーマンスをチューニングできます。

ハッシュキーは、インデックス付けされたカラムの実際の値を表します。インデックスのハッシュの目的は、インデックスされた値を特定するためのローの検索、ロード、アンパックという負荷の高い処理を避けることです。必要なだけの実際のローデータを 1 つのロー ID に含めることで、これらの処理を回避します。

ロー ID を使用することで、Ultra Light がデータベースファイル内の実際のローデータを見つけることができます。ハッシュサイズを 0 に設定する (インデックスのハッシュを無効にする) と、インデックスエントリにはこのロー ID だけが含まれます。ハッシュサイズを 0 以外の値に設定すると、ハッシュキーも使用されます。ハッシュキーには、そのローの変換されたデータすべてまたはその一部を含めることが可能であり、ロー ID とともにインデックスのページに格納されます。

ハッシュキーに含まれるローデータの量は、次の要素によって決まります。

- 設定した最大ハッシュサイズプロパティ。
- カラムのデータ型に実際に必要とされるデータサイズ。

ハッシュの例

インデックスのハッシュの値は、インデックス付けされたカラムの実際のローデータの順序を保持しています。たとえば、Employees テーブルの LastName カラムにインデックスを作成した場合、4 つの名前が次の順序で表示されます。

- Anders
- Anderseck
- Andersen
- Anderson

最初の 6 文字をハッシュした場合、これらのロー値のハッシュキーは次のようになります。

- Anders
- Anders
- Anders
- Anders

これらのエントリは同じに見えますが、リストの最初の **Anders** は実際のロー値 を表すために使用されます。リストの最後の **Anderson** は実際のロー値 を表すために使用されます。

次の文を考えてみます。

```
SELECT *
FROM Employees
WHERE LastName = 'Andersen'
```

Employees テーブルに Andersen に似た名前ばかりが数多く含まれている場合、ハッシュキーではパフォーマンスが向上するほどの一意性がありません。この場合、Ultra Light はこの文の条件に実際に一致するハッシュキーを決定できません。インデックスのハッシュキーの重複があると、Ultra Light が次の処理を行う必要が生じます。

1. 目的のロー ID に一致するテーブルローを探す。
2. データをロードしてアンパックし、値を評価します。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

パフォーマンスが向上するのは、Ultra Light が適正な数のユニークなハッシュを認識可能であり、クエリ条件の評価がそのままインデックス自体につながる場合のみです。たとえば、Employees テーブルに何千という名前があっても、6 文字によるハッシュで十分なパフォーマンス向上が得られます。ただし、Employees テーブルに Anders* で始まる名前が突出して含まれている場合は、少なくとも 7 文字でハッシュして、ユニークなキーの程度を向上させる必要があります。これにより、この例の冒頭に挙げた 4 つの名前は、次のハッシュキーで表されるようになります。

- Anders
- Anderse
- Anderse
- Anderso

この例では、4 つローの値のうち、アンパックして評価する必要があるのは、4 つではなく 2 つになります。

関連情報

最適なハッシュサイズ制限 [531 ページ]
Ultra Light インデックスの追加 [63 ページ]
Ultra Light max_hash_size 作成オプション [155 ページ]

1.14.2.3 最適なハッシュサイズ制限

Ultra Light のデフォルトの最大ハッシュサイズである 4 バイトは、ほとんどの場合に適しています。

値を大きくすると、ロー ID とともに含めるデータを増やすことができます。ただしそうすると、インデックスのサイズが大きくなり、複数ページに断片化される場合があります。この変更によって、結果的にデータベースのサイズが大きくなる可能性があります。最大ハッシュサイズが大きくなることによる影響は、テーブルに含まれるローの数によって異なります。たとえば、ローの数が少なければ、インデックスのハッシュキーは、大きくてもインデックスページに収まる可能性があります。その場合、インデックスの断片化は起こりません。

最適なハッシュサイズを選択するには、データ型、ローのデータ、データベースのサイズ (特にテーブルに含まれるローの数が多い場合) を考慮してください。

最適なハッシュサイズを選択したかどうかを確認する唯一の方法は、ターゲットデバイスで Ultra Light クライアントアプリケーションに対してベンチマークテストを実行することです。データベースサイズ自体の変化に加え、さまざまなハッシュサイズがアプリケーションとクエリのパフォーマンスに及ぼす影響についても確認する必要があります。

インデックス化されるカラムに、共通のプレフィクスがない文字列など、適切な値の分散がある場合は、インデックス構造が大きくなっても、インデックスのハッシュによって挿入、更新、削除、検索のパフォーマンスが改善されます。ハッシュされたインデックスは、ハッシュを使用して最初にローを特定してから、ダイレクトロー比較で同じハッシュ値のローを区別します。ハッシュサイズが十分に大きければ、ハッシュをユニークに識別できるので、ローの読み出しと比較は必要ありません。ただし、ページサイズに比べてハッシュサイズが大きすぎると、インデックスに多数のデータベースページが必要になる可能性があります。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

データ型

カラム内の値全体をハッシュする場合は、次の表で、データ型ごとに必要なサイズを確認してください。Ultra Light は、必要な場合にのみ最大ハッシュサイズを使用し、指定した最大ハッシュサイズを超えることはありません。カラムのデータ型でフルバイト制限が使用されていなければ、小さい方のハッシュサイズが使用されます。

データ型	値全体のハッシュに使用されるバイト数
LONG VARCHAR、DOUBLE、FLOAT、REAL、LONG BINARY、ST_GEOMETRY	ハッシュなし
BIT、TINYINT	1
SMALLINT	2
INTEGER、DATE	4
BIGINT、DATETIME、TIME、TIMESTAMP、TIMESTAMP WITH TIME ZONE	8
DECIMAL、NUMERIC	ほぼ正確に 2 等分されます。
CHAR、VARCHAR	文字列全体をハッシュするには、バイト単位の最大ハッシュサイズが、カラムの宣言されたサイズと一致する必要があります。UTF-8 でエンコードされたデータベースでは、宣言されたサイズを常に 2 倍にします。ただし、最大値は 32 バイトです。 たとえば、UTF-8 でエンコードされていないデータベースでカラムを VARCHAR(10) と宣言した場合は、必要なサイズは 10 バイトです。しかし、同じカラムを UTF-8 でエンコードされたデータベースで宣言した場合は、文字列全体のハッシュに使用されるサイズは 20 バイトです。
BINARY、VARBINARY	バイト単位の最大ハッシュサイズは、カラムの宣言されたサイズと一致する必要があります。 たとえば、カラムを BINARY(30) と宣言した場合は、必要なサイズは 30 バイトです。
UNIQUEIDENTIFIER	16

たとえば、それぞれ INTEGER と BINARY (20) を宣言した 2 つのカラムの複合インデックスに対して最大ハッシュサイズを 6 バイトに設定した場合、データ型のサイズ要件に基づいて、次の変更が生じます。

- INTEGER カラムのローの値全体がハッシュされ、インデックスに格納されます。これは、整数のデータ型のハッシュに必要なのは 4 バイトだけだからです。
- ハッシュの最初の 4 バイトは INTEGER カラムで使用されるので、BINARY カラムは最初の 2 バイトだけがハッシュされ、インデックスに格納されます。残りの 2 バイトで、BINARY カラムの適切な量がハッシュされない場合は、最大ハッシュサイズを大きくしてください。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

ローのデータ

データベースに格納されているデータのローの値も、インデックスのハッシュの有効性に影響します。

たとえば、特定のカラムのエントリ間で共通のプレフィクスを共有している場合は、プレフィクスだけがハッシュされるサイズを選択するとハッシュの効果がなくなる可能性があります。この場合、共通のプレフィクスがハッシュされるサイズよりも大きいサイズを選択する必要があります。共通のプレフィクスが長い場合は、値をハッシュしないことも検討してください。

ユニークでないインデックスに重複する値が数多く含まれており、Ultra Light が値全体をハッシュできない場合は、ハッシュによるパフォーマンスの向上は期待できません。

データベースサイズ

各インデックスページには固定のオーバヘッドがありますが、ページのほとんどの領域は実際のインデックスエントリに使用されます。ハッシュサイズを大きくすると、各インデックスエントリが大きくなり、1 ページに収まるエントリの数が少なくなります。大規模なテーブルになると、大規模なハッシュを使用するインデックスの方が、小規模のハッシュを使用するテーブルやハッシュを使用しないテーブルより、使用するページ数が多くなります。必要なページ数が増えると、データベースサイズが増加し、パフォーマンスが低下します。通常、このようなパフォーマンスの低下が生じるのは、キャッシュで保持できるページ数が固定されていて、Ultra Light でページのスワップが発生するためです。

次の表に、インデックスに含まれるデータの格納に必要なページ数にハッシュサイズがどう影響するかの概要を示します。

テーブル	ページサイズ	ハッシュサイズ	エントリ数	必要なページ
テーブル A	4 KB	0	1200	3 ページ
テーブル B	4 KB	32 バイト	116	3 ページ
テーブル C	4 KB	32 バイト	1200 エントリ	11 ページ

ハッシュサイズの設定

最大ハッシュサイズは次の 2 つの方法で設定できます。

- データベースでの最大サイズのデフォルトを格納するには、データベースの作成時に max_hash_size 作成パラメータを設定します。デフォルトでインデックスをハッシュしない場合は、この値を 0 に設定します。インデックスをハッシュする場合は、この値を 32 バイト以内の任意の値に変更するか、Ultra Light のデフォルト値 4 バイトを使用できます。
- 新しいインデックスの作成時に特定のハッシュサイズを設定してデフォルト値を上書きします。次のいずれかの方法を使用します。
  - SQL Central で、新しいインデックスを作成するときに [最大ハッシュサイズ] プロパティを設定します。
  - SQL で、CREATE TABLE 文または CREATE INDEX 文で WITH MAX HASH SIZE 句を使用します。

関連情報

SQL データ型 [278 ページ]

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

Ultra Light インデックスの追加 [63 ページ]
Ultra Light max_hash_size 作成オプション [155 ページ]
CREATE INDEX 文 [Ultra Light] [490 ページ]
CREATE TABLE 文 [Ultra Light] [495 ページ]

1.14.2.4 Ultra Light の実行プラン

Ultra Light の実行プランは、クエリ実行時のテーブルとインデックスへのアクセス方法を示します。

Ultra Light には、クエリオプティマイザが含まれています。このオプティマイザは、クエリの効率的なプランの生成を試みる、Ultra Light ランタイムの内部コンポーネントです。オプティマイザは、テンポラリテーブルを使用して中間結果が格納されないようにしたり、クエリで 2 つのテーブルがジョインされるときに、テーブルの関連するサブセットだけがアクセスされるようにしたりします。

オプティマイザの上書き

オプティマイザでは、常に最も効率的なアクセスプランが目標とされますが、特に多数の可能性がある複雑なクエリでは、この目標の達成は保証されません。極端な場合、`OPTION (FORCE ORDER)` 句をクエリに追加することによって、Ultra Light が選択するテーブル順序を上書きして、クエリに出現する順序でテーブルにアクセスすることもできます。このオプションは一般的には使用しないことをおすすめします。パフォーマンスが低い場合、通常は、適切なインデックスを作成して実行速度を上げるようにしてください。

> **注記**
>
> クエリを使用してデータを更新しない場合は、クエリで FOR READ ONLY 句を指定してください。こうすると、パフォーマンスが向上することがあります。

このセクションの内容:

オプティマイザで使用するアクセス方法の決定 [535 ページ]
Ultra Light オプティマイザは、クエリの最適化に使用するインデックスを選択する際に、優れた最適化方式を取ります。

実行プランを確認する場合 [537 ページ]
実行プランは、クエリの処理時にデータベースサーバが何を考慮するかを示します

実行プランの表示 [538 ページ]
Interactive SQL を使用して、準備文の実行方法をまとめた Ultra Light プランを表示します。テキストプランは、Interactive SQL のプランビューアに表示されます。

実行プランの解釈 [539 ページ]
Ultra Light のショートプランは、クエリのアクセス方法をテキスト形式で短くまとめたものです。

 © 2016 SAP SE or an SAP affiliate company.All rights reserved.

関連情報

SELECT 文 [Ultra Light] [516 ページ]

1.14.2.4.1 オプティマイザで使用するアクセス方法の決定

Ultra Light オプティマイザは、クエリの最適化に使用するインデックスを選択する際に、優れた最適化方式を取ります。

ただし、単純なクエリの場合、オプティマイザがクエリパフォーマンスの最適化に使用するインデックスをあらかじめ決めることや、そもそもインデックスを使用するかどうかを決めることは、簡単ではありません。複雑さが増すにつれて、選択するインデックスはクエリで必要な句によって変わってきます。通常、FOR READ ONLY 句が存在することで、オプティマイザが、クエリパフォーマンスの向上のためにインデックスではなくダイレクトテーブルスキャンを選択することがあります。

クエリを最適化する場合、オプティマイザはクエリの要件を確認し、パフォーマンスの向上に使用できるインデックスがないかどうかをチェックします。どのインデックスを使用してもパフォーマンス向上が望めない場合、オプティマイザはインデックスをスキャンせず、テンポラリテーブルかダイレクトページスキャンを使用します。したがって、インデックスを使用して実験し、生成された実行プランを頻繁にチェックして、次の事項を確認する必要があります。

- オプティマイザによって使用されないインデックスを保持していないこと。
- 作成されるテンポラリテーブルの数を最小限に抑えていること。

複雑なクエリの場合には、使用されるインデックスを予測するのはさらに困難です。たとえば、クエリに WHERE 述部があり、ORDER BY 句に加えて GROUP BY 句が存在する場合、インデックスが 1 つだけではクエリの探索条件を満たさない可能性があります。したがって、WHERE 述部の選択性要件を満たすインデックスを作成していても、オプティマイザがそれを使用しない可能性があります。代わりに、オプティマイザは、最も多くの処理を必要とする ORDER BY 条件でのパフォーマンスが最も優れているインデックスを使用する可能性があります。

実行プランのチェック

実行プランは、API 呼び出しを使用してプログラムでチェックしたり、Interactive SQL のプランビューアでチェックしたりできます。

インデックスが使用されていない場合

実行プランは次のように表示されます。

```
scan(T)
```

テンポラリテーブルが使用されている場合

実行プランは次のように表示されます。

```
temp [scan(T)]
```

インデックスが使用されている場合

インデックス名が実行プランに含まれている場合は次のように表示されます。

```
scan (T, index_name)
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

このセクションの内容:

テンポラリテーブルの管理 [536 ページ]
テンポラリテーブルは、アクセスプランによって、実行中にデータを格納するために、一時的なテーブル、つまりテンポラリワークテーブルとして使用されます。このテーブルが存在するのは、アクセスプランの実行中のみです。

1.14.2.4.1.1 テンポラリテーブルの管理

テンポラリテーブルは、アクセスプランによって、実行中にデータを格納するために、一時的なテーブル、つまりテンポラリワークテーブルとして使用されます。このテーブルが存在するのは、アクセスプランの実行中のみです。

オプティマイザは、クエリ結果を返すことを目的としたテンポラリテーブル作成を回避しようとします。これは、最初のローを返す前に、テンポラリテーブル全体を移植する必要があるからです。インデックスが存在する場合、オプティマイザはまずこのインデックスを使用しようとします。テンポラリテーブルを作成するのは最後の手段としてのみです。

一般的に、テンポラリテーブルが使用されるのは、使用可能なメモリに中間結果が収まらない場合です。たとえば、次のような場合です。

- サブクエリをアクセスプランの早い段階で評価する必要がある場合
- テンポラリテーブルのデータが単一の接続に対してのみ保持される場合
- クエリにインデックス以外のカラムに対する ORDER BY 句が含まれている場合
- クエリにインデックス以外のカラムに対する GROUP BY 句が含まれている場合

作成したインデックスでテンポラリテーブルの作成を回避できるかどうかを予測することは困難です。そのため、クエリプランをチェックして、作成したインデックスが実際に Ultra Light クエリオプティマイザで使用されているかどうかを確認する必要があります。

テンポラリテーブルの使用を避けるには、ORDER BY 句または GROUP BY 句で使用されるカラムでインデックスを使用します。

このセクションの内容:

ダイレクトページスキャン [537 ページ]
Ultra Light では、データベースページから直接情報にアクセスする方が効率的な場合、インデックススキャンの代わりにダイレクトページスキャンを使用します。

関連情報

実行プランの解釈 [539 ページ]
実行プランを確認する場合 [537 ページ]
Ultra Light TEMP_DIR 接続パラメータ [195 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.14.2.4.1.1.1 ダイレクトページスキャン

Ultra Light では、データベースページから直接情報にアクセスする方が効率的な場合、インデックススキャンの代わりにダイレクトページスキャンを使用します。

ダイレクトページスキャンが使用されるのは、オプティマイザが次の条件を確認した場合のみです。

- 結果を効率的に返すことができる既存のインデックスがありません。
- 更新の実行にクエリが使用されません。たとえば、文が FOR READ ONLY (FOR 句が指定されていない場合のデフォルト設定) であると宣言されている場合、またはデータが更新されないことが明らかなようにクエリが記述されている場合です。

Ultra Light では、ローが格納されているページからローを直接読み込むので、クエリ結果は順序に関係なく返されます。その後のクエリ結果の順序は予想できません。ローの順序を予測可能で決定的にするには、ORDER BY 句を使用して一貫性のある順序で結果を取得する必要があります。ただし、順序が重要ではない場合は、ORDER BY 句を省略することで、クエリのパフォーマンスを向上できます。

> **注記**
>
> アプリケーションのプログラムに Ultra Light API の UL テーブルを使用している場合、ダイレクトページスキャンは使用できません。

Ultra Light がいつページを直接スキャンするか、または結果を返すのにどのインデックスが使用されたかを確認できます。

関連情報

オプティマイザで使用するアクセス方法の決定 [535 ページ]
インデックススキャンの作成と保持 [529 ページ]

1.14.2.4.2 実行プランを確認する場合

実行プランは、クエリの処理時にデータベースサーバが何を考慮するかを示します

次の情報が必要な場合に、Interactive SQL で実行プランを確認します。

- 結果を返すために使用されるインデックス。インデックススキャンオブジェクトにテーブルの名前と、そのテーブルで使用されているインデックスが含まれます。
- 結果を返すためにテンポラリテーブルが使用されるかどうか。テンポラリテーブルは、Ultra Light のテンポラリファイルに書き込まれます。
- テーブルがジョインされる順序。この情報により、パフォーマンスへの影響を確認できます。
- クエリの実行が遅い理由、またはクエリの実行が遅くならないかどうか。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

関連情報

Ultra Light TEMP_DIR 接続パラメータ [195 ページ]

1.14.2.4.3 実行プランの表示

Interactive SQL を使用して、準備文の実行方法をまとめた Ultra Light プランを表示します。テキストプランは、Interactive SQL のプランビューアに表示されます。

コンテキスト

Ultra Light では、実行プランはプランをテキスト形式で短くまとめたものです。他のプランのタイプはサポートされていません。ただし、プランは短く、情報が 1 行にまとめてあるので、簡単に比較できます。

手順

1. ツール > プランビューア を開くをクリックします。
2. *SQL* ウィンドウ枠で、クエリを入力します。
3. プランの取得をクリックして、指定した SQL 文のプランを生成します。

結果

プランビューアの下ウィンドウ枠にテキストプランが表示されます。

例

次の文を考えてみます。

```
SELECT I.inv_no, I.name, T.quantity, T.prod_no
FROM Invoice I, Transactions T
WHERE I.inv_no = T.inv_no
```

この文により、下記のようなプランが生成されます。

```
join[scan(Invoice,primary),index-scan(Transactions,secondary)]
```

このプランは、primary というインデックスを使用して Invoice テーブルのすべてのローを読み込むことによって、ジョイン操作を実行することを示します。次に、Transaction テーブルの secondary というインデックスを使用して、inv_no カラムが一致するローのみを読み込みます。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

関連情報

実行プランの解釈 [539 ページ]

1.14.2.4.4 実行プランの解釈

Ultra Light のショートプランは、クエリのアクセス方法をテキスト形式で短くまとめたものです。

ショートプランを読み込むには、テーブルのジョイン操作またはスキャン操作が実装される方法を理解している必要があります。

スキャン操作の場合

1 つのオペランドで表されます。1 つのテーブルのみに適用され、インデックスが使用されます。テーブル名とインデックス名は操作名の後に丸カッコ ((と)) で表されます。

その他の操作

1 つまたは複数のオペランドで表されます。このオペランド自体もプランの場合があります。Ultra Light では、これらのオペランドは、角カッコ ([と]) で囲まれた、カンマ区切りのリストです。

操作リスト

次の表に、Ultra Light でサポートされている操作を示します。

演算	説明
count(*)	テーブルのローの数を数えます。
distinct[`plan`]	クエリの DISTINCT 部分を実装し、重複するローを比較、排除します。基本となるプランでローがソートされるときに使用され、重複、連続するローが排除されます。2 つの連続するローが一致する場合、最初のローだけが結果セットに追加されます。
dummy	操作は実行されません。次の 2 つの場合にのみ使用されます。 • FROM 句で DUMMY を指定した場合。 • クエリに FROM 句がない場合。
filter[`plan`]	基本となるプランによって指定される各ローに探索条件を実行します。true と評価されたローだけが、結果セットの一部として転送されます。
group-by[`plan`]	GROUP BY の結果を集約したものを作成し、グループ化されたデータの複数の行をソートします。ローは、発生する順序で表示され、連続するローを比較してグループ化されます。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

演算	説明
group-single[plan]	1 行だけが返されることがわかっている場合にのみ、GROUP BY の結果を集約したものを作成します。
keyset[plan]	テンポラリテーブル内のローの作成に使用されたローを記録し、Ultra Light で元の行を更新できるようにします。これらのローを更新しない場合は、クエリで FOR READ ONLY 句を使用してこの操作をなくします。
index-scan(table-name、index-name)	テーブルの一部だけを読み出します。開始ローはインデックスを使用して検索します。
join[plan、plan]	2 つのプラン間で内部ジョインを行います。
lojoin[plan、plan]	2 つのプラン間で左外部ジョインを行います。
like-scan(table-name、index-name)	テーブルの一部だけを読み出します。開始ローはインデックスを使用してパターン一致で検索します。
rowlimit[plan]	移植されたローに対してローの制限処理を行います。ローの制限は、SELECT 文の TOP n または FIRST 句によって設定されます。
scan(table-name、index-name)	インデックスが示す順序でテーブル全体を読み出します。
sub-query[plan]	サブクエリの開始を示します。
temp[plan]	基本となるプラン内のローからテンポラリテーブルを作成します。Ultra Light では、基本となるローをソートする必要があり、そのために使用できるインデックスが見つからなかった場合にテンポラリテーブルが使用されます。 インデックスを追加して、テンポラリテーブルを不要にできます。ただし、インデックスを追加すると、インデックス対象のテーブル内のローの挿入または同期に必要な時間が長くなります。
union-all[plan, ..., plan]	基本となるプランで生成されるローに対して UNION ALL 操作を行います。

1.14.3 挿入と更新のパフォーマンスのヒント

Ultra Light データベースの挿入と更新のパフォーマンス向上に役立ついくつかのトピックについて説明します。

このセクションの内容:

トランザクションとローのステータス管理 [541 ページ]
Ultra Light は、データベース内でデータのほかにステータス情報を保持します。

単一のトランザクションまたはグループ化されたトランザクションのフラッシュ [544 ページ]
コミットされたトランザクションのフラッシュを遅延させることで、Ultra Light でのリカバリポイントを選択できます。

 © 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.14.3.1 トランザクションとローのステータス管理

Ultra Light は、データベース内でデータのほかにステータス情報を保持します。

ステータス情報を追跡して格納することにより、次の項目が管理されます。

- 同時接続数: Ultra Light で必要に応じてリソースを共有できるようにするためです。
- 同期の進行状況のステータス: 同期が正常に実行されるかを確認するためです。
- ローのステータス: 同期間でのデータの変更内容を追跡することでデータの整合性を維持するためです。
- トランザクション: データをいつ、どのようにコミットするかを決定するためです。Ultra Light では、1 つのトランザクションは完全に処理されるかまったく処理されないかのどちらかです。
- リカバリとバックアップの情報: オペレーティングシステムのクラッシュ、およびストレージカードの取り出しやデバイスのリセットなどの Ultra Light の実行中のエンドユーザ操作からデータを保護するためです。

このセクションの内容:

Ultra Light の同時実行性 [541 ページ]
Ultra Light の同時実行性には、いくつかの要素があります。

Ultra Light データベースのローのステータス管理 [542 ページ]
Ultra Light はローステータス情報を管理します。テーブルとローのステータスの追跡は、データ同期時に特に重要です。

Ultra Light でのトランザクション処理 [543 ページ]
トランザクションは、自動的に実行される処理の論理セットです。つまり、トランザクションのすべての処理がデータベースに格納されるか、トランザクションの処理が 1 つもデータベースに格納されないかのどちらかです。

関連情報

Ultra Light 同期クライアント [10 ページ]
Ultra Light データベースのバックアップとリカバリ [50 ページ]

1.14.3.1.1 Ultra Light の同時実行性

Ultra Light の同時実行性には、いくつかの要素があります。

複数の Ultra Light データベースへのアクセス

1 つのアプリケーションで、複数のデータベースへの接続を開くことができます。

複数のアプリケーション

一度に 1 つのプロセスによって Ultra Light データベースを 1 つのみ開くことができます。Ultra Light エンジンを使用すると、複数のアプリケーションを操作できます。

複数のスレッド

Ultra Light はマルチスレッドのアプリケーションをサポートします。複数のスレッドを使用し、各スレッドが同じまたは異なるデータベースに接続できる、単一のアプリケーションを作成できます。特定の接続 (または SQLCA) に一度にアクセス

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

できるスレッドは 1 つのみです (Ultra Light はスレッド対応で、接続はスレッド非依存ですが、単一の接続または SQLCA に関する同時要求をすることはできません)。一般的に、特定のデータベースに関する要求は直列化されますが、同期は例外です。別々のデータベースに関する要求は同時に実行されます。

複数のトランザクション/要求

各接続は、進行中のトランザクションを一度に 1 つ使用できます。トランザクションは、1 つの要求または複数の要求で構成されます。トランザクションの実行中に行われたデータの変更は、トランザクションがコミットされるまでデータベースで永続化されません。トランザクションで行われたデータの変更は、すべてコミットされるか、すべてロールバックされるかのどちらかです。

同期

アップロードおよびダウンロードでは、読み込み/書き込みアクセスが許可されます。ただし、アプリケーションがローを変更してからダウンロードによってこのローが変更されようとすると、ダウンロードが失敗し、ロールバックが行われます。同期時のデータへのアクセスを無効にするには、Disable Concurrency 同期パラメータを使用します。

Ultra Light では、すべてのプラットフォームで再開可能なダウンロードがサポートされます。再開可能なダウンロードを使用すると、ネットワークの問題またはアプリケーションを終了する必要があることに起因して停止しているダウンロードを完了することができます。

関連情報

Ultra Light データベースの制限事項 [15 ページ]
Ultra Light でのトランザクション処理 [543 ページ]
Ultra Light クライアント [71 ページ]
Additional Parameters 同期パラメータ [92 ページ]

1.14.3.1.2 Ultra Light データベースのローのステータス管理

Ultra Light はローステータス情報を管理します。テーブルとローのステータスの追跡は、データ同期時に特に重要です。

Ultra Light データベースのローのステータスを記録するために、内部マーカが使用されます。ローのステータスは、トランザクション処理、リカバリ、同期を制御します。アプリケーションがローを挿入、更新、または削除すると、Ultra Light はその操作内容、およびその操作を実行した接続を反映するようにローのステータスを変更します。トランザクションがコミットされると、トランザクションの影響を受けるすべてのローのステータスが変更され、コミットが反映されます。コミット中に予期せぬ障害が発生すると、トランザクション全体がロールバックされます。次のリストは、その動作をまとめたものです。

削除が実行された場合

対象の各ローのステータスが、削除されたことを反映して変更されます。ロールバックによって削除が実行されない場合は、ローの元のステータスがリストアされます。

削除がコミットされた場合

対象のローがメモリから削除されない場合もあります。一度も同期が行われていないローは削除されます。同期が実行されたローは削除されません。削除操作を行うには、まず統合データベースと同期させる必要があるからです。次の同期が実行された後、ローはメモリから削除されます。

ローが更新された場合

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

ローの新しいバージョンが作成されます。古いローと新しいローのステータスが変更され、古いローは表示されなくなり、新しいローが表示されます。
ローの更新がコミットされた場合

トランザクションがコミットされると、トランザクションの影響を受けるすべてのローのステータスが変更され、コミットが反映されます。更新が同期されると、ローの古いバージョンと新しいバージョンの両方で競合が検出され、解決が行われます。その後、古いローがデータベースから削除され、新しいローが通常のローになります。
ローが追加された場合

ローがデータベースに追加され、コミット未実行のマークが付きます。
追加されたローがコミットされた場合

ローにコミット済みのマークが付き、統合データベースとの同期が必要であると通知されます。

関連情報

Ultra Light データベースのバックアップとリカバリ [50 ページ]
単一のトランザクションまたはグループ化されたトランザクションのフラッシュ [544 ページ]
Ultra Light でのトランザクション処理 [543 ページ]

1.14.3.1.3 Ultra Light でのトランザクション処理

トランザクションは、自動的に実行される処理の論理セットです。つまり、トランザクションのすべての処理がデータベースに格納されるか、トランザクションの処理が 1 つもデータベースに格納されないかのどちらかです。

Ultra Light ランタイムへの Ultra Light アプリケーションのアクセスは直列化されます。複数のトランザクションを同時に開くことは可能ですが、Ultra Light では一度に 1 つのトランザクションしか処理されません。これにより、アプリケーションでは以下の現象が発生しません。

- トランザクションがブロックされること (デッドロック)。Ultra Light が既存のローロックに基づいて要求をブロックすることはありません。このような事態になった場合、Ultra Light はすぐにエラーを返します。
- 未処理の変更を上書きすること。トランザクションは、他のトランザクションの未処理の変更を上書きすることはできません。トランザクションによってローが変更されると、Ultra Light は、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まで、このローをロックします。ロックによって、他のトランザクションはローを読み込むことはできても、ローを変更できなくなります。

> **注記**
>
> Ultra Light J および C++ の API を除くすべての Ultra Light API は、オートコミットモードで動作します。
>
> オートコミットモードでは、Ultra Light は各操作の後にコミットを実行します。一部の API はデフォルトでオートコミットを使用します。これらのインタフェースのいずれかを使用している場合、複数操作トランザクションを使用するには、オートコミットをオフに設定します。オートコミットをオフに設定する方法は、使用しているプログラミングインタフェースによって異なります。ほとんどのインタフェースでは、これは接続オブジェクトのプロパティです。

たとえば、A と B という 2 つのアプリケーションが 1 つのデータベースの同じローを読み取っており、どちらもそのデータに基づいて当該ローのカラムの 1 つに入る新しい値を計算するとします。A がローを新しい値で更新し、B が同じローを変更しよう

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

とした場合、B はエラーになります。ロックされたローを変更しようとするとエラー SQLCODE SQLE_LOCKED が設定され、削除されたローを変更しようとするとエラー SQLE_NOTFOUND が設定されます。したがって、データの変更を試行した後には SQLCODE 値をチェックするようアプリケーションをプログラムしてください。

関連情報

エラー処理 [625 ページ]
Ultra Light.NET でのエラー処理 [577 ページ]
Ultra Light C++ でのトランザクションの管理 [624 ページ]
Ultra Light.NET でのトランザクション管理 [576 ページ]

1.14.3.2 単一のトランザクションまたはグループ化されたトランザクションのフラッシュ

コミットされたトランザクションのフラッシュを遅延させることで、Ultra Light でのリカバリポイントを選択できます。

リカバリポイントを使用すると、Ultra Light がコミットを記憶領域に解放したときに、トランザクションに含まれている SQL 文のサブセットがトリガする追加操作オーバヘッドを制御できます。

デフォルトでは、Ultra Light は操作ベースのデフォルト処理を使用しており、トランザクションはコミットされるとすぐに、個別に記憶領域にフラッシュされます。配備環境によっては、このような頻度の高い処理では負荷が重くなり、トランザクションスループットが制限される可能性があります。このデフォルト処理によるパフォーマンスの低下を防ぐには、ステータスベースの方法を選択できます。特にオートコミット操作に依存しているアプリケーションでは、この方法によって、コミットされたトランザクションの記憶領域へのフラッシュに必要な追加オーバヘッドを遅延させます。

チェックポイントでの制御

独自のチェックポイントを設定し、これを使用して、それまでの期間に実行された作業を解放します。チェックポイントは、単一トランザクション内にも、複数のトランザクションにわたるものでも、必要に応じていくつでも使用できます。

グループ化して制御

トランザクションカウントスレッショルドかタイムアウトスレッショルドまたはその両方を使用して、実行した作業を解放できます。

ステータスに応じてコミットのフラッシュを遅延させる方が、パフォーマンスへの影響が少なく、アプリケーションの設計もすっきりします。これは、アプリケーションが Ultra Light からの応答を待機する必要がないためです。コミットフラッシュを遅延させることで、作業が完全には完了していないデータへのきめ細かい制御が可能になり、トランザクションへの影響が最小限に抑えられます。たとえば、販売アプリケーションでは、注文が、すべての項目が追加されたり承認されたりする前に、別のアプリケーションに対しても使用可能になります。

ただし、コミットフラッシュが遅延されるトランザクションのリカバリ性を考慮しておくことが重要です。解放されていないトランザクションはリカバリできません。したがって、アプリケーションのデータ整合性とパフォーマンスとの間のトレードオフを評価しておく必要があります。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

関連情報

Ultra Light COMMIT_FLUSH 接続パラメータ [183 ページ]
Ultra Light commit_flush_count オプション [テンポラリ] [203 ページ]
Ultra Light commit_flush_timeout オプション [テンポラリ] [204 ページ]
CHECKPOINT 文 [Ultra Light] [488 ページ]

1.14.4 Ultra Light のベンチマークに関するヒント

ベンチマークテストは、一般的に、プリケーション開発サイクルの運用段階に入る前に行われます。

このフェーズでは、アプリケーションで Ultra Light データベースをテストして、両方のコンポーネントができるだけ効率的に連携することを確認する必要があります。テストで期待どおりのパフォーマンス効率が達成されなかった場合は、データベースのチューニングかアプリケーションの最適化、またはその両方を実行して、ベンチマーク結果を向上させることができます。

> **注記**
>
> Ultra Light の配備が Mobile Link 同期環境内に含まれている場合、必ず同期パフォーマンスもテストしてください。

このセクションの内容:

ベンチマークテストの種類 [545 ページ]
ベンチマークテストを行うことにより、数種類のパフォーマンスを観察できます。

手法 [547 ページ]
ベンチマークテストは、準備、作成、および実行という 3 つの段階から成ります。

1.14.4.1 ベンチマークテストの種類

ベンチマークテストを行うことにより、数種類のパフォーマンスを観察できます。

- SQL 文:Ultra Light データベースは、SQL クエリを効率的に処理して、結果をすばやく返すように最適化されています。それでも、データベースのパフォーマンスを向上させるには、重要なクエリが適切に実行されているかどうかを確認する必要があります。
- 同期:Mobile Link 同期のスループットを最大限にするには、複数の同期が同時に発生し、効率よく実行することが重要になります。
- インデックス
- テーブル設計
- アプリケーションコード
- デバイス設定:たとえば、Ultra Light の配備ロケーションとして、外部フラッシュメモリとデバイスの内部メモリをそれぞれ使用して比較します。
- データベース設定:たとえば、キャッシュサイズ、ページサイズ、予約サイズ、インデックス、ハッシュサイズなどを変えてみます。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

- データのスループット (トランザクション数/秒に基づく):一般的に、Ultra Light は大量データ入力処理用のデータベースではありませんが、業務の要件によっては、このベンチマークについてのテストを実行する場合があります。
- ソフトウェアの変更:2 つの異なるバージョンの Ultra Light 間、または 2 つの異なるバージョンのアプリケーション間で、ソフトウェアの変更による影響をテストする必要があります。

このセクションの内容:

SQL クエリのテスト [546 ページ]
実行できるクエリベンチマークテストには、代表 SQL ベンチマークと対象を限定した SQL ベンチマークの 2 種類があります。

1.14.4.1.1 SQL クエリのテスト

実行できるクエリベンチマークテストには、代表 SQL ベンチマークと対象を限定した SQL ベンチマークの 2 種類があります。

代表 SQL ベンチマーク

この種類のテストでは、日常的な業務においてアプリケーションによって実行される、一般的なトランザクションの代表的な文を選択してテストする必要があります。各アプリケーションにはさまざまな業務用途があるため、アプリケーションごとに異なるベンチマークテストが必要となります。たとえば、計測アプリケーションでは、1 つの基本的な INSERT 文をテストするのみですむ場合があります。一方、モバイル営業支援アプリケーションでは、複数の INSERT 文以外に、複数の SELECT 文と、必要に応じて UPDATE 文をテストする場合があります。

アプリケーションでのクエリの量によって、現実的にテスト可能な範囲が制限されます。クエリ処理が多すぎる場合は、目的の SQL ベンチマークテストの実行が制限されることがあります。

対象を限定した SQL ベンチマーク

アプリケーションで多数の文を使用する場合は、必要に応じて、次の文の一部またはすべてを含めて、テストの範囲を絞ることができます。

- 最も頻繁に使用する文。
- 大量のデータを処理する文。
- 緊急の要件を持つ文。
- アプリケーションを使用する業務状況で最も重要となる文。
- 最も複雑な文。たとえば、最大数のテーブルジョインが含まれる文や、多くのサブクエリが使用される文です。これらの種類の文ででは、大量のデバイスリソースが使用されることがあります。これらの文については、使用頻度が低くても、デバイスの容量を超えていないことを確認してください。
- インデックスでサポートされていない文。
- 大量のメモリリソースを使用する文。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.14.4.2 手法

ベンチマークテストは、準備、作成、および実行という 3 つの段階から成ります。

1. 準備フェーズ
   ベンチマークテストを開始する前に、データベース設計を最終決定し、アプリケーション開発の条件を整備します。
2. 作成フェーズ
   Ultra Light の配備において予測されるエンドユーザの動作を模倣するカスタムプログラムを構築できます。
3. 実行および分析フェーズ
   データベースの各種要素を微調整し、その変更の結果を記録して分析できるようにします。すべての変更から最大限の効果を得ることができるまで、テストを繰り返します。

このセクションの内容:

準備フェーズ [547 ページ]
準備フェーズでは、データベースとアプリケーションを適切なベンチマーク候補と見なせる状態にし、テストで達成したい目標を決定できます。

作成フェーズ [548 ページ]
信頼できる結果が得られるテストを作成します。信頼できるテストでない場合、一定期間における複数のテスト結果を適切に比較できません。

ベンチマークテストの実行 [549 ページ]
ベンチマークテストを実行します。

1.14.4.2.1 準備フェーズ

準備フェーズでは、データベースとアプリケーションを適切なベンチマーク候補と見なせる状態にし、テストで達成したい目標を決定できます。

準備フェーズで行うことは次のとおりです。

1. データベースの論理設計を完了します。
   次のことを確認します。
   - テーブルを作成し、代表的なデータを移植していること。
   - これらのテーブル内のデータをより効果的に取得するためのインデックスを作成していること。
2. データベースとアプリケーションの両方の物理配備環境を準備します。配備環境は、最終的な運用環境を正確に表している必要があります。つまり、ラボと運用環境で同じプラットフォーム/デバイスタイプ上の同じメモリおよびディスク設定を共有する必要があります。
3. アプリケーションプログラミングフェーズを進める条件が整っていることを確認します。パフォーマンスの欠陥ではなく最適化を追求するようにしてください。ただし、テストの結果として欠陥が見つかる場合もあります。
   すべてのデータベースクエリがデータにアクセスでき、必要な量のデータが返される必要があります。運用環境でデータのソートが必要な場合は、クエリにもこのデータが含まれていることを確認してください。そうしないと、アプリケーションで代表的なメモリ要件を正確にテストできません。
4. データベースとアプリケーションのコピーを最終的なディスクロケーションに配備します。
5. 確認して調整したいデータベースパフォーマンスの要素を決定します。

これで、データベースやアプリケーションに対するベンチマークテストを実行できます。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

関連情報

ベンチマークテストの種類 [545 ページ]

1.14.4.2.2 作成フェーズ

信頼できる結果が得られるテストを作成します。信頼できるテストでない場合、一定期間における複数のテスト結果を適切に比較できません。

効果的で信頼性の高いベンチマークテストを作成するには、次の要因を考慮します。

目標

パフォーマンス比率を取得しますか。それとも、データベースに対するコマンドの処理に必要な期間を確認しますか。前者については、SQL パフォーマンスをテスト場合、設定された時間間隔が経過するまで、1 つ以上の文を繰り返し実行することがあります。このテストでは、スループット比率を確認できます。スループット比率を簡単に表すと次のようになります。

```
statement-number / time-interval = throughput ratio
```

環境

基準とするテスト環境を確立し、その設計とスコープを記録します。同じ条件で同じテストを実行できないと、そのテストの結果を適切に比較できません。また、ベンチマークテストの一環でラボで使用するハードウェアとソフトウェアが運用環境のものと一致している必要があります。

状態

信頼できるベンチマークテストでは、各反復を常に同じアクションで開始します。サードパーティアプリケーションを Ultra Light と同時に動作させる必要があるかどうかを判断してください。これらのアプリケーションがパフォーマンスに影響を及ぼす場合は、ベンチマークテストの設計に追加します。実行する必要のないサードパーティアプリケーションは、必ず完全に終了してください。アプリケーション/プロセスが最小限に抑えられ、アイドル状態になっていても、メモリは引き続き使用されるため、適切な結果を得ることができない場合があります。

結果

ベンチマークの結果は、テストの各反復の後に一貫した方法で残します。一定期間に収集した結果が特定の傾向を示すことがあり、これにより、データベースかアプリケーション、またはその両方にどのような変更を加えると Ultra Light のパフォーマンスを改善できるかを特定しやすくなることがあります。

タイミングメカニズム

ベンチマークテストでは、ユーザアクションをシミュレートします。したがって、通常、これらのアクションで経過した実行時間を追跡します。タイミングメカニズムが体系化されており、実行時間がテストの結果に正確に反映されていることを確認してください。

関連情報

SQL クエリのテスト [546 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.14.4.2.3 ベンチマークテストの実行

ベンチマークテストを実行します。

コンテキスト

ベンチマークテストの実行は、テストを繰り返し実行してデータベースをチューニングするフェーズです。このフェーズでは、データベースに何らか (たとえば、1 つ以上のデータベースプロパティまたは接続パラメータの値) の変更を加えた後、テストを再実行して変更の結果を確認します。

次の手順は、効果を最大化するために、さまざまなデータベースプロパティや接続パラメータをテストすることを想定しています。テストが必要なすべてのパラメータのテストが完了するまで、この手順を繰り返してください。

> **注記**
>
> Ultra Light の配備の負荷および目的に対して重要な意味を持つプロパティまたはパラメータのみを選択してください。

手順

1. テストの最初の反復を実行することによって、基準を作成します。この場合、さまざまなデータベースプロパティや接続パラメータをテストするため、可能な範囲で Ultra Light のデフォルト値を使用します。
2. 標準的なテストの実行を開始します。1 回のテストでチューニングするデータベースプロパティまたは接続パラメータは 1 つのみとします。このように制限することによって、収集する結果がそのアプローチの中で体系化され、チューニング作業において効果が最大化されたタイミングを判断しやすくなります。
3. ベンチマークプログラムからの出力には、次の内容が含まれます。
   - 各テストの識別子またはラベル
   - プログラムの実行の反復
   - チェック対象の要素の名前とその要素をどのように変更したか
   - 記録された経過時間

   たとえば、他のデータベースパラメータをテストできたとしても、ページサイズ、キャッシュサイズ、予約サイズのみを変更するようにテストを制限した場合、次の例に示すようなテーブルに出力が保存されます。

   ```
   PROP/PARM             VALUES
   TEST NUMBER           001     002     003
   page_size             1       2       8
   CACHE_SIZE            128     256     512
   RESERVE_SIZE          128     256     512

   STMT ID               EXECUTION (seconds)
     01                  01.55   01.50   01.49
     02                  02.01   02.20   01.59
     03                  00.33   00.55   00.44
   ```
4. 反復が完了したら、それ以降の実行結果に意図しない影響を及ぼさないよう、データベースを基準のステータスに戻します。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

結果

ベンチマークテストの結果に応じて、次のいずれかを実行します。

- パフォーマンスが向上した場合は、同じプロパティまたはパラメータの値を変更して、テストを再実行します。パフォーマンスがそれ以上向上しなくなるまで、この値のチューニングを続けてください。
- パフォーマンスが低下した場合は、プロパティまたはパラメータの値を前の値に戻します。

次のステップ

別の新しいプロパティまたはパラメータをテストします。

1.15 Ultra Light トラブルシューティング

Ultra Light データベースの問題をトラブルシューティングする際に役立ついくつかのトピックについて説明します。

このセクションの内容:

Ultra Light エンジンを起動できない [551 ページ]
START 接続パラメータを使用し、次の定義で Ultra Light エンジンを起動しましたが、クライアントが SQLE_UNABLE_TO_CONNECT_OR_START を返します。

アップグレード後にデータベースに接続できない [551 ページ]
Ultra Light をアップグレードしました。管理ツールを使用して空のデータベースを作成できることがわかりました。しかし、SQL Central を使用して空のデータベースまたはその他のデータベース (`CustDB.udb` を含む) に接続しようとすると、エラーが発生します。ただし、SQL Anywhere データベースとの接続は問題なく動作します。

Ultra Light データベースの破損 [552 ページ]
データベースが破損している場合、エラーメッセージが送信されます。

データベースのサイズが安定しない [553 ページ]
アプリケーションが複数のクライアントデバイス間のラージバイナリオブジェクトを数多く収集して、この情報を統合データベースに同期し、その後、各クライアントデバイスから同期されたデータが削除されます。ただし、データベースからデータが削除されても、データベースのサイズは大きいままです。

新しい Ultra Light データベースへの ASCII データのインポート [554 ページ]
新しい Ultra Light データベースを作成しましたが、インポートできない `.csv` ASCII データファイルがあります。

以前のバージョンのユーティリティが動作する [555 ページ]
Ultra Light 17 のインストールを完了しました。しかし、Ultra Light のユーティリティを実行しようとすると、以前のバージョンが起動します。

結果セットに予測できない変更が生じる [555 ページ]
クエリを実行すると、実行するたびに結果セットが変わります。

Ultra Light エンジンクライアントでエラー -764 が発生する [556 ページ]
Windows Mobile デバイスで Ultra Light エンジンを実行しようとして、クライアントから -764 エラーが返されます。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.15.1 Ultra Light エンジンを起動できない

START 接続パラメータを使用し、次の定義で Ultra Light エンジンを起動しましたが、クライアントが SQLE_UNABLE_TO_CONNECT_OR_START を返します。

```
START="¥Program Files¥uleng17.exe"
```

説明

引用符の位置が正しくありません。

推奨事項

このパラメータを機能させるには、最初の引用符の前に ¥ 記号が必要です。たとえば、次のようにしてこのパス内のスペースを明確に記述します。

```
START=¥"Program Files¥uleng17.exe"
```

または

```
START='"¥Program Files¥uleng17.exe"'
```

1.15.2 アップグレード後にデータベースに接続できない

Ultra Light をアップグレードしました。管理ツールを使用して空のデータベースを作成できることがわかりました。しかし、SQL Central を使用して空のデータベースまたはその他のデータベース (`CustDB.udb` を含む) に接続しようとすると、エラーが発生します。ただし、SQL Anywhere データベースとの接続は問題なく動作します。

説明

SQL Anywhere のすべてのアプリケーションとプロセスを閉じていませんでした。このため、Ultra Light のプラグインが正しくインストールされていません。

推奨事項

SQL Anywhere を削除して再インストールします。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

1. SQL Central、Interactive SQL、実行中のすべてのデータベースエンジンを閉じます。
2. 次のコマンドを実行します。

```
dbisql -terminate
```

```
scjview -terminate
```

3. Windows のタスク マネージャを開き、`scjview.exe` プロセスと `dbisql.exe` プロセスを閉じます。
4. 最新バージョンの Ultra Light を再インストールします。

1.15.3 Ultra Light データベースの破損

データベースが破損している場合、エラーメッセージが送信されます。

現象

次の場合は、Ultra Light データベースが破損している可能性があります。

- 次のエラーが生成されます。
  - SQLE_DEVICE_ERROR
  - SQLE_DATABASE_ERROR (他の問題による現象の可能性もあり)
  - SQLE_MEMORY_ERROR (他の問題による現象の可能性もあり)
- クラッシュするか、無効なクエリ結果が返されます。

説明

一般的な破損の原因は 2 つあります。

- 頻繁の高い原因としては、ファイルの格納時にデバイスに問題が発生したため、ファイルの内容が誤って変更されることを挙げることができます。通常、この問題が発生すると、Ultra Light データベースはすぐに機能を停止します。
- 頻度の低い原因としては、Ultra Light コードのエラーによってインデックスの正常な管理に失敗することを挙げることができます。クエリの結果に対する変更を予期することが難しいため、これらの問題は長期間発見されない可能性があります。

推奨事項

チェックサムは、Ultra Light データベースのオフラインの破損を検出するために使用します。チェックサムによって、重要なページの破損が原因で他のデータが破損するのを防ぐことができます。チェックサムが一致しなかった場合、Ultra Light データベースでページがロードされるときに、Ultra Light によってすぐにデータベースが停止され、致命的なエラーがレポートされます。このエラーは訂正できません。代わりに、次の手順を実行してください。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1. エラーをレポートします。これは、破損の原因となる一連のイベントがわかっている場合、およびエラーに再現性がある場合に役立ちます。
2. データが必要な場合は、Ultra Light データベースの内容をファイルにアンロードします。
3. 新しい Ultra Light データベースを作成します。
4. アンロードしたデータを同期またはロードすることによって、データを再移植します。

関連情報

Ultra Light checksum_level 作成オプション [143 ページ]

1.15.4 データベースのサイズが安定しない

アプリケーションが複数のクライアントデバイス間のラージバイナリオブジェクトを数多く収集して、この情報を統合データベースに同期し、その後、各クライアントデバイスから同期されたデータが削除されます。ただし、データベースからデータが削除されても、データベースのサイズは大きいままです。

説明

このことは問題となります。デバイスのリソースは限られているため、ファイルサイズは慎重に管理する必要があります。

データベースのサイズが大きくなるのは、データベース内のデータ量が増える場合のみです。ただし、いったん拡大すると、データベースファイルはそのサイズを維持し、自動的に縮小しません。ファイルの空き領域は、内部的に管理されます。

推奨事項

同期しないテーブルに STOP SYNCHRONIZATION DELETE 文または TRUNCATE 文を使用しないようにします。代わりに、同期しないテーブルには FROM `table-name` 句を含む DELETE 文を使用します。

同期後のデータベースを再作成します。

1. デバイスに配備するデータベースを作成します。
2. クライアントデバイスで必要とされるスキーマを定義する DDL 文の SQL スクリプトを作成します。
3. データを同期します。
4. データベースを削除します。
5. 新しい空のデータベースを作成し、ALTER DATABASE SCHEMA FROM FILE 文で標準のデータベーススキーマを使用します。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

関連情報

Ultra Light データベーススキーマのアップグレードの配備 [125 ページ]
STOP SYNCHRONIZATION DELETE 文 [Ultra Light] [521 ページ]
TRUNCATE TABLE 文 [Ultra Light] [523 ページ]
DELETE 文 [Ultra Light] [503 ページ]
ALTER DATABASE SCHEMA FROM FILE 文 [Ultra Light] [479 ページ]

1.15.5 新しい Ultra Light データベースへの ASCII データのインポート

新しい Ultra Light データベースを作成しましたが、インポートできない .csv ASCII データファイルがあります。

説明

Ultra Light の管理ツールでは、.csv フォーマットはサポートされていません。

推奨事項

次のいずれかの方法を試すことができます。

- Interactive SQL (dbisql) を使用してデータをインポートします。Ultra Light データベースに接続し、データ > インポート の順にクリックします。または、Ultra Light データベースに接続し、INPUT 文を実行します (この文は、Ultra Light の PreparedStatement オブジェクトでは使用できません)。

  > **注記**
  >
  > Ultra Light にはプライマリキーが必要です。Interactive SQL ではユーザのテーブルを作成できますが、そのプライマリキーは自動的に作成されません。この目的のために作成した空の Ultra Light データベースに必ず接続してください。

- バッチプロセスの一部としてこの機能を組み込む場合は、独自のコードを作成してください。

関連情報

Ultra Light 用 Interactive SQL ユーティリティ (dbisql) [209 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

1.15.6 以前のバージョンのユーティリティが動作する

Ultra Light 17 のインストールを完了しました。しかし、Ultra Light のユーティリティを実行しようとすると、以前のバージョンが起動します。

説明

Ultra Light の複数のバージョンが同じコンピュータにインストールされている場合は、管理ツールを使用するとき、システムのパスに注意してください。インストールでは、最も新しくインストールされたバージョンの実行プログラムのフォルダがシステムパスの最後に追加されるため、ソフトウェアの最新バージョンをインストールしたにもかかわらず、以前にインストールしたバージョンが実行される場合があります。

推奨事項

この問題にはさまざまな対処方法があります。

1.15.7 結果セットに予測できない変更が生じる

クエリを実行すると、実行するたびに結果セットが変わります。

説明

得られる結果セットを注意深く確認します。セットの結果は本当に違いますか。または、実行のたびに最も効率的な順序で結果が返されているだけではありませんか。ローに最後にアクセスした日時やその他の要因によって、クエリを実行するたびに選択される順序が変わることがあります。

推奨事項

予測可能な順序または一貫性のある順序で結果セットが返されるようにする場合は、SELECT 文に ORDER BY 句が含まれていることを確認します。結果セットで返される結果が依然として不適切な場合は、データベースが破損している可能性があります。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

関連情報

SELECT 文 [Ultra Light] [516 ページ]
Ultra Light データベースの破損 [552 ページ]

1.15.8 Ultra Light エンジンクライアントでエラー -764 が発生する

Windows Mobile デバイスで Ultra Light エンジンを実行しようとして、クライアントから -764 エラーが返されます。

適用対象

Windows Mobile

説明

-764 のエラーは、エンジンが見つからないため、起動できなかったことを意味します。

推奨事項

次のいずれかのアクションを検討します。

- 推奨される配備ロケーション `¥Windows` フォルダにエンジンを再配備することを検討します。Ultra Light は、自動的にこのロケーションでエンジンのファイルを検索します。
- 別のロケーションにエンジンをインストールした場合は、接続コードに START 接続パラメータを使用していることを確認します。
- START 接続パラメータを使用しており、エンジンへのパスが正しい場合は、パス名の特殊文字に正しいエスケープシーケンスを使用していることを確認します。
  たとえば、次のコードは変更が必要となる場合があります。

```
ULConnection conn = new ULConnection(@"dbf=¥Program Files¥HelloEngine
¥HelloEngine.udb;
START=¥Windows¥uleng17.exe")
```

  場合によっては、次のようなコードに変更します。

```
ULConnection conn = new ULConnection(@"dbf=¥¥¥"Program Files ¥"¥¥HelloEngine¥
¥HelloEngine.udb;
START=¥¥Windows¥¥uleng17.exe");
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

関連情報

Ultra Light エンジンの起動 [127 ページ]
Ultra Light START 接続パラメータ [194 ページ]

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

2 Ultra Light - .NET プログラミング

このマニュアルでは、Ultra Light.NET プログラミングインタフェースについて説明します。Ultra Light.NET を使用すると、Microsoft Windows コンピュータや、ハンドヘルドデバイス、モバイルデバイス、埋め込みデバイス用のデータベースアプリケーションを開発し、配備することができます。

このセクションの内容:

システムの要件とサポート対象プラットフォーム [558 ページ]
Ultra Light.NET は、Windows Mobile デバイスをサポートします。データベースの開発には、サードパーティソフトウェアが必要です。

Ultra Light.NET アプリケーション開発 [559 ページ]
Ultra Light.NET API には、Sap.Data.UltraLite ネームスペースがあります。このネームスペースには、Ultra Light への ADO.NET インタフェースが用意されています。業界標準モデルに基づいて構築でき、非常によく似た SQL Anywhere ADO.NET インタフェースへのマイグレーションパスが提供されるという利点があります。

チュートリアル: Ultra Light.NET を使用した Windows Mobile アプリケーションの構築 [583 ページ]
このチュートリアルでは、Microsoft Visual Studio を使用して Windows Mobile 用の Ultra Light アプリケーションをビルドする手順について説明します。また、このチュートリアルは Sap.Data.UltraLite ネームスペースに用意されている Microsoft ADO.NET インタフェースを使用し、Microsoft .NET 3.5 Compact Framework 上で動作します。

Ultra Light .NET API リファレンス [601 ページ]
.NET Framework 2.0 および .NET Compact Framework 2.0 用 Ultra Light.NET データプロバイダの API を使用して、Ultra Light の.NET アプリケーションを作成することができます。

2.1 システムの要件とサポート対象プラットフォーム

Ultra Light.NET は、Windows Mobile デバイスをサポートします。データベースの開発には、サードパーティソフトウェアが必要です。

開発プラットフォーム

Ultra Light.NET を使用してアプリケーションを開発するためには、次の要件を満たす必要があります。

- サポートされているデスクトップバージョンの Microsoft Windows
- Microsoft Visual Studio 2005 または Microsoft Visual Studio 2008
- Windows Mobile デバイスを使用する場合は、.NET Compact Framework バージョン 2 以降

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

ターゲットプラットフォーム

Ultra Light.NET でサポートしているターゲットプラットフォームは次のとおりです。

- Windows 上で動作する Microsoft .NET Compact Framework バージョン 2.0 以降と .NET Framework 2.0 以降
- Windows Mobile デバイスを使用する場合は、Microsoft .NET Compact Framework バージョン 2 以降

2.2 Ultra Light.NET アプリケーション開発

Ultra Light.NET API には、Sap.Data.UltraLite ネームスペースがあります。このネームスペースには、Ultra Light への ADO.NET インタフェースが用意されています。業界標準モデルに基づいて構築でき、非常によく似た SQL Anywhere ADO.NET インタフェースへのマイグレーションパスが提供されるという利点があります。

.NET Compact Framework は、Windows Mobile 用の Microsoft .NET ランタイムコンポーネントです。複数のプログラミング言語をサポートします。Visual Basic.NET または C# を使用して、Ultra Light.NET を使用するアプリケーションを構築できます。

> **注記**
>
> Ultra Light.NET API の代わりに Ultra Light C++ API を使用して Windows Mobile デバイスおよびデスクトップ用のアプリケーションを作成できます。

このセクションの内容:

Microsoft Visual Studio の SQL Anywhere ツール [560 ページ]
Microsoft Visual Studio 統合は Ultra Light.NET に対してサポートされています。Microsoft Visual Studio 2005 以降では、Microsoft Visual Studio サーバエクスプローラーから SQL Anywhere 統合ツールにアクセスできます。

Ultra Light データベースの接続の設定 [560 ページ]
データベースのデータを操作するには、Ultra Light アプリケーションをデータベースに接続する必要があります。

Ultra Light.NET での SQL 文を使用したデータの作成と修正 [562 ページ]
Ultra Light アプリケーションは、SQL 文またはテーブル API を使用してテーブルデータにアクセスできます。

Ultra Light.NET での ULTable クラスを使用したデータの作成と修正 [568 ページ]
Ultra Light アプリケーションは、SQL 文または ULTable クラスを使用してテーブルデータにアクセスできます。

Ultra Light.NET でのトランザクション管理 [576 ページ]
Ultra Light のトランザクション処理は、データベース内のデータの整合性を保証します。トランザクションは、作業の論理単位です。トランザクション全体が実行されるか、トランザクション内の文がどれも実行されないかのいずれかです。

Ultra Light.NET でのスキーマ情報 [576 ページ]
テーブル API のオブジェクトは、テーブル、カラム、インデックス、同期の各パブリケーションを表します。各オブジェクトには、そのオブジェクトの構造情報へアクセスするための Schema プロパティがあります。

Ultra Light.NET でのエラー処理 [577 ページ]
.NET の標準エラー処理機能を使用して、エラーを処理できます。ほとんどの Ultra Light メソッドは、ULException エラーをスローします。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

Ultra Light.NET での Mobile Link データ同期 [577 ページ]
Ultra Light データベースを中央の統合データベースと同期します。同期には、SQL Anywhere に付属の Mobile Link 同期ソフトウェアが必要です。

Ultra Light.NET アプリケーションを配備する方法 [579 ページ]
Ultra Light.NET アプリケーションは、Windows Mobile および Windows に配備できます。Windows Mobile に配備する場合、Ultra Light.NET は .NET Compact Framework を必要とします。Windows に配備する場合は、.NET Framework を必要とします。Ultra Light.NET は、Microsoft ActiveSync 同期もサポートしています。

関連情報

Ultra Light C++ アプリケーション開発 [604 ページ]

2.2.1 Microsoft Visual Studio の SQL Anywhere ツール

Microsoft Visual Studio 統合は Ultra Light.NET に対してサポートされています。Microsoft Visual Studio 2005 以降では、Microsoft Visual Studio サーバエクスプローラーから SQL Anywhere 統合ツールにアクセスできます。

2.2.2 Ultra Light データベースの接続の設定

データベースのデータを操作するには、Ultra Light アプリケーションをデータベースに接続する必要があります。

> **注記**
> この章のサンプルコードは Microsoft C# で記述されています。サポートされている他の開発ツールを使用している場合は、このマニュアルの記述を適宜読み替えてください。

ULConnection オブジェクトの使用

ほとんどのアプリケーションは、Ultra Light データベースへの単一接続を使用し、接続を開いたままにしておきます。複数の接続が必要になるのは、マルチスレッドデータアクセスの場合だけです。そのため、ULConnection オブジェクトは、アプリケーションに対してグローバルに宣言するのが最も効果的です。

ULConnection オブジェクトの次のプロパティは、アプリケーションのグローバルな動作を管理します。

コミット動作

デフォルトでは、Ultra Light.NET アプリケーションは AutoCommit モードに設定されています。Insert 文、Update 文、Delete 文はすべて、すぐにデータベースにコミットされます。アプリケーションでのトランザクションの開始を定義するには、ULConnection.BeginTransaction を使用します。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

ユーザ認証

メソッドを使用すると、アプリケーションのユーザ ID とパスワードをそれぞれデフォルト値の DBA と sql から別の値に変更できます。各 Ultra Light データベースには、4 つまでのユーザ ID を定義できます。

同期

Connection オブジェクトから、同期を管理するオブジェクトのセットにアクセスできます。

テーブル

Ultra Light テーブルには、Connection オブジェクトのメソッドを使用してアクセスします。

コマンド

動的 SQL 文の実行を処理し、結果セットをナビゲーションするために、オブジェクトのセットが提供されています。

マルチスレッドアプリケーション

ULConnection オブジェクトと、このオブジェクトから作成されるすべてのオブジェクトは、それぞれ単一のスレッドで使用してください。アプリケーションが Ultra Light データベースにアクセスするのに複数のスレッドを必要とする場合は、スレッドごとに個別の接続が必要です。たとえば、独立したスレッドから同期を実行するアプリケーションを設計する場合、同期用に別の接続を使用し、その接続を独立したスレッドから開く必要があります。

このセクションの内容:

Ultra Light.NET を使用した Ultra Light データベースへの接続 [562 ページ]
ULConnectionParms オブジェクトを使用して、`mydata.udb` という名前の Ultra Light データベースに接続します。

関連情報

Ultra Light.NET でのトランザクション管理 [576 ページ]
Ultra Light.NET での Mobile Link データ同期 [577 ページ]
Ultra Light.NET での ULTable クラスを使用したデータの作成と修正 [568 ページ]
Ultra Light.NET での SQL 文を使用したデータの作成と修正 [562 ページ]

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

2.2.2.1 Ultra Light.NET を使用した Ultra Light データベースへの接続

ULConnectionParms オブジェクトを使用して、`mydata.udb` という名前の Ultra Light データベースに接続します。

手順

1. ULConnection オブジェクトを宣言します。

   ```
   ULConnection conn;
   ```

2. 既存のデータベースへの接続を開きます。

   接続パラメータは、接続文字列として指定するか、ULConnectionParms オブジェクトを使用して指定します。

   ```
   ULConnectionParms parms = new ULConnectionParms();
   parms.DatabaseOnDesktop = "mydata.udb";
   conn = new ULConnection( parms.ToString() );
   conn.Open();
   ```

結果

`mydata.udb` データベースへの接続が確立されます。

次のステップ

データベース接続を使用して、データの作成、変更、または削除の SQL 操作を実行します。接続が開いている接状態で既存のデータベースオプションを変更できます。完了後、ULConnection オブジェクトを閉じます。

2.2.3 Ultra Light.NET での SQL 文を使用したデータの作成と修正

Ultra Light アプリケーションは、SQL 文またはテーブル API を使用してテーブルデータにアクセスできます。

SQL 文を使用して、次のタスクを実行できます。

- ローの挿入、削除、更新
- クエリの実行と結果セットのローの取得
- 結果セットのローのスクロール

このセクションの内容:

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

Ultra Light.NET での INSERT、UPDATE、DELETE を使用したデータ修正 [563 ページ]
Ultra Light では、SQL データ操作言語の操作を実行できます。この操作は、ULCommand.ExecuteNonQuery メソッドを使用して実行します。

Ultra Light.NET での SELECT を使用したデータの検索 [567 ページ]
SELECT 文を実行して Ultra Light データベースから情報を取得し、返される結果セットを処理します。

結果セットスキーマの説明 [567 ページ]
ULDataReader.GetSchemaTable メソッドと ULDataReader.Schema プロパティを使用すると、カラム名、カラムの総数、カラムスケール、カラムサイズ、カラムの SQL 型など、結果セットに関する情報を取得できます。

Ultra Light.NET での SQL 結果セットのナビゲーション [568 ページ]
ULDataReader オブジェクトに関連したメソッドを使用して、結果セット内をナビゲーションできます。

関連情報

Ultra Light.NET での ULTable クラスを使用したデータの作成と修正 [568 ページ]

2.2.3.1 Ultra Light.NET での INSERT、UPDATE、DELETE を使用したデータ修正

Ultra Light では、SQL データ操作言語の操作を実行できます。この操作は、ULCommand.ExecuteNonQuery メソッドを使用して実行します。

このセクションの内容:

Ultra Light.NET を使用したテーブル内のローの挿入 [564 ページ]
SQL 文のパラメータのプレースホルダは、? 文字を使用して指定します。INSERT、UPDATE、DELETE では必ず、コマンドのパラメータでの順序位置に従ってそれぞれの ? が参照されます。たとえば、最初の ? は 0、2 番目の ? は 1 のようになります。

Ultra Light.NET を使用したテーブル内のローの更新 [565 ページ]
SQL 文のパラメータのプレースホルダは、? 文字を使用して指定します。INSERT、UPDATE、DELETE では必ず、コマンドのパラメータでの順序位置に従ってそれぞれの ? が参照されます。たとえば、最初の ? は 0、2 番目の ? は 1 のようになります。

Ultra Light.NET でのテーブル内のローの削除 [566 ページ]
SQL 文のパラメータのプレースホルダは、? 文字を使用して指定します。INSERT、UPDATE、DELETE では必ず、コマンドのパラメータでの順序位置に従ってそれぞれの ? が参照されます。たとえば、最初の ? は 0、2 番目の ? は 1 のようになります。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

2.2.3.1.1 Ultra Light.NET を使用したテーブル内のローの挿入

SQL 文のパラメータのプレースホルダは、? 文字を使用して指定します。INSERT、UPDATE、DELETE では必ず、コマンドのパラメータでの順序位置に従ってそれぞれの ? が参照されます。たとえば、最初の ? は 0、2 番目の ? は 1 のようになります。

手順

1. ULCommand を宣言します。

```
ULCommand cmd;
```

2. SQL 文を ULCommand オブジェクトに割り当てます。

```
cmd = conn.CreateCommand();
cmd.CommandText = "INSERT INTO MyTable(MyColumn) values (?)";
```

3. 文の入力パラメータ値を割り当てます。

   次のコードは、文字列パラメータを示します。

```
String newValue;
cmd.Parameters.Clear();
// assign value
cmd.Parameters.Add("", newValue);
```

4. 文を実行します。

   戻り値は、文に影響されたローの数を示します。

```
int rowsInserted = cmd.ExecuteNonQuery();
```

5. 明示的なトランザクションを使用している場合は、変更をコミットします。

```
myTransaction.Commit();
```

結果

MyTable に新しいローが追加されます。MyColumn の値は空の文字列です。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

2.2.3.1.2 Ultra Light.NET を使用したテーブル内のローの更新

SQL 文のパラメータのプレースホルダは、? 文字を使用して指定します。INSERT、UPDATE、DELETE では必ず、コマンドのパラメータでの順序位置に従ってそれぞれの ? が参照されます。たとえば、最初の ? は 0、2 番目の ? は 1 のようになります。

手順

1. ULCommand を宣言します。

```
ULCommand cmd;
```

2. ULCommand オブジェクトに文を割り当てます。

```
cmd = conn.CreateCommand();
cmd.CommandText = "UPDATE MyTable SET MyColumn1 = ? WHERE MyColumn2 = ?";
```

3. 文の入力パラメータ値を割り当てます。

```
String newValue;
String oldValue;
cmd.Parameters.Clear();
// assign values
cmd.Parameters.Add("", newValue);
cmd.Parameters.Add("", oldValue);
```

4. 文を実行します。

```
int rowsUpdated = cmd.ExecuteNonQuery();
```

5. 明示的なトランザクションを使用している場合は、変更をコミットします。

```
myTransaction.Commit();
```

結果

MyTable のローエントリが更新されます。MyColumn1 の値は空の文字列です。このシナリオでは、MyColumn2 の値も空の文字列に設定されます。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

2.2.3.1.3 Ultra Light.NET でのテーブル内のローの削除

SQL 文のパラメータのプレースホルダは、? 文字を使用して指定します。INSERT、UPDATE、DELETE では必ず、コマンドのパラメータでの順序位置に従ってそれぞれの ? が参照されます。たとえば、最初の ? は 0、2 番目の ? は 1 のようになります。

手順

1. ULCommand を宣言します。

```
ULCommand cmd;
```

2. ULCommand オブジェクトに文を割り当てます。

```
cmd = conn.CreateCommand();
cmd.CommandText = "DELETE FROM MyTable WHERE MyColumn = ?";
```

3. 文の入力パラメータ値を割り当てます。

```
String deleteValue;
cmd.Parameters.Clear();
// assign value
cmd.Parameters.Add("", deleteValue);
```

4. 文を実行します。

```
int rowsDeleted = cmd.ExecuteNonQuery();
```

5. 明示的なトランザクションを使用している場合は、変更をコミットします。

```
myTransaction.Commit();
```

結果

MyTable のローエントリが削除されます。このテーブルの MyColumn 値は空の文字列です。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

2.2.3.2 Ultra Light.NET での SELECT を使用したデータの検索

SELECT 文を実行して Ultra Light データベースから情報を取得し、返される結果セットを処理します。

手順

1. クエリを保持する ULCommand オブジェクトを宣言します。

```
ULCommand cmd;
```

2. このオブジェクトに文を割り当てます。

```
cmd = conn.CreateCommand();
cmd.CommandText = "SELECT MyColumn FROM MyTable";
```

3. 文を実行します。

   クエリの結果は、数種類のオブジェクトの 1 つとして返されます。この例では、ULDataReader オブジェクトが使用されています。

```
ULDataReader customerNames = cmd.ExecuteReader();
int fc = customerNames.FieldCount();
while( customerNames.MoveNext() ) {
  for ( int i = 0; i < fc; i++ ) {
    System.Console.Write(customerNames.GetString( i ) + " " );
  }
  System.Console.WriteLine();
}
```

結果

SELECT 文の結果に含まれる文字列はその後コマンドプロンプトに出力されます。

2.2.3.3 結果セットスキーマの説明

ULDataReader.GetSchemaTable メソッドと ULDataReader.Schema プロパティを使用すると、カラム名、カラムの総数、カラムスケール、カラムサイズ、カラムの SQL 型など、結果セットに関する情報を取得できます。

例

次のサンプルコードは、ULDataReader.Schema プロパティと ResultSet.Schema プロパティを使用して、スキーマ情報をコマンドプロンプトに表示する方法を示しています。

```
for ( int i = 0; i < MyResultSet.Schema.GetColumnCount(); i++ ) {
      System.Console.WriteLine( MyResultSet.Schema.GetColumnName(i)
                                + " "
                                + MyResultSet.Schema.GetColumnSQLType(i) );
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

```
}
```

2.2.3.4 Ultra Light.NET での SQL 結果セットのナビゲーション

ULDataReader オブジェクトに関連したメソッドを使用して、結果セット内をナビゲーションできます。

結果セットオブジェクトは、結果セットをナビゲーションする次のメソッドを提供します。

MoveAfterLast

最後のローの後に移動します。

MoveBeforeFirst

最初のローの前に移動します。

MoveFirst

最初のローに移動します。

MoveLast

最後のローに移動します。

MoveNext

次のローに移動します。

MovePrevious

前のローに移動します。

MoveRelative(offset)

現在のローを基準にして、オフセットが指定した数だけローを移動します。オフセット値を正の値で指定すると、現在の結果セットのカーソル位置から前方に移動します。負の値で指定すると後方に移動します。オフセット値が 0 の場合、カーソルは移動しませんが、ローバッファが再配置されます。

2.2.4 Ultra Light.NET での ULTable クラスを使用したデータの作成と修正

Ultra Light アプリケーションは、SQL 文または ULTable クラスを使用してテーブルデータにアクセスできます。

テーブル API を使用して、次のタスクを実行できます。

- テーブルのローのスクロール
- 現在のローの値へのアクセス
- find メソッドと lookup メソッドを使用したテーブルのローの検索
- ローの挿入、削除、更新

このセクションの内容:

Ultra Light.NET でのローのナビゲーション [569 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

Ultra Light.NET は、幅広いナビゲーション作業を行うために、テーブルをナビゲーションする複数のメソッドを提供します。

Ultra Light のモード [570 ページ]
Ultra Light モードは、バッファ内の値の使用目的を指定します。Ultra Light には、デフォルトモードに加えて、次の 4 つの操作モードがあります。

Ultra Light.NET でのローの挿入 [571 ページ]
ローの挿入手順は、ローの更新手順とほぼ同じです。ただし、挿入操作の場合は、テーブル内のローをあらかじめ指定する必要はありません。

ローの更新 [571 ページ]
デフォルトでは、Ultra Light.NET は AutoCommit モードで動作するため、更新は永続的な記憶領域のローに即時適用されます。AutoCommit モードを無効にした場合は、コミット操作を実行するまで、更新は適用されません。

ローの検索 [572 ページ]
Ultra Light には、データを操作するための操作モードがいくつかあります。これらのモードのうちの 2 つ (検索モードとルックアップモード) は、検索に使用されます。Table オブジェクトには、テーブル内の特定のローを検索するために、これらのモードに対応するメソッドがあります。

ローの取得 [574 ページ]
ローの操作は、ULTable クラスのさまざまなメソッドを使用して管理できます。

Ultra Light.NET でのローの削除 [575 ページ]
ローの削除手順は、ローの挿入や更新よりも簡単です。挿入モードや更新モードに対応する削除モードはありません。

関連情報

Ultra Light.NET での SQL 文を使用したデータの作成と修正 [562 ページ]

2.2.4.1 Ultra Light.NET でのローのナビゲーション

Ultra Light.NET は、幅広いナビゲーション作業を行うために、テーブルをナビゲーションする複数のメソッドを提供します。

テーブルオブジェクトは、テーブルをナビゲーションする次のメソッドを提供します。

MoveAfterLast

最後のローの後に移動します。
MoveBeforeFirst

最初のローの前に移動します。
MoveFirst

最初のローに移動します。
MoveLast

最後のローに移動します。
MoveNext

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

次のローに移動します。
MovePrevious

前のローに移動します。
MoveRelative(offset)

現在のローを基準にして、オフセットが指定した数だけローを移動します。オフセット値を正の値で指定すると、現在のテーブルのカーソル位置から前方に移動します。負の値で指定すると後方に移動します。オフセット値が 0 の場合、カーソルは移動しませんが、ローバッファが再配置されます。

例

次のサンプルコードは、MyTable テーブルを開き、各ローの MyColumn カラムの値を表示します。

```
ULTable t = conn.ExecuteTable( "MyTable" );
int colID = t.GetOrdinal( "MyColumn" );
while ( t.MoveNext() ){
   System.Console.WriteLine( t.GetString( colID ) );
}
```

テーブルオブジェクトを開くと、テーブルのローがアプリケーションに公開されます。デフォルトでは、ローはプライマリキー値の順に並んでいますが、テーブルを開くときにインデックスを指定すると特定の順序でローにアクセスできます。

例

次のコードは、ix_col インデックスで順序付けられた MyTable テーブルの最初のローに移動します。

```
ULTable t = conn.ExecuteTable( "MyTable", "ix_col" );
t.MoveFirst();
```

2.2.4.2 Ultra Light のモード

Ultra Light モードは、バッファ内の値の使用目的を指定します。Ultra Light には、デフォルトモードに加えて、次の 4 つの操作モードがあります。

挿入モード

Insert メソッドを呼び出すと、バッファ内のデータが新しいローとしてテーブルに追加されます。

更新モード

Update メソッドを呼び出すと、現在のローがバッファ内のデータに置き換えられます。

検索モード

find メソッドの 1 つが呼び出されたときに、値がバッファ内のデータに正確に一致するローの検索に使用されます。

ルックアップモード

いずれかの lookup メソッドが呼び出されたときに、バッファ内のデータと一致するか、それより大きい値のローを検索します。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

2.2.4.3 Ultra Light.NET でのローの挿入

ローの挿入手順は、ローの更新手順とほぼ同じです。ただし、挿入操作の場合は、テーブル内のローをあらかじめ指定する必要はありません。

テーブルへのローの挿入順序に意味はありません。

例

次のコードでは、新しいローが挿入されます。

```
t.InsertBegin();
t.SetInt( id, 3 );
t.SetString( lname, "Carlo" );
t.Insert();
```

カラムの値を設定しない場合、そのカラムにデフォルト値があるときはデフォルト値が使用されます。カラムにデフォルトがない場合は、次のエントリが使用されます。

- NULL 入力可のカラムの場合は NULL
- NULL 入力不可の数値カラムの場合は 0
- NULL 入力不可の文字カラムの場合は空の文字列
- 明示的に値を NULL に設定するには、SetDBNull メソッドを使用します。

更新操作の場合は、コミットを実行したときに、永続的な記憶領域のデータベースに挿入が適用されます。AutoCommit モードでは、Insert メソッドの一部として Commit が実行されます。

2.2.4.4 ローの更新

デフォルトでは、Ultra Light.NET は AutoCommit モードで動作するため、更新は永続的な記憶領域のローに即時適用されます。AutoCommit モードを無効にした場合は、コミット操作を実行するまで、更新は適用されません。

警告

ローのプライマリキー値は更新できません。ローを削除し、代わりに新しいローを追加します。

このセクションの内容:

Ultra Light.NET を使用したテーブル内のローの更新 [572 ページ]
Update メソッドを使用してテーブル内のローを更新します。

関連情報

Ultra Light.NET でのトランザクション管理 [576 ページ]

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

2.2.4.4.1 Ultra Light.NET を使用したテーブル内のローの更新

Update メソッドを使用してテーブル内のローを更新します。

手順

1. 更新するローに移動します。

   ローに移動するには、テーブルをスクロールするか、find メソッドまたは lookup メソッドを使用してテーブルを検索します。
2. 更新モードを開始します。

   たとえば、次の指示は、テーブルで更新モードを開始します。

   ```
   t.UpdateBegin();
   ```
3. 更新するローの新しい値を設定します。

   たとえば、次の指示は、バッファ内の id カラムを 3 に設定します。

   ```
   t.SetInt( id , 3);
   ```
4. Update を実行します。

   ```
   t.Update();
   ```

結果

現在のローが更新されます。Table オブジェクトを開いたときに指定したインデックスのカラム値を変更した場合は、現在のローの定義が解除されます。

2.2.4.5 ローの検索

Ultra Light には、データを操作するための操作モードがいくつかあります。これらのモードのうちの 2 つ (検索モードとルックアップモード) は、検索に使用されます。Table オブジェクトには、テーブル内の特定のローを検索するために、これらのモードに対応するメソッドがあります。

> **注記**
>
> find メソッドや lookup メソッドを使用して検索されるカラムは、テーブルを開くのに使用されたインデックスにあることが必要です。

find メソッド

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

Table オブジェクトを開いたときに指定したソート順に基づいて、指定された検索値と正確に一致する最初のローに移動します。検索値が見つからない場合は、最初のローの前、または最後のローの後ろに位置設定されます。

lookup メソッド

Table オブジェクトを開いたときに指定したソート順に基づいて、指定された検索値と一致するか、それより大きい値の最初のローに移動します。

このセクションの内容:

検索とルックアップのメソッドによるローの検索 [573 ページ]
検索とルックアップのメソッドを使用して ULTable オブジェクトのローを検索します。

2.2.4.5.1 検索とルックアップのメソッドによるローの検索

検索とルックアップのメソッドを使用して ULTable オブジェクトのローを検索します。

手順

1. 検索モードまたはルックアップモードを開始します。

   モードを開始するには、テーブルオブジェクトのメソッドを呼び出します。たとえば、次のコードは検索モードを開始します。

   ```
   t.FindBegin();
   ```

2. 検索値を設定します。

   検索値は、現在のローの値を設定することで設定します。これらの値の設定は、データベースではなく、現在のローを保持しているバッファにのみ影響します。たとえば、次のコードは、バッファの値を Kaminski に設定します。

   ```
   int lname = t.GetOrdinal( "lname" );
   t.SetString( lname, "Kaminski" );
   ```

3. ローを検索します。

   適切なメソッドを使用して検索を実行します。たとえば、次の指示は、現在のインデックスで指定された値と正確に一致する最初のローを検索します。

   マルチカラムインデックスの場合、最初のカラムの値が常に使用され、他のカラムは省略できます。

   ```
   t.FindFirst();
   ```

4. ローの次のインスタンスを検索します。

   適切なメソッドを使用して検索を実行します。検索操作の場合は、FindNext でインデックス内のパラメータの次のインスタンスを検索します。ルックアップ操作では、MoveNext で次のインスタンスを検索します。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

結果

カーソルが目的のローをポイントします。

次のステップ

そのローのデータの削除または変更など、ローに対する操作を実行します。

2.2.4.6 ローの取得

ローの操作は、ULTable クラスのさまざまなメソッドを使用して管理できます。

Table オブジェクトは、次のいずれかの位置に常に置かれています。

- テーブルの最初のローの前
- テーブルのいずれかのローの上
- テーブルの最後のローの後ろ

Table オブジェクトがローの上に置かれている場合は、そのデータ型に適したメソッドセットを使用して、各カラムの値を取得したり、変更したりできます。

カラム値の取得

Table オブジェクトは、カラム値を取得するメソッドセットを提供します。これらのメソッドは、カラム ID を引数として取ります。

例

次のコードは、lname カラムの値を取得します。このカラムの値は文字列です。

```
int lname = t.GetOrdinal( "lname" );
string lastname = t.GetString( lname );
```

次のコードは、cust_id カラムの値を取得します。このカラムの値は整数です。

```
int cust_id = t.GetOrdinal( "cust_id" );
int id = t.GetInt( cust_id );
```

カラム値の変更

値を取り出すメソッド以外に、値を設定するメソッドもあります。値を設定するメソッドは、カラム ID と値を引数として取ります。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

例

たとえば、次のコードは、lname カラムの値を Kaminski に設定します。

```
t.SetString( lname, "Kaminski" );
```

これらのプロパティへの値の割り当てによって、データベース内のデータの値が変更されることはありません。現在の位置がテーブルの最初のローの前でも、テーブルの最後のローの後ろであっても、プロパティに値を割り当てることができます。ただし、これらの位置に現在のローが置かれている場合に、たとえば、変数へのプロパティの割り当てなどによって、データにアクセスしようとするとエラーになります。

```
// This code is incorrect
t.MoveBeforeFirst();
id = t.GetInt( cust_id );
```

値のキャスト

選択するメソッドは、割り当てるデータ型に一致させてください。データ型に互換性がある場合は、Ultra Light が自動的にデータベースのデータ型をキャストするため、getString メソッドを使用して整数値を文字列変数にフェッチしたりできます。

関連情報

CAST 関数 [データ型変換] [339 ページ]
CONVERT 関数 [データ型変換] [347 ページ]

2.2.4.7 Ultra Light.NET でのローの削除

ローの削除手順は、ローの挿入や更新よりも簡単です。挿入モードや更新モードに対応する削除モードはありません。

削除するローにカーソルを移動してから、Table.Delete メソッドを実行して、ローを削除します。

例

次のコードは、テーブル内の最初のローを削除する方法を示しています。

```
t.MoveFirst();
t.Delete();
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

2.2.5 Ultra Light.NET でのトランザクション管理

Ultra Light のトランザクション処理は、データベース内のデータの整合性を保証します。トランザクションは、作業の論理単位です。トランザクション全体が実行されるか、トランザクション内の文がどれも実行されないかのいずれかです。

デフォルトでは、Ultra Light.NET は AutoCommit モードで動作するため、挿入、更新、削除はそれぞれ独立したトランザクションとして実行されます。操作が完了すると、データベースに変更が加えられます。

複数文のトランザクションを使用するには、ULConnection.BeginTransaction を呼び出して ULTransaction クラスのオブジェクトを作成する必要があります。たとえば、2 つの口座間で資金を移動するアプリケーションでは、振込元口座からの引き落としと振込先口座への振り込みとを合わせて 1 つの操作として完了する必要があります。完了できない場合は、両方とも完了しないでください。

接続が有効なトランザクションを処理した場合、ULTransaction.Commit 文を実行して、トランザクションを完了し、データベースへの変更をコミットする必要があります。一連の更新を中止する場合は、ULTransaction.Rollback 文を実行して、トランザクションのすべての操作を取り消してロールバックする必要があります。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と、次に ULConnection.BeginTransaction を呼び出すまで、接続は AutoCommit モードに戻ります。

たとえば、次のコードフラグメントは、(デフォルトのオートコミット動作を回避しながら) 複数の操作を伴うトランザクションを設定する方法を示しています。

```
// Assuming an already open connection named  conn
ULTransaction txn = conn.BeginTransaction(IsolationLevel.ReadUncommitted);
// Perform transaction operations here
txn.Commit();
```

> **注記**
>
> Ultra Light は、IsolationLevel 列挙の ReadCommitted メンバーと ReadUncommitted メンバーをサポートしています。

一部の SQL 文 (特にデータベースの構造を変更する文) は、保留中のトランザクションをすべてコミットします。処理中に自動的にトランザクションをコミットする SQL 文の例として、CREATE TABLE と ALTER TABLE を挙げることができます。

2.2.6 Ultra Light.NET でのスキーマ情報

テーブル API のオブジェクトは、テーブル、カラム、インデックス、同期の各パブリケーションを表します。各オブジェクトには、そのオブジェクトの構造情報へアクセスするための Schema プロパティがあります。

API によるスキーマの変更はできません。スキーマに関する情報の取得のみが可能です。

次のスキーマオブジェクトと情報にアクセスできます。

ULDatabaseSchema

データベース内のテーブルの数と名前、日付と時刻の形式などのグローバルプロパティを公開します。

ULDatabaseSchema オブジェクトを取得するには、ULConnection.Schema を呼び出します。

ULTableSchema

このテーブル内のカラムとインデックスの数と名前。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

ULTableSchema オブジェクトを取得するには、ULTable.Schema を呼び出します。

ULIndexSchema

インデックス内のカラムに関する情報。インデックスには直接に対応するデータがないため、個別の Index クラスはなく、ULIndexSchema クラスのみが存在します。

ULIndexSchema オブジェクトを取得するには、ULTableSchema.GetIndex、ULTableSchema.GetOptimalIndex、ULTableSchema.GetPrimaryKey のいずれかのメソッドを呼び出します。

2.2.7 Ultra Light.NET でのエラー処理

.NET の標準エラー処理機能を使用して、エラーを処理できます。ほとんどの Ultra Light メソッドは、ULException エラーをスローします。

ULException.NativeError を使用して、エラーに割り当てられた ULSQLCode 値を取得できます。ULException には Message プロパティがあり、エラーの説明文の取得に使用できます。ULSQLCode エラーは、エラータイプを示す負の番号です。

同期後は、接続の SyncResult プロパティを使用して詳細なエラー情報を取得できます。たとえば、次の例は、同期時に発生したエラーをレポートする方法を示しています。

```
public void Sync() {
    try {
        _conn.Synchronize( this );
        _inSync = false;
    }
    catch( ULException uEx ) {
        if( uEx.NativeError == ULSQLCode.SQLE_MOBILINK_COMMUNICATIONS_ERROR ) {
            MessageBox.Show(
                "StreamErrorCode = " +
                      _conn.SyncResult.StreamErrorCode.ToString() + "¥r¥n"
                    + "StreamErrorParameters = " +
                      _conn.SyncResult.StreamErrorParameters + "¥r¥n"
                    + "StreamErrorSystem = " +
                      _conn.SyncResult.StreamErrorSystem + "¥r¥n"
            );
        }
        else {
            MessageBox.Show(uEx.Message);
        }
    }
    catch(System.Exception ex ) {
        MessageBox.Show(ex.Message);
    }
}
```

2.2.8 Ultra Light.NET での Mobile Link データ同期

Ultra Light データベースを中央の統合データベースと同期します。同期には、SQL Anywhere に付属の Mobile Link 同期ソフトウェアが必要です。

CustDB サンプルアプリケーションには、同期の実例もあります。詳細については、SQL Anywhere17 インストール環境の `Samples¥UltraLite.NET¥CustDB` サブディレクトリを参照してください。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

Ultra Light.NET は、TCP/IP、HTTP、HTTPS、TLS (トランスポートレイヤセキュリティ) の各同期をサポートしています。同期は、Ultra Light アプリケーションによって開始されます。常に、SyncParms オブジェクトのプロパティを使用して同期を制御します。

このセクションの内容:

C# アプリケーションでの同期の開始 [578 ページ]
C# アプリケーションで同期を開始するには、SyncParms オブジェクトを使用します。

Ultra Light.NET での Microsoft ActiveSync 同期の設定 [579 ページ]
Microsoft ActiveSync 同期を Ultra Light.NET アプリケーションに追加し、エンドユーザのコンピュータ上の Microsoft ActiveSync で使用するアプリケーションを登録できます。

関連情報

Ultra Light クライアント [71 ページ]

2.2.8.1 C# アプリケーションでの同期の開始

C# アプリケーションで同期を開始するには、SyncParms オブジェクトを使用します。

```
private void Sync( ULConnection conn )
{
    // Sync
    try
    {
         // setup to synchronize a publication named "high_priority"
         conn.SyncParms.Publications = "high_priority";

         // Set the synchronization parameters
         conn.SyncParms.Version     = "Version1";
         conn.SyncParms.StreamParms = "";
         conn.SyncParms.Stream      = ULStreamType.TCPIP;
         conn.SyncParms.UserName    = "51";
         conn.Synchronize();
    }
    catch (System.Exception t)
         {
           MessageBox.Show("Exception: " + t.Message);
         }
}
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

2.2.8.2 Ultra Light.NET での Microsoft ActiveSync 同期の設定

Microsoft ActiveSync 同期を Ultra Light.NET アプリケーションに追加し、エンドユーザのコンピュータ上の Microsoft ActiveSync で使用するアプリケーションを登録できます。

Microsoft ActiveSync 同期を開始できるのは、Microsoft ActiveSync 自体だけです。デバイスがクレードルにある場合や、[Microsoft ActiveSync] ウィンドウで同期が選択された場合に、Microsoft ActiveSync は同期を開始します。

Microsoft ActiveSync が同期を開始すると、Ultra Light アプリケーションがまだ実行されていない場合は Microsoft ActiveSync 用 Mobile Link プロバイダが Ultra Light アプリケーションを起動し、メッセージを送信します。Mobile Link プロバイダからのメッセージを受信および処理するには、ULActiveSyncListener オブジェクトをアプリケーションに実装します。アプリケーションは、SetActiveSyncListener メソッドを使用してリスナオブジェクトを指定する必要があります。この場合における MyAppClassName は、アプリケーションの一意の Microsoft Windows クラス名です。

```
dbMgr.SetActiveSyncListener( "MyAppClassName", listener );
```

Ultra Light は、Microsoft ActiveSync メッセージを受信すると、指定されたリスナの ActiveSyncInvoked メソッドを別のスレッド上で呼び出します。マルチスレッドの問題を回避するには、ActiveSyncInvoked メソッドでユーザインタフェースにイベントを通知してください。

マルチスレッドアプリケーションを使用する場合は、スレッドごとに別々の接続を使用し、Microsoft C# の *lock* キーワードまたは Microsoft Visual Basic .NET の *SyncLock* キーワードを使用して、アプリケーションの共有オブジェクトにアクセスします。ActiveSyncInvoked メソッドは、使用する接続の SyncParms.Stream に ULStreamType.ACTIVE_SYNC を指定してから ULConnection.Synchronize を呼び出します。

アプリケーションを登録するときは、次のパラメータを設定します。

クラス名

Connection.SetActiveSyncListener メソッドでアプリケーションが使用したクラス名と同じクラス名

2.2.9 Ultra Light.NET アプリケーションを配備する方法

Ultra Light.NET アプリケーションは、Windows Mobile および Windows に配備できます。Windows Mobile に配備する場合、Ultra Light.NET は .NET Compact Framework を必要とします。Windows に配備する場合は、.NET Framework を必要とします。Ultra Light.NET は、Microsoft ActiveSync 同期もサポートしています。

このセクションの内容:

Windows Mobile への Ultra Light.NET アプリケーションの配備 [580 ページ]
適切な作成パラメータ、接続パラメータ、同期パラメータ、プロトコルオプション、参照、配備ファイルを指定して、Ultra Light.NET アプリケーションが正常に実行されるようにします。

Windows Mobile への Ultra Light.NET アプリケーションの配備 (Ultra Light エンジン) [581 ページ]
適切な作成パラメータ、接続パラメータ、同期パラメータ、プロトコルオプション、参照、配備ファイルを指定して、Ultra Light.NET アプリケーションが正常に実行されるようにします。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

関連情報

Ultra Light アプリケーションのビルドと配備の仕様 [118 ページ]

2.2.9.1 Windows Mobile への Ultra Light.NET アプリケーションの配備

適切な作成パラメータ、接続パラメータ、同期パラメータ、プロトコルオプション、参照、配備ファイルを指定して、Ultra Light.NET アプリケーションが正常に実行されるようにします。

手順

1. 次のパラメータを指定します。
   - 難読化を使用している場合は、データベース作成時に作成パラメータ *obfuscate=1* を設定します。
   - AES 暗号化を使用する場合は、データベース作成時または接続時に接続パラメータ *DBKEY*=`encryption-key` を設定します。
2. Ultra Light アプリケーションで同期を使用する場合は、パラメータを適切に設定してください。

同期タイプ	パラメータの設定
TCP/IP	*Stream* 同期パラメータを tcpip に設定します。
HTTP	*Stream* 同期パラメータを http に設定します。
RSA TLS	*Stream* 同期パラメータを tls に設定します。
RSA HTTPS	*Stream* 同期パラメータを https に設定します。

3. RSA エンドツーエンド暗号化を使用している場合は、プロトコルオプション *e2ee_public_key*=`key-file` を設定します。
4. ZLIB 圧縮を使用している場合は、プロトコルオプション *compression=zlib* を設定します。
5. 参照の追加先:
   - Sap.Data.UltraLite
   - Sap.Data.UltraLite.resources
6. 次のファイルを配備します。
   - *%SQLANY17%*`¥UltraLite¥UltraLite.NET¥Assembly¥V2¥Sap.Data.UltraLite.dll`
   - *%SQLANY17%*`¥UltraLite¥UltraLite.NET¥Assembly¥V2¥en¥Sap.Data.UltraLite.resources.dll`
   - `ulnet`*17*`.dll`。Windows Mobile の場合は、*%SQLANY17%*`¥UltraLite¥UltraLite.NET¥CE¥Arm50` にあります。Windows の場合は、*%SQLANY17%*`¥UltraLite¥UltraLite.NET¥x64` または *%SQLANY17%*`¥UltraLite¥UltraLite.NET¥win32` にあります。
7. アプリケーションに適したファイルを配備します。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

- ZLIB 圧縮を使用している場合は、`mlczlib17.dll`。
- RSA TLS、RSA HTTPS、または RSA E2EE を使用している場合は、`mlcrsa17.dll`。

Windows Mobile の場合は、ファイルは *%SQLANY17%*`¥UltraLite¥CE¥Arm.50` にあります。Windows の場合は、ファイルは *%SQLANY17%*`¥UltraLite¥Windows¥x64` または *%SQLANY17%*`¥UltraLite¥Windows¥x86` にあります。

結果

Ultra Light.NET アプリケーションは、配備先の Windows デスクトップまたは Windows Mobile デバイスで正常に稼動します。

次のステップ

アプリケーションを配備した Windows デスクトップまたは Windows Mobile デバイスに Ultra Lite データベースを配備するか、配備したアプリケーションと一緒に新しいデータベースを作成します。

関連情報

Ultra Light データベースの配備方法 [124 ページ]
Ultra Light アプリケーションのビルドと配備の仕様 [118 ページ]

2.2.9.2 Windows Mobile への Ultra Light.NET アプリケーションの配備 (Ultra Light エンジン)

適切な作成パラメータ、接続パラメータ、同期パラメータ、プロトコルオプション、参照、配備ファイルを指定して、Ultra Light.NET アプリケーションが正常に実行されるようにします。

手順

1. 次のパラメータを指定します。
   - 難読化を使用している場合は、データベース作成時に作成パラメータ *obfuscate=1* を設定します。
   - AES 暗号化を使用する場合は、データベース作成時または接続時に接続パラメータ *DBKEY*`=encryption-key` を設定します。
2. Ultra Light アプリケーションで同期を使用する場合は、パラメータを適切に設定してください。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

同期タイプ	パラメータの設定
TCP/IP	*Stream* 同期パラメータを tcpip に設定します。
HTTP	*Stream* 同期パラメータを http に設定します。
RSA TLS	*Stream* 同期パラメータを tls に設定します。
RSA HTTPS	*Stream* 同期パラメータを https に設定します。

3. RSA エンドツーエンド暗号化を使用している場合は、プロトコルオプション *e2ee_public_key*=`key-file` を設定します。
4. ZLIB 圧縮を使用している場合は、プロトコルオプション *compression=zlib* を設定します。
5. 参照の追加先:
   - Sap.Data.UltraLite
   - Sap.Data.UltraLite.resources
6. 次のファイルを配備します。
   - *%SQLANY17%*`¥UltraLite¥UltraLite.NET¥Assembly¥V2¥Sap.Data.UltraLite.dll`
   - *%SQLANY17%*`¥UltraLite¥UltraLite.NET¥Assembly¥V2¥en¥Sap.Data.UltraLite.resources.dll`
   - `ulnetclient`*17*`.dll`。Windows Mobile の場合は、*%SQLANY17%*`¥UltraLite¥UltraLite.NET¥CE¥Arm50` にあります。Windows の場合は、*%SQLANY17%*`¥UltraLite¥UltraLite.NET¥x64` または *%SQLANY17%*`¥UltraLite¥UltraLite.NET¥win32` にあります。
7. アプリケーションに適したファイルを配備します。
   - *uleng17*`.exe`
   - ZLIB 圧縮を使用している場合は、`mlczlib`*17*`.dll`。
   - RSA TLS、RSA HTTPS、または RSA E2EE を使用している場合は、`mlcrsa`*17*`.dll`。

   Windows Mobile の場合は、ファイルは *%SQLANY17%*`¥UltraLite¥CE¥Arm.50` にあります。Windows の場合は、ファイルは *%SQLANY17%*`¥UltraLite¥Windows¥x64` または *%SQLANY17%*`¥UltraLite¥Windows¥x86` にあります。

結果

Ultra Light.NET アプリケーションは Ultra Light エンジンを使用し、配備先の Windows デスクトップまたは Windows Mobile デバイスで正常に稼動します。

次のステップ

アプリケーションを配備した Windows デスクトップまたは Windows Mobile デバイスに Ultra Lite データベースを配備するか、配備したアプリケーションと一緒に新しいデータベースを作成します。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

関連情報

Ultra Light データベースの配備方法 [124 ページ]
Ultra Light アプリケーションのビルドと配備の仕様 [118 ページ]

2.3 チュートリアル: Ultra Light.NET を使用した Windows Mobile アプリケーションの構築

このチュートリアルでは、Microsoft Visual Studio を使用して Windows Mobile 用の Ultra Light アプリケーションをビルドする手順について説明します。また、このチュートリアルは Sap.Data.UltraLite ネームスペースに用意されている Microsoft ADO.NET インタフェースを使用し、Microsoft .NET 3.5 Compact Framework 上で動作します。

前提条件

このチュートリアルは、次のことを前提にしています。

- C# プログラミング言語または Microsoft Visual Basic プログラミング言語に精通しています。
- SQL Central の Ultra Light プラグインを使用して Ultra Light データベースを作成する方法を知っています。
- Microsoft Visual Studio がコンピュータにインストール済みです。また、Microsoft Visual Studio に精通しています。このチュートリアルは Microsoft Visual Studio 2008 を使用してテストされており、Microsoft Visual Studio のその他のバージョンとは多少異なる Microsoft Visual Studio アクションやプロシージャに基づいている場合があります。
- Microsoft から Windows Mobile 5.0 SDK 以降をインストール済みです。
- 使用中のモバイルデバイスに .NET 3.5 Compact Framework をインストール済みです。

コンテキスト

このチュートリアルの目的は、Microsoft Visual Studio 環境での Ultra Light アプリケーションの開発プロセスについて、知識と経験を得ることです。

このチュートリアルには、Microsoft Visual Basic アプリケーションと Microsoft Visual C# アプリケーションのコードが含まれています。

Ultra Light ソフトウェアをインストールした Windows コンピュータに Microsoft Visual Studio がインストール済みの場合、Ultra Light のインストールプロセスは Microsoft Visual Studio の存在を検出し、必要な統合手順を実行します。Microsoft Visual Studio を Ultra Light の後にインストールする場合、または Microsoft Visual Studio の新しいバージョンをインストールする場合、次の手順に従って、コマンドプロンプトから手動で Ultra Light を Microsoft Visual Studio に統合する必要があります。

- Microsoft Visual Studio が実行されていないことを確認します。
- Microsoft Visual Studio 2005 以降の場合は、*%SQLANY17%*`¥UltraLite¥UltraLite.NET¥Assembly¥v2¥` フォルダから `installULNet.exe` を実行します。この作業には管理者権限が必要です。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

1. レッスン 1: Microsoft Visual Studio プロジェクトの作成 [584 ページ]
   このレッスンでは、新しい Microsoft Visual Studio アプリケーションを作成し、設定します。プログラミング言語として Microsoft Visual Basic と C# のどちらも使用できます。
2. レッスン 2: Ultra Light データベースの作成 [587 ページ]
   このレッスンでは、デスクトップ PC の SQL Central を使用して Ultra Light データベースを作成します。
3. レッスン 3: アプリケーションへのデータベース接続コントロールの追加 [589 ページ]
   このレッスンでは、Ultra Light データベースへの接続を確立する Ultra Light.NET アプリケーションにコントロールを追加します。
4. レッスン 4: データの挿入、更新、削除 [591 ページ]
   このレッスンでは、動的 SQL を使用するアプリケーションにコードを追加し、データベースのデータを変更します。
5. レッスン 5: アプリケーションの構築と配備 [596 ページ]
   このレッスンでは、アプリケーションを構築し、リモートデバイスまたはエミュレータに配備します。
6. C# チュートリアルのコードリスト [597 ページ]
   この章で説明したチュートリアルプログラムの完全なコードを次に示します。
7. Microsoft Visual Basic チュートリアルのコードリスト [599 ページ]
   この章で説明したチュートリアルプログラムの完全なコードを次に示します。

関連情報

データベース作成ウィザードでの Ultra Light データベースの作成 [28 ページ]

2.3.1 レッスン 1: Microsoft Visual Studio プロジェクトの作成

このレッスンでは、新しい Microsoft Visual Studio アプリケーションを作成し、設定します。プログラミング言語として Microsoft Visual Basic と C# のどちらも使用できます。

前提条件

このレッスンでは、必要なソフトウェアをすでにインストールしていることを前提としています。

コンテキスト

このチュートリアルでは、C# アプリケーションを設計している場合はファイルがディレクトリ `C:¥tutorial¥uldotnet¥CSApp` にあり、Microsoft Visual Basic アプリケーションを設計している場合はファイルがディレクトリ `C:¥tutorial¥uldotnet¥VBApp` にあることを前提とします。別の名前のディレクトリを使用する場合は、チュートリアルを通じてそのディレクトリを使用してください。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

手順

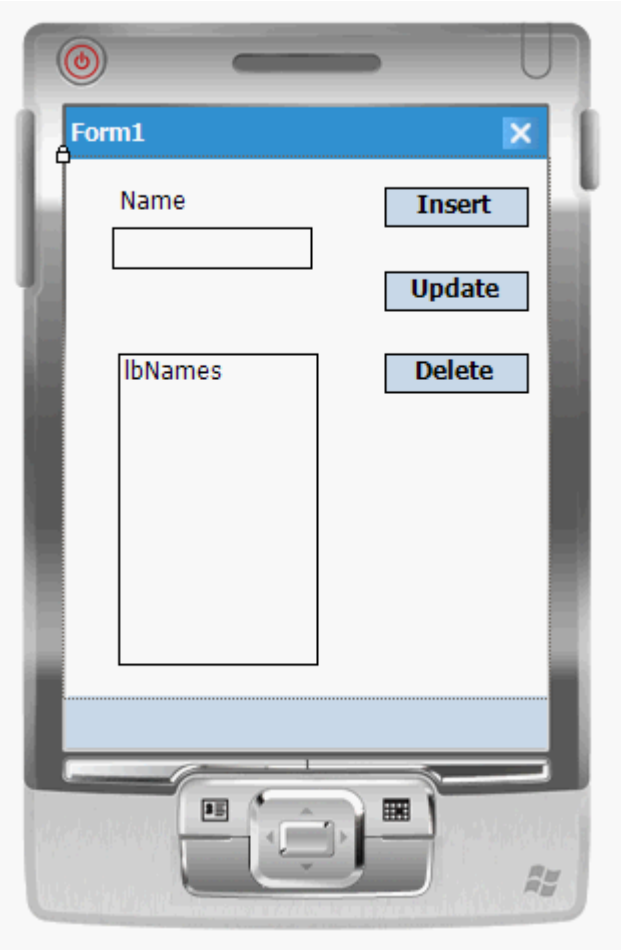
1. Microsoft Visual Studio プロジェクトの作成
   - Microsoft Visual Studio で、**ファイル** > **新規作成** > **プロジェクト** をクリックします。
   - **新しいプロジェクト**ウィンドウが表示されます。左ウィンドウ枠で、*Visual Basic* フォルダまたは *Visual C#* フォルダを選択します。プロジェクトタイプとして、**スマートデバイス**をクリックします。
     右ウィンドウ枠で、**スマートデバイスプロジェクト**をクリックし、Microsoft Visual Basic と C# のどちらをプログラミング言語として使用するかに応じて、プロジェクトに VBApp または CSApp という名前を付けます。
   - `C:¥tutorial¥uldotnet` の場所を入力し、*OK* をクリックします。
   - ターゲットプラットフォームとして *Windows Mobile 5.0 Pocket PC SDK*、ターゲット .NET Compact Framework バージョンとして *.NET Compact Framework バージョン 3.5* をクリックします。*OK* をクリックします。
2. プロジェクトに参照を追加します。
   - Sap.Data.UltraLite アセンブリと、関連するリソースをプロジェクトに追加します。
     1. **プロジェクト**メニューで、**参照の追加**をクリックします。
     2. 利用可能な参照のリストで、*Sap.Data.UltraLite* と *Sap.Data.UltraLite EN* (英語の場合) をクリックします。*OK* をクリックして、選択済みコンポーネントのリストに追加します。
        使用する言語が英語でない場合は、**参照**をクリックして、SQL Anywhere インストール環境の `UltraLite¥UltraLite.NET¥ce¥Assembly¥v2¥xx` サブディレクトリで *Sap.Data.UltraLite xx* を見つけます。この場合、xx は、言語を表す 2 文字の略語 (英語の場合は *en*、など) です。
        `Sap.Data.UltraLite.resources.dll` をクリックして、**開く**をクリックします。
   - Ultra Light コンポーネントをプロジェクトにリンクします。
     ここで、リンクをコンポーネントに追加したことと、コンポーネントを開いていないことを確認します。
     1. **プロジェクト**メニューで、**既存項目の追加**をクリックし、SQL Anywhere インストール環境の `UltraLite¥UltraLite.NET¥ce` サブディレクトリを参照します。
     2. *Objects of Type* リストで、**実行ファイル**をクリックします。
     3. 使用している Windows Mobile デバイスのプロセッサに対応するフォルダを開きます。Microsoft Visual Studio 2005 以降の場合は、`arm.50` フォルダを開きます。`ulnet17.dll` をクリックし、**追加**ボタンの矢印をクリックして、**リンクとして追加**をクリックします。
3. アプリケーションのフォームを作成します。

   Microsoft Visual Studio のツールボックスパネルが表示されていない場合は、メインメニューで、**表示** > **ツールボックス** をクリックします。オブジェクトを選択して目的のロケーションのフォームにドラッグすることで、次のビジュアルコンポーネントをフォームに追加します。

タイプ	設計 - 名前
Button	btnInsert
Button	btnUpdate
Button	btnDelete
TextBox	txtName
ListBox	lbNames
Label	laName

   フォームは、次の図のようになります。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.



4. ソリューションをビルドおよび配置します。

   ソリューションをビルドおよび配置し、Microsoft Visual Studio プロジェクトが正しく設定されたことを確認します。

   a. ビルドメニューで、ソリューションのビルドをクリックします。プロジェクトが正常にビルドされたことを確認します。Microsoft Visual Basic アプリケーションをビルドしている場合、次の警告は無視してかまいません。

      ```
      Referenced assembly 'Sap.Data.UltraLite.resources' is a localized satellite
      assembly
      ```

   b. デバッグメニューで、デバッグの開始をクリックします。

      これによって、アプリケーションがモバイルデバイスまたはエミュレータに配置され、起動されます。アプリケーションがエミュレータまたはデバイスの `¥Program Files¥VBApp` または `¥Program Files¥CSApp` に配置されます(使用しているプロジェクト名による)。

      配置には時間がかかる場合があります。

   c. アプリケーションがエミュレータまたはターゲットデバイスに配置され、設計したフォーム (*Form1*) が正しく表示されることを確認します。

   d. エミュレータまたはターゲットデバイスのアプリケーションを終了します。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

結果

Ultra Light.NET API が新しい Windows Mobile アプリケーションで機能します。

次のステップ

次のレッスンに進みます。

タスクの概要: チュートリアル: Ultra Light.NET を使用した Windows Mobile アプリケーションの構築 [583 ページ]

次のタスク: レッスン 2: Ultra Light データベースの作成 [587 ページ]

2.3.2 レッスン 2: Ultra Light データベースの作成

このレッスンでは、デスクトップ PC の SQL Central を使用して Ultra Light データベースを作成します。

前提条件

このチュートリアルのこれまでのレッスンを完了している必要があります。

このチュートリアルの冒頭に記載されているロールと権限を持っている必要があります。

手順

1. スタート > プログラム > *SQL Anywhere17* > 管理ツール > *SQL Central* の順にクリックします。
2. SQL Central の Ultra Light プラグインを使用して、アプリケーションと同じディレクトリにデータベースを作成します。

   ツールメニューで、*Ultra Light 17* > データベースの作成 をクリックします。

   通常、SQL Central によって設定されるデフォルトのデータベース特性のままで問題ありません。特性は次のとおりです。

   データベースファイル名

   `c:¥tutorial¥uldotnet¥VBApp¥VBApp.udb` または `c:¥tutorial¥uldotnet¥CSApp¥CSApp.udb` (アプリケーションタイプによって異なります)。

   DBA ユーザ ID とパスワード

   このマニュアルでは、例としてそれぞれ DBA と sql に設定します。

   照合順

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

デフォルトの照合を使用します。
文字列の比較で大文字と小文字を区別する

このオプションはオフにしてください。

完了をクリックして、Ultra Light データベースに接続します。

3. SQL Central ツリービューのテーブルフォルダアイコンを強調表示してから、ファイル > 新規 > テーブル をクリックして、新しい Ultra Light テーブルを作成します。特性は次のとおりです。

   テーブル名

   Names と入力します。

   カラム

   次の属性を持つカラムを **Names** テーブルに作成します。

カラム名	データ型 (サイズ)	NULL	ユニーク	デフォルト値
ID	整数	いいえ	はい (プライマリキー)	グローバルオートインクリメント
名前	Varchar(30)	いいえ	いいえ	なし

   プライマリキー

   ID カラムをプライマリキーとして指定します。

4. SQL Central を終了し、データベースファイルが指定のディレクトリに作成されていることを確認します。
5. 初期化済みの (空の) データベースファイルを Microsoft Visual Studio プロジェクトにリンクし、データベースファイルがアプリケーションコードとともにデバイスに配備されるようにします。
   - *Visual Studio* メニューで、プロジェクト > 既存項目の追加 をクリックします。
   - *Objects of Type* がすべてのファイルに設定されていることを確認します。データベースファイルを作成したディレクトリを参照し、アプリケーションタイプに応じてファイル `VBApp.udb` または `CSApp.udb` をクリックします。
   - 追加ボタンの矢印をクリックし、リンクとして追加をクリックします。
   - Solution Explorer フレームで、プロジェクトに追加したデータベースファイル名を右クリックし、プロパティをクリックします。
     プロパティのパネルで、ビルドアクションプロパティをコンテンツに設定し、出力ディレクトリにコピープロパティを常にコピーするに設定します。

結果

Ultra Light データベースが作成されます。

次のステップ

次のレッスンに進みます。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

タスクの概要: チュートリアル: Ultra Light.NET を使用した Windows Mobile アプリケーションの構築 [583 ページ]

前のタスク: レッスン 1: Microsoft Visual Studio プロジェクトの作成 [584 ページ]

次のタスク: レッスン 3: アプリケーションへのデータベース接続コントロールの追加 [589 ページ]

関連情報

データベース作成ウィザードでの Ultra Light データベースの作成 [28 ページ]

2.3.3 レッスン 3: アプリケーションへのデータベース接続コントロールの追加

このレッスンでは、Ultra Light データベースへの接続を確立する Ultra Light.NET アプリケーションにコントロールを追加します。

前提条件

このチュートリアルのこれまでのレッスンを完了している必要があります。

このチュートリアルの冒頭に記載されているロールと権限を持っている必要があります。

手順

1. フォームをダブルクリックして、ソースファイル (`Form1.cs` または `Form1.vb`) を開きます。
2. コードを追加して、Sap.Data.UltraLite ネームスペースをインポートします。

   ファイルの先頭行に次の文を追加します。

   ```
   //Microsoft Visual C#
   using Sap.Data.UltraLite;
   ```

   ```
   'Microsoft Visual Basic
   Imports Sap.Data.UltraLite
   ```
3. グローバル変数をフォーム宣言に追加します。

   Microsoft Visual C# の場合は、フォームコンポーネントを記述したコードと、最初のメソッド宣言までの間に、次のコードを追加します。

   ```
   //Microsoft Visual C#
   ```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

```
private ULConnection Conn;
private int[] ids;
```

Microsoft Visual Basic の場合は、Form1 クラスの先頭に次のコードを追加します。

```
'Microsoft Visual Basic
Dim Conn As ULConnection
Dim ids() As Integer
```

これらの変数は次のように使用します。

ULConnection

Connection オブジェクトは、データベースへの接続時に実行されるすべてのアクションのルートオブジェクトです。

ids

ids 配列を使用して、クエリの実行後に返される ID カラム値を格納します。

ListBox コントロール自体によって順序番号にアクセスできますが、ローが削除されると、これらの番号は ID カラムの値とは異なります。このため、ID カラム値を個別に格納してください。

4. フォームの空白領域をダブルクリックして、Form1_Load メソッドを作成します。

   このメソッドでは次のタスクを実行します。

   - ulConnectionParms1 コントロールで設定されている接続パラメータを使用して、データベースへの接続を開きます。
   - RefreshListBox メソッドを呼び出します (このチュートリアルの後半で定義)。
   - エラーが発生したら、エラーメッセージを印刷 (表示) します。SQL Anywhere エラーの場合は、エラーコードも出力されます。

   C# の場合は、次のコードを Form1_Load メソッドに追加します。

```
//Microsoft Visual C#
try {
    String ConnString = "dbf=¥¥Program Files¥¥CSApp¥¥CSApp.udb";
    Conn = new ULConnection( ConnString );
    Conn.Open();
    Conn.DatabaseID = 1;
    RefreshListBox();
}
catch ( System.Exception t ) {
    MessageBox.Show( "Exception: " + t.Message);
}
```

   Microsoft Visual Basic の場合は、次のコードを Form1_Load メソッドに追加します。

```
'Microsoft Visual Basic
Try
    Dim ConnString as String = "dbf=¥Program Files¥VBApp¥VBApp.udb"
    Conn = New ULConnection( ConnString )
    Conn.Open()
    Conn.DatabaseID = 1
    RefreshListBox()
Catch
    MsgBox("Exception: " + err.Description)
End Try
```

5. プロジェクトをビルドします。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

ビルドメニューで、ソリューションのビルドをクリックします。この段階で、単一のエラーレポートを受け取ることがあります。たとえば、C# では、RefreshListBox がまだ宣言されていないため、`error CS0103: The name 'RefreshListBox' does not exist in the current context.` が表示されます。次のレッスンではこの関数を追加します。

他のエラーが表示された場合は、先へ進む前にエラーを修正します。C# における大文字と小文字の違いなどの一般的なエラーをチェックします。たとえば、Ultra Light、ULConnection は大文字と小文字が正確に一致している必要があります。Microsoft Visual Basic では、レッスン 3 で説明した Imports Sap.Data.UltraLite 文を含める必要があります。

結果

アプリケーションが設定され、Ultra Light データベースに接続されます。

タスクの概要: チュートリアル: Ultra Light.NET を使用した Windows Mobile アプリケーションの構築 [583 ページ]

前のタスク: レッスン 2: Ultra Light データベースの作成 [587 ページ]

次のタスク: レッスン 4: データの挿入、更新、削除 [591 ページ]

2.3.4 レッスン 4: データの挿入、更新、削除

このレッスンでは、動的 SQL を使用するアプリケーションにコードを追加し、データベースのデータを変更します。

前提条件

このチュートリアルのこれまでのレッスンを完了している必要があります。

このチュートリアルの冒頭に記載されているロールと権限を持っている必要があります。

コンテキスト

このレッスンでは、リストボックスを管理するサポートメソッドを作成します。この手順は、以降の手順で使用されるデータ操作メソッドに必要です。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

手順

1. フォームを右クリックし、コードの表示をクリックします。
2. Form1 クラスのメソッドを追加し、リストボックスを更新してデータを移植します。このメソッドでは、次のタスクを実行します。
   - リストボックスをクリアします。
   - ULCommand オブジェクトをインスタンス化し、データベース内の Names テーブルのデータを返す SELECT クエリに割り当てます。
   - クエリを実行し、ULDataReader として結果セットを返します。
   - 結果セットの行数と同じ長さの整数配列をインスタンス化します。
   - ULDataReader で返される名前をリストボックスに移植し、ULDataReader で返される ID を整数配列に移植します。
   - ULDataReader を閉じます。
   - エラーが発生した場合は、エラーメッセージが出力されます。SQL エラーの場合は、エラーコードも出力されます。

   C# の場合は、Form1 クラスのメソッドとして次のコードをアプリケーションに追加します。

```
//Microsoft Visual C#
private void RefreshListBox(){
    try{
        long NumRows;
        int I = 0;
        lbNames.Items.Clear();
        using( ULCommand cmd = Conn.CreateCommand() ){
            cmd.CommandText = "SELECT ID, Name FROM Names";
            using( ULDataReader dr = cmd.ExecuteReader()){
                dr.MoveBeforeFirst();
                NumRows = dr.RowCount;
                ids = new int[ NumRows ];
                while (dr.MoveNext())
                {
                    lbNames.Items.Add(
                    dr.GetString(1));
                    ids[ I ] = dr.GetInt32(0);
                    I++;
                }
            }
            txtName.Text = " ";
        }
    }
    catch( Exception err ){
        MessageBox.Show(
        "Exception in RefreshListBox: " + err.Message );
    }
}
```

   Microsoft Visual Basic の場合は、Form1 クラスのメソッドとして次のコードをアプリケーションに追加します。

```
'Microsoft Visual Basic
Private Sub RefreshListBox()
    Try
        Dim cmd As ULCommand = Conn.CreateCommand()
        Dim I As Integer = 0
        lbNames.Items.Clear()
        cmd.CommandText = "SELECT ID, Name FROM Names"
        Dim dr As ULDataReader = cmd.ExecuteReader()
        ReDim ids(dr.RowCount)
        While (dr.MoveNext)
            lbNames.Items.Add(dr.GetString(1))
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

```
            ids(I) = dr.GetInt32(0)
            I = I + 1
        End While
        dr.Close()
        txtName.Text = " "
    Catch ex As Exception
        MsgBox(ex.ToString)
    End Try
End Sub
```

3. プロジェクトをビルドします。

   プロジェクトのビルドによってエラーが発生しないようにします。

4. [フォームデザイン] タブで、*Insert* をダブルクリックして、btnInsert_Click メソッドを作成します。このメソッドでは、次のタスクを実行します。
   - ULCommand オブジェクトをインスタンス化し、テキストボックス内の値をデータベースに挿入する INSERT 文に割り当てます。
   - 文を実行します。
   - ULCommand オブジェクトを破棄します。
   - リストボックスを再表示します。
   - エラーが発生した場合は、エラーメッセージが出力されます。SQL エラーの場合は、エラーコードも出力されます。

   Microsoft C# の場合は、次のコードを btnInsert_Click メソッドに追加します。

```
//Microsoft Visual C#
try {
    long RowsInserted;
    using( ULCommand cmd = Conn.CreateCommand() ) {
        cmd.CommandText =
            "INSERT INTO Names(name) VALUES (?)";
        cmd.Parameters.Add("", txtName.Text);
        RowsInserted = cmd.ExecuteNonQuery();
    }
    RefreshListBox();
}
catch( Exception err ) {
    MessageBox.Show("Exception: " + err.Message );
}
```

   Microsoft Visual Basic の場合は、次のコードを btnInsert_Click メソッドに追加します。

```
'Microsoft Visual Basic
Try
    Dim RowsInserted As Long
    Dim cmd As ULCommand = Conn.CreateCommand()
    cmd.CommandText = "INSERT INTO Names(name) VALUES (?)"
    cmd.Parameters.Add("", txtName.Text)
    RowsInserted = cmd.ExecuteNonQuery()
    cmd.Dispose()
    RefreshListBox()
Catch
    MsgBox("Exception: " + Err.Description)
End Try
```

5. [フォームデザイン] タブで、*Update* をダブルクリックして、btnUpdate_Click メソッドを作成します。このメソッドでは、次のタスクを実行します。
   - ULCommand オブジェクトをインスタンス化し、関連 ID に基づいて、テキストボックス内の値をデータベースに挿入する UPDATE 文に割り当てます。
   - 文を実行します。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

- ULCommand オブジェクトを破棄します。
- リストボックスを再表示します。
- エラーが発生した場合は、エラーメッセージが出力されます。SQL エラーの場合は、エラーコードも出力されます。

Microsoft C# の場合は、次のコードを btnUpdate_Click メソッドに追加します。

```
//Microsoft Visual C#
try {
    long RowsUpdated;
    int updateID = ids[ lbNames.SelectedIndex ];
    using( ULCommand cmd = Conn.CreateCommand() ){
        cmd.CommandText =
            "UPDATE Names SET name = ? WHERE id = ?" ;
        cmd.Parameters.Add("", txtName.Text );
        cmd.Parameters.Add("", updateID);
        RowsUpdated = cmd.ExecuteNonQuery();
    }
    RefreshListBox();
}
catch( Exception err ) {
    MessageBox.Show(
        "Exception: " + err.Message);
}
```

Microsoft Visual Basic の場合は、次のコードを btnUpdate_Click メソッドに追加します。

```
'Microsoft Visual Basic
Try
    Dim RowsUpdated As Long
    Dim updateID As Integer = ids(lbNames.SelectedIndex)
    Dim cmd As ULCommand = Conn.CreateCommand()
    cmd.CommandText = "UPDATE Names SET name = ? WHERE id = ?"
    cmd.Parameters.Add("", txtName.Text)
    cmd.Parameters.Add("", updateID)
    RowsUpdated = cmd.ExecuteNonQuery()
    cmd.Dispose()
    RefreshListBox()
Catch
    MsgBox("Exception: " + Err.Description)
End Try
```

6. [フォームデザイン] タブで、*Delete* をダブルクリックして、btnDelete_Click メソッドを作成します。コードを追加し、次のタスクを実行します。
   - ULCommand オブジェクトをインスタンス化し、DELETE 文に割り当てます。DELETE 文は、整数配列 ids からの関連 ID に基づいて、選択した行をデータベースから削除します。
   - 文を実行します。
   - ULCommand オブジェクトを破棄します。
   - リストボックスを再表示します。
   - エラーが発生した場合は、エラーメッセージが表示されます。SQL エラーの場合は、エラーコードも表示されます。

   Microsoft C# の場合は、次のコードを btnDelete_Click メソッドに追加します。

```
//Microsoft Visual C#
try{
    long RowsDeleted;
    int deleteID = ids[lbNames.SelectedIndex];
    using( ULCommand cmd = Conn.CreateCommand() ){
        cmd.CommandText =
            "DELETE From Names WHERE id = ?" ;
        cmd.Parameters.Add("", deleteID);
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

```
            RowsDeleted = cmd.ExecuteNonQuery ();
        }
        RefreshListBox();
    }
    catch( Exception err ) {
        MessageBox.Show("Exception: " + err.Message );
    }
```

Microsoft Visual Basic の場合は、次のコードを btnDelete_Click メソッドに追加します。

```
'Microsoft Visual Basic
Try
    Dim RowsDeleted As Long
    Dim deleteID As Integer = ids(lbNames.SelectedIndex)
    Dim cmd As ULCommand = Conn.CreateCommand()
    cmd.CommandText = "DELETE From Names WHERE id = ?"
    cmd.Parameters.Add("", deleteID)
    RowsDeleted = cmd.ExecuteNonQuery()
    cmd.Dispose()
    RefreshListBox()
Catch
    MsgBox("Exception: " + Err.Description)
End Try
```

7. アプリケーションをビルドし、正常にコンパイルされることを確認します。

結果

Windows Mobile アプリケーションが設定され、Ultra Light データベースに対するデータ操作を実行できます。

次のステップ

次のレッスンに進みます。

タスクの概要: チュートリアル: Ultra Light.NET を使用した Windows Mobile アプリケーションの構築 [583 ページ]

前のタスク: レッスン 3: アプリケーションへのデータベース接続コントロールの追加 [589 ページ]

次のタスク: レッスン 5: アプリケーションの構築と配備 [596 ページ]

関連情報

Ultra Light.NET での ULTable クラスを使用したデータの作成と修正 [568 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

2.3.5 レッスン 5: アプリケーションの構築と配備

このレッスンでは、アプリケーションを構築し、リモートデバイスまたはエミュレータに配備します。

前提条件

このチュートリアルのこれまでのレッスンを完了している必要があります。

このチュートリアルの冒頭に一覧されているロールと権限を持っている必要があります。

手順

1. ソリューションをビルドします。

   アプリケーションがエラーなくビルドされることを確認します。
2. 配置ターゲットを選択します。

   配置ターゲットは、アプリケーションに含める `ulnet17.dll` のバージョンと一致している必要があります。
3. ▶ デバッグ ▶ 開始 をクリックします。

   これにより、アプリケーションが格納された実行可能ファイルがビルドされ、エミュレータに配置されます。このプロセスには時間がかかることがあります。アプリケーションを実行する前に .NET Compact Framework を配置する必要がある場合は、特に時間がかかります。
4. エラーがレポートされた場合は、次のチェックリストを使用して、配置が正常に完了したかどうかをチェックします。
   - アプリケーションが `¥Program Files¥appname` に配置されていることを確認します。`appname` は、レッスン 1 でアプリケーションに付けた名前 (CSApp または VBApp) です。
   - アプリケーションコード内のデータベースファイルへのパスが正しいことを確認します。
   - データベースファイルをプロジェクトに追加するときに [リンクファイル] を選択し、[ビルドアクション] を [コンテンツ] に設定し、[出力ディレクトリにコピー] を [常にコピーする] に設定したことを確認します。これらのオプションを適切に設定しなかった場合は、ファイルがデバイスに配置されません。
   - ターゲットプラットフォームに対して適切なバージョンの `ulnet17.dll` への参照を追加したこと、または Windows Mobile インストーラを実行したことを確認します。Windows Mobile 5.0 より前のバージョンの Windows Mobile で、エミュレータと実際のデバイスの切り替えを行う場合は、使用するライブラリのバージョンを変更する必要があります。
   - エミュレータの状態を保存しないでエミュレータを終了できます。アプリケーションの再配置を行うと、必要なすべてのファイルがエミュレータにコピーされ、バージョン上の問題がないことが確認されます。
5. アプリケーションのテストを行います。
   a. データベースにデータを挿入します。

      テキストボックスに名前を入力し、*Insert* をクリックします。名前がリストボックスに表示されます。
   b. データベース内のデータを更新します。

      リストボックスで名前をクリックします。テキストボックスに新しい名前を入力します。*Update* をクリックします。

      新しい名前が、リストボックスの古い名前の代わりに表示されます。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

c. データベースからデータを削除します。

リストで名前をクリックします。*Delete* をクリックします。

名前はリストに表示されなくなります。

結果

アプリケーションのテストが終了し、配備できるようになります。

タスクの概要: チュートリアル: Ultra Light.NET を使用した Windows Mobile アプリケーションの構築 [583 ページ]

前のタスク: レッスン 4: データの挿入、更新、削除 [591 ページ]

次のトピック: C# チュートリアルのコードリスト [597 ページ]

2.3.6 C# チュートリアルのコードリスト

この章で説明したチュートリアルプログラムの完全なコードを次に示します。

```
using Sap.Data.UltraLite;
using System;
using System.Linq;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Text;
using System.Windows.Forms;
namespace CSApp
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }
        private ULConnection Conn;
        private int[] ids;
        private void Form1_Load(object sender, EventArgs e)
        {
            try
            {
                String ConnString = "dbf=¥¥Program Files¥¥CSApp¥¥CSApp.udb";
                Conn = new ULConnection(ConnString);
                Conn.Open();
                Conn.DatabaseID = 1;
                RefreshListBox();
            }
            catch (System.Exception t)
            {
                MessageBox.Show("Exception: " + t.Message);
            }
        }
```


© 2016 SAP SE or an SAP affiliate company. All rights reserved.

```
        private void RefreshListBox()
        {
            try
            {
                long NumRows;
                int I = 0;
                lbNames.Items.Clear();
                using (ULCommand cmd = Conn.CreateCommand())
                {
                    cmd.CommandText = "SELECT ID, Name FROM Names";
                    using (ULDataReader dr = cmd.ExecuteReader())
                    {
                        dr.MoveBeforeFirst();
                        NumRows = dr.RowCount;
                        ids = new int[NumRows];
                        while (dr.MoveNext())
                        {
                            lbNames.Items.Add(
                            dr.GetString(1));
                            ids[i] = dr.GetInt32(0);
                            I++;
                        }
                    }
                    txtName.Text = " ";
                }
            }
            catch (Exception err)
            {
                MessageBox.Show(
                "Exception in RefreshListBox: " + err.Message);
            }
        }
        private void btnInsert_Click(object sender, EventArgs e)
        {
            try
            {
                long RowsInserted;
                using (ULCommand cmd = Conn.CreateCommand())
                {
                    cmd.CommandText =
                        "INSERT INTO Names(name) VALUES (?)";
                    cmd.Parameters.Add("", txtName.Text);
                    RowsInserted = cmd.ExecuteNonQuery();
                }
                RefreshListBox();
            }
            catch (Exception err)
            {
                MessageBox.Show("Exception: " + err.Message);
            }
        }
        private void btnUpdate_Click(object sender, EventArgs e)
        {
            try
            {
                long RowsUpdated;
                int updateID = ids[lbNames.SelectedIndex];
                using (ULCommand cmd = Conn.CreateCommand())
                {
                    cmd.CommandText =
                        "UPDATE Names SET name = ? WHERE id = ?";
                    cmd.Parameters.Add("", txtName.Text);
                    cmd.Parameters.Add("", updateID);
                    RowsUpdated = cmd.ExecuteNonQuery();
                }
                RefreshListBox();
            }
            catch (Exception err)
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

```
            {
                MessageBox.Show(
                    "Exception: " + err.Message);
            }
        }
        private void btnDelete_Click(object sender, EventArgs e)
        {
            try
            {
                long RowsDeleted;
                int deleteID = ids[lbNames.SelectedIndex];
                using (ULCommand cmd = Conn.CreateCommand())
                {
                    cmd.CommandText =
                        "DELETE From Names WHERE id = ?";
                    cmd.Parameters.Add("", deleteID);
                    RowsDeleted = cmd.ExecuteNonQuery();
                }
                RefreshListBox();
            }
            catch (Exception err)
            {
                MessageBox.Show("Exception: " + err.Message);
            }
        }
    }
}
```

親トピック チュートリアル: Ultra Light.NET を使用した Windows Mobile アプリケーションの構築 [583 ページ]

前のタスク: レッスン 5: アプリケーションの構築と配備 [596 ページ]

次のトピック: Microsoft Visual Basic チュートリアルのコードリスト [599 ページ]

2.3.7 Microsoft Visual Basic チュートリアルのコードリスト

この章で説明したチュートリアルプログラムの完全なコードを次に示します。

```
Imports Sap.Data.UltraLite
Public Class Form1
    Dim Conn As ULConnection
    Dim ids() As Integer
    Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As
      System.EventArgs) Handles MyBase.Load
        Try
            Dim ConnString As String = "dbf=¥Program Files¥VBApp¥VBApp.udb"
            Conn = New ULConnection(ConnString)
            Conn.Open()
            Conn.DatabaseID = 1
            RefreshListBox()
        Catch
            MsgBox("Exception: " + Err.Description)
        End Try
    End Sub
    Private Sub RefreshListBox()
        Try
            Dim cmd As ULCommand = Conn.CreateCommand()
            Dim I As Integer = 0
            lbNames.Items.Clear()
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

```
            cmd.CommandText = "SELECT ID, Name FROM Names"
            Dim dr As ULDataReader = cmd.ExecuteReader()
            ReDim ids(dr.RowCount)
            While (dr.MoveNext)
                lbNames.Items.Add(dr.GetString(1))
                ids(I) = dr.GetInt32(0)
                I = I + 1
            End While
            dr.Close()
            txtName.Text = " "
        Catch ex As Exception
            MsgBox(ex.ToString)
        End Try
    End Sub
    Private Sub btnInsert_Click(ByVal sender As System.Object, ByVal e As
      System.EventArgs) Handles btnInsert.Click
        Try
            Dim RowsInserted As Long
            Dim cmd As ULCommand = Conn.CreateCommand()
            cmd.CommandText = "INSERT INTO Names(name) VALUES (?)"
            cmd.Parameters.Add("", txtName.Text)
            RowsInserted = cmd.ExecuteNonQuery()
            cmd.Dispose()
            RefreshListBox()
        Catch
            MsgBox("Exception: " + Err.Description)
        End Try
    End Sub
    Private Sub btnUpdate_Click(ByVal sender As System.Object, ByVal e As
      System.EventArgs) Handles btnUpdate.Click
        Try
            Dim RowsUpdated As Long
            Dim updateID As Integer = ids(lbNames.SelectedIndex)
            Dim cmd As ULCommand = Conn.CreateCommand()
            cmd.CommandText = "UPDATE Names SET name = ? WHERE id = ?"
            cmd.Parameters.Add("", txtName.Text)
            cmd.Parameters.Add("", updateID)
            RowsUpdated = cmd.ExecuteNonQuery()
            cmd.Dispose()
            RefreshListBox()
        Catch
            MsgBox("Exception: " + Err.Description)
        End Try
    End Sub
    Private Sub btnDelete_Click(ByVal sender As System.Object, ByVal e As
      System.EventArgs) Handles btnDelete.Click
        Try
            Dim RowsDeleted As Long
            Dim deleteID As Integer = ids(lbNames.SelectedIndex)
            Dim cmd As ULCommand = Conn.CreateCommand()
            cmd.CommandText = "DELETE From Names WHERE id = ?"
            cmd.Parameters.Add("", deleteID)
            RowsDeleted = cmd.ExecuteNonQuery()
            cmd.Dispose()
            RefreshListBox()
        Catch
            MsgBox("Exception: " + Err.Description)
        End Try
    End Sub
End Class
```

親トピック チュートリアル: Ultra Light.NET を使用した Windows Mobile アプリケーションの構築 [583 ページ]

前のトピック: C# チュートリアルのコードリスト [597 ページ]

 © 2016 SAP SE or an SAP affiliate company.All rights reserved.

2.4 Ultra Light .NET API リファレンス

.NET Framework 2.0 および .NET Compact Framework 2.0 用 Ultra Light.NET データプロバイダの API を使用して、Ultra Light の.NET アプリケーションを作成することができます。

SQL Anywhere の ADO.NET 用データプロバイダで利用できない Ultra Light.NET 拡張については、この API リファレンスでは **UL 拡張:** と示します。

Ultra Light.NET の Ultra Light エンジンのランタイムを使用するには、RuntimeType プロパティを適切な値に設定してから、他の Ultra Light.NET API を使用します。

アセンブリは、Sap.Data.UltraLite.resources という名前のサテライトリソースアセンブリを使用します。メインアセンブリは、このリソースアセンブリを次の順序で国別に検索します。

- CultureInfo.CurrentUICulture
- CultureInfo.CurrentCulture
- EN

Sap.Data.UltraLite ネームスペースでよく使用される高レベルクラスのいくつかについて次のリストにまとめて示します。

ULConnection

各 ULConnection オブジェクトは、Ultra Light データベースとの接続を表します。ULConnection オブジェクトは 1 つまたは複数作成できます。

ULTable

各 ULTable オブジェクトを使用して、単一テーブル内のデータにアクセスできます。

ULCommand オブジェクト

各 ULCommand オブジェクトには、データベースに対して実行される SQL 文が格納されます。

ULDataReader オブジェクト

各 ULDataReader オブジェクトには、単一のクエリの結果セットが格納されます。

ULSyncParms

ULSyncParms オブジェクトを使用して、Ultra Light データベースを Mobile Link サーバと同期させます。

この章の多くのプロパティとメソッドは、.NET Framework Data Provider for OLE DB (System.Data.OleDb) とよく似ています。詳細と例については、Microsoft .NET Framework のマニュアルを参照してください。

ネームスペース

```
Sap.Data.UltraLite
```

> **注記**
>
> API リファレンスマニュアルをお探しですか。マニュアルをローカルにインストールした場合は、Windows のスタートメニューを使用してアクセスするか (Microsoft Windows)、`C:¥Program Files¥SQL Anywhere 17¥Documentation` にナビゲートします。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

また、DocCommentXchange の Web で、SAP SQL Anywhere API リファレンスマニュアルにアクセスすることもできます。http://dcx.sap.com

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

3 Ultra Light - C++ プログラミング

このマニュアルでは、Ultra Light C++ のプログラミングインタフェースについて説明します。Ultra Light を使用すると、ハンドヘルドデバイス、モバイルデバイス (Apple の iPhone や iPad、Google Android など)、埋め込みデバイス用にデータベースアプリケーションを開発し、これらのデバイスに配備することができます。

このセクションの内容:

システムの要件とサポート対象プラットフォーム [603 ページ]
Ultra Light C++ は、Microsoft Windows Mobile、Linux、Apple iOS などさまざまなプラットフォームをサポートしています。Ultra Light データベースの開発には、サードパーティソフトウェアが必要です。

アプリケーション開発 [604 ページ]
C++ インタフェースによって、Ultra Light 開発者に以下の利点が提供されます。

チュートリアル:C++ API を使用した Windows アプリケーションの構築 [675 ページ]
このチュートリアルでは、Ultra Light C++ アプリケーションを開発するプロセスを説明します。アプリケーションは Windows デスクトップオペレーティングシステム用に開発され、コマンドプロンプトで実行されます。

API リファレンス [686 ページ]
Ultra Light C++ API を使用してモバイルアプリケーションを開発します。

3.1 システムの要件とサポート対象プラットフォーム

Ultra Light C++ は、Microsoft Windows Mobile、Linux、Apple iOS などさまざまなプラットフォームをサポートしています。Ultra Light データベースの開発には、サードパーティソフトウェアが必要です。

開発プラットフォーム

Ultra Light C++ を使用してアプリケーションを開発するには、以下が必要です。

- Microsoft Windows、Linux、または Mac デスクトップ (開発プラットフォーム)
- サポートされている Microsoft または GNU の C++ コンパイラ

ターゲットプラットフォーム

Ultra Light C++ は、次のターゲットプラットフォームをサポートしています。

- Apple iOS

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

- Apple Mac OS X
- Linux
- Embedded Linux
- Microsoft Windows
- Microsoft Windows Mobile

プラットフォームバージョンに関する最新情報については、http://scn.sap.com/docs/DOC-35654#UL を参照してください。

3.2 アプリケーション開発

C++ インタフェースによって、Ultra Light 開発者に以下の利点が提供されます。

- 専有容量が小さく、高パフォーマンスで、ネイティブな同期を実装したデータベースストア
- C++ 言語の優れた機能、効率、柔軟性
- アプリケーションを Microsoft Windows Mobile、Microsoft Windows デスクトッププラットフォーム、Linux、および Apple iOS に展開する機能

すべての Ultra Light C++ インタフェースは、同じ Ultra Light ランタイムエンジンを使用します。したがって、各 API は同じ基本機能へのアクセスを提供します。

このセクションの内容:

Ultra Light C++ アプリケーション開発 [604 ページ]
Ultra Light C++ は、データベース機能を備えており、Windows Mobile デバイス、Windows デスクトッププラットフォーム、Linux デスクトップ、Embedded Linux、Apple iOS デバイスとの同期が可能です。

Embedded SQL を使用した Ultra Light C++ アプリケーション開発 [635 ページ]
Embedded SQL Ultra Light アプリケーションのデータベースアクセスコードを作成できます。

Windows Mobile 向け Ultra Light アプリケーション開発 [667 ページ]
Microsoft Visual Studio 2005 以降は、Windows Mobile 環境用のアプリケーション開発に使用できます。

3.2.1 Ultra Light C++ アプリケーション開発

Ultra Light C++ は、データベース機能を備えており、Windows Mobile デバイス、Windows デスクトッププラットフォーム、Linux デスクトップ、Embedded Linux、Apple iOS デバイスとの同期が可能です。

このセクションの内容:

Ultra Light C++ アプリケーション開発のクイックスタートガイド [605 ページ]
Ultra Light C++ の開発には、アプリケーションで ULDatabaseManager オブジェクトを使用します。

Apple iOS と Mac OS X での考慮事項 [606 ページ]
Apple iOS または Mac OS X プラットフォームを開発する場合、設計および開発に関していくつかの決定を行う必要があります。

 © 2016 SAP SE or an SAP affiliate company.All rights reserved.

Ultra Light データベースへの接続 [607 ページ]
Ultra Light アプリケーションをデータベースに接続しないと、データを操作できません。

Ultra Light C++ での SQL 文を使用したデータの作成と修正 [609 ページ]
Ultra Light アプリケーションは、SQL 文を実行するか ULTable クラスを使用してテーブルデータにアクセスできます。

Ultra Light C++ での ULTable クラスを使用したデータの作成と修正 [617 ページ]
Ultra Light アプリケーションは、SQL 文を実行するか ULTable クラスを使用してテーブルデータにアクセスできます。

Ultra Light C++ でのトランザクションの管理 [624 ページ]
トランザクションは、データベースを変更するために最初の文によって暗黙的に開始されますが、明示的にコミットまたはロールバックする必要があります。

Ultra Light C++ のスキーマ情報 [624 ページ]
結果セットまたはデータベース構造の定義は、プログラムを使用して取得できます。これらの記述はスキーマ情報と呼ばれます。この情報は、Ultra Light C API スキーマクラスから使用できます。

エラー処理 [625 ページ]
Ultra Light C++ API には、エラー情報の取得に使用する ULError オブジェクトが含まれています。API のさまざまなメソッドが、メソッド呼び出しが成功したかどうかを示すブール値を返します。場合によっては、エラーが発生したときに NULL が返されます。

Ultra Light C++ での Mobile Link データの同期 [625 ページ]
Ultra Light アプリケーションでは、データを中央のデータベースに同期できます。同期には、SQL Anywhere に付属の Mobile Link 同期ソフトウェアが必要です。

Ultra Light データベース接続の切断 [626 ページ]
使用しなくなったソフトウェアリソースを開放し、アプリケーションがデータベースに接続されているあいだ Ultra Light データベースファイルが使用中のままにならないようにします。

Ultra Light C++ アプリケーションの構築と配備 [626 ページ]
Ultra Light エンジンを使用しない C++ アプリケーションを作成する場合、静的な Ultra Light ランタイムライブラリにリンクさせるか、または Windows および Windows Mobile でインポートライブラリにリンクさせて、アプリケーションの起動時に Ultra Light ランタイムコードを動的にロードすることができます。

3.2.1.1 Ultra Light C++ アプリケーション開発のクイックスタートガイド

Ultra Light C++ の開発には、アプリケーションで ULDatabaseManager オブジェクトを使用します。

以下の手順は、通常、Ultra Light C++ API を使用したアプリケーションを作成するときに使用されます。

1. ULDatabaseManager オブジェクトを初期化します。
2. (省略可) Ultra Light ランタイムライブラリの機能を有効にします。
3. Ultra Light データベースを使用します。既存のデータベースへの接続の確立、新しいデータベースの作成、既存のデータベースの削除、または既存のデータベースのファイルに破損がないかどうかの検証を実行できます。
4. ULDatabaseManager オブジェクトをファイナライズします。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

ULDatabaseManager オブジェクトは、アプリケーション内で 1 回のみ初期化し、アプリケーションが終了するときにファイナライズしてください。ULDatabaseManager クラスのすべてのメソッドは静的です。Ultra Light アプリケーションでのエラー情報を取得するには、ULError クラスを使用します。

関連情報

Ultra Light C++ アプリケーションの構築と配備 [626 ページ]

3.2.1.2 Apple iOS と Mac OS X での考慮事項

Apple iOS または Mac OS X プラットフォームを開発する場合、設計および開発に関していくつかの決定を行う必要があります。

開発環境

Apple iOS と Mac OS X の開発環境は Xcode です。

ビルド設定

Ultra Light のヘッダファイルとライブラリを参照するには、SQL Anywhere インストールフォルダの場所にユーザ定義のビルド設定セットを作成すると便利です。たとえば、`SQLANY_ROOT` を `/Applications/SQLAnywhere17` に設定します。この設定を作成するには、プロジェクトエディタのビルドウィンドウ枠を開き、ユーザ定義の設定を追加をクリックして、名前と値を入力します。

インクルードファイル

Ultra Light のインクルードファイルが検索されるようにするには、`$(SQLANY_ROOT)/sdk/include` を、ユーザヘッダ検索パスの `(USER_HEADER_SEARCH_PATHS)` ビルド設定に追加します。

サポートされない Mobile Link クライアントネットワークプロトコルオプション

Apple iOS 用、Mac OS X 用、またはその両方の Ultra Light は、以下の Mobile Link クライアントネットワークプロトコルオプションをサポートしません。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

- certificate_company
- certificate_unit
- client_port
- identity
- identity_password
- network_leave_open
- network_name

暗号化

FIPS 認定の標準暗号化方式はサポートされていません。

Mobile Link サーバを使用して Apple iOS と Mac OS X の Ultra Light クライアントを同期させるとき、エンドツーエンド暗号化を使用するには、(PEM パブリックキーファイルではなく) パブリックキーを PEM でエンコードされた X509 証明書にカプセル化する必要があります。E2EE プライベートキーを使用して PEM でエンコードされた X509 証明書を作成する場合は、証明書作成ユーティリティ createcert を使用してください。

iPhone 向けの Ultra Light アプリケーションを開発する場合は、Xcode プロジェクトの Resources フォルダに証明書を含める必要があります。trusted_certificates または e2ee_public_key オプションが割り当てられている場合には、Ultra Light 同期ロジックは、iPhone 開発パッケージのメインリソースバンドル (mainBundle) の証明書を検索します。

iPhone アプリケーションのデバッグ

Xcode デバッガ (GDB) では、ステップスルーと `longjmp()` 呼び出しでのブレークがサポートされています。通常、アプリケーションでは `longjmp` を使用しませんが、Ultra Light ランタイムライブラリでは内部的に使用されます (エラーが通知されたときなど)。これにより、アプリケーションコードのトレース中に Ultra Light 呼び出しが出現した場合に、問題が発生することがあります。Ultra Light 呼び出しのステップオーバで、デバッガからエラーを受け取った場合は、プログラムを再起動して、問題のある行の後にブレークポイントを設定し、問題のある行をステップオーバする代わりに *Continue* コマンドを使用します。デバッガは次のブレークポイントで停止しますが、`longjmp` 呼び出しに関連する問題を回避ができるため、同じ効果があります。この問題が発生する可能性が最も高いのは、OpenConnection を使用して既存のデータベースを開く場合、またはデータベースが存在しないかどうかを判別する場合です (データベースが存在しない場合、エラーが通知されます)。

3.2.1.3 Ultra Light データベースへの接続

Ultra Light アプリケーションをデータベースに接続しないと、データを操作できません。

ULDatabaseManager クラスは、データベースへの接続を開くために使用されます。ULDatabaseManager クラスは、接続が確立されると Null ではない ULConnection オブジェクトを返します。以下のタスクを実行するには、ULConnection オブジェクトを使用します。

- トランザクションのコミットまたはロールバック。
- データの Mobile Link サーバとの同期。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

- データベース内のテーブルへのアクセス。
- SQL 文の処理。
- アプリケーション内のエラーの処理。

データベースファイルに対して書き込み可能なパスが指定されていることを確認します。たとえば、`NSSearchPathForDirectoriesInDomains` メソッドを使用して、`NSDocumentDirectory` を問い合わせます。

> **注記**
>
> サンプルコードは、*%SQLANYSAMP17%*`¥UltraLite¥CustDB¥` ディレクトリにあります。

マルチスレッドアプリケーション

各接続と、それを基に作成されるすべてのオブジェクトは、単一のスレッドで使用してください。アプリケーションが Ultra Light データベースにアクセスするのに複数のスレッドを必要とする場合は、スレッドごとに個別の接続が必要です。

このセクションの内容:

Ultra Light C++ を使用した Ultra Light データベースへの接続 [608 ページ]
ULDatabaseManager オブジェクトを使用して、`sample.udb` という名前の Ultra Light データベースを作成するか、またはこのデータベースに接続します。

3.2.1.3.1 Ultra Light C++ を使用した Ultra Light データベースへの接続

ULDatabaseManager オブジェクトを使用して、`sample.udb` という名前の Ultra Light データベースを作成するか、またはこのデータベースに接続します。

手順

1. ULDatabaseManager オブジェクトを初期化し、次のコードを使用して Ultra Light の機能を有効にします。

```
if( !ULDatabaseManager::Init() ) {
    return 0;
}
ULDatabaseManager::EnableAesDBEncryption();

// Use ULDatabaseManager.Fini() when terminating the app.
```

2. 次のコードを使用して、既存のデータベースへの接続を開きます。指定のデータベースファイルが存在しない場合は、新しいデータベースを作成します。

```
ULConnection * conn;
ULError ulerr;
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

```
conn = ULDatabaseManager::OpenConnection( "dbf=sample.udb;dbkey=aBcD1234",
&ulerr );
if( conn == NULL ) {
    if( ulerr.GetSQLCode() == SQLE_ULTRALITE_DATABASE_NOT_FOUND ) {
        conn =
ULDatabaseManager::CreateDatabase( "dbf=sample.udb;dbkey=aBcD1234", &ulerr );
        if( conn == NULL ) {
            // write code that uses ulerr to determine what happened
            return 0;
        }
        // add code to create the schema for your database
    } else {
        // write code that uses ulerr to determine what happened
        return 0;
    }
}
assert( conn != NULL );
```

この手順では、接続が成功しなかった場合にエラー情報が保持される ULError オブジェクトを宣言します。

結果

`sample.udb` データベースへの接続が確立されます。

3.2.1.4 Ultra Light C++ での SQL 文を使用したデータの作成と修正

Ultra Light アプリケーションは、SQL 文を実行するか ULTable クラスを使用してテーブルデータにアクセスできます。

SQL 文を使用して、次のタスクを実行できます。

- ローの挿入、削除、更新
- 結果セットのローの取得
- 結果セットのローのスクロール

このセクションの内容:

Ultra Light C++ での INSERT、UPDATE、DELETE を使用したデータ修正 [610 ページ]
Ultra Light では、ExecuteStatement メソッド (ULPreparedStatement クラスのメンバー) を使用して、SQL データ操作を実行できます。

Ultra Light C++ での SELECT を使用したデータの検索 [615 ページ]
SELECT 文を実行して Ultra Light データベースから情報を取得し、返される結果セットを処理します。

スキーマの説明の作成と取得 [616 ページ]
GetResultSetSchema メソッドを使用すると、カラム名、カラムの総数、カラムスケール、カラムサイズ、カラム SQL 型など、結果セットに関するスキーマ情報を取得できます。

Ultra Light C++ での SQL 結果セットのナビゲーション [616 ページ]
ULResultSet クラスに関連付けられているメソッドを使用して、結果セット内をナビゲーションすることができます。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

関連情報

Ultra Light C++ での ULTable クラスを使用したデータの作成と修正 [617 ページ]

3.2.1.4.1 Ultra Light C++ での INSERT、UPDATE、DELETE を使用したデータ修正

Ultra Light では、ExecuteStatement メソッド (ULPreparedStatement クラスのメンバー) を使用して、SQL データ操作を実行できます。

このセクションの内容:

Ultra Light C++ を使用したテーブル内のローの挿入 [610 ページ]
Ultra Light では、疑問符 (?) を使用して query パラメータを示します。INSERT 文、UPDATE 文、DELETE 文では必ず、準備文での並び順に従ってそれぞれの ? が参照されます。たとえば、最初の ? はパラメータ 1、2 番目の ? はパラメータ 2、のようになります。

Ultra Light C++ でのテーブル内のローの削除 [612 ページ]
Ultra Light では、疑問符 (?) を使用して query パラメータを示します。INSERT 文、UPDATE 文、DELETE 文では必ず、準備文での並び順に従ってそれぞれの ? が参照されます。たとえば、最初の ? はパラメータ 1、2 番目の ? はパラメータ 2、のようになります。

テーブル内のローの更新 [613 ページ]
Ultra Light では、疑問符 (?) を使用して query パラメータを示します。INSERT 文、UPDATE 文、DELETE 文では必ず、準備文での並び順に従ってそれぞれの ? が参照されます。たとえば、最初の ? はパラメータ 1、2 番目の ? はパラメータ 2、のようになります。

3.2.1.4.1.1 Ultra Light C++ を使用したテーブル内のローの挿入

Ultra Light では、疑問符 (?) を使用して query パラメータを示します。INSERT 文、UPDATE 文、DELETE 文では必ず、準備文での並び順に従ってそれぞれの ? が参照されます。たとえば、最初の ? はパラメータ 1、2 番目の ? はパラメータ 2、のようになります。

手順

1. 次のコードを使用して、ULPreparedStatement を宣言します。

```
ULPreparedStatement * prepStmt;
```

2. SQL 文の実行を準備します。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

次のコードは、INSERT 文の実行を準備します。

```
prepStmt = conn->PrepareStatement("INSERT INTO MyTable(MyColumn1) VALUES (?)");
```

3. 文を準備する際に、エラーをチェックします。

   たとえば、SQL 構文のエラーのチェックには次のコードが有用です。

```
if( prepStmt == NULL ) {
    const ULError * ulerr;
    ulerr = conn->GetLastError();
    // write code to handle the error
    return;
}
```

4. 準備した文内の疑問符 (?) を置き換える値を設定します。

   次のコードは、エラーチェック時に疑問符 (?) を "some value" に設定します。たとえば、パラメータの順序が準備文内のパラメータ数の範囲外である場合は、エラーがキャッチされます。

```
if( !prepStmt->SetParameterString(1, "some value") ) {
    const ULError * ulerr;
    ulerr = conn->GetLastError();
    // write code to handle the error
    return;
}
```

5. 準備文を実行し、データをデータベースに挿入します。

   次のコードでは、文の実行後に発生する可能性があるエラーをチェックします。たとえば、ユニークインデックスで重複したインデックス値が検出されると、エラーが返されます。

```
bool success;
success = prepStmt->ExecuteStatement();
if( !success ) {
    const ULError * ulerr;
    ulerr = conn->GetLastError();
    // write code to handle the error
} else {
    // Use the following line if you are interested in the number of rows
inserted ...
    ul_u_long rowsInserted = prepStmt->GetRowsAffectedCount();
}
```

6. 準備した文のリソースをクリーンアップします。

   次のコードは、準備した文のオブジェクトで使用されるリソースを解放します。Close メソッドが呼び出された後は、このオブジェクトにはアクセスできなくなります。

```
prepStmt->Close();
```

7. データベースにデータをコミットします。

   次のコードは、データベースにデータを保存し、データの損失を防ぎます。デバイスアプリケーションがコミットの呼び出しを完了する前にアプリケーションが予期せず終了すると、手順 5 のデータは失われます。

```
conn->Commit();
```


© 2016 SAP SE or an SAP affiliate company. All rights reserved.

結果

MyColumn1 の値が文字列 "some value" に設定されている場合、MyTable に新しいローが追加されます。

3.2.1.4.1.2 Ultra Light C++ でのテーブル内のローの削除

Ultra Light では、疑問符 (?) を使用して query パラメータを示します。INSERT 文、UPDATE 文、DELETE 文では必ず、準備文での並び順に従ってそれぞれの ? が参照されます。たとえば、最初の ? はパラメータ 1、2 番目の ? はパラメータ 2、のようになります。

手順

1. 次のコードを使用して、ULPreparedStatement を宣言します。

   ```
   ULPreparedStatement * prepStmt;
   ```

2. SQL 文の実行を準備します。

   次のコードは、DELETE 文の実行を準備します。

   ```
   prepStmt = conn->PrepareStatement("DELETE FROM MyTable(MyColumn1) VALUES (?)");
   ```

3. 文を準備する際に、エラーをチェックします。

   たとえば、SQL 構文のエラーのチェックには次のコードが有用です。

   ```
   if( prepStmt == NULL ) {
       const ULError * ulerr;
       ulerr = conn->GetLastError();
       // write code to handle the error
       return;
   }
   ```

4. 準備した文内の疑問符 (?) を置き換える値を設定します。

   次のコードは、エラーチェック時に疑問符 (?) を 7 に設定します。たとえば、パラメータの順序が準備文内のパラメータ数の範囲外である場合は、エラーがキャッチされます。

   ```
   ul_s_long value_to_delete = 7;
   if( !prepStmt->SetParameterInt(1, value_to_delete) ) {
       const ULError * ulerr;
       ulerr = conn->GetLastError();
       // write code to handle the error.
       return;
   }
   ```

5. 準備文を実行し、データベースからデータを削除します。

   次のコードでは、文の実行後に発生する可能性があるエラーをチェックします。たとえば、外部キーが参照しているローを削除しようとすると、エラーが返されます。

   ```
   bool success;
   ```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

```
success = prepStmt->ExecuteStatement();
if( !success ) {
    const ULError * ulerr;
    ulerr = conn->GetLastError();
    // write code to handle the error
} else {
    // Use the following line if you are interested in the number of rows
deleted ...
    ul_u_long rowsDeleted = prepStmt->GetRowsAffectedCount();
}
```

6. 準備した文のリソースをクリーンアップします。

   次のコードは、準備した文のオブジェクトで使用されるリソースを解放します。Close メソッドが呼び出された後は、このオブジェクトにはアクセスできなくなります。

   ```
   prepStmt->Close();
   ```

7. データベースにデータをコミットします。

   次のコードは、データベースにデータを保存し、データの損失を防ぎます。デバイスアプリケーションがコミットの呼び出しを完了する前にアプリケーションが予期せず終了すると、手順 5 のデータは失われます。

   ```
   conn->Commit();
   ```

結果

MyTable の MyColumn 値が 7 である場合、このテーブルのローエントリが削除されます。

3.2.1.4.1.3 テーブル内のローの更新

Ultra Light では、疑問符 (?) を使用して query パラメータを示します。INSERT 文、UPDATE 文、DELETE 文では必ず、準備文での並び順に従ってそれぞれの ? が参照されます。たとえば、最初の ? はパラメータ 1、2 番目の ? はパラメータ 2、のようになります。

手順

1. 次のコードを使用して、ULPreparedStatement を宣言します。

   ```
   ULPreparedStatement * prepStmt;
   ```

2. SQL 文の実行を準備します。

   次のコードは、UPDATE 文の実行を準備します。

   ```
   prepStmt = conn->PrepareStatement("UPDATE MyTable SET MyColumn = ? WHERE
   MyColumn = ?");
   ```


© 2016 SAP SE or an SAP affiliate company. All rights reserved.

3. 文を準備する際に、エラーをチェックします。

   たとえば、SQL 構文のエラーのチェックには次のコードが有用です。

```
if( prepStmt == NULL ) {
    const ULError * ulerr;
    ulerr = conn->GetLastError();
    // write code to handle the error
    return;
}
```

4. 準備した文内の疑問符 (?) を置き換える値を設定します。

   次のコードは、エラーチェック時に ? 文字を整数値に設定します。たとえば、パラメータの順序が準備文内のパラメータ数の範囲外である場合は、エラーがキャッチされます。

```
bool success;
success = prepStmt->SetParameterInt( 1, 25 );
if( success ) {
    success = prepStmt->SetParameterInt( 2, -1 );
}
if( !success ) {
    const ULError * ulerr;
    ulerr = conn->GetLastError();
    // write code to handle the error
    return;
}
```

5. 準備文を実行し、データベースのデータを更新します。

   次のコードでは、文の実行後に発生する可能性があるエラーをチェックします。たとえば、ユニークインデックスで重複したインデックス値が検出されると、エラーが返されます。

```
success = prepStmt->ExecuteStatement();
if( !success ) {
    const ULError * ulerr;
    ulerr = conn->GetLastError();
    // write code to handle the error
} else {
    // if you are interested in the number of rows updated ...
    ul_u_long rowsUpdated = prepStmt->GetRowsAffectedCount();
}
```

6. 準備した文のリソースをクリーンアップします。

   次のコードは、準備した文のオブジェクトで使用されるリソースを解放します。Close メソッドが呼び出された後は、このオブジェクトにはアクセスできなくなります。

```
prepStmt->Close();
```

7. データベースにデータをコミットします。

   次のコードは、データベースにデータを保存し、データの損失を防ぎます。デバイスアプリケーションがコミットの呼び出しを完了する前にアプリケーションが予期せず終了すると、手順 5 のデータは失われます。

```
conn->Commit();
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

結果

このシナリオでは、MyColumn 値が -1 である場合、My Table のローエントリが更新されます。この値は 25 に更新されます。

3.2.1.4.2 Ultra Light C++ での SELECT を使用したデータの検索

SELECT 文を実行して Ultra Light データベースから情報を取得し、返される結果セットを処理します。

手順

1. 次のコードを使用して、必要な変数を宣言します。

```
ULPreparedStatement * prepStmt;
ULResultSet * resultSet;
```

2. SQL 文の実行を準備します。

   次のコードは、SELECT 文の実行を準備します。

```
prepStmt = conn->PrepareStatement("SELECT MyColumn1 FROM MyTable");
```

3. 文を準備する際に、エラーをチェックします。

   たとえば、SQL 構文のエラーのチェックには次のコードが有用です。

```
if( prepStmt == NULL ) {
    const ULError * ulerr;
    ulerr = conn->GetLastError();
    // write code to handle the error
    return;
}
```

4. SQL を実行し、クエリの結果を移動するために使用できる結果セットオブジェクトを返します。

```
resultSet = prepStmt->ExecuteQuery();
if( resultSet == NULL ) {
    const ULError * ulerr;
    ulerr = conn->GetLastError();
    // write code to handle the error
    prepStmt->Close();
    return;
}
```

5. Next メソッドを呼び出すことによって、ローをトラバースします。結果を文字列として保存し、バッファに格納します。

   Next メソッドによって、結果セットの次のローに移動します。呼び出しが true を返す場合、ULResultSet オブジェクトはローに位置付けられます。それ以外の場合 (呼び出しが false を返す場合)、すべてのローがトラバースされます。

```
while( resultSet->Next() ) {
    char buffer[ 100 ];
    resultSet->GetString( 1, buffer, 100 );
    printf( "MyColumn = %s¥n", buffer );
```


© 2016 SAP SE or an SAP affiliate company. All rights reserved.

```
}
```

6. 準備文オブジェクトと結果セットオブジェクトのリソースをクリーンアップします。

   Close メソッドを呼び出した後に、準備文オブジェクトにアクセスしないでください。

```
resultSet->Close();
prepStmt->Close();
```

結果

SELECT 文の結果に含まれる文字列はその後コマンドプロンプトに出力されます。

3.2.1.4.3 スキーマの説明の作成と取得

GetResultSetSchema メソッドを使用すると、カラム名、カラムの総数、カラムスケール、カラムサイズ、カラム SQL 型など、結果セットに関するスキーマ情報を取得できます。

例

次のサンプルコードは、GetResultSetSchema メソッドを使用して、スキーマ情報をコマンドプロンプトに表示する方法を示しています。

```
const char * name;
int column_count;
const ULResultSetSchema & rss = prepStmt->GetResultSetSchema();
int column_count = rss.GetColumnCount();
for( int i = 1; i < column_count; i++ ) {
    name = rss.GetColumnName( i );
    printf( "id = %d, name = %s¥n", i, name );
}
```

この例では、必要な変数が宣言され、ULResultSetSchema オブジェクトが割り当てられます。結果セットオブジェクト自体から ULResultSetSchema オブジェクトを取得することは可能ですが、この例は、文が準備されてクエリが実行される前に、スキーマを利用する方法を示しています。結果セット内のローの数がカウントされ、各カラムの名前が表示されます。

3.2.1.4.4 Ultra Light C++ での SQL 結果セットのナビゲーション

ULResultSet クラスに関連付けられているメソッドを使用して、結果セット内をナビゲーションすることができます。

結果セットクラスは、結果セットをナビゲーションする次のメソッドを提供します。

AfterLast

カーソルを最後のローの直後に配置します。

BeforeFirst

最初のローの直前に配置します。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

First

最初のローに移動します。
Last

最後のローに移動します。
Next

次のローに移動します。
Previous

前のローに移動します。
Relative (offset)

現在のローを基準にして、符号付きオフセット値で指定された数だけローを移動します。オフセット値を正の値で指定すると、現在の結果セットのポインタ位置から前方に移動します。負の値で指定すると後方に移動します。オフセット値が 0 の場合、カーソルは現在のロケーションから移動しませんが、ローバッファが再配置されます。

3.2.1.5 Ultra Light C++ での ULTable クラスを使用したデータの作成と修正

Ultra Light アプリケーションは、SQL 文を実行するか ULTable クラスを使用してテーブルデータにアクセスできます。

ULTable API を使用して、次のタスクを実行できます。

- テーブルのローのスクロール
- 現在のローの値へのアクセス
- Find メソッドと Lookup メソッドを使用したテーブルのローの検索
- ローの挿入、削除、更新

> ⚠ 警告
>
> ローのプライマリキーは更新しないでください。ローを削除し、代わりに新しいローを追加します。

このセクションの内容:

Ultra Light C++ でのローのナビゲーション [618 ページ]
Ultra Light C++ API は、幅広いナビゲーション作業を行うために、テーブルをナビゲーションする複数のメソッドを提供します。

Ultra Light のモード [619 ページ]
Ultra Light モードは、バッファ内の値の使用方法を指定します。

Ultra Light C++ でのローの挿入 [620 ページ]
ローの挿入手順は、ローの更新手順とほぼ同じです。ただし、挿入操作の場合は、テーブル内のローをあらかじめ指定する必要はありません。

ローの更新 [620 ページ]
Update メソッドを使用してテーブル内のローを更新します。

ロー検索の検索モードとルックアップモード [621 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

Ultra Light には、データを操作するための操作モードがいくつかあります。検索では、これらのモードのうち、検索とルックアップの 2 つを使用できます。ULTable オブジェクトには、テーブル内の特定のローを検索するために、これらのモードに対応するメソッドがあります。

現在のローの値へのアクセス [622 ページ]
ローのカラム値を管理するには、ULTable オブジェクトを使用します。

Ultra Light C++ でのローの削除 [623 ページ]
ローの削除手順は、ローの挿入や更新よりも簡単です。挿入モードや更新モードに対応する削除モードはありません。

関連情報

Ultra Light C++ での SQL 文を使用したデータの作成と修正 [609 ページ]

3.2.1.5.1 Ultra Light C++ でのローのナビゲーション

Ultra Light C++ API は、幅広いナビゲーション作業を行うために、テーブルをナビゲーションする複数のメソッドを提供します。

ULTable オブジェクトは、テーブルをナビゲーションする次のメソッドを提供します。

AfterLast

カーソルを最後のローの直後に配置します。

BeforeFirst

最初のローの直前に配置します。

First

最初のローに移動します。

Last

最後のローに移動します。

Next

次のローに移動します。

Previous

前のローに移動します。

Relative (offset)

現在のローを基準にして、符号付きオフセット値で指定された数だけローを移動します。オフセット値を正の値で指定すると、現在の結果セットのポインタ位置から前方に移動します。負の値で指定すると後方に移動します。オフセット値が 0 の場合、カーソルは現在のロケーションから移動しませんが、ローバッファが再配置されます。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

例

次の例は、MyTable テーブルを開き、各ローの MyColumn カラムの値を表示します。

```
char buffer[ 100 ];
ul_column_num column_id;
ULTable * tbl = conn->OpenTable( "MyTable" );
if( tbl == NULL ) {
    const ULError * ulerr;
    ulerr = conn->GetLastError();
    // write code to handle the error
    return;
}
column_id = tbl->GetTableSchema().GetColumnID( "MyColumn" );
if( column_id == 0 ) {
    // the column "MyColumn" likely does not exist.  Handle the error.
    tbl->Close();
    return;
}
while( tbl->Next() ) {
    tbl->GetString( column_id, buffer, 100 );
    printf( "%s¥n", buffer );
}
tbl->Close();
```

ULTable オブジェクトを開くと、テーブルのローがアプリケーションに公開されます。デフォルトでは、ローはプライマリキー値の順に並んでいますが、テーブルを開くときにインデックスを指定すると特定の順序でローにアクセスできます。

例

次の例は、ix_col インデックスで順序付けられた MyTable テーブルの最初のローに移動します。

```
ULTable * tbl = conn->OpenTable( "MyTable", "ix_col" );
```

3.2.1.5.2 Ultra Light のモード

Ultra Light モードは、バッファ内の値の使用方法を指定します。

Ultra Light のモードは次のいずれかに設定できます。

挿入モード

Insert メソッドを呼び出すと、バッファ内のデータが新しいローとしてテーブルに追加されます。

更新モード

Update メソッドを呼び出すと、現在のローがバッファ内のデータに置き換えられます。

検索モード

Find メソッドの 1 つが呼び出されたときに、値がバッファ内のデータに正確に一致するローの検索が検索されます。

ルックアップモード

いずれかの Lookup メソッドが呼び出されたときに、バッファ内のデータと一致するか、それより大きい値のローが検索されます。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

モードを設定するには、モードを設定するための対応メソッドを呼び出します。たとえば InsertBegin、UpdateBegin、FindBegin などです。

3.2.1.5.3 Ultra Light C++ でのローの挿入

ローの挿入手順は、ローの更新手順とほぼ同じです。ただし、挿入操作の場合は、テーブル内のローをあらかじめ指定する必要はありません。

カラムの値を設定しない場合、そのカラムにデフォルト値があるときはデフォルト値が使用されます。カラムにデフォルトがない場合は、次のエントリが使用されます。

- NULL 入力可のカラムの場合は NULL
- NULL 入力不可の数値カラムの場合は 0
- NULL 入力不可の文字カラムの場合は空の文字列
- 明示的に値を NULL に設定するには、SetNull メソッドを使用します。

例

次のコードは、新しいローの挿入を示しています。

```
ULTable * tbl = conn->OpenTable("MyTable");
bool success;
tbl->InsertBegin();  // enter "Insert mode"
tbl->SetInt("id", 3);
tbl->SetString("lname", "Smith");
tbl->SetString("fname", "Mary");
success = tbl->Insert();
conn->Commit();
tbl->Close();
```

この例では、tbl 変数は MyTable を開くように設定されています。各カラムの値は、現在のローバッファに設定されます。カラムは名前または ID で参照できます。Insert メソッドによって、テンポラリローバッファ値がデータベースに挿入されます。結果がコミットされて表示されます。リソースは Close メソッドによって解放されます。

3.2.1.5.4 ローの更新

Update メソッドを使用してテーブル内のローを更新します。

手順

1. 更新するローに移動します。

   テーブルをスクロールするか、find メソッドと lookup メソッドを使用してテーブルを検索し、ローに移動できます。
2. 更新モードを開始します。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

たとえば、次の指示は、テーブル tbl 上で更新モードを開始します。

```
tbl->UpdateBegin();
```

3. 更新するローの新しい値を設定します。たとえば、次の指示は、バッファ内の id カラムを 3 に設定します。

```
tbl->SetInt("id", 3);
```

4. Update を実行します。

```
tbl->Update();
```

> ⚠ 警告
>
> Find メソッドと Update メソッドを使用する場合は、検索条件に含まれるカラムを更新した後、ポインタが予期した位置にない場合があります。場合によっては、SQL 文を使用してローを更新できます。

結果

更新操作が終了すると、更新したローが現在のローになります。

次のステップ

データベースに追加の SQL 操作を実行してから、database. executed への変更をコミットするために Commit メソッドを実行します。

関連情報

Ultra Light C++ でのトランザクションの管理 [624 ページ]

3.2.1.5.5 ロー検索の検索モードとルックアップモード

Ultra Light には、データを操作するための操作モードがいくつかあります。検索では、これらのモードのうち、検索とルックアップの 2 つを使用できます。ULTable オブジェクトには、テーブル内の特定のローを検索するために、これらのモードに対応するメソッドがあります。

> ℹ 注記
>
> Find メソッドと Lookup メソッドを使用して検索されるカラムは、テーブルを開くのに使用されたインデックスにあることが必要です。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

Find メソッド

ULTable オブジェクトを開いたときに指定したソート順に基づいて、指定された検索値と正確に一致する最初のローに移動します。検索値が見つからない場合は、最初のローの前、または最後のローの後ろに位置設定されます。

lookup メソッド

ULTable オブジェクトを開いたときに指定したソート順に基づいて、指定された検索値と一致するか、それより大きい値の最初のローに移動します。

例

この例は、次の SQL 文を使用して作成された MyTable という名前のテーブルを使用します。

```
CREATE TABLE MyTable( id int primary key, lname char(100), fname char(100) )
CREATE INDEX ix_lname ON MyTable ( lname )
```

次のコードは、lname カラムが "Smith" であるローの fname カラムの内容をすべて表示します。

```
ULTable * tbl = conn->OpenTable( "MyTable", "ix_lname" );
char buffer[ 100 ];
bool found;
tbl->FindBegin();  // enter "Find mode"
tbl->SetString( "lname", "Smith" );  // set pointer row buffer to "Smith"
found = tbl->FindFirst();
while( found ) {
    tbl->GetString( 3, buffer, 100 );
    printf( "%s¥n", buffer );
    found = tbl->FindNext();
}
tbl->Close();
```

この例では、tbl 変数は ix_lname インデックスを使用して MyTable を開くように設定されています。これにより、lname カラムと同じ順序でローが返されます。ULTable オブジェクトは、検索を実行するとき、ローバッファ内の値を使用します。このバッファは "Smith" として指定されています。このことは、SetString メソッドによって定義されています。FindFirst メソッドは、lname が "Smith" に設定されている最初のローからトラバーサルが開始されることを示します。lname が "Smith" に設定されているローがない場合、ポインタはテーブルの最後のローの後に位置付けられます。GetString メソッドによって fname が取得されます。これは、fname のカラム ID が 3 であるためです。次に、結果が表示され、リソースが解放されます。

3.2.1.5.6 現在のローの値へのアクセス

ローのカラム値を管理するには、ULTable オブジェクトを使用します。

ULTable オブジェクトは、次のいずれかの位置に常に置かれています。

- テーブルの最初のローの前
- テーブルのいずれかのローの上
- テーブルの最後のローの後ろ

ULTable オブジェクトがローの上に置かれている場合は、そのデータ型に適したメソッドセットを使用して、そのローのカラムの値を取得したり、変更したりできます。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

カラム値の取得

ULTable オブジェクトは、カラム値を取得するメソッドセットを提供します。これらのメソッドは、カラム名またはカラム ID を引数として取ります。

次の例は、テーブルの最初のカラムが age であることを前提として、開いているテーブルから age 値を取得する 2 つの方法を示します。

```
ul_s_long age1 = tbl->GetInt( 1 );
ul_s_long age2 = tbl->GetInt( "age" );
assert( age1 == age2 );
```

値をループで取得する場合、値の取得にカラム ID を使用するバージョンの方が、パフォーマンスが優れています。

カラム値の変更

値を取り出すメソッド以外に、値を設定するメソッドもあります。値を設定するメソッドは、カラム名またはカラム ID と値を引数として取ります。

たとえば、次のコードは、テーブルの最初のカラムが lname であることを前提として、文字列カラム lname と fname を持つローに文字列値を設定する 2 つの方法を示しています。

```
tbl->SetString( 1, last_name );
tbl->SetString( "fname", first_name );
```

カラムの値を設定することにより、データベースのデータが直接変更されることはありません。位置がテーブルの最初のローの前または最後のローの後ろにある場合でも、カラムに値を割り当てることができます。現在のローが定義されていないときに、データにアクセスしようとしないでください。たとえば、次の例でカラムの値をフェッチしようとすることは不正です。

```
// This code is incorrect
tbl->BeforeFirst();
tbl = tbl.GetInt( cust_id );
```

値のキャスト

選択するメソッドは、割り当てるデータ型に一致させてください。データ型に互換性がある場合は、Ultra Light が自動的にデータベースのデータ型をキャストするため、GetString メソッドを使用して整数値を文字列変数にフェッチしたりできます。

3.2.1.5.7 Ultra Light C++ でのローの削除

ローの削除手順は、ローの挿入や更新よりも簡単です。挿入モードや更新モードに対応する削除モードはありません。

削除するローにカーソルを移動してから、ULTable.Delete メソッドを実行する方法で、ローを削除します。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

例

次のコードは、テーブル内の最初のローを削除する方法を示しています。

```
tbl->First();
tbl->Delete();
```

3.2.1.6 Ultra Light C++ でのトランザクションの管理

トランザクションは、データベースを変更するために最初の文によって暗黙的に開始されますが、明示的にコミットまたはロールバックする必要があります。

トランザクションをコミットするには、ULConnection.Commit メソッドを使用します。

トランザクションをロールバックするには、ULConnection.Rollback メソッドを使用します。

3.2.1.7 Ultra Light C++ のスキーマ情報

結果セットまたはデータベース構造の定義は、プログラムを使用して取得できます。これらの記述はスキーマ情報と呼ばれます。この情報は、Ultra Light C API スキーマクラスから使用できます。

注記

Ultra Light C API を使用してスキーマを変更することはできません。スキーマ情報の取得のみが可能です。

次のスキーマオブジェクトと情報にアクセスできます。

ULResultSetSchema

クエリ、またはテーブル内のデータの定義です。識別子、名前、各カラムの型情報、およびテーブル内のカラム数を公開します。ULResultSetSchema クラスは、以下のクラスから取得できます。

- ULPreparedStatement
- ULResultSet
- ULTable

ULDatabaseSchema

データベース内のテーブルとパブリケーションの数と名前、日付と時刻の形式などのグローバルプロパティを公開します。ULDatabaseSchema クラスは、ULConnection クラスから取得できます。

ULTableSchema

カラムとインデックスの設定に関する情報を公開します。ULTableSchema クラスのカラム情報は、ULResultSetSchema クラスから利用できる情報を補完します。たとえば、カラムにデフォルト値があるかどうか、または NULL 値を許可するかどうかを判別できます。ULTableSchema クラスは、ULTable クラスから取得できます。

ULIndexSchema

インデックス内のカラムに関する情報を返します。ULIndexSchema クラスは、ULTableSchema クラスから取得できます。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

ポインタとして返される ULDatabaseSchema、ULTableSchema、および ULIndexSchema クラスと異なり、ULResultSetSchema クラスは定数参照として返されます。定数参照を返すクラスは閉じられませんが、ポインタとして返されるクラスは閉じる必要があります。

次のコードは、スキーマクラスの正しい閉じ方と不正な閉じ方を示しています。

```
// This code demonstrates proper use of the ULResultSetSchema class:
const ULResultSetSchema & rss = prepStmt->GetResultSetSchema();
c_count = prepStmt->GetSchema().GetColumnCount();

// This code demonstrates proper use of the ULDatabaseSchema class:
ULDatabaseSchema * dbs = conn->GetResultSetSchema();
t_count = dbs->GetTableCount();
dbs->Close();  // This line is required.
// This code demonstrates improper use of the ULDatabaseSchema class
// because the object needs to be closed using the Close method:
t_count = conn->GetResultSetSchema()->GetTableCount();
```

3.2.1.8 エラー処理

Ultra Light C++ API には、エラー情報の取得に使用する ULError オブジェクトが含まれています。API のさまざまなメソッドが、メソッド呼び出しが成功したかどうかを示すブール値を返します。場合によっては、エラーが発生したときに NULL が返されます。

ULConnection オブジェクトには、ULError オブジェクトを返す GetLastError メソッドが含まれています。

エラーを診断するには、SQLCode を使用します。SQLCode に加えて、GetParameterCount メソッドと GetParameter メソッドを使用すると、エラーに関する追加情報を提供する追加のパラメータが存在するかどうかを判別できます。

Ultra Light では、明示的なエラー処理に加えて、エラーコールバック関数をサポートしています。コールバック関数を登録すると、Ultra Light エラーが発生するたびに関数が呼び出されます。コールバック関数がアプリケーションフローを制御することはありませんが、すべてのエラーを通知することができます。コールバック関数を使用すると、アプリケーションの開発中やデバッグ中は特に効果的です。

関連情報

チュートリアル:C++ API を使用した Windows アプリケーションの構築 [675 ページ]

3.2.1.9 Ultra Light C++ での Mobile Link データの同期

Ultra Light アプリケーションでは、データを中央のデータベースに同期できます。同期には、SQL Anywhere に付属の Mobile Link 同期ソフトウェアが必要です。

Ultra Light C++ API は、TCP/IP、TLS、HTTP、HTTPS 通信による同期をサポートします。同期は、Ultra Light アプリケーションによって開始されます。接続オブジェクトのメソッドとプロパティは、同期を制御するために使用できます。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

3.2.1.10 Ultra Light データベース接続の切断

使用しなくなったソフトウェアリソースを開放し、アプリケーションがデータベースに接続されているあいだ Ultra Light データベースファイルが使用中のままにならないようにします。

手順

1. Close メソッドを呼び出して、リソースを解放します。

   アプリケーションでデータベースへの接続が必要でなくなったら、次のコードを使用します。

   ```
   if( conn != NULL ) {
       conn->Close( &ulerr );
   }
   ```

2. Fini メソッドを呼び出して、ULDatabaseManager オブジェクトをファイナライズします。

   アプリケーションを閉じる場合、次のコードを使用します。

   ```
   ULDatabaseManager.Fini();
   ```

結果

データベースへの接続が閉じられ、リソースが開放されます。

3.2.1.11 Ultra Light C++ アプリケーションの構築と配備

Ultra Light エンジンを使用しない C++ アプリケーションを作成する場合、静的な Ultra Light ランタイムライブラリにリンクさせるか、または Windows および Windows Mobile でインポートライブラリにリンクさせて、アプリケーションの起動時に Ultra Light ランタイムコードを動的にロードすることができます。

リンカオプションおよびコンパイラオプションによる Linux 開発用ランタイムの構築とリンク

`libulrt.a` のリンカ/コンパイラオプションは、以下のとおりです。

```
-L<$SQLANY17>/ultralite/linux/x86/586/lib -lulrt -lulbase
```

エンジンのオプション:

```
-L<$SQLANY17>/ultralite/linux/x86/586/lib -lulrtc -lulbase
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

ヘッダコマンドラインのオプション:

```
-I<$SQLANY17>/sdk/include
```

このセクションの内容:

Windows Mobile 用の Ultra Light アプリケーションの配備 (静的リンク) [627 ページ]
適切な作成パラメータ、接続パラメータ、同期パラメータ、プロトコルオプション、リンクライブラリ、メソッド呼び出し、配備ファイルを指定して、Ultra Light C++ アプリケーションが Windows および Windows Mobile デバイスで正常に実行されるようにします。

Windows Mobile 用の Ultra Light アプリケーションの配備 (動的リンク) [629 ページ]
適切な作成パラメータ、接続パラメータ、同期パラメータ、プロトコルオプション、リンクライブラリ、メソッド呼び出し、配備ファイルを指定して、Ultra Light C++ アプリケーションが正常に実行されるようにします。

Windows Mobile 用の Ultra Light アプリケーションの配備 (Ultara Light エンジン) [631 ページ]
適切な作成パラメータ、接続パラメータ、同期パラメータ、プロトコルオプション、リンクライブラリ、メソッド呼び出し、配備ファイルを指定して、Ultra Light C++ アプリケーションが Windows および Windows Mobile デバイスで正常に実行されるようにします。

Mac OS X または iOS 用 Ultra Light アプリケーションの配備 [632 ページ]
適切な作成パラメータ、接続パラメータ、同期パラメータ、プロトコルオプション、メソッド呼び出し、配備ファイルを指定して、Ultra Light アプリケーションが Mac コンピュータ、iPhone、または iPad で正常に実行されるようにします。

Linux 用 Ultra Light アプリケーションの配備 [634 ページ]
適切な作成パラメータ、接続パラメータ、同期パラメータ、プロトコルオプション、メソッド呼び出し、配備ファイルを指定して、Ultra Light アプリケーションが Linux で正常に実行されるようにします。

3.2.1.11.1 Windows Mobile 用の Ultra Light アプリケーションの配備 (静的リンク)

適切な作成パラメータ、接続パラメータ、同期パラメータ、プロトコルオプション、リンクライブラリ、メソッド呼び出し、配備ファイルを指定して、Ultra Light C++ アプリケーションが Windows および Windows Mobile デバイスで正常に実行されるようにします。

手順

1. 次のパラメータを指定します。
   - 難読化を使用している場合は、データベース作成時に作成パラメータ *obfuscate=1* を設定します。
   - AES 暗号化を使用する場合は、データベース作成時または接続時に接続パラメータ *DBKEY*=`encryption-key` を設定します。
2. Ultra Light アプリケーションで同期を使用する場合は、パラメータを適切に設定してください。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

同期タイプ	パラメータの設定
TCP/IP	*Stream* 同期パラメータを tcpip に設定します。
HTTP	*Stream* 同期パラメータを http に設定します。
RSA TLS	*Stream* 同期パラメータを tls に設定します。
RSA HTTPS	*Stream* 同期パラメータを https に設定します。

3. RSA エンドツーエンド暗号化を使用している場合は、プロトコルオプション *e2ee_public_key*=`key-file` を設定します。
4. ZLIB 圧縮を使用している場合は、プロトコルオプション *compression=zlib* を設定します。
5. 次のファイルとリンクします。
   - `ulrt.lib`
   - `ulbase.lib`
   - RSA TLS、または RSA HTTPS の同期を使用している場合は `ulrsa.lib`

   Windows Mobile の場合、これらのファイルは *%SQLANY17%*`¥UltraLite¥CE¥Arm.50¥Lib` にあります。
   Windows の場合は、ファイルは *%SQLANY17%*`¥UltraLite¥Windows¥x64¥Lib¥VS9` または *%SQLANY17%*`¥UltraLite¥Windows¥x86¥Lib¥VS9` にあります。
6. Ultra Light アプリケーションで次のメソッドを呼び出します。
   - AES 暗号化を使用している場合は ULDatabaseManager.EnableAesDBEncryption メソッド
7. 次のメソッドが、Ultra Light アプリケーションで使用される同期タイプに対して呼び出されるようにします。

同期タイプ	パラメータの設定
TCP/IP	EnableTcpipSynchronization メソッドを呼び出します。
HTTP	EnableHttpSynchronization メソッドを呼び出します。
RSA TLS	EnableTlsSynchronization メソッドと EnableRsaSyncEncryption メソッドを呼び出します。
RSA HTTPS	EnableHttpsSynchronization メソッドと EnableRsaSyncEncryption メソッドを呼び出します。

結果

Ultra Light C++ アプリケーションは静的リンクを使用し、配備先の Windows デスクトップまたは Windows Mobile デバイスで正常に稼動します。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

次のステップ

アプリケーションを配備した Windows デスクトップまたは Windows Mobile デバイスに Ultra Lite データベースを配備するか、配備したアプリケーションと一緒に新しいデータベースを作成します。

3.2.1.11.2 Windows Mobile 用の Ultra Light アプリケーションの配備 (動的リンク)

適切な作成パラメータ、接続パラメータ、同期パラメータ、プロトコルオプション、リンクライブラリ、メソッド呼び出し、配備ファイルを指定して、Ultra Light C++ アプリケーションが正常に実行されるようにします。

手順

1. 次のパラメータを指定します。
   - 難読化を使用している場合は、データベース作成時に作成パラメータ *obfuscate=1* を設定します。
   - AES 暗号化を使用する場合は、データベース作成時または接続時に接続パラメータ *DBKEY*=`encryption-key` を設定します。
2. Ultra Light アプリケーションで同期を使用する場合は、パラメータを適切に設定してください。

同期タイプ	パラメータの設定
TCP/IP	*Stream* 同期パラメータを tcpip に設定します。
HTTP	*Stream* 同期パラメータを http に設定します。
RSA TLS	*Stream* 同期パラメータを tls に設定します。
RSA HTTPS	*Stream* 同期パラメータを https に設定します。

3. RSA エンドツーエンド暗号化を使用している場合は、プロトコルオプション *e2ee_public_key*=`key-file` を設定します。
4. ZLIB 圧縮を使用している場合は、プロトコルオプション *compression=zlib* を設定します。
5. 次のファイルとリンクします。
   - `ulbase.lib`
   - `ulimp.lib`

   Windows Mobile の場合、これらのファイルは *%SQLANY17%*`¥UltraLite¥CE¥Arm.50¥Lib` にあります。Windows の場合は、ファイルは *%SQLANY17%*`¥UltraLite¥Windows¥x64¥Lib¥VS9` または *%SQLANY17%*`¥UltraLite¥Windows¥x86¥Lib¥VS9` にあります。
6. `ulimp.lib` ライブラリにリンクする場合は、コンパイル時に UL_USE_DLL プリプロセッサマクロを定義してください。たとえば、次のように指定します。

```
-DUL_USE_DLL
```


© 2016 SAP SE or an SAP affiliate company. All rights reserved.

7. Ultra Light アプリケーションで次のメソッドを呼び出します。
   - AES 暗号化を使用している場合は ULDatabaseManager.EnableAesDBEncryption メソッド
8. 次のメソッドが、Ultra Light アプリケーションで使用される同期タイプに対して呼び出されるようにします。

同期タイプ	パラメータの設定
TCP/IP	EnableTcpipSynchronization メソッドを呼び出します。
HTTP	EnableHttpSynchronization メソッドを呼び出します。
RSA TLS	EnableTlsSynchronization メソッドと EnableRsaSyncEncryption メソッドを呼び出します。
RSA HTTPS	EnableHttpsSynchronization メソッドと EnableRsaSyncEncryption メソッドを呼び出します。

9. 次のファイルを配備します。
   - `ulrt`*`17`*`.dll`
   - ZLIB 圧縮を使用している場合は、`mlczlib`*`17`*`.dll`。
   - RSA TLS、RSA HTTPS、または RSA E2EE を使用している場合は、`mlcrsa`*`17`*`.dll`。

   Windows Mobile の場合は、ファイルは *%SQLANY17%*`¥UltraLite¥CE¥Arm.50` にあります。Windows の場合は、ファイルは *%SQLANY17%*`¥UltraLite¥Windows¥x64` または *%SQLANY17%*`¥UltraLite¥Windows¥x86` にあります。

結果

Ultra Light C++ アプリケーションは動的リンクを使用し、配備先の Windows デスクトップまたは Windows Mobile デバイスで正常に稼動します。

次のステップ

アプリケーションを配備した Windows デスクトップまたは Windows Mobile デバイスに Ultra Lite データベースを配備するか、配備したアプリケーションと一緒に新しいデータベースを作成します。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

3.2.1.11.3 Windows Mobile 用の Ultra Light アプリケーションの配備 (Ultara Light エンジン)

適切な作成パラメータ、接続パラメータ、同期パラメータ、プロトコルオプション、リンクライブラリ、メソッド呼び出し、配備ファイルを指定して、Ultra Light C++ アプリケーションが Windows および Windows Mobile デバイスで正常に実行されるようにします。

手順

1. 次のパラメータを指定します。
   - 難読化を使用している場合は、データベース作成時に作成パラメータ *obfuscate=1* を設定します。
   - AES 暗号化を使用する場合は、データベース作成時または接続時に接続パラメータ *DBKEY*=`encryption-key` を設定します。
2. Ultra Light アプリケーションで同期を使用する場合は、パラメータを適切に設定してください。

同期タイプ	パラメータの設定
TCP/IP	*Stream* 同期パラメータを tcpip に設定します。
HTTP	*Stream* 同期パラメータを http に設定します。
RSA TLS	*Stream* 同期パラメータを tls に設定します。
RSA HTTPS	*Stream* 同期パラメータを https に設定します。

3. RSA エンドツーエンド暗号化を使用している場合は、プロトコルオプション *e2ee_public_key*=`key-file` を設定します。
4. ZLIB 圧縮を使用している場合は、プロトコルオプション *compression=zlib* を設定します。
5. 次のファイルとリンクします。
   - `ulrtc.lib`
   - `ulbase.lib`

   Windows Mobile の場合、これらのファイルは *%SQLANY17%*`¥UltraLite¥CE¥Arm.50¥Lib` にあります。Windows の場合は、ファイルは *%SQLANY17%*`¥UltraLite¥Windows¥x64¥Lib¥VS9` または *%SQLANY17%*`¥UltraLite¥Windows¥x86¥Lib¥VS9` にあります。
6. 次のメソッドが、Ultra Light アプリケーションで使用される同期タイプに対して呼び出されるようにします。

同期タイプ	パラメータの設定
TCP/IP	EnableTcpipSynchronization メソッドを呼び出します。
HTTP	EnableHttpSynchronization メソッドを呼び出します。
RSA TLS	EnableTlsSynchronization メソッドと EnableRsaSyncEncryption メソッドを呼び出します。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

同期タイプ	パラメータの設定
RSA HTTPS	EnableHttpsSynchronization メソッドと EnableRsaSyncEncryption メソッドを呼び出します。

7. 次のファイルを配備します。
   - *uleng17*.exe.
   - ZLIB 圧縮を使用している場合は、mlczlib*17*.dll。
   - RSA TLS、RSA HTTPS、または RSA E2EE を使用している場合は、mlcrsa*17*.dll。

   Windows Mobile の場合は、ファイルは *%SQLANY17%*¥UltraLite¥CE¥Arm.50 にあります。Windows の場合は、ファイルは *%SQLANY17%*¥UltraLite¥Windows¥x64 または *%SQLANY17%*¥UltraLite¥Windows¥x86 にあります。

結果

Ultra Light C++ アプリケーションは Ultra Light エンジンを使用し、配備先の Windows デスクトップまたは Windows Mobile デバイスで正常に稼働します。

次のステップ

アプリケーションを配備した Windows デスクトップまたは Windows Mobile デバイスに Ultra Lite データベースを配備するか、配備したアプリケーションと一緒に新しいデータベースを作成します。

3.2.1.11.4 Mac OS X または iOS 用 Ultra Light アプリケーションの配備

適切な作成パラメータ、接続パラメータ、同期パラメータ、プロトコルオプション、メソッド呼び出し、配備ファイルを指定して、Ultra Light アプリケーションが Mac コンピュータ、iPhone、または iPad で正常に実行されるようにします。

手順

1. Mac OS X の場合、次のランタイムライブラリファイルを Xcode プロジェクトに追加します。
   - */Applications/SQLAnywhere17/System*/ultralite/macosx/x86_64/libulrt.a
   - */Applications/SQLAnywhere17/System*/ultralite/macosx/x86_64/libulbase.a
2. iOS の場合、Ultra Light ランタイムライブラリにリンクするには、次のいずれかを行います。

   install-dir/ultralite/iphone/libulrt.a を Xcode の *Frameworks* グループに追加します。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

または

Other Linker Flags (OTHER_LDFLAGS) のビルド設定に次の指定を追加します。

```
-L$(SQLANY_ROOT)/ultralite/iphone
-lulrt
```

ここで、SQLANY_ROOT は、SQL Anywhere インストールディレクトリに設定されたカスタムビルド設定を表します。

Ultra Light ランタイムは、インストール後にビルドしてください。install-dir/ultralite/iphone/readme.txt の指示に従ってください。

3. Xcode プロジェクトに適切なフレームワークを追加します。
   - Mac OS X の場合、CoreFoundation.framework、CoreServices.framework、および Security.framework
   - iOS の場合、CFNetwork.framework および Security.framework
4. 次のパラメータを指定します。
   - 難読化を使用している場合は、データベース作成時に作成パラメータ *obfuscate=1* を設定します。
   - AES 暗号化を使用する場合は、データベース作成時または接続時に接続パラメータ *DBKEY*=encryption-key を設定します。
5. Ultra Light アプリケーションで同期を使用する場合は、パラメータを適切に設定してください。

同期タイプ	パラメータの設定
TCP/IP	*Stream* 同期パラメータを tcpip に設定します。
HTTP	*Stream* 同期パラメータを http に設定します。
RSA TLS	*Stream* 同期パラメータを tls に設定します。
RSA HTTPS	*Stream* 同期パラメータを https に設定します。

6. RSA E2EE 暗号化を使用している場合は、プロトコルオプション *e2ee_public_key*=key-file を設定します。
7. ZLIB 圧縮を使用している場合は、プロトコルオプション *compression=zlib* を設定します。
8. AES 暗号化を使用している場合は、ULDatabaseManager.EnableAesDBEncryption メソッドを呼び出します。
9. 次のメソッドが、Ultra Light アプリケーションで使用される同期タイプに対して呼び出されるようにします。

同期タイプ	パラメータの設定
TCP/IP	EnableTcpipSynchronization メソッドを呼び出します。
HTTP	EnableHttpSynchronization メソッドを呼び出します。
RSA TLS	EnableTlsSynchronization メソッドと EnableRsaSyncEncryption メソッドを呼び出します。
RSA HTTPS	EnableHttpsSynchronization メソッドと EnableRsaSyncEncryption メソッドを呼び出します。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

結果

Ultra Light アプリケーションは、Mac OS X デスクトップまたは iOS デバイスで正常に稼動します。

次のステップ

アプリケーションが配備された Mac デスクトップ、iPhone、または iPad に Ultra Light データベースを配備するか、または配備したアプリケーションで新しいデータベースを作成します。

関連情報

Apple iOS と Mac OS X での考慮事項 [606 ページ]

3.2.1.11.5 Linux 用 Ultra Light アプリケーションの配備

適切な作成パラメータ、接続パラメータ、同期パラメータ、プロトコルオプション、メソッド呼び出し、配備ファイルを指定して、Ultra Light アプリケーションが Linux で正常に実行されるようにします。

手順

1. 次のパラメータを指定します。
   - 難読化を使用している場合は、データベース作成時に作成パラメータ *obfuscate=1* を設定します。
   - AES 暗号化を使用する場合は、データベース作成時または接続時に接続パラメータ *DBKEY*=`encryption-key` を設定します。
2. Ultra Light アプリケーションで同期を使用する場合は、パラメータを適切に設定してください。

同期タイプ	パラメータの設定
TCP/IP	*Stream* 同期パラメータを tcpip に設定します。
HTTP	*Stream* 同期パラメータを http に設定します。
RSA TLS	*Stream* 同期パラメータを tls に設定します。
RSA HTTPS	*Stream* 同期パラメータを https に設定します。

3. RSA または FIPS 140-2 RSA エンドツーエンド暗号化を使用している場合は、プロトコルオプション *e2ee_public_key*=`key-file` を設定します。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

4. ZLIB 圧縮を使用している場合は、プロトコルオプション *compression=zlib* を設定します。
5. 次のファイルとリンクします。
   - `libulrt.a`
   - `libulbase.a`
   - RSA TLS、RSA HTTPS、または RSA E2EE を使用している場合は、`libulrsa.a`。

   これらのファイルは、*/opt/sqlanywhere17*`/ultralite/linux/x86/586/lib` にあります。
6. AES 暗号化を使用している場合は、ULDatabaseManager.EnableAesDBEncryption メソッドを呼び出します。
7. 次のメソッドが、Ultra Light アプリケーションで使用される同期タイプに対して呼び出されるようにします。

同期タイプ	パラメータの設定
TCP/IP	EnableTcpipSynchronization メソッドを呼び出します。
HTTP	EnableHttpSynchronization メソッドを呼び出します。
RSA TLS	EnableTlsSynchronization メソッドと EnableRsaSyncEncryption メソッドを呼び出します。
RSA HTTPS	EnableHttpsSynchronization メソッドと EnableRsaSyncEncryption メソッドを呼び出します。

結果

Ultra Light アプリケーションは配備先の Linux コンピュータで正常に稼動します。

次のステップ

アプリケーションが配備された Linux コンピュータに Ultra Light データベースを配備するか、または配備したアプリケーションで新しいデータベースを作成します。

3.2.2 Embedded SQL を使用した Ultra Light C++ アプリケーション開発

Embedded SQL Ultra Light アプリケーションのデータベースアクセスコードを作成できます。

このセクションの内容:

Ultra Light Embedded SQL アプリケーション開発のクイックスタートガイド [636 ページ]
Embedded SQL アプリケーションを開発するときは、SQL 文に標準の C または C++ ソースコードを混在させます。Embedded SQL アプリケーションを開発するには、C または C++ のプログラミング言語に精通していることが必要です。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

Embedded SQL の例 [637 ページ]
Embedded SQL は、C/C++ プログラムコードと擬似コードの組み合わせの環境です。

SQL Communications Area の初期化 [639 ページ]
SQLCA (SQL Communications Area) とは、アプリケーションとデータベースの間で、統計情報とエラーをやりとりするのに使用されるメモリ領域です。SQLCA は、アプリケーションとデータベース間の通信リンクのハンドルとして使用されます。

Embedded SQL を使用した Ultra Light データベース接続 [641 ページ]
Embedded SQL アプリケーションから Ultra Light データベースに接続するには、SQLCA を初期化した後、コードに EXEC SQL CONNECT 文を指定します。

ホスト変数 [642 ページ]
Embedded SQL アプリケーションでは、ホスト変数を使用してデータベースと値をやり取りします。ホスト変数とは、宣言セクションにおいて SQL プリプロセッサが認識する C 変数です。

データのフェッチ [652 ページ]
Embedded SQL でデータをフェッチするには SELECT 文を使用します。

ユーザ認証 [656 ページ]
ユーザ認証は、ULGrantConnectTo メソッドと ULRevokeConnectFrom メソッドを使用して制御できます。

Ultra Light Embedded SQL でのデータの暗号化 [657 ページ]
Ultra Light Embedded SQL を使用して、Ultra Light データベースを暗号化したり、難読化したりできます。

Embedded SQL アプリケーションの同期の設定 [658 ページ]
多くの Ultra Light アプリケーションにとって、同期は重要な機能です。Embedded SQL API の構造体のメンバーは、Ultra Lite C++ API と似ています。

Embedded SQL アプリケーションの構築 [665 ページ]
Embedded SQL アプリケーションを構築するには、いくつかの設定に関する知識が必要です。

関連情報

Ultra Light Embedded SQL API リファレンス [689 ページ]

3.2.2.1 Ultra Light Embedded SQL アプリケーション開発のクイックスタートガイド

Embedded SQL アプリケーションを開発するときは、SQL 文に標準の C または C++ ソースコードを混在させます。Embedded SQL アプリケーションを開発するには、C または C++ のプログラミング言語に精通していることが必要です。

Embedded SQL アプリケーションの開発プロセスは、次のとおりです。

1. Ultra Light データベースを設計します。
2. 通常 `.sqc` という拡張子の付いた Embedded SQL ソースファイルにソースコードを記述します。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

ソースコードにデータアクセスが必要な場合は、EXEC SQL キーワードに続いて SQL 文を指定して実行します。次に例を示します。

```
EXEC SQL BEGIN DECLARE SECTION
   int cost
   char pname[31];
EXEC SQL END DECLARE SECTION
EXEC SQL SELECT price, prod_name
   INTO :cost, :pname
   FROM ULProduct
   WHERE prod_id= :pid;
```

3. .sqc ファイルの前処理を実行します。
   SQL Anywhere には SQL プリプロセッサ (sqlpp) が用意されており、.sqc ファイルを読み込んで .cpp ファイルを生成します。これらのファイルには Ultra Light ランタイムライブラリへの関数呼び出しが格納されています。
4. .cpp ファイルをコンパイルします。
5. .cpp ファイルをリンクします。
   コンパイルしたファイルは、Ultra Light ランタイムライブラリにリンクする必要があります。

関連情報

Embedded SQL アプリケーションの構築 [665 ページ]

3.2.2.2 Embedded SQL の例

Embedded SQL は、C/C++ プログラムコードと擬似コードの組み合わせの環境です。

従来の C/C++ コードの間に存在する擬似コードは、SQL 文のサブセットです。プリプロセッサは、Embedded SQL 文を、アプリケーションを作成するためにコンパイルされる実際のコードの一部である関数呼び出しに変換します。

Embedded SQL プログラムの非常に簡単な例を次に示します。従業員 195 の姓を変更して Ultra Light データベースレコードを更新します。

```
#include <stdio.h>
EXEC SQL INCLUDE SQLCA;
main( )
{
   db_init( &sqlca );
   EXEC SQL WHENEVER SQLERROR GOTO error;
   EXEC SQL CONNECT "DBA" IDENTIFIED BY "sql";
   EXEC SQL UPDATE employee
      SET emp_lname = 'Johnson'
      WHERE emp_id = 195;
   EXEC SQL COMMIT;
   EXEC SQL DISCONNECT;
   db_fini( &sqlca );
   return( 0 );
   error:
      printf( "update unsuccessful: sqlcode = %ld¥n",
         sqlca.sqlcode );
      return( -1 );
}
```


© 2016 SAP SE or an SAP affiliate company. All rights reserved.

この例は実際に使用するプログラムと比べてかなり簡単な内容ですが、Embedded SQL アプリケーションにおける次のような共通事項が示されています。

- 各 SQL 文には、キーワード EXEC SQL のプレフィクスが付いています。
- 各 SQL 文は、セミコロンで終わる。
- 標準の SQL では実装されていない Embedded SQL 文があります。INCLUDE SQLCA 文はその一例です。
- Embedded SQL では、SQL 文のほかに、特定のタスクを実行するためにライブラリ関数が提供されます。関数 db_init と db_fini は、ライブラリ関数呼び出しの 2 例です。

初期化

前述のサンプルコードでは、Ultra Light データベースのデータを操作する前に含める必要のある初期化文が示されています。

1. 次のコマンドを使用して、SQLCA (SQL Communication Area) を定義します。

   ```
   EXEC SQL INCLUDE SQLCA;
   ```

   この定義は、最初の Embedded SQL 文でなければならないため、通常はインクルードリストの最後に記述します。アプリケーションに複数の .sqc ファイルがある場合は、各ファイルにこの行を含めます。
2. 最初のデータベース処理として、db_init という Embedded SQL ライブラリ関数を呼び出す必要があります。この関数は、Ultra Light ランタイムライブラリを初期化します。この呼び出しの前に実行できるのは、Embedded SQL の定義文だけです。
3. Ultra Light データベースに接続するには、SQL CONNECT 文を使用する必要があります。

終了の準備

前述のサンプルコードでは、終了の準備に必要となる呼び出しの順序も示しています。

1. 未処理の変更をコミットまたはロールバックします。
2. データベースを切断します。
3. db_fini というライブラリメソッドを呼び出して、SQL の作業を終了します。

終了時に、コミットされていないデータベースの変更は、すべて自動的にロールバックされます。

エラー処理

この例では、SQL と C コード間の対話はまったくありません。C コードはプログラムのフロー制御だけを行います。WHENEVER 文はエラーチェックに使用されています。エラーアクション (この例では GOTO) は、いずれかの SQL 文がエラーになると実行されます。

このセクションの内容:

Embedded SQL プログラム構造 [639 ページ]

 © 2016 SAP SE or an SAP affiliate company.All rights reserved.

Embedded SQL 文は、必ず、EXEC SQL で始まり、セミコロンで終わります。

関連情報

db_init メソッド [691 ページ]

3.2.2.2.1 Embedded SQL プログラム構造

Embedded SQL 文は、必ず、EXEC SQL で始まり、セミコロンで終わります。

Embedded SQL 文の途中に、C 言語の通常のコメントを記述できます。

Embedded SQL を使用する C プログラムでは、ソースファイル内のどの Embedded SQL 文よりも前に、必ず次の文を置きます。

```
EXEC SQL INCLUDE SQLCA;
```

プログラムで実行される最初の Embedded SQL 文は、CONNECT 文である必要があります。CONNECT 文は、Ultra Light データベースへの接続を確立するために使用される接続パラメータを指定します。

Embedded SQL コマンドには C プログラムコードを生成しないものや、データベースとのやりとりをしないものもあります。このようなコマンドは、CONNECT 文の前に記述できます。よく使われるのは、INCLUDE 文と、エラー処理を指定する WHENEVER 文です。

3.2.2.3 SQL Communications Area の初期化

SQLCA (SQL Communications Area) とは、アプリケーションとデータベースの間で、統計情報とエラーをやりとりするのに使用されるメモリ領域です。SQLCA は、アプリケーションとデータベース間の通信リンクのハンドルとして使用されます。

データベースとやりとりするデータベースライブラリ関数には、SQLCA が明示的に渡されます。また、Embedded SQL 文でも必ず暗黙のうちに渡されます。

生成コードには、SQLCA グローバル変数が 1 つ定義されています。プリプロセッサは、このグローバル SQLCA 変数の外部参照を生成します。外部参照の名前は sqlca、型は SQLCA です。実際のグローバル変数は、インポートライブラリ内で宣言されています。

SQLCA 型はヘッダファイル *%SQLANY17%*¥SDK¥Include¥sqlca.h に定義されています。

アプリケーションでデータベースを操作するには、SQLCA (`EXEC SQL INCLUDE SQLCA;`) を宣言した後、db_init を呼び出して SQLCA を渡すことによって SQL Communication Area を初期化する必要があります。

```
db_init( &sqlca );
```


© 2016 SAP SE or an SAP affiliate company. All rights reserved.

SQLCA にはエラーコードが入る

SQLCA を参照すると、特定のエラーコードの検査ができます。データベースへの要求がエラーになると、フィールド sqlcode にエラーコードが入ります。sqlcode などの SQLCA のフィールドを参照するために、マクロが定義されています。

このセクションの内容:

Ultra Lite C++ で使用する SQLCA フィールド [640 ページ]
SQLCA には、SQLCA 構造体を説明する複数のフィールドが含まれます。

3.2.2.3.1 Ultra Lite C++ で使用する SQLCA フィールド

SQLCA には、SQLCA 構造体を説明する複数のフィールドが含まれます。

sqlcaid

SQLCA 構造体の ID として文字列 SQLCA が格納される 8 バイトの文字フィールド。このフィールドはデバッグ時にメモリの中身を見るときに役立ちます。

sqlcabc

long 型整数。SQLCA 構造体の長さ (バイト単位) が入ります。

sqlcode

long 型整数。データベースが検出した要求エラーのエラーコードが入ります。エラーコードの定義はヘッダファイル *%SQLANY17%*¥SDK¥Include¥sqlerr.h にあります。エラーコードは、0 (ゼロ) は成功、正の値は警告、負の値はエラーを示します。

SQLCODE マクロを使用してこのフィールドに直接アクセスできます。

sqlerrml

sqlerrmc フィールドの情報の長さ。

Ultra Light アプリケーションでは、このフィールドは使用されません。

sqlerrmc

エラーメッセージに挿入する 1 つ以上の文字列。エラーメッセージにプレースホルダ文字列 (%1) があると、このフィールドの文字列と置換されます。

Ultra Light アプリケーションでは、このフィールドは使用されません。

sqlerrp

予約済み。

sqlerrd

long 型整数の汎用配列。

sqlwarn

予約済み。

Ultra Light アプリケーションでは、このフィールドは使用されません。

sqlstate

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

SQLSTATE ステータス値。

Ultra Light アプリケーションでは、このフィールドは使用されません。

3.2.2.4 Embedded SQL を使用した Ultra Light データベース接続

Embedded SQL アプリケーションから Ultra Light データベースに接続するには、SQLCA を初期化した後、コードに EXEC SQL CONNECT 文を指定します。

CONNECT 文の形式は次のとおりです。

```
EXEC SQL CONNECT USING
'uid=user-name;pwd=password;dbf=database-filename';
```

接続文字列 (一重引用符で囲まれている) には、追加のデータベース接続パラメータが含まれることがあります。

アプリケーションの中で複数のデータベース接続を使用する場合、複数の SQLCA を使用することもできれば、1 つの SQLCA で複数の接続を管理することもできます。

単一の SQLCA の使用

1 つの SQLCA でデータベースへの複数の接続を管理できます。

各 SQLCA はアクティブな接続、つまり現在の接続を持ちますが、その接続は変更が可能です。コマンドを実行する前に、SET CONNECTION 文でコマンドの実行対象となる接続を指定します。

このセクションの内容:

複数の SQLCA を使用した複数のデータベース接続の管理 [641 ページ]
Embedded SQL 文の SET SQLCA を使用して、SQL プリプロセッサにデータベース要求で特定の SQLCA を使用することを伝えます。

3.2.2.4.1 複数の SQLCA を使用した複数のデータベース接続の管理

Embedded SQL 文の SET SQLCA を使用して、SQL プリプロセッサにデータベース要求で特定の SQLCA を使用することを伝えます。

手順

1. db_init への呼び出しによって、プログラムで使用される各 SQLCA を初期化します。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

2. プログラムの先頭かヘッダファイルに置いて、SQLCA 参照がそのタスク固有のデータを指すようにします。

```
EXEC SQL SET SQLCA 'task_data->sqlca';
```

この文はコードをまったく生成しないので、パフォーマンスに影響を与えません。

結果

プリプロセッサ内部の状態が変更され、指定の文字列で SQLCA を参照するようにします。

次のステップ

db_fini メソッドへの呼び出しによってアプリケーションを閉じると、リソースがクリーンアップされます。

関連情報

db_init メソッド [691 ページ]

3.2.2.5 ホスト変数

Embedded SQL アプリケーションでは、ホスト変数を使用してデータベースと値をやり取りします。ホスト変数とは、宣言セクションにおいて SQL プリプロセッサが認識する C 変数です。

このセクションの内容:

ホスト変数の宣言 [643 ページ]
宣言セクション内にホスト変数を配置して、ホスト変数を定義します。通常の C 変数宣言を BEGIN DECLARE SECTION 文と END DECLARE SECTION 文で囲むことで、ホスト変数を宣言します。

データ型 [643 ページ]
プログラムとデータベースサーバ間で情報を転送するには、それぞれのデータ項目についてデータ型を設定します。ホスト変数は、サポートされる任意のデータ型について作成できます。

Ultra Light C++ でのホスト変数の使用方法 [646 ページ]
ホスト変数は、いくつかの特別な場合に使用できます。

ホスト変数のスコープ [647 ページ]
ホスト変数の宣言セクションは、C 変数を宣言できる通常の場所であれば、C の関数のパラメータの宣言セクションも含め、どこにでも記述できます。

ホスト変数としての式 [648 ページ]
SQL プリプロセッサはポインタや参照式を認識しないため、ホスト変数は単純な名前でなければなりません。

 © 2016 SAP SE or an SAP affiliate company.All rights reserved.

C++ でのホスト変数 [649 ページ]
一般に、クラスを別のヘッダファイルで頻繁に宣言するときに便利です。

インジケータ変数 [650 ページ]
インジケータ変数とは、特定のホスト変数に関する補足的な情報を保持する C 変数のことです。ホスト変数は、データのやりとりをするときに使用できます。NULL 値を扱うには、インジケータ変数を使用します。

3.2.2.5.1 ホスト変数の宣言

宣言セクション内にホスト変数を配置して、ホスト変数を定義します。通常の C 変数宣言を BEGIN DECLARE SECTION 文と END DECLARE SECTION 文で囲むことで、ホスト変数を宣言します。

ホスト変数を SQL 文で使用するときは、変数名にコロン (:) をプレフィクスとして付けなければなりません。これは、SQL プリプロセッサが、(宣言済みの) ホスト変数が参照されていることを認識しており、SQL 文の中で他の識別子とホスト変数を区別するためです。

ホスト変数は、どの SQL 文でも値定数の代わりに使用できます。データベースサーバがこのコマンドを実行すると、ホスト変数の値がホスト変数から読み込まれたり、逆にホスト変数に書き込まれたりします。ホスト変数をテーブル名やカラム名の代わりに使用することはできません。

SQL プリプロセッサは、宣言セクションの外では C 言語コードをスキャンしません。変数の初期化は宣言セクション内で行うことも可能ですが、typedef 型と構造体は使用できません。

INSERT コマンドでホスト変数を使用するサンプルコードです。プログラム側で変数に値を設定してから、データベースに挿入しています。

```
/* Declare fields for personal data. */
EXEC SQL BEGIN DECLARE SECTION;
   long employee_number = 0;
   char employee_name[50];
   char employee_initials[8];
   char employee_phone[15];
EXEC SQL END DECLARE SECTION;
/* Fill variables with appropriate values. */
/* Insert a row in the database. */
EXEC SQL INSERT INTO Employee
   VALUES (:employee_number, :employee_name,
      :employee_initials, :employee_phone );
```

3.2.2.5.2 データ型

プログラムとデータベースサーバ間で情報を転送するには、それぞれのデータ項目についてデータ型を設定します。ホスト変数は、サポートされる任意のデータ型について作成できます。

ホスト変数として使用できる C のデータ型は非常に限られています。また、ホスト変数の型には、対応する C の型がないものもあります。

`sqlca.h` ヘッダファイルで定義したマクロは、VARCHAR、FIXCHAR、BINARY、DECIMAL、または SQLDATETIME 型のホスト変数を宣言するのに使用できます。これらのマクロは次のように使用します。

```
EXEC SQL BEGIN DECLARE SECTION;
```


© 2016 SAP SE or an SAP affiliate company. All rights reserved.

```
    DECL_VARCHAR( 10 ) v_varchar;
    DECL_FIXCHAR( 10 ) v_fixchar;
    DECL_BINARY( 4000 ) v_binary;
    DECL_DECIMAL( 10, 2 ) v_packed_decimal;
    DECL_DATETIME v_datetime;
EXEC SQL END DECLARE SECTION;
```

プリプロセッサは宣言セクション内のこれらのマクロを認識し、変数を適切な型として処理します。

次のデータ型が、Embedded SQL プログラミングインタフェースでサポートされます。

16 ビット符号付き整数

```
short int I;
unsigned short int I;
```

32 ビット符号付き整数

```
long int l;
unsigned long int l;
```

4 バイト浮動小数点数

```
float f;
```

8 バイト浮動小数点数

```
double d;
```

パック 10 進数

```
DECL_DECIMAL(p,s)
typedef struct TYPE_DECIMAL {
   char array[1];
} TYPE_DECIMAL;
```

ブランクが埋め込まれ、NULL で終了された文字列

```
char a[n]; /* n > 1 */
char *a; /* n = 2049 */
```

C 言語の配列には NULL ターミネータが必要であるため、char a[n] データ型は、CHAR(n - 1) SQL データ型にマッピングされます。SQL データ型は、n - 1 文字まで保持できます。

> **注記**
>
> SQL プリプロセッサでは、**char** のポインタは 2049 バイトの文字配列を指しており、この文字配列が 2048 文字と NULL ターミネータを十分に保持できるとみなされます。つまり、char* データ型は、CHAR(2048) SQL 型にマッピングされます。この制限を超えると、アプリケーションによるメモリ破損が発生する場合があります。
>
> 16 ビットコンパイラの場合、単純に 2049 バイトを確保するとプログラムのスタックオーバーフローを引き起こすこともあります。この問題を避けるため、宣言した配列を必要に応じて関数のパラメータとして使用し、SQL プリプロセッサに配列のサイズを通知するようにしてください。WCHAR と TCHAR も char と同じように機能します。

NULL で終了された Unicode、またはワイド文字列

文字ごとに 2 バイトの領域を占有するため、Unicode 文字を含めることができます。

```
WCHAR a[n]; /* n > 1 */
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

システムに依存し、NULL で終了された文字列

Unicode を使用するシステム (Microsoft Windows Mobile など) では、TCHAR の文字セットは WCHAR と同じです。それ以外の場合、TCHAR は char と同じです。TCHAR データ型は、どちらかのシステムで文字列を自動的にサポートするように設計されています。

```
TCHAR a[n]; /* n > 1 */
```

ブランクが埋め込まれた固定長文字列

```
char a; /* n  = 1 */
DECL_FIXCHAR(n) a; /* n >= 1 */
```

2 バイトの長さフィールドを持つ可変長文字列

データベースサーバに情報を渡す場合は、長さフィールドを設定してください。データベースサーバから情報をフェッチする場合は、サーバが長さフィールドを設定します (埋め込みは行われません)。

```
DECL_VARCHAR(n) a; /* n >= 1 */
typedef struct VARCHAR {
   a_sql_ulen len;
   TCHAR array[1];
} VARCHAR;
```

2 バイトの長さフィールドを持つ可変長バイナリデータ

データベースサーバに情報を渡す場合は、長さフィールドを設定してください。データベースサーバから情報をフェッチする場合は、サーバが長さフィールドを設定します。

```
DECL_BINARY(n) a; /* n >= 1 */
typedef struct BINARY {
  a_sql_ulen len;
   unsigned char array[1];
} BINARY;
```

タイムスタンプの各部分に対応するフィールドを持つ SQLDATETIME 構造体

```
DECL_DATETIME a;
typedef struct SQLDATETIME {
   unsigned short year; /* for example: 1999 */
   unsigned char month; /* 0-11 */
   unsigned char day_of_week; /* 0-6, 0 = Sunday */
   unsigned short day_of_year; /* 0-365 */
   unsigned char day; /* 1-31 */
   unsigned char hour; /* 0-23 */
   unsigned char minute; /* 0-59 */
   unsigned char second; /* 0-59 */
   unsigned long microsecond; /* 0-999999 */
} SQLDATETIME;
```

SQLDATETIME 構造体は、型が DATE、TIME、TIMESTAMP (または、いずれかの型に変換できるもの) のフィールドを取り出すのに使用されます。アプリケーションは、日付に関して独自のフォーマットで処理をすることがあります。この構造体の中のデータをフェッチすると、このデータを簡単に操作できます。また、型が DATE、TIME、TIMESTAMP のフィールドは、文字型であれば、どの型でもフェッチと更新が可能です。

SQLDATETIME 構造体を介してデータベースに日付、時刻、またはタイムスタンプを入力しようとすると、day_of_year と day_of_week メンバーは無視されます。

DT_LONGVARCHAR

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

長い可変長文字データ。マクロによって、構造体が次のように定義されます。

```
#define DECL_LONGVARCHAR( size ) ¥
  struct { a_sql_uint32    array_len;    ¥
           a_sql_uint32    stored_len;   ¥
           a_sql_uint32    untrunc_len;  ¥
           char            array[size+1];¥
         }
```

32 KB を超えるデータには、DECL_LONGVARCHAR 構造体を使用できます。データは、全体を一度にフェッチする方法と、GET DATA 文を使用して分割してフェッチする方法があります。また、サーバに対しても、全体を一度に送信する方法と、SET 文を使用してデータベース変数に追加することで分割して送信する方法があります。データは、NULL で終了しません。

DT_LONGBINARY

長いバイナリデータ。マクロによって、構造体が次のように定義されます。

```
#define DECL_LONGBINARY( size )  ¥
  struct { a_sql_uint32    array_len;    ¥
           a_sql_uint32    stored_len;   ¥
           a_sql_uint32    untrunc_len;  ¥
           char            array[size];  ¥
         }
```

32 KB を超えるデータには、DECL_LONGBINARY 構造体を使用できます。データは、全体を一度にフェッチする方法と、GET DATA 文を使用して分割してフェッチする方法があります。また、サーバに対しても、全体を一度に送信する方法と、SET 文を使用してデータベース変数に追加することで分割して送信する方法があります。

これらの構造体は *%SQLANY17%*¥SDK¥Include¥sqlca.h ファイルに定義されています。VARCHAR 型、BINARY 型、TYPE_DECIMAL 型は、データ格納領域が長さ 1 の文字配列のため、ホスト変数の宣言には向いていません。しかし、動的な変数の割り付けや他の変数の型変換を行うのには有効です。

データベースの DATE 型と TIME 型

データベースのさまざまな DATE 型と TIME 型に対応する、Embedded SQL インタフェースのデータ型はありません。これらの型は、SQLDATETIME 構造体または文字列を使用してフェッチと更新を行います。

データベースの LONG VARCHAR 型と LONG BINARY 型に対応する、Embedded SQL インタフェースのデータ型はありません。

3.2.2.5.3 Ultra Light C++ でのホスト変数の使用方法

ホスト変数は、いくつかの特別な場合に使用できます。

- SELECT、INSERT、UPDATE、DELETE 文で数値定数または文字列定数を記述できる場所。
- SELECT または FETCH 文の INTO 句。
- CONNECT、DISCONNECT、SET CONNECT 文では、ユーザ ID、パスワード、接続名、データベース名の代わりにホスト変数を使用できます。

ホスト変数は、テーブル名、カラム名の代わりには使用できません。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

3.2.2.5.4 ホスト変数のスコープ

ホスト変数の宣言セクションは、C 変数を宣言できる通常の場所であれば、C の関数のパラメータの宣言セクションも含め、どこにでも記述できます。

C 変数は通常のスコープを持っています (定義されたブロック内で使用可能)。ただし、SQL プリプロセッサは C コードをスキャンしないため、C ブロックを重視しません。

プリプロセッサはすべてのホスト変数をグローバルとみなす

SQL プリプロセッサから見ると、ホスト変数はその宣言に従って、ソースモジュールに対してグローバルに認識されています。2 つのホスト変数が同じ名前を持つことはできません。この規則の例外として、2 つのホスト変数が同じ型 (必要な長さを含む) の場合、同じ名前を持つことができます。

ホスト変数ごとにユニークな名前を付けることが最善の方法であるといえます。

例

SQL プリプロセッサは C コードを解析できません。そのため、ホスト変数がどこで宣言されたかにかかわらず、宣言に従ってグローバルに認識されているとみなします。

```
// Example demonstrating poor coding
EXEC SQL BEGIN DECLARE SECTION;
   long emp_id;
EXEC SQL END DECLARE SECTION;
long getManagerID( void )
{
   EXEC SQL BEGIN DECLARE SECTION;
      long manager_id = 0;
   EXEC SQL END DECLARE SECTION;
   EXEC SQL SELECT manager_id
            INTO :manager_id
            FROM employee
            WHERE emp_number = :emp_id;
   return( manager_number );
}
void setManagerID( long manager_id )
{
   EXEC SQL UPDATE employee
            SET manager_number = :manager_id
            WHERE emp_number = :emp_id;
}
```

このコードは、`getManagerID` 内の文を処理するとき、SQL プリプロセッサが `setManagerID` 内の宣言に依存しているので、機能はするもののわかりにくくなっています。このコードを次のように書き換えます。

```
// Rewritten example
#if 0
   // Declarations for the SQL preprocessor
   EXEC SQL BEGIN DECLARE SECTION;
      long emp_id;
      long manager_id;
   EXEC SQL END DECLARE SECTION;
#endif
long getManagerID( long emp_id )
{
```


© 2016 SAP SE or an SAP affiliate company. All rights reserved.

```
    long manager_id = 0;
    EXEC SQL SELECT manager_id
             INTO :manager_id
             FROM employee
             WHERE emp_number = :emp_id;
    return( manager_number );
}
void setManagerID( long emp_id, long manager_id )
{
    EXEC SQL UPDATE employee
             SET manager_number = :manager_id
             WHERE emp_number = :emp_id;
}
```

SQL プリプロセッサは、これらのディレクティブを無視するため、#if ディレクティブにあるホスト変数の宣言を調べます。それに対し、プロシージャ内の宣言は、DECLARE SECTION 内にないため、無視されます。これとは逆に、C コンパイラは #if ディレクティブ内の宣言を無視し、プロシージャ内の宣言を使用します。

これらの宣言が機能するのは、同じ名前を持つ変数が同じ型を持つように宣言されているときだけです。

3.2.2.5.5 ホスト変数としての式

SQL プリプロセッサはポインタや参照式を認識しないため、ホスト変数は単純な名前でなければなりません。

たとえば、次の文では、SQL プリプロセッサがドット演算子を理解しないため、正しく機能しません。同じ構文が、SQL では違う意味を持ちます。

```
// Incorrect statement:
EXEC SQL SELECT LAST sales_id INTO :mystruct.mymember;
```

上記の構文は使用できませんが、次の方法で式を使用できます。

- SQL 宣言セクションを #if 0 プリプロセッサディレクティブで囲む。SQL プリプロセッサはプリプロセッサディレクティブを無視するため、この宣言を読み込み、残りのモジュールで使用します。
- マクロをホスト変数と同じ名前で定義します。#if ディレクティブがあるため C コンパイラからは SQL 宣言セクションが見えず、競合が起きません。マクロがホスト変数と同じ型であると評価されることを確認してください。

次のコードでは、SQL プリプロセッサから `host_value` 式を隠す方法が示されています。

```
#include <sqlerr.h>
#include <stdio.h>
EXEC SQL INCLUDE SQLCA;
typedef struct my_struct {
   long     host_field;
} my_struct;
#if 0
   // Because it ignores #if preprocessing directives,
   // SQLPP reads the following declaration.
   EXEC SQL BEGIN DECLARE SECTION;
      long    host_value;
   EXEC SQL END DECLARE SECTION;
#endif
// Make C/C++ recognize the 'host_value' identifier
// as a macro that expands to a struct field.
#define host_value my_s.host_field
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

SQLPP プロセッサは条件付きコンパイルのディレクティブを無視するため、host_value は long ホスト変数として扱われ、その後ホスト変数として使用されたときにその名前を生成します。C/C++ コンパイラはこの生成されたファイルを処理し、このように名前が使用されている場合には my_s.host_field に置き換えます。

前述の宣言を使用した状態で、次のように host_field にアクセスできます。

```
void main( void )
{
   my_struct      my_s;
   db_init( &sqlca );
   EXEC SQL CONNECT "DBA" IDENTIFIED BY "SQL";
   EXEC SQL DECLARE my_table_cursor CURSOR FOR
      SELECT int_col FROM my_table order by int_col;
   EXEC SQL OPEN my_table_cursor;
   for( ; ; ) {
      // :host_value references my_s.host_field
      EXEC SQL FETCH NEXT AllRows INTO :host_value;
      if( SQLCODE == SQLE_NOTFOUND ) {
         break;
      }
      printf( "%ld¥n", my_s.host_field );
   }
   EXEC SQL CLOSE my_table_cursor;
   EXEC SQL DISCONNECT;
   db_fini( &sqlca );
}
```

同じ方法で、他の lvalue をホスト変数として使用できます。

- ポインタの間接参照

  ```
  *ptr
  p_struct->ptr
  (*pp_struct)->ptr
  ```

- 配列参照

  ```
  my_array[ I ]
  ```

- 任意の複雑な lvalue

3.2.2.5.6 C++ でのホスト変数

一般に、クラスを別のヘッダファイルで頻繁に宣言するときに便利です。

たとえば、このヘッダファイルには、次のような my_class の宣言が含まれています。

```
typedef short a_bool;
#define  TRUE  ((a_bool)(1==1))
#define  FALSE ((a_bool)(0==1))
public class {
   long  host_member;
   my_class();    // Constructor
   ~my_class();      // Destructor
   a_bool FetchNextRow( void );
      // Fetch the next row into host_member
} my_class;
```


© 2016 SAP SE or an SAP affiliate company. All rights reserved.

この例では、各メソッドは、Embedded SQL ソースファイルに実装されます。簡単な変数だけがホスト変数として使用されます。あるクラスのデータメンバーへのアクセスに、前の項で説明した方法を使用できます。

```
EXEC SQL INCLUDE SQLCA;
#include "my_class.hpp"
#if 0
   // Because it ignores #if preprocessing directives,
   // SQLPP reads the following declaration.
   EXEC SQL BEGIN DECLARE SECTION;
      long  this_host_member;
   EXEC SQL END DECLARE SECTION;
#endif
// Macro used by the C++ compiler only.
#define this_host_member this->host_member
my_class::my_class()
{
   EXEC SQL DECLARE my_table_cursor CURSOR FOR
      SELECT int_col FROM my_table order by int_col;
   EXEC SQL OPEN my_table_cursor;
}
my_class::~my_class()
{
   EXEC SQL CLOSE my_table_cursor;
}
a_bool my_class::FetchNextRow( void )
{
   // :this_host_member references this->host_member
   EXEC SQL FETCH NEXT AllRows INTO :this_host_member;
   return( SQLCODE != SQLE_NOTFOUND );
}
void main( void )
{
   db_init( &sqlca );
   EXEC SQL CONNECT "DBA" IDENTIFIED BY "SQL";
   {
      my_class mc; // Created after connecting.
      while( mc.FetchNextRow() ) {
         printf( "%ld¥n", mc.host_member );
      }
   }
   EXEC SQL DISCONNECT;
   db_fini( &sqlca );
}
```

この例では、SQL プリプロセッサに `this_host_member` を宣言しますが、マクロで C++ がこの宣言を `this->host_member` に変換します。この変換が行われないと、プリプロセッサはこの変数の型を知ることができません。C/C++ コンパイラは通常重複した宣言を黙認しません。`#if` ディレクティブは、2 番目の宣言をコンパイラから隠しますが、SQL プリプロセッサからは見える状態を保ちます。

複数の宣言は便利ですが、各宣言が同じ型に同じ変数名を割り当てるようにしてください。プリプロセッサは、C 言語を完全に解析することができないため、宣言に従ってホスト変数がグローバルに認識されているとみなします。

3.2.2.5.7 インジケータ変数

インジケータ変数とは、特定のホスト変数に関する補足的な情報を保持する C 変数のことです。ホスト変数は、データのやりとりをするときに使用できます。NULL 値を扱うには、インジケータ変数を使用します。

インジケータ変数は a_sql_len 型のホスト変数で、SQL 文では通常のホスト変数の直後に書きます。NULL 値を検出したり指定したりするため、SQL 文では通常のホスト変数の直後にインジケータ変数を記述します。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

例

たとえば、次の INSERT 文では、:ind_phone がインジケータ変数です。

```
EXEC SQL INSERT INTO Employee
   VALUES (:employee_number, :employee_name,
   :employee_initials, :employee_phone:ind_phone );
```

フェッチ時または実行時にデータベースサーバからローを受信しなかった場合 (エラーが発生したか、結果セットの末尾に到達した場合)、インジケータの値は変更されません。

注記

32 ビット長および 64 ビット長とインジケータを今後使用できるように、Embedded SQL インジケータ変数での short int の使用は推奨されません。代わりに a_sql_len を使用してください。

インジケータ変数の値

次の表は、インジケータ変数の使用法をまとめたものです。

インジケータの値	データベースに渡す値	データベースから受け取る値
0	ホスト変数値	NULL でない値をフェッチした値
-1	NULL 値	NULL 値をフェッチした値

このセクションの内容:

NULL を扱うためのインジケータ変数 [651 ページ]
SQL での NULL を同じ名前の C 言語の定数と混同しないでください。SQL 言語では、NULL は属性が不明であるか情報が適切でないかのいずれかを表します。C 言語の定数は、ポイント先がメモリのロケーションではないポインタ値を表します。

3.2.2.5.7.1 NULL を扱うためのインジケータ変数

SQL での NULL を同じ名前の C 言語の定数と混同しないでください。SQL 言語では、NULL は属性が不明であるか情報が適切でないかのいずれかを表します。C 言語の定数は、ポイント先がメモリのロケーションではないポインタ値を表します。

SQL Anywhere のマニュアルで使用されている NULL は、上記の SQL データベースでの意味を指します。C 言語の定数を指す場合は、null ポインタ (小文字) のように表記されます。

NULL は、カラムに定義されるどのデータ型の値とも同じではありません。インジケータ変数がデータベースに NULL 値を渡すか、NULL の結果をを受け取る必要があります。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

NULL を挿入する場合のインジケータ変数

INSERT 文は、次のようにインジケータ変数を含むことができます。

```
EXEC SQL BEGIN DECLARE SECTION;
short int employee_number;
char employee_name[50];
char employee_initials[6];
char employee_phone[15];
a_sql_len ind_phone;
EXEC SQL END DECLARE SECTION;
/* set values of employee number, name,
   initials, and phone number */
if( /* phone number is known */ ) {
   ind_phone =  0;
} else {
   ind_phone = -1; /* NULL */
}
EXEC SQL INSERT INTO Employee
   VALUES (:employee_number, :employee_name,
   :employee_initials, :employee_phone:ind_phone );
```

インジケータ変数の値が -1 の場合は、NULL が書き込まれます。値が 0 の場合は、employee_phone の実際の値が書き込まれます。

NULL をフェッチする場合のインジケータ変数

インジケータ変数は、データをデータベースから受け取るときにも使用されます。この場合は、NULL 値がフェッチされた (インジケータが負) ことを示すために使用されます。NULL 値がデータベースからフェッチされたときにインジケータ変数が渡されない場合は、SQLE_NO_INDICATOR エラーが発生します。

関連情報

SQL Communications Area の初期化 [639 ページ]

3.2.2.6 データのフェッチ

Embedded SQL でデータをフェッチするには SELECT 文を使用します。

返されるローには、次の 2 つの場合があります。

1. SELECT 文がローをまったく返さないか、1 つだけローを返す場合。
2. SELECT 文が複数のローを返す場合。

このセクションの内容:

シングルローフェッチ [653 ページ]

 © 2016 SAP SE or an SAP affiliate company.All rights reserved.

"シングルロークエリ" がデータベースから取り出すローの数は多くても 1 つだけです。

複数ローのフェッチ [654 ページ]
カーソルは、結果セットに複数のローがあるクエリからローを取り出すために使用されます。カーソルは、SQL クエリ結果セットのためのハンドルつまり識別子であり、結果セット内の位置を示します。

3.2.2.6.1 シングルローフェッチ

"シングルロークエリ" がデータベースから取り出すローの数は多くても 1 つだけです。

シングルロークエリの SELECT 文では、INTO 句が select リストの後、FROM 句の前にきます。INTO 句には、select リストの各項目の値を受け取るホスト変数のリストを指定します。select リスト項目と同数のホスト変数を指定してください。ホスト変数と一緒に、結果が NULL であることを示すインジケータ変数も指定できます。

SELECT 文が実行されると、データベースサーバは結果を取り出して、ホスト変数に格納します。

- クエリが複数のローを返すと、データベースサーバは SQLE_TOO_MANY_RECORDS エラーを返す。
- クエリがローを返さなかった場合、警告 SQLE_NOTFOUND が返されます。

例

たとえば、次のコードは employee テーブルから正しくローをフェッチできた場合は 1 を、ローが存在しない場合は 0 を、エラーが発生した場合は -1 を返します。

```
EXEC SQL BEGIN DECLARE SECTION;
   long int    emp_id;
   char        name[41];
   char        sex;
   char        birthdate[15];
   a_sql_len   ind_birthdate;
EXEC SQL END DECLARE SECTION;
int find_employee( long employee )
{
   emp_id = employee;
   EXEC SQL SELECT emp_fname || ' ' || emp_lname,
         sex, birth_date
      INTO :name, :sex, birthdate:ind_birthdate
      FROM "DBA".employee
      WHERE emp_id = :emp_id;
   if( SQLCODE == SQLE_NOTFOUND ) {
      return( 0 ); /* employee not found */
   } else if( SQLCODE < 0 ) {
      return( -1 ); /* error */
   } else {
      return( 1 ); /* found */
   }
}
```

関連情報

SQL Communications Area の初期化 [639 ページ]

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

3.2.2.6.2 複数ローのフェッチ

カーソルは、結果セットに複数のローがあるクエリからローを取り出すために使用されます。カーソルは、SQL クエリ結果セットのためのハンドルつまり識別子であり、結果セット内の位置を示します。

Ultra Light アプリケーションのカーソルは、常に WITH HOLD オプションを使用して開かれます。自動的に閉じられることはありません。CLOSE 文を使用して、各カーソルを明示的に閉じます。

次の手順でカーソルを管理します。

1. DECLARE 文を使って、特定の SELECT 文のためのカーソルを宣言します。
2. OPEN 文を使って、カーソルを開きます。
3. FETCH 文を使用して、一度に 1 つのローをカーソルから取り出します。
4. 警告 SQLE_NOTFOUND が返されるまで、ローをフェッチします。エラーコードと警告のコードは、SQL Communications Area 構造体で定義される変数 SQLCODE で返されます。
5. CLOSE 文を使ってカーソルを閉じます。

次は、簡単なカーソル使用の例です。

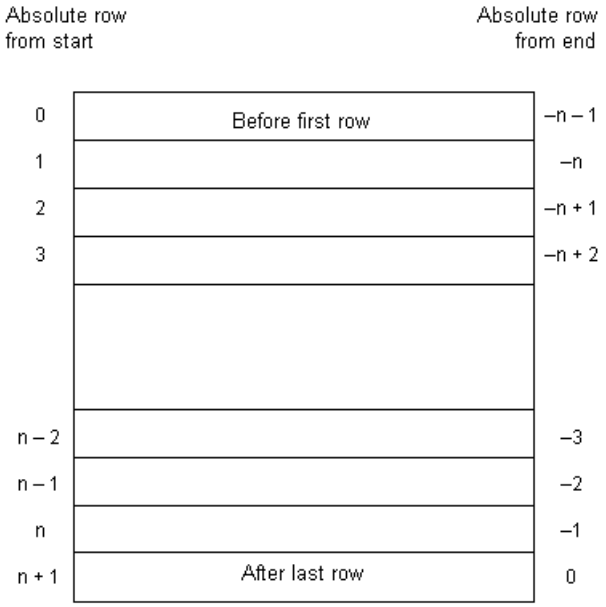
```
void print_employees( void )
{
   int status;
   EXEC SQL BEGIN DECLARE SECTION;
   char name[50];
   char sex;
   char birthdate[15];
   a_sql_len   ind_birthdate;
   EXEC SQL END DECLARE SECTION;
   /* 1. Declare the cursor. */
   EXEC SQL DECLARE C1 CURSOR FOR
      SELECT emp_fname || ' ' || emp_lname,
               sex, birth_date
      FROM "DBA".employee
      ORDER BY emp_fname, emp_lname;
   /* 2. Open the cursor. */
   EXEC SQL OPEN C1;
   /* 3. Fetch each row from the cursor. */
   for( ;; ) {
      EXEC SQL FETCH C1 INTO :name, :sex,
            :birthdate:ind_birthdate;
      if( SQLCODE == SQLE_NOTFOUND ) {
         break; /* no more rows */
      } else if( SQLCODE < 0 ) {
         break; /* the FETCH caused an error */
      }
      if( ind_birthdate < 0 ) {
         strcpy( birthdate, "UNKNOWN" );
      }
      printf( "Name: %s Sex: %c Birthdate:
               %s¥n",name, sex, birthdate );
   }
   /* 4. Close the cursor. */
   EXEC SQL CLOSE C1;
}
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

カーソル位置

カーソルは、次のいずれかの位置にあります。

- ローの上
- 最初のローの前
- 最後のローの後



カーソル内のローの順序

カーソル内のローの順序を制御するには、そのカーソルを定義する SELECT 文に ORDER BY 句を含めます。この句を省略すると、ローの順序が制御できなくなります。

明示的に順序を定義しない場合は、SQLE_NOTFOUND が返される前に一度だけ、フェッチごとに各ローが結果セットに返されます。

カーソルの再配置

カーソルを開くと、最初のローの前に置かれます。FETCH 文が自動的にカーソル位置を進めます。最後のローより後で FETCH 文を実行しようとすると、SQLE_NOTFOUND エラーが発生します。これは、ローの連続処理を完了するための信号として利用できます。

カーソルはクエリ結果の先頭または末尾を基準にした絶対位置に再配置できます。また、カーソルの現在位置を基準にした相対位置にも移動できます。カーソルの現在位置のローを更新または削除するために、特別な位置付け型の UPDATE 文と

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

DELETE 文があります。先頭のローの前か、末尾のローの後にカーソルがある場合、SQLE_NOTFOUND エラーが返されます。

明示的な位置付けを行って予期しない結果が出るのを避けるには、そのカーソルを定義する SELECT 文に ORDER BY 句を含めます。

カーソルにローを挿入するには、PUT 文を使用します。

更新後のカーソル位置

開かれたカーソルがアクセスしている情報を変更した場合は、そのローをもう一度フェッチして表示するのが最適な方法です。単一のローの表示にカーソルが使用されている場合は、FETCH RELATIVE 0 が現在のローを再度フェッチします。現在のローが削除されていた場合は、次のローがカーソルからフェッチされます。ローがこれ以上ない場合は、SQLE_NOTFOUND が返されます。

テンポラリテーブルがカーソルに使用されている場合、基本となるテーブルに挿入されたローは、カーソルが閉じられて再び開かれるまでまったく表示されません。通常、SQL プリプロセッサが生成したコードを検査したり、テンポラリテーブルが使用される条件に精通していたりしないかぎり、SELECT 文にテンポラリテーブルが含まれているかどうかを検出するのは困難です。ORDER BY 句で使用されるカラムにインデックスを設定することによって、通常はテンポラリテーブルを回避できます。

非テンポラリテーブルに対する挿入、更新、削除は、カーソル位置に影響を及ぼすことがあります。Ultra Light では、新しくローにデータが挿入されたり、ローが新しく削除されてデータがなくなったりしている場合には、その後の FETCH 操作に影響を及ぼします。これは、テンポラリテーブルが使用されていない場合、カーソルローを一度に 1 つだけ表示するためです。(一部の) ローが単一のテーブルから選択されている簡単な例では、挿入または更新されたローが SELECT 文の選択基準を満たす場合、そのローはカーソルの結果セットに表示されます。同様に、結果セットに表示されたローが新しく削除されると、そのローは結果セットに表示されなくなります。

3.2.2.7 ユーザ認証

ユーザ認証は、ULGrantConnectTo メソッドと ULRevokeConnectFrom メソッドを使用して制御できます。

次のコードは、ユーザ認証を制御する方法を示しています。

```
//Embedded SQL
app() {
   ...
/* Declare fields */
   EXEC SQL BEGIN DECLARE SECTION;
      char uid[31];
      char pwd[31];
   EXEC SQL END DECLARE SECTION;
   db_init( &sqlca );
   ...
   EXEC SQL CONNECT "DBA" IDENTIFIED BY "sql";
   if( SQLCODE == SQLE_NOERROR ) {
      printf("Enter new user ID and password¥n" );
      scanf( "%s %s", uid, pwd );
      ULGrantConnectTo( &sqlca,
         UL_TEXT( uid ), UL_TEXT( pwd ) );
      if( SQLCODE == SQLE_NOERROR ) {
         // new user added: remove DBA
         ULRevokeConnectFrom( &sqlca, UL_TEXT("DBA") );
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

```
      }
      EXEC SQL DISCONNECT;
   }
   // Prompt for password
    printf("Enter user ID and password¥n" );
    scanf( "%s %s", uid, pwd );
    EXEC SQL CONNECT :uid IDENTIFIED BY :pwd;
```

このコードでは、次のタスクを実行します。

1. db_init を呼び出してデータベースの機能を開始します。
2. デフォルトのユーザ ID とパスワードを使用して接続します。
3. 接続に成功したら、新しいユーザを追加します。
4. 新しいユーザが追加されたら、Ultra Light データベースから DBA を削除します。
5. 切断します。更新されたユーザ ID とパスワードがデータベースに追加されます。
6. 更新されたユーザ ID とパスワードを使用して接続します。

3.2.2.8 Ultra Light Embedded SQL でのデータの暗号化

Ultra Light Embedded SQL を使用して、Ultra Light データベースを暗号化したり、難読化したりできます。

暗号化

Ultra Light データベースを (SQL Central などを使用して) 作成する場合は、オプションで暗号化キーを指定できます。暗号化キーは、データベースの暗号化に使用されます。データベースが暗号化されると、その後のすべての接続で暗号化キーの指定が必要になります。指定されたキーが元の暗号化キーと照合され、キーが一致しないと接続は失敗します。

暗号化キーには簡単に推測できる値を選択しないでください。キーの長さは任意ですが、短いと推測されやすいため、一般的には長い方が適しています。数字、文字、特殊文字を組み合わせると、キーは推測されにくくなります。

キーにはセミコロンを含めないでください。キー自体を引用符で囲まないでください。キーを引用符で囲むと、引用符がキーの一部と認識されてしまいます。

暗号化された Ultra Light への一般的な接続手順は次のとおりです。

1. EXEC SQL CONNECT 文に指定されている接続文字列で暗号化キーを指定します。
2. 暗号化キーは、key= 接続文字列パラメータの形で指定します。
   このキーは、データベースに接続するたびに指定する必要があります。キーを忘れた場合はデータベースにまったくアクセスできなくなります。
3. 間違ったキーを使用して暗号化されたデータベースを開こうとしてみてください。
   暗号化されたデータベースを開こうとして、間違ったキーが渡されると、db_init が ul_false を返し、SQLCODE -840 が設定されます。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

暗号化キーを変更します。

データベースの暗号化キーは変更できます。既存のキーを使用してアプリケーションをデータベースに接続してから、変更を行ってください。

ULChangeEncryptionKey メソッドの引数として新しいキーを入力します。

難読化

データベースの暗号化の代わりに、データベースの難読化という方法があります。難読化とは、データベースのデータを簡単にマスキングすることで、低レベルのファイル検証ユーティリティを使用してデータベース内のデータが見られても内容がわからないようにすることです。しかしながら難読化は、巧妙で容赦ないデータへのアクセス試行に対してセキュアではありません。難読化はデータベース作成オプションの 1 つで、データベースの作成時に指定する必要があります。

3.2.2.9 Embedded SQL アプリケーションの同期の設定

多くの Ultra Light アプリケーションにとって、同期は重要な機能です。Embedded SQL API の構造体のメンバーは、Ultra Lite C++ API と似ています。

Ultra Light アプリケーションを統合データベースの最新状態と同期する論理は、アプリケーション自体にはありません。統合データベースに格納されている同期スクリプトは、Mobile Link サーバと Ultra Light ランタイムライブラリとともに、変更のアップロード時に変更をどのように処理するかを制御し、ダウンロードする変更はどれかを決定します。

同期ごとの詳細は、同期パラメータのセットによって制御されます。これらのパラメータは、構造体に収集された後、メソッド呼び出しの引数として渡され、同期されます。このメソッドの概要は、どの開発モデルでも同じです。

アプリケーションに同期機能を追加するには、通常、次の手順を行います。

1. 同期パラメータが格納された構造体を初期化します。
2. アプリケーションのパラメータ値を割り当てます。
3. 同期メソッドを呼び出し、構造体またはオブジェクトを引数として指定します。
   同期するときに、コミットされていない変更がないことを確認してください。

このセクションの内容:

同期パラメータの初期化 [659 ページ]
同期パラメータは、ul_sync_info 構造体に格納されます。

同期の呼び出し [659 ページ]
同期を呼び出す方法は、ターゲットプラットフォームと同期ストリームによって細かく異なります。

コミットされた変更と同期 [660 ページ]
Ultra Light データベースは、同期のときに変更をコミットしないでおくことはできません。

アプリケーションの初期データ [660 ページ]
Ultra Light アプリケーションは、一般的に、使用前にデータが必要です。同期でアプリケーションにデータをダウンロードできます。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

同期通信エラー [660 ページ]
SQLCODE を使用して、Mobile Link の同期エラーをレポートできます。

同期のモニタとキャンセル [661 ページ]
Ultra Light アプリケーションから、同期のモニタおよびキャンセルができます。

3.2.2.9.1 同期パラメータの初期化

同期パラメータは、ul_sync_info 構造体に格納されます。

ul_sync_info 構造体のメンバーは、初期化時には未定義です。構造体のメンバーの初期値にパラメータを設定し、特別なメソッドを呼び出します。同期パラメータは、Ultra Light ヘッダファイル *%SQLANY17%*¥SDK¥Include¥ulglobal.h で宣言された構造体で定義されます。

例

次の例は、ULInitSyncInfo メソッドで同期パラメータを初期化する方法を示しています。

```
ul_sync_info synch_info;
ULInitSyncInfo( &synch_info );
```

例

次のコードは、TCP/IP の同期を開始します。Mobile Link ユーザ名は Betty Best、パスワードは TwentyFour、スクリプトバージョンは default です。Mobile Link サーバはホストコンピュータ test.internal で実行されており、ポート 2439 を使用しています。

```
ul_sync_info synch_info;
ULInitSyncInfo( &synch_info );
synch_info.user_name = UL_TEXT("Betty Best");
synch_info.password = UL_TEXT("TwentyFour");
synch_info.version = UL_TEXT("default");
synch_info.stream = ULSocketStream();
synch_info.stream_parms =
   UL_TEXT("host=test.internal;port=2439");
ULSynchronize( &sqlca, &synch_info );
```

3.2.2.9.2 同期の呼び出し

同期を呼び出す方法は、ターゲットプラットフォームと同期ストリームによって細かく異なります。

同期処理が機能するのは、Ultra Light アプリケーションを実行するデバイスが Mobile Link サーバと通信できる場合だけです。プラットフォームによっては、デバイスを、クレードルに置くかまたは適切なケーブルを使用してサーバコンピュータに接続して、物理的に接続する必要があります。同期を完了できない場合は、エラー処理コードをアプリケーションに追加します。

同期を呼び出すには、ULInitSyncInfo メソッドを呼び出して、同期パラメータを初期化してから、ULSynchronize メソッドを呼び出して、同期化します。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

同期呼び出しでは、その同期固有の情報が記述されたパラメータのセットを保持している構造体が必要です。使用される特定のパラメータは、ストリームによって異なります。

3.2.2.9.3 コミットされた変更と同期

Ultra Light データベースは、同期のときに変更をコミットしないでおくことはできません。

Ultra Light データベースの同期をとろうとした時点で、コミットされていないトランザクションが接続にあると、同期は失敗し、例外がスローされ、SQLE_UNCOMMITTED_TRANSACTIONS エラーが設定されます。このエラーコードは、Mobile Link サーバログにも表示されます。

3.2.2.9.4 アプリケーションの初期データ

Ultra Light アプリケーションは、一般的に、使用前にデータが必要です。同期でアプリケーションにデータをダウンロードできます。

アプリケーションが最初に実行されたとき、他のアクションが行われる前に必要なデータがすべてダウンロードされるように、アプリケーションに論理を追加します。

> **注記**
>
> アプリケーションを段階別に開発すると、エラーが発見しやすくなります。プロトタイプの開発中に、テストとデモンストレーションを目的としたデータを得るため、一時的に INSERT 文をアプリケーションで使用します。プロトタイプが正常に動作することを確認したら、一時的な INSERT 文を同期を実行するコードに置き換えます。

3.2.2.9.5 同期通信エラー

SQLCODE を使用して、Mobile Link の同期エラーをレポートできます。

次のサンプルコードは、Embedded SQL アプリケーションから通信エラーを処理する方法を示しています。

```
if( psqlca->sqlcode == SQLE_MOBILINK_COMMUNICATIONS_ERROR ) {
     printf( "  Stream error information:¥n"
             "    stream_error_code = %ld¥t(ss_error_code)¥n"
             "    error_string      = ¥"%s¥"¥n"
             "    system_error_code = %ld¥n",
     (long)info.stream_error.stream_error_code,
     info.stream_error.error_string,
     (long)info.stream_error.system_error_code );
}
```

SQLE_MOBILINK_COMMUNICATIONS_ERROR は、通信エラーを表す一般的なエラーコードです。

Ultra Light のサイズを小さくするために、ランタイムによるレポートは、メッセージではなく数値で行われます。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

3.2.2.9.6 同期のモニタとキャンセル

Ultra Light アプリケーションから、同期のモニタおよびキャンセルができます。

同期のモニタ

- 同期構造体 (ul_synch_info) の observer メンバーのコールバック関数の名前を指定します。
- 同期関数かメソッドを呼び出して同期を開始します。
- Ultra Light が、同期のステータスが変更するたびにコールバック関数を呼び出します。

次のコードは、Embedded SQL アプリケーションでこのタスクのシーケンスをどのように実装できるかを示しています。

```
ULInitSyncInfo( &info );
info.user_name = m_EmpIDStr;
...
//The info parameter of ULSynchronize() contains
// a pointer to the observer function
info.observer = ObserverFunc;
ULSynchronize( &sqlca, &info );
```

このセクションの内容:

同期ステータス情報 [661 ページ]
同期をモニタするコールバック関数は、ul_sync_status 構造体をパラメータとして取ります。

3.2.2.9.6.1 同期ステータス情報

同期をモニタするコールバック関数は、ul_sync_status 構造体をパラメータとして取ります。

sent.inserts

これまでにアップロードされた挿入済みローの数。
sent.updates

これまでにアップロードされた更新済みローの数。
sent.deletes

これまでにアップロードされた削除済みローの数。
sent.bytes

これまでにアップロードされたバイト数。
received.inserts

これまでにダウンロードされた挿入済みローの数。
received.updates

これまでにダウンロードされた更新済みローの数。
received.deletes

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

これまでにダウンロードされた削除済みローの数。
received.bytes

これまでにダウンロードされたバイト数。
info

ul_sync_info 構造体にポインタを返します。
db_table_count

データベース内のテーブルの数を返します。
table_id

現在アップロードまたはダウンロードされているテーブルの番号 (1 から始まります)を返します。同期されないテーブルがある場合には、この番号で値がスキップされることがあります。また、番号が必ず増加するとはかぎりません。
table_name[]

現在のテーブルの名前を返します。
table_name_w2[]

現在のテーブルの名前 (ワイド文字バージョン)を返します。このフィールドには、Windows (デスクトップまたは Mobile) 環境の場合のみ値が入ります。
sync_table_count

同期中のテーブルの数を返します。
sync_table_index

アップロードまたはダウンロードされているテーブルの番号を返します。この番号は 1 から始まり sync_table_count の値で終わります。同期されていないテーブルがある場合には、この番号で値がスキップされることがあります。
state

以下のステータスのいずれかを表します。

UL_SYNC_STATE_STARTING

同期処理はまだ行われていません。
UL_SYNC_STATE_CONNECTING

同期ストリームは構築されていますが、まだ開かれていません。
UL_SYNC_STATE_RESUMING_DOWNLOAD

部分的なダウンロードを再開しようとすることを示すオプションの状態。成功した場合、同期は UL_SYNC_STATE_RECEIVING_TABLE ステータスに進みます。
UL_SYNC_STATE_SENDING_HEADER

同期ストリームが開かれ、ヘッダが送信されようとしています。
UL_SYNC_STATE_SENDING_CHECK_SYNC_REQUEST

前回のアップロードの状態が不明なため、ステータスをチェックする要求が送信されます。
UL_SYNC_STATE_WAITING_CHECK_SYNC_REQUEST

クライアントは、サーバが同期チェック要求に応答するのを待機しています。
UL_SYNC_STATE_PROCESSING_CHECK_SYNC_REQUEST

同期チェック要求への応答が受け取られ、処理されています。
UL_SYNC_STATE_SENDING_TABLE

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

テーブルが送信されています。
UL_SYNC_STATE_SENDING_DATA

スキーマ情報またはデータが送信されています。
UL_SYNC_STATE_FINISHING_UPLOAD

アップロード処理が完了し、コミットが実行されています。
UL_SYNC_STATE_WAITING_UPLOAD_ACK

クライアントは、サーバがアップロードの受信を確認するのを待機しています。
UL_SYNC_STATE_PROCESSING_UPLOAD_ACK

サーバがアップロードの受信を確認しました。
UL_SYNC_STATE_WAITING_FOR_DOWNLOAD

クライアントは、サーバがダウンロードの送信を開始するのを待機しています。
UL_SYNC_STATE_RECEIVING_TABLE

テーブルを受信しています。
UL_SYNC_STATE_RECEIVING_DATA

スキーマ情報またはデータを受信しています。
UL_SYNC_STATE_COMMITTING_DOWNLOAD

ダウンロード処理が完了し、コミットが実行されています。
UL_SYNC_STATE_SENDING_DOWNLOAD_ACK

ダウンロード完了の確認が送信されています。
UL_SYNC_STATE_DISCONNECTING

同期ストリームが閉じられようとしています。
UL_SYNC_STATE_DONE

同期は正常に完了しました。
UL_SYNC_STATE_ERROR

同期は完了しましたが、エラーが発生しました。
UL_SYNC_STATE_ROLLING_BACK_DOWNLOAD

ダウンロード中にエラーが発生し、ダウンロードがロールバックされています。

stop

同期を中断するには、このメンバーを true に設定します。SQL 例外の SQLE_INTERRUPTED が設定され、通信エラーが発生したかのように同期が停止します。observer は、適切なクリーンアップを実行するように、常に DONE または ERROR のステータスで呼び出されます。

flags

現在の状態に関連する追加情報を示す、現在の同期フラグを返します。

user_data

引数として ULSetSynchronizationCallback 関数に渡されるユーザデータオブジェクトを返します。

sqlca

接続のアクティブな SQLCA へのポインタを返します。

current_download_row_count

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

これまでにダウンロードされたローの数を返します。この数には、received.inserts、received.updates、または received.deletes に含まれていない、重複するローが含まれます。

total_download_row_count

ダウンロードで受信するローの合計数を返します。この数には、received.inserts、received.updates、または received.deletes に含まれていない、重複するローが含まれます。

例

次のコードは、ごく簡単な observer 関数を示しています。

```
extern void __stdcall ObserverFunc(
    p_ul_sync_status status )
{
    switch( status->state ) {
        case UL_SYNC_STATE_STARTING:
            printf( "Starting¥n");
            break;
        case UL_SYNC_STATE_CONNECTING:
            printf( "Connecting¥n"  );
            break;
        case UL_SYNC_STATE_SENDING_HEADER:
            printf( "Sending Header¥n" );
            break;
        case UL_SYNC_STATE_SENDING_TABLE:
            printf( "Sending Table %d of %d¥n",
                    status->tableIndex + 1,
                    status->tableCount );
            break;
        case UL_SYNC_STATE_RECEIVING_TABLE:
            printf( "Receiving Table %d of %d¥n",
                    status->tableIndex + 1,
                    status->tableCount );
            break;
        case UL_SYNC_STATE_SENDING_DOWNLOAD_ACK:
            printf( "Sending Download Ack¥n" );
            break;
        case UL_SYNC_STATE_DISCONNECTING:
            printf( "Disconnecting¥n" );
            break;
        case UL_SYNC_STATE_DONE:
            printf( "Done¥n" );
            break;
...
```

この observer 関数では、2 つのテーブルが同期されたときに次のような出力が生成されます。

```
Starting
Connecting
Sending Header
Sending Table 1 of 2
Sending Table 2 of 2
Receiving Upload Ack
Receiving Table 1 of 2
Receiving Table 2 of 2
Sending Download Ack
Disconnecting
Done
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

CustDB の例

observer 関数の例は CustDB サンプルアプリケーションに含まれています。CustDB を使った実装では、同期の進捗状況を示すウィンドウが表示されます。ユーザはそのウィンドウで同期をキャンセルすることができます。ユーザインタフェースコンポーネントは observer 関数のプラットフォームを指定します。

CustDB サンプルコードは、*%SQLANYSAMP17%*¥UltraLite¥CustDB ディレクトリにあります。observer 関数は、CustDB ディレクトリのプラットフォーム固有のサブディレクトリに保管されています。

3.2.2.10 Embedded SQL アプリケーションの構築

Embedded SQL アプリケーションを構築するには、いくつかの設定に関する知識が必要です。

このセクションの内容:

一般的な構築手順 [665 ページ]
Ultra Light Embedded SQL アプリケーションの構築は、SQL プリプロセッサの実行、ソースコードのコンパイル、およびその後のオブジェクトファイルへのリンクから成ります。

Embedded SQL 開発用の開発ツールの設定 [666 ページ]
多くの開発ツールは依存モデルを使用しています。これは makefile として表現されることもあり、ソースファイルのタイムスタンプが、ターゲットファイル (通常はオブジェクトファイル) のタイムスタンプと比較され、ターゲットファイルを再生成すべきかどうかが決定されます。

3.2.2.10.1 一般的な構築手順

Ultra Light Embedded SQL アプリケーションの構築は、SQL プリプロセッサの実行、ソースコードのコンパイル、およびその後のオブジェクトファイルへのリンクから成ります。

1. 各 Embedded SQL ソースファイルに対して SQL プリプロセッサを実行します。
   SQL プリプロセッサは sqlpp コマンドラインユーティリティです。Embedded SQL ソースファイルの前処理を実行し、アプリケーションにコンパイルする C++ ソースファイルを生成します。

   > ⚠ 警告
   >
   > sqlpp は出力ファイルをその内容に関係なく上書きします。出力ファイルの名前が、どのソースファイルの名前とも一致していないことを確認してください。sqlpp は、デフォルトでは、ソースファイルのサフィックスを .cpp に変更して出力ファイル名を作成します。一致するファイル名があるかどうかはっきり分からない場合は、ソースファイルの名前に従って出力ファイルの名前を明示的に指定してください。

2. 各 C++ ソースファイルを、選択したターゲットプラットフォームに合わせてコンパイルします。次のファイルを含めます。
   - SQL プリプロセッサが生成した各 C++ ファイル
   - アプリケーションで必要な追加の C または C++ ソースファイル
3. これらすべてのオブジェクトファイルを Ultra Light ランタイムライブラリとともにリンクします。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

3.2.2.10.2 Embedded SQL 開発用の開発ツールの設定

多くの開発ツールは依存モデルを使用しています。これは makefile として表現されることもあり、ソースファイルのタイムスタンプが、ターゲットファイル (通常はオブジェクトファイル) のタイムスタンプと比較され、ターゲットファイルを再生成すべきかどうかが決定されます。

Ultra Light の開発では、開発プロジェクトで SQL 文が変更されると、生成コードを再生成する必要があります。SQL 文はリファレンスデータベースに保存されるため、個々のソースファイルのタイムスタンプには変更が反映されません。

このセクションの内容:

SQL プリプロセッサの実行 [666 ページ]
Microsoft Visual C++.用のプリプロセッサを実行する命令を追加することにより、SQL プリプロセッサを依存ベースの構築環境に組み込みます。

3.2.2.10.2.1 SQL プリプロセッサの実行

Microsoft Visual C++.用のプリプロセッサを実行する命令を追加することにより、SQL プリプロセッサを依存ベースの構築環境に組み込みます。

手順

1. .sqc ファイルを開発プロジェクトに追加します。

   開発プロジェクトは、開発ツールで定義されています。
2. 各 .sqc ファイルのカスタム構築規則を追加します。
   - カスタム構築規則により SQL プリプロセッサを実行してください。Microsoft Visual C++ の場合、構築規則には次のコマンドを含めてください (すべて 1 行に入力)。

     ```
     "%SQLANY17%¥Bin32¥sqlpp.exe" -q -u $(InputPath) $(InputName).cpp
     ```

     SQLANY17 は SQL Anywhere インストールフォルダを指す環境変数です。
   - コマンドの出力を $(InputName).cpp に設定します。
3. .sqc ファイルをコンパイルし、生成された .cpp ファイルを開発プロジェクトに追加します。

   生成されたファイルはソースファイルではありませんが、プロジェクトに追加する必要があります。これは、依存性と構築のオプションを設定できるようにするためです。
4. 生成された .cpp ファイルごとに、プリプロセッサの定義を設定します。
   - [全般] または [プリプロセッサ] で、[プリプロセッサ] の定義に UL_USE_DLL を追加します。
   - [プリプロセッサ] に、$(SQLANY*17*)¥SDK¥Include と、インクルードパスに必要なインクルードフォルダを、カンマ区切りのリストとして追加します。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

結果

Microsoft Visual C++ 開発用に SQL プリプロセッサが設定されます。

3.2.3 Windows Mobile 向け Ultra Light アプリケーション開発

Microsoft Visual Studio 2005 以降は、Windows Mobile 環境用のアプリケーション開発に使用できます。

Windows Mobile を対象にしたアプリケーションでは、wchar_t のデフォルト設定を使用し、`¥Program Files ¥SQLAny17¥ultralite¥ce¥arm.50¥lib¥` 内にある Ultra Light ランタイムライブラリでリンクする必要があります。

ほとんどの Windows Mobile ターゲットプラットフォームのエミュレータで、アプリケーションをテストできます。

このセクションの内容:

CustDB のサンプルアプリケーション [667 ページ]
CustDB は、Microsoft Visual Studio ソリューションとして提供される簡単な販売管理アプリケーションです。CustDB は SQL Anywhere インストールディレクトリの *%SQLANYSAMP17%*`¥UltraLite¥` ディレクトリにあります。

永続的なデータ [669 ページ]
Ultra Light データベースは Windows Mobile ファイルシステムに格納されます。

アプリケーションに対するクラス名の割り当て [669 ページ]
MFC を使用している場合は、アプリケーションに固有のクラス名を割り当てます。ActiveSync で使用するアプリケーションを登録するには、ウィンドウクラス名を指定します。

Windows Mobile の同期 [671 ページ]
Microsoft Windows Mobile 上の Ultra Light アプリケーションは、Microsoft ActiveSync、TCP/IP、および HTTP ストリームを介して同期できます。

3.2.3.1 CustDB のサンプルアプリケーション

CustDB は、Microsoft Visual Studio ソリューションとして提供される簡単な販売管理アプリケーションです。CustDB は SQL Anywhere インストールディレクトリの *%SQLANYSAMP17%*`¥UltraLite¥` ディレクトリにあります。

> **注記**
>
> サンプルプロジェクトでは、可能なかぎり環境変数を使用しています。アプリケーションを正しく構築するために、プロジェクトの調整が必要になることもあります。問題が発生した場合は、Microsoft Visual C++ のフォルダに足りないファイルがないかどうか調べ、適切なフォルダ設定を追加してみてください。

このセクションの内容:

CustDB サンプルアプリケーションの構築 [668 ページ]

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

CustDB のサンプルアプリケーションを構築すると、Ultra Light データベースとのインタフェースとしてのアプリケーションの機能がわかります。

関連情報

Embedded SQL アプリケーションの構築 [665 ページ]

3.2.3.1.1 CustDB サンプルアプリケーションの構築

CustDB のサンプルアプリケーションを構築すると、Ultra Light データベースとのインタフェースとしてのアプリケーションの機能がわかります。

手順

1. Microsoft Visual Studio を起動します。
2. プロジェクトファイルを開きます。このファイルは、*%SQLANYSAMP17%*¥UltraLite¥CustDB ディレクトリにあります。
3. ビルド > 構成マネージャの設定 をクリックしてターゲットプラットフォームを設定します。

   使用するアクティブソリューションプラットフォームを設定します。
4. アプリケーションを構築します。
   - ビルド > ソリューションの配置 をクリックして CustDB をビルドおよび展開します。

   アプリケーションがビルドされると、リモートデバイスに自動的にアップロードされます。
5. Mobile Link サーバを起動します。
   - Mobile Link サーバを起動するには、スタート > プログラム > *SQL Anywhere17* > *Mobile Link* > 同期サーバのサンプル をクリックします。
6. CustDB アプリケーションを実行します。

   CustDB アプリケーションを実行する前に、custdb データベースをデバイスのルートフォルダにコピーする必要があります。*%SQLANYSAMP17%*¥UltraLite¥CustDB¥custdb.udb というデータベースファイルをデバイスのルートにコピーします。

   デバイスまたはシミュレータ上で、¥Program Files の下のフォルダにある CustDB.exe を実行します。

結果

CustDB アプリケーションがロードされます。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

3.2.3.2 永続的なデータ

Ultra Light データベースは Windows Mobile ファイルシステムに格納されます。

デフォルトは、¥UltraLiteDB¥ul_store.udb です。この設定は、永続的保存ファイルのフルパス名を指定する file_name 接続パラメータを使用して上書きできます。

Ultra Light ランタイムは file_name パラメータへの代入を行いません。ファイル名を有効化するためにディレクトリの作成が必要な場合は、db_init を呼び出す前にディレクトリが作成されることをアプリケーション側で確認してください。

たとえば、フラッシュメモリストレージカードを使用する場合は、ストレージカードを検索し、そのストレージカードの名前の前にフォルダ名を追加します。次に例を示します。

```
file_name = "¥¥Storage Card¥¥My Documents¥¥flash.udb"
```

3.2.3.3 アプリケーションに対するクラス名の割り当て

MFC を使用している場合は、アプリケーションに固有のクラス名を割り当てます。ActiveSync で使用するアプリケーションを登録するには、ウィンドウクラス名を指定します。

前提条件

クラス名の割り当ては開発時に行うので、その方法については、アプリケーション開発ツールのマニュアルが主な情報源になります。

手順

1. デフォルトのクラスに基づいて、ダイアログボックスのカスタムウィンドウクラスを作成して登録します。

   アプリケーションの起動コードに、次のコードを追加します。ダイアログを作成する前に、このコードを実行する必要があります。

   ```
   WNDCLASS wc;
   if( ! GetClassInfo( NULL, L"Dialog", &wc ) ) {
    AfxMessageBox( L"Error getting class info" );
   }
   wc.lpszClassName = L"MY_APP_CLASS";
   if( ! AfxRegisterClass( &wc ) ) {
    AfxMessageBox( L"Error registering class" );
   }
   ```

   MY_APP_CLASS は、アプリケーションのユニークなクラス名です。
2. アプリケーションのメインダイアログにするダイアログを決めます。

   MFC アプリケーションウィザードを使ってプロジェクトを作成した場合、ダイアログ名は通常 MyAppDlg です。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

3. メインダイアログのリソース ID を調べて記録しておきます。

   リソース ID は、IDD_MYAPP_DIALOG のような、一般形式の定数です。
4. アプリケーションの実行中は常にメインダイアログが開かれていることを確認します。

   アプリケーションの InitInstance メソッドに、次のコードを追加します。

   ```
   m_pMainWnd = &dlg;
   ```

   このコードにより、メインダイアログ dlg を閉じたときにアプリケーションも確実に閉じるようになります。詳細については、CWinThread::m_pMainWnd に関する Microsoft のマニュアルを参照してください。

   アプリケーションの実行中にダイアログが閉じられてしまう場合には、他のダイアログのウィンドウクラスも変更してください。
5. 変更内容を保存します。
6. プロジェクトのリソースファイルを変更します。

   メモ帳などのテキストエディタで、リソースファイル (拡張子は .rc) を開きます。
7. メインダイアログのリソース ID を見つけます。

   メインダイアログの定義を変更して、次の例のように、新しいウィンドウクラスが使用されるようにします。変更は、CLASS 行の追加だけです。

   ```
   IDD_MYAPP_DIALOG DIALOG DISCARDABLE  0, 0, 139, 103
   STYLE WS_POPUP | WS_VISIBLE | WS_CAPTION
   EXSTYLE WS_EX_APPWINDOW | WS_EX_CAPTIONOKBTN
   CAPTION "MyApp"
   FONT 8, "System"
   CLASS "MY_APP_CLASS"
   BEGIN
          LTEXT   "TODO: Place dialog controls here.",IDC_STATIC,13,33,112,17
   END
   ```

   MY_APP_CLASS は、前の手順で使用したウィンドウクラスの名前です。
8. .rc ファイルを保存します。
9. 同期メッセージを取得するコードを追加します。

結果

アプリケーション固有のクラス名が作成されます。

関連情報

メインダイアログクラスでの ActiveSync 同期の追加 [672 ページ]

 © 2016 SAP SE or an SAP affiliate company.All rights reserved.

3.2.3.4 Windows Mobile の同期

Microsoft Windows Mobile 上の Ultra Light アプリケーションは、Microsoft ActiveSync、TCP/IP、および HTTP ストリームを介して同期できます。

Windows Mobile では、初期化のときに `user_name` パラメータと `stream_parms` パラメータを UL_TEXT() マクロで囲んでください。これは、コンパイル環境がユニコードワイド文字であるためです。

このセクションの内容:

Ultra Light C++ での Microsoft ActiveSync 同期 [671 ページ]
Microsoft ActiveSync は、Microsoft 社のソフトウェアで、Windows を実行しているデスクトップコンピュータと、そのコンピュータに接続された Windows Mobile ハンドヘルドデバイスとの間のデータの同期を処理します。Ultra Light は Microsoft ActiveSync バージョン 3.5 以降をサポートしています。

Ultra Light C++ での Microsoft ActiveSync 同期 (Windows API) [672 ページ]
直接 Windows API でプログラミングしている場合は、アプリケーションの WindowProc 関数を使用して、Mobile Link プロバイダからのメッセージを処理します。メッセージを受信したかどうかを判断するには、ULIsSynchronizeMessage 関数を使用します。

メインダイアログクラスでの ActiveSync 同期の追加 [672 ページ]
メインダイアログクラスで同期メッセージを取得します。

アプリケーションクラスでの ActiveSync 同期の追加 [674 ページ]
アプリケーションクラスで同期メッセージを取得します。

Windows Mobile からの TCP/IP、HTTP、HTTPS 同期 [675 ページ]
TCP/IP、HTTP、または HTTPS では、同期のタイミングをアプリケーションが制御します。ユーザが同期を要求できるように、アプリケーションにはメニュー項目かユーザインタフェースを用意してください。

3.2.3.4.1 Ultra Light C++ での Microsoft ActiveSync 同期

Microsoft ActiveSync は、Microsoft 社のソフトウェアで、Windows を実行しているデスクトップコンピュータと、そのコンピュータに接続された Windows Mobile ハンドヘルドデバイスとの間のデータの同期を処理します。Ultra Light は Microsoft ActiveSync バージョン 3.5 以降をサポートしています。

Microsoft ActiveSync を使用する場合、同期を開始できるのは、Microsoft ActiveSync 自体だけです。デバイスがクレードルにある場合や、[Microsoft ActiveSync] ウィンドウで同期が選択された場合に、Microsoft ActiveSync は自動的に同期を開始します。Mobile Link プロバイダは、アプリケーションを起動し (まだ動作していなかった場合)、アプリケーションにメッセージを送ります。

Microsoft ActiveSync プロバイダは wParam パラメータを使用します。wParam の値 1 は、アプリケーションが Microsoft ActiveSync 用 Mobile Link プロバイダによって起動されたことを示します。同期の完了後、アプリケーションは自動的に停止しなければなりません。Microsoft ActiveSync 用 Mobile Link プロバイダが呼び出したときに、すでにアプリケーションが実行されていた場合、wParam は 0 です。アプリケーションを実行したままにする場合は、wParam パラメータを無視できます。

同期の追加方法は、Windows API を直接使用しているか、Microsoft Foundation Classes を使用しているかによって異なります。ここでは、両方の開発モデルについて説明します。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

3.2.3.4.2 Ultra Light C++ での Microsoft ActiveSync 同期 (Windows API)

直接 Windows API でプログラミングしている場合は、アプリケーションの WindowProc 関数を使用して、Mobile Link プロバイダからのメッセージを処理します。メッセージを受信したかどうかを判断するには、ULIsSynchronizeMessage 関数を使用します。

次の例は、メッセージの処理方法を示しています。

```
LRESULT CALLBACK WindowProc( HWND hwnd,
         UINT uMsg,
         WPARAM wParam,
         LPARAM lParam )
{
  if( ULIsSynchronizeMessage( uMsg ) ) {
    DoSync();
    if( wParam == 1 ) DestroyWindow( hWnd );
    return 0;
  }
  switch( uMsg ) {
  // code to handle other windows messages
  default:
    return DefWindowProc( hwnd, uMsg, wParam, lParam );
  }
  return 0;
}
```

DoSync は実際に ULSynchronize を呼び出すメソッドです。

3.2.3.4.3 メインダイアログクラスでの ActiveSync 同期の追加

メインダイアログクラスで同期メッセージを取得します。

前提条件

アプリケーションを開発するには、Microsoft Foundation Classes を使用する必要があります。

コンテキスト

メッセージの通知用に、アプリケーションのカスタムウィンドウクラス名を作成し、登録しておいてください。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

手順

1. 登録したメッセージを追加し、メッセージハンドラを宣言します。

   メインダイアログのソースファイルでメッセージマップを検索します (名前は `CMyAppDlg.cpp` と同じ形式です)。次の例のように、登録したメッセージを *static* を使用して追加し、メッセージハンドラを ON_REGISTERED_MESSAGE を使用して宣言します。

   ```
   static UINT WM_ULTRALITE_SYNC_MESSAGE =
    ::RegisterWindowMessage( UL_AS_SYNCHRONIZE );
   BEGIN_MESSAGE_MAP(CMyAppDlg, CDialog)
     //{{AFX_MSG_MAP(CMyAppDlg)
     //}}AFX_MSG_MAP
    ON_REGISTERED_MESSAGE( WM_ULTRALITE_SYNC_MESSAGE,
     OnDoUltraLiteSync )
   END_MESSAGE_MAP()
   ```

2. メッセージハンドラを実装します。

   次のシグネチャで、メインダイアログクラスにメソッドを追加します。ActiveSync 用 Mobile Link プロバイダがアプリケーションの同期を要求するときは、常にこのメソッドが自動的に実行されます。このメソッドで、ULSynchronize メソッドを呼び出してください。

   ```
   LRESULT CMyAppDlg::OnDoUltraLiteSync(
      WPARAM wParam,
      LPARAM lParam
   );
   ```

   この関数の戻り値は 0 にしてください。

結果

メインダイアログクラスによって同期が実行されます。

関連情報

アプリケーションに対するクラス名の割り当て [669 ページ]

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

3.2.3.4.4 アプリケーションクラスでの ActiveSync 同期の追加

アプリケーションクラスで同期メッセージを取得します。

前提条件

アプリケーションを開発するには、Microsoft Foundation Classes を使用する必要があります。

メッセージの通知用に、アプリケーションのカスタムウィンドウクラス名を作成し、登録しておいてください。

手順

1. アプリケーションクラスのクラスウィザードを開きます。
2. メッセージリストで、*PreTranslateMessage* を強調表示して、関数の追加をクリックします。
3. コードの編集をクリックします。

   PreTranslateMessage メソッドが表示されます。
4. 次のように PreTranslateMessage メソッドを変更します。

```
BOOL CMyApp::PreTranslateMessage(MSG* pMsg)
{
 if( ULIsSynchronizeMessage(pMsg->message) ) {
  DoSync();
  // close application if launched by provider
  if( pMsg->wParam == 1 ) {
   ASSERT( AfxGetMainWnd() != NULL );
   AfxGetMainWnd()->SendMessage( WM_CLOSE );
  }
  return TRUE; // message has been processed
 }
 return CWinApp::PreTranslateMessage(pMsg);
}
```

   DoSync メソッドが ULSynchronize メソッドを呼び出す。

結果

PreTranslateMessage メソッドが同期を実行します。

関連情報

アプリケーションに対するクラス名の割り当て [669 ページ]

 © 2016 SAP SE or an SAP affiliate company.All rights reserved.

3.2.3.4.5 Windows Mobile からの TCP/IP、HTTP、HTTPS 同期

TCP/IP、HTTP、または HTTPS では、同期のタイミングをアプリケーションが制御します。ユーザが同期を要求できるように、アプリケーションにはメニュー項目かユーザインタフェースを用意してください。

3.3 チュートリアル:C++ API を使用した Windows アプリケーションの構築

このチュートリアルでは、Ultra Light C++ アプリケーションを開発するプロセスを説明します。アプリケーションは Windows デスクトップオペレーティングシステム用に開発され、コマンドプロンプトで実行されます。

前提条件

このチュートリアルは、次のことを前提にしています。

- C++ プログラミング言語に精通しています。
- C++ コンパイラがマシンにインストールされています。
- データベース作成ウィザードを使用して Ultra Light データベースを作成する方法を知っています。

コンテキスト

このチュートリアルの目的は、Ultra Light C++ アプリケーションの開発プロセスについて、知識を得ることです。

このチュートリアルは、Microsoft Visual C++ を使用した開発を基にしていますが、任意の C++ 開発環境を使用できます。

このチュートリアルは、コードをコピーして貼り付ける場合、約 30 分で終了します。この章の最後には、このチュートリアルで説明しているプログラムの完全なソースコードを掲載しています。

1. レッスン 1: データベースの作成とデータベースへの接続 [676 ページ]
   このレッスンでは、Ultra Light データベースを作成します。次に、作成したデータベースにアクセスする C++ アプリケーションを記述、コンパイル、実行します。
2. レッスン 2:データベースへのデータの挿入 [679 ページ]
   このレッスンでは、データベースにデータを追加します。
3. レッスン 3:テーブルのローの選択とリスト [681 ページ]
   このレッスンでは、テーブルからローを取得し、それらをコマンドラインで出力します。
4. レッスン 4:アプリケーションへの同期の追加 [682 ページ]
   このレッスンでは、アプリケーションに同期コードを追加し、Mobile Link サーバを起動し、アプリケーションを実行して、統合されたデータベースと同期させます。
5. チュートリアルのコードリストの確認 [684 ページ]
   これまでの項で説明したチュートリアルプログラムの完全なコードを次に示します。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

3.3.1 レッスン 1: データベースの作成とデータベースへの接続

このレッスンでは、Ultra Light データベースを作成します。次に、作成したデータベースにアクセスする C++ アプリケーションを記述、コンパイル、実行します。

前提条件

このレッスンでは、必要なソフトウェアをすでにインストールしていることを前提としています。

手順

1. VCINSTALLDIR 環境変数に Microsoft Visual C++ インストール環境のルートフォルダを設定します (変数がまだ存在しない場合)。
2. `%VCINSTALLDIR%¥VC¥atlmfc¥src¥atl` を INCLUDE 環境変数に追加します。
3. このチュートリアルで作成されるファイルを保存するフォルダを作成します。

   以降、このチュートリアルではこのディレクトリが `C:¥tutorial¥cpp¥` であることを前提に説明します。別の名前のディレクトリを作成した場合は、`C:¥tutorial¥cpp¥` の代わりにそのディレクトリを使用してください。
4. SQL Central で Ultra Light を使用して、デフォルトの特性を持つデータベース `ULCustomer.udb` を新しいディレクトリに作成します。
5. ULCustomer という名前のテーブルをデータベースに追加します。次の ULCustomer テーブル仕様を使用します。

カラム名	データ型 (サイズ)	カラムの NULL 値の許可	デフォルト値	プライマリキー
cust_id	整数	いいえ	autoincrement	昇順
cust_name	varchar(30)	いいえ	なし	

6. SQL Central でデータベースから切断します。そうしないと、実行ファイルが接続できません。
7. Microsoft Visual C++ で、ファイル > 新規作成 をクリックします。
8. ファイルタブで、C++ ソースファイルをクリックします。
9. チュートリアルのディレクトリに、そのファイルを `customer.cpp` として保存します。
10. Ultra Light ライブラリをインクルードします。

    以下のコードを `customer.cpp` にコピーします。

```
#include <tchar.h>
#include <stdio.h>
#include "ulcpp.h"
#define MAX_NAME_LEN 100
#define MAX_ERROR_LEN 256
```

11. データベースに接続するための接続パラメータを定義します。

    次のコードでは、接続パラメータはハードコードされています。実際のアプリケーションでは、ロケーションは実行時に指定されることもあります。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

以下のコードを `customer.cpp` にコピーします。

```
static ul_char const * ConnectionParms =
    "UID=DBA;PWD=sql;DBF=C:¥¥tutorial¥¥cpp¥¥ULCustomer.udb";
```

> **注記**
>
> ファイル名ロケーションの文字列にバックスラッシュ文字が含まれる場合は、バックスラッシュ文字をもう 1 つ追加してエスケープする必要があります。

12. アプリケーションでデータベースエラーを処理するメソッドを定義します。

    Ultra Light は、アプリケーションにエラーを通知するためのコールバックメカニズムを備えています。開発環境において、このメソッドは予期しないエラーを処理するメカニズムとして便利です。運用アプリケーションには、あらゆる一般的なエラー状況を処理するコードが含まれているのが普通です。アプリケーションでは、Ultra Light メソッドを呼び出すごとにエラー確認するか、エラーコールバック関数を使用するかを選択します。

    コールバック関数のサンプルコードを次に示します。

```
ul_error_action UL_CALLBACK_FN MyErrorCallBack(
    const ULError *  error,
    ul_void *        user_data )
{
    ul_error_action rc;
    an_sql_code code = error->GetSQLCode();

    (void) user_data;
    switch( code ){
        // The following error is used for flow control - don't report it here
        case SQLE_NOTFOUND:
            rc = UL_ERROR_ACTION_CONTINUE;
        break;
        default:
            if (code >= 0) { // warning or success
                rc = UL_ERROR_ACTION_DEFAULT;
            } else { // negative is real error
                ul_char etext[ MAX_ERROR_LEN ];
                error->GetString( etext, MAX_ERROR_LEN );
                _tprintf( "Error %ld: %s¥n", code, etext );
                rc = UL_ERROR_ACTION_CANCEL;
            }
        break;
     }
    return rc;
 }
```

    Ultra Light では、ほとんどの場合はエラー SQLE_NOTFOUND を使用してアプリケーションのフローを制御しています。このエラーが通知されると、結果セットのループの終了がマークされます。上記の汎用エラーハンドラは、このエラー条件に対してはメッセージを出力しません。

13. データベースの接続を開くメソッドを定義します。

```
static ULConnection * open_conn( void ) {
    ULConnection * conn = ULDatabaseManager::OpenConnection( ConnectionParms );
    if( conn == UL_NULL ) {
        _tprintf("Unable to open existing database.¥n");
    }
    return conn;
}
```

    データベースファイルがなかった場合は、エラーメッセージが表示されます。そうでない場合は、接続が確立されます。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

14. 次のタスクを実行する main メソッドを実装します。
    - エラー処理メソッドを登録します。
    - データベースへの接続を開きます。
    - データベースとの接続を閉じ、データベースマネージャを終了します。

```
int main() {
    ULConnection *    conn;
    ULDatabaseManager::Init();
    ULDatabaseManager::SetErrorCallback( MyErrorCallBack, NULL );
    conn = open_conn();
    if ( conn == UL_NULL ) {
        ULDatabaseManager::Fini();
        return 1;
    }
    // Main processing code goes here ...
    conn->Close();
    ULDatabaseManager::Fini();
    return 0;
}
```

15. ソースファイルのコンパイルとリンクを行います。

    ソースファイルのコンパイル方法は、コンパイラによって異なります。以下の手順は、makefile で Microsoft Visual C++ コマンドラインコンパイラを使用する場合です。

    a. コマンドプロンプトを開き、チュートリアルフォルダに変更します。
    b. `makefile` という名前の makefile を作成します。
    c. makefile で、フォルダをインクルードパスに追加します。

```
IncludeFolders=/I"$(SQLANY17)¥SDK¥Include"
```

    d. makefile で、フォルダをライブラリパスに追加します。

```
LibraryFolders=/LIBPATH:"$(SQLANY17)¥UltraLite¥Windows¥x86¥Lib¥vs9"
```

    e. makefile で、ライブラリをリンカオプションに追加します。

```
Libraries=ulimp.lib
```

    Ultra Light ランタイムライブラリの名前は `ulimp.lib` です。

    f. makefile で、コンパイラオプションを設定します。次のように、1 行でこれらのオプションを入力してください。

```
CompileOptions=/c /nologo /W3 /Od /Zi /DWIN32 /DUL_USE_DLL
```

    g. makefile で、次のようにアプリケーションのリンク命令を追加します。

```
customer.exe: customer.obj
   link /NOLOGO /DEBUG customer.obj $(LibraryFolders) $(Libraries)
```

    h. makefile で、次のようにアプリケーションのコンパイル命令を追加します。

```
customer.obj: customer.cpp
   cl $(CompileOptions) $(IncludeFolders) customer.cpp
```

    i. `vsvars32.bat` を実行します。

```
%VCINSTALLDIR%¥Tools¥vsvars32.bat
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

j. makefile を実行します。

```
nmake
```

`customer.exe` という実行ファイルが作成されます。

16. アプリケーションを実行します。

コマンドプロンプトで **`customer`** と入力します。

結果

アプリケーションがデータベースに接続したら、切断します。エラーメッセージが表示されなければ、アプリケーションは正常に実行されています。

次のステップ

次のレッスンに進みます。

タスクの概要: チュートリアル:C++ API を使用した Windows アプリケーションの構築 [675 ページ]

次のタスク: レッスン 2:データベースへのデータの挿入 [679 ページ]

関連情報

エラー処理 [625 ページ]

3.3.2 レッスン 2:データベースへのデータの挿入

このレッスンでは、データベースにデータを追加します。

前提条件

このチュートリアルのこれまでのレッスンを完了している必要があります。

このチュートリアルの冒頭に一覧されているロールと権限を持っている必要があります。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

手順

1. 以下のメソッドを、customer.cpp の main メソッドの直前に追加します。

```
static bool do_insert( ULConnection * conn )
{
    ULTable * table = conn->OpenTable( "ULCustomer" );
    if( table == UL_NULL ) {
        _tprintf( "Table not found: ULCustomer¥n" );
        return false;
    }
    if( table->GetRowCount() == 0 ) {
        _tprintf( "Inserting one row.¥n" );
        table->InsertBegin();
        table->SetString( "cust_name", "New Customer" );
        table->Insert();
        conn->Commit();
    } else {
        _tprintf( "The table has %lu rows¥n", table->GetRowCount() );
    }
    table->Close();
    return true;
}
```

   このメソッドでは次のタスクを実行します。

   - connection->OpenTable() メソッドを使用してテーブルを開きます。テーブルの操作を実行するには、Table オブジェクトを開いてください。
   - テーブルが空の場合は、ローを 1 つ追加します。このコードでは、ローを挿入するために、InsertBegin メソッドを使用して挿入モードに変更し、必要な各カラムに値を設定し、挿入を実行してデータベースにローを追加します。
   - テーブルが空でない場合は、テーブルのロー数をレポートします。
   - Table オブジェクトを閉じ、関連付けられているリソースを解放します。
   - 操作が正常に完了したことを示すブール値が返されます。

2. 作成した do_insert メソッドを呼び出します。

   次の行を、main() メソッドの conn->Close 呼び出しの直前に追加します。

```
do_insert(conn);
```

3. nmake を実行してアプリケーションをコンパイルします。
4. コマンドプロンプトで **customer** と入力してアプリケーションを実行します。

結果

アプリケーションが実行され、ULCustomer テーブルにデータを挿入できます。

次のステップ

次のレッスンに進みます。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

タスクの概要: チュートリアル:C++ API を使用した Windows アプリケーションの構築 [675 ページ]

前のタスク: レッスン 1: データベースの作成とデータベースへの接続 [676 ページ]

次のタスク: レッスン 3:テーブルのローの選択とリスト [681 ページ]

3.3.3 レッスン 3:テーブルのローの選択とリスト

このレッスンでは、テーブルからローを取得し、それらをコマンドラインで出力します。

前提条件

このチュートリアルのこれまでのレッスンを完了している必要があります。

このチュートリアルの冒頭に一覧されているロールと権限を持っている必要があります。

手順

1. 以下のメソッドを、`customer.cpp` の do_insert メソッドの直後に追加します。

```
static bool do_select( ULConnection * conn )
{
    ULTable * table = conn->OpenTable( "ULCustomer" );
    if( table == UL_NULL ) {
        return false;
    }
    ULTableSchema * schema = table->GetTableSchema();
    if( schema == UL_NULL ) {
        table->Close();
        return false;
    }
    ul_column_num id_cid =
        schema->GetColumnID( "cust_id" );
    ul_column_num cname_cid =
        schema->GetColumnID( "cust_name" );
    schema->Close();
    _tprintf( "¥n¥nTable 'ULCustomer' row contents:¥n" );
    while( table->Next() ) {
        ul_char cname[ MAX_NAME_LEN ];
        table->GetString( cname_cid, cname, MAX_NAME_LEN );
        _tprintf( "id=%d, name=%s ¥n", (int)table->GetInt(id_cid), cname );
    }
    table->Close();
    return true;
}
```

このメソッドでは、次のタスクを実行します。

- Table オブジェクトを開きます。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

- カラム識別子を取得します。
- 現在の位置を、テーブル内にある最初のローの前に設定します。
  テーブルに対するすべての操作は、現在の位置で実行されます。現在の位置は、最初のローの前、テーブルのいずれかのローの上、または最後のローの後ろです。この例のように、デフォルトでは、ローはプライマリキー値 (cust_id) に基づいて順序付けられています。別の順序でローを並べ替えるには、Ultra Light データベースにインデックスを追加し、そのインデックスを使用してテーブルを開きます。
- 各ローに対して、cust_id と cust_name の値が書き出されます。ループは、最後のローの後に Next メソッドが false を返すまで繰り返されます。
- Table オブジェクトを閉じます。

2. 次の行を、main メソッドの insert メソッドの呼び出しの直後に追加します。

```
do_select(conn);
```

3. `nmake` を実行してアプリケーションをコンパイルします。
4. コマンドプロンプトで `customer` と入力してアプリケーションを実行します。

結果

ULCustomer テーブル内のすべての顧客 ID と顧客名のリストが出力されます。

次のステップ

次のレッスンに進みます。

タスクの概要: チュートリアル:C++ API を使用した Windows アプリケーションの構築 [675 ページ]

前のタスク: レッスン 2:データベースへのデータの挿入 [679 ページ]

次のタスク: レッスン 4:アプリケーションへの同期の追加 [682 ページ]

3.3.4 レッスン 4:アプリケーションへの同期の追加

このレッスンでは、アプリケーションに同期コードを追加し、Mobile Link サーバを起動し、アプリケーションを実行して、統合されたデータベースと同期させます。

前提条件

このチュートリアルのこれまでのレッスンを完了している必要があります。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

このチュートリアルの冒頭に一覧されているロールと権限を持っている必要があります。

コンテキスト

前のレッスンで作成した Ultra Light データベースは、Ultra Light 17 Sample データベースと同期します。Ultra Light 17 の Sample データベースの ULCustomer テーブルのカラムには、作成したローカル Ultra Light データベースの customer テーブルのカラムが含まれます。

手順

1. 以下のメソッドを `customer.cpp` に追加します。

```
static bool do_sync( ULConnection * conn )
{
    ul_sync_info info;
    ul_stream_error * se = &info.stream_error;

    ULDatabaseManager::EnableTcpipSynchronization();
    conn->InitSyncInfo( &info );
    info.stream = "TCPIP";
    info.version = "custdb 17.0";
    info.user_name = "50";
    info.download_only = true;
    if( !conn->Synchronize( &info ) ) {
        _tprintf( "Synchronization error ¥n" );
        _tprintf( "  stream_error_code is '%lu'¥n", se->stream_error_code );
        _tprintf( "  system_error_code is '%ld'¥n", se->system_error_code );
        _tprintf( "  error_string is '" );
        _tprintf( "%s", se->error_string );
        _tprintf( "'¥n" );
        return false;
    }
    return true;
}
```

   このメソッドでは、次のタスクを実行します。

   - EnableTcpipSynchronization を呼び出して、TCP/IP 通信を有効にします。同期は、HTTP、HTTPS、および TLS を使用しても実行できます。
   - スクリプトバージョンを設定します。Mobile Link 同期は、統合データベースに保存されているスクリプトによって制御されます。スクリプトバージョンは、使用するスクリプトセットを識別します。
   - Mobile Link ユーザ名を設定します。この値は、Mobile Link サーバでの認証に使用されます。アプリケーションによっては、Mobile Link のユーザ ID と Ultra Light データベースのユーザ ID を同じに設定しますが、これらの ID はあくまでも別のものです。
   - download_only パラメータを true に設定します。デフォルトでは、Mobile Link 同期は双方向です。このアプリケーションでは、テーブルのローがサンプルデータベースにアップロードされないように、ダウンロード専用同期を使用します。

2. do_select メソッド呼び出し後に、次のコード行を main メソッドに配置します。

```
do_sync(conn);
```


© 2016 SAP SE or an SAP affiliate company. All rights reserved.

3. nmake を実行してアプリケーションをコンパイルします。
4. Mobile Link サーバを起動します。

   コマンドプロンプトで次のコマンドを実行します。

   ```
   mlsrv17 -c "dsn=SQL Anywhere 17 CustDB;uid=ml_server;pwd=sql" -v -vr -vs -zu+ -o
   custdbASA.log
   ```

   -zu+ オプションを指定すると、ユーザの自動追加が行われます。-v+ オプションを指定すると、すべてのメッセージについて冗長ロギングがオンになります。
5. コマンドプロンプトで customer と入力してアプリケーションを実行します。

結果

Mobile Link サーバのメッセージウィンドウでは、同期の進行状況を示すステータスメッセージが表示されます。同期が正しく行われると、最後に Synchronization complete というメッセージが表示されます。

タスクの概要: チュートリアル:C++ API を使用した Windows アプリケーションの構築 [675 ページ]

前のタスク: レッスン 3:テーブルのローの選択とリスト [681 ページ]

次のトピック: チュートリアルのコードリストの確認 [684 ページ]

3.3.5 チュートリアルのコードリストの確認

これまでの項で説明したチュートリアルプログラムの完全なコードを次に示します。

```
#include <tchar.h>
#include <stdio.h>
#include "ulcpp.h"
#define MAX_NAME_LEN 100
#define MAX_ERROR_LEN 256
static ul_char const * ConnectionParms =
    "UID=DBA;PWD=sql;DBF=c:¥¥tutorial¥¥cpp¥¥ULCustomer.udb";
ul_error_action UL_CALLBACK_FN MyErrorCallBack(
    const ULError *  error,
    ul_void *        user_data )
{
    ul_error_action rc;
    an_sql_code code = error->GetSQLCode();

    (void) user_data;
    switch( code ){
        // The following error is used for flow control - don't report it here
        case SQLE_NOTFOUND:
            rc = UL_ERROR_ACTION_CONTINUE;
        break;
        default:
            if (code >= 0) { // warning or success
                rc = UL_ERROR_ACTION_DEFAULT;
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

```
            } else { // negative is real error
                ul_char etext[ MAX_ERROR_LEN ];
                error->GetString( etext, MAX_ERROR_LEN );
                _tprintf( "Error %ld: %s¥n", code, etext );
                rc = UL_ERROR_ACTION_CANCEL;
            }
        break;
     }
    return rc;
}
static ULConnection * open_conn( void ) {
    ULConnection * conn = ULDatabaseManager::OpenConnection( ConnectionParms );
    if( conn == UL_NULL ) {
        _tprintf( "Unable to open existing database.¥n" );
    }
    return conn;
}
static bool do_insert( ULConnection * conn ) {
    ULTable * table = conn->OpenTable( "ULCustomer" );
    if( table == UL_NULL ) {
        _tprintf( "Table not found: ULCustomer¥n" );
        return false;
    }
    if( table->GetRowCount() == 0 ) {
        _tprintf( "Inserting one row.¥n" );
        table->InsertBegin();
        table->SetString( "cust_name", "New Customer" );
        table->Insert();
        conn->Commit();
    } else {
        _tprintf( "The table has %lu rows¥n",
        table->GetRowCount() );
    }
    table->Close();
    return true;
}
static bool do_select( ULConnection * conn )
{
    ULTable * table = conn->OpenTable( "ULCustomer" );
    if( table == UL_NULL ) {
        return false;
    }
    ULTableSchema * schema = table->GetTableSchema();
    if( schema == UL_NULL ) {
        table->Close();
        return false;
    }
    ul_column_num id_cid =
        schema->GetColumnID( "cust_id" );
    ul_column_num cname_cid =
        schema->GetColumnID( "cust_name" );
    schema->Close();
    _tprintf( "¥n¥nTable 'ULCustomer' row contents:¥n" );
    while( table->Next() ) {
        ul_char cname[ MAX_NAME_LEN ];
        table->GetString( cname_cid, cname, MAX_NAME_LEN );
        _tprintf( "id=%d, name=%s ¥n", (int)table->GetInt(id_cid), cname );
    }
    table->Close();
    return true;
}
static bool do_sync( ULConnection * conn )
{
    ul_sync_info info;
    ul_stream_error * se = &info.stream_error;

    ULDatabaseManager::EnableTcpipSynchronization();
    conn->InitSyncInfo( &info );
```


© 2016 SAP SE or an SAP affiliate company. All rights reserved.

```
    info.stream = "TCPIP";
    info.version = "custdb 12.0";
    info.user_name = "50";
    info.download_only = true;
    if( !conn->Synchronize( &info ) ) {
        _tprintf( "Synchronization error ¥n" );
        _tprintf( "  stream_error_code is '%lu'¥n", se->stream_error_code );
        _tprintf( "  system_error_code is '%ld'¥n", se->system_error_code );
        _tprintf( "  error_string is '" );
        _tprintf( "%s", se->error_string );
        _tprintf( "'¥n" );
        return false;
    }
    return true;
}
int main()
{
    ULConnection *    conn;
    ULDatabaseManager::Init();
    ULDatabaseManager::SetErrorCallback( MyErrorCallBack, NULL );
    conn = open_conn();
    if( conn == UL_NULL ){
        ULDatabaseManager::Fini();
        return 1;
    }

    // Main processing code goes here ...
    do_insert( conn );
    do_select( conn );
    do_sync( conn );
    conn->Close();
    ULDatabaseManager::Fini();
    return 0;
}
```

親トピック チュートリアル:C++ API を使用した Windows アプリケーションの構築 [675 ページ]

前のタスク: レッスン 4:アプリケーションへの同期の追加 [682 ページ]

3.4 API リファレンス

Ultra Light C++ API を使用してモバイルアプリケーションを開発します。

このセクションの内容:

Ultra Light C++ 共通 API リファレンス [687 ページ]
関数とマクロを Embedded SQL または C++ インタフェースで使用します。

Ultra Light C++ API リファレンス [689 ページ]
Ultra Light C++ は豊富な API オブジェクトを提供します。

Ultra Light Embedded SQL API リファレンス [689 ページ]
Embedded SQL アプリケーションは複数の Ultra Light 関数をサポートします。

 © 2016 SAP SE or an SAP affiliate company.All rights reserved.

3.4.1 Ultra Light C++ 共通 API リファレンス

関数とマクロを Embedded SQL または C++ インタフェースで使用します。

ヘッダファイル

- `ulglobal.h`

> **注記**
>
> API リファレンスマニュアルをお探しですか。マニュアルをローカルにインストールした場合は、Windows のスタートメニューを使用してアクセスするか (Microsoft Windows)、`C:¥Program Files¥SQL Anywhere 17¥Documentation` にナビゲートします。
>
> また、DocCommentXchange の Web で、SAP SQL Anywhere API リファレンスマニュアルにアクセスすることもできます。http://dcx.sap.com

このセクションの内容:

Ultra Light C++ アプリケーションのマクロとコンパイラディレクティブ [687 ページ]
Embedded SQL および C++ アプリケーションは豊富なディレクティブをサポートしています。これらのディレクティブは、特に指定のない限り両方の API に適用されます。

3.4.1.1 Ultra Light C++ アプリケーションのマクロとコンパイラディレクティブ

Embedded SQL および C++ アプリケーションは豊富なディレクティブをサポートしています。これらのディレクティブは、特に指定のない限り両方の API に適用されます。

コンパイラディレクティブは、次の場所で指定できます。

- コンパイラのコマンドライン。一般にディレクティブは /D オプションを使用して設定します。たとえば、ユーザ認証を使用する Ultra Light アプリケーションをコンパイルする場合、Microsoft Visual C++ の makefile は、次のようになります。

  ```
  CompileOptions=/c /DPRWIN32 /Od /Zi /DWIN32 /DUL_USE_DLL
  IncludeFolders= ¥
  /I"$(VCDIR)¥include" ¥
  /I"$(SQLANY17)¥SDK¥Include"
  sample.obj: sample.cpp
   cl $(CompileOptions) $(IncludeFolders) sample.cpp
  ```

  `VCDIR` および `SQLANY17` は対応するインストールディレクトリを指す環境変数です。
- ユーザインタフェースのコンパイラ設定ウィンドウ。
- ソースコード。ディレクティブは `#define` 文を使用して指定します。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

このセクションの内容:

UL_USE_DLL マクロ [688 ページ]
静的ランタイムライブラリではなく、ランタイムライブラリ DLL を使用するようにアプリケーションを設定します。

UNDER_CE マクロ [688 ページ]
デフォルトで、このマクロはすべての新しい Microsoft Visual C++ スマートデバイスプロジェクトで定義されています。

3.4.1.1.1 UL_USE_DLL マクロ

静的ランタイムライブラリではなく、ランタイムライブラリ DLL を使用するようにアプリケーションを設定します。

備考

Windows Mobile アプリケーションと Windows アプリケーションに適用されます。

3.4.1.1.2 UNDER_CE マクロ

デフォルトで、このマクロはすべての新しい Microsoft Visual C++ スマートデバイスプロジェクトで定義されています。

備考

Windows Mobile アプリケーションに適用されます。

例

```
/D UNDER_CE
```

関連情報

Windows Mobile 向け Ultra Light アプリケーション開発 [667 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

3.4.2 Ultra Light C++ API リファレンス

Ultra Light C++ は豊富な API オブジェクトを提供します。

次に、よく使用される API オブジェクトの一部を示します。

ULDatabaseManager

データベースと接続を管理する方法を提供します。

ULConnection

Ultra Light データベースへの接続を表します。ULConnection オブジェクトは 1 つまたは複数作成できます。

ULTable

データベースのテーブルへの直接アクセスを提供します。

ULPreparedStatement、ULResultSet、ULResultSetSchema

動的 SQL 文の作成、クエリの記述、INSERT、UPDATE、DELETE 文の実行、プログラムによるデータベースの結果セットの制御を行います。

ヘッダファイル

- `ulcpp.h`

> **注記**
>
> API リファレンスマニュアルをお探しですか。マニュアルをローカルにインストールした場合は、Windows のスタートメニューを使用してアクセスするか (Microsoft Windows)、`C:¥Program Files¥SQL Anywhere 17¥Documentation` にナビゲートします。
>
> また、DocCommentXchange の Web で、SAP SQL Anywhere API リファレンスマニュアルにアクセスすることもできます。http://dcx.sap.com

3.4.3 Ultra Light Embedded SQL API リファレンス

Embedded SQL アプリケーションは複数の Ultra Light 関数をサポートします。

この章で説明する関数のプロトタイプは、EXEC SQL INCLUDE SQLCA コマンドを使用してインクルードします。

ヘッダファイル

- `mlfiletransfer.h`

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

- `ulprotos.h`

> **注記**
>
> **API リファレンスマニュアルをお探しですか。マニュアルをローカルにインストールした場合は、Windows のスタートメニューを使用してアクセスするか (Microsoft Windows)、`C:¥Program Files¥SQL Anywhere 17¥Documentation` にナビゲートします。**
>
> **また、DocCommentXchange の Web で、SAP SQL Anywhere API リファレンスマニュアルにアクセスすることもできます。** http://dcx.sap.com

このセクションの内容:

db_fini メソッド [690 ページ]
Ultra Light ランタイムライブラリで使用したリソースを解放します。

db_init メソッド [691 ページ]
Ultra Light ランタイムライブラリを初期化します。

関連情報

Embedded SQL を使用した Ultra Light C++ アプリケーション開発 [635 ページ]

3.4.3.1 db_fini メソッド

Ultra Light ランタイムライブラリで使用したリソースを解放します。

> **構文**

```
unsigned short db_fini( SQLCA * sqlca );
```

戻り値

- 処理中にエラーが発生した場合は 0。エラー情報は SQLCA に設定されます。
- エラーが発生しなかった場合は 0 以外。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

備考

db_fini が呼び出された後に、他の Ultra Light ライブラリ呼び出しをしたり、Embedded SQL コマンドを実行したりしないでください。

使用する SQLCA ごとに 1 回ずつ db_fini を呼び出します。

関連情報

db_init メソッド [691 ページ]

3.4.3.2 db_init メソッド

Ultra Light ランタイムライブラリを初期化します。

構文

```
unsigned short db_init( SQLCA * sqlca );
```

戻り値

- 処理中にエラーが発生した場合は 0 (たとえば永続ストアの初期化時)。エラー情報は SQLCA に設定されます。
- エラーが発生しなかった場合は 0 以外。Embedded SQL コマンドと関数の使用を開始できます。

備考

この関数は、他の Ultra Light ライブラリを呼び出す前、および Embedded SQL コマンドを実行する前に呼び出す必要があります。

通常は、この関数を 1 回だけ呼び出して、ヘッダファイル `sqlca.h` に定義されているグローバル変数 sqlca のアドレスを渡してください。アプリケーションに複数の実行パスがある場合、複数の db_init を呼び出すことができます。ただし、それぞれに別個の sqlca ポインタが必要です。この別個の SQLCA ポインタには、ユーザが定義したものを使用するか、db_fini で解放されたグローバル SQLCA を使用します。

マルチスレッドアプリケーションでは、各スレッドは、別個の SQLCA を獲得するために db_init を呼び出します。同じ SQLCA を使用する接続とトランザクションは、1 つのスレッドで続けて実行してください。

SQLCA を初期化すると、その前の ULEnable 関数呼び出しによる設定がすべてリセットされます。SQLCA を再初期化する場合は、アプリケーションで必要な ULEnable 関数をすべて発行する必要があります。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

関連情報

db_fini メソッド [690 ページ]

 © 2016 SAP SE or an SAP affiliate company.All rights reserved.

4 Ultra Light - Java プログラミング

このマニュアルでは、Ultra Light J のプログラミングインタフェースについて説明します。Ultra Light J を使用すると、Android デバイス用のデータベースアプリケーションを開発し、配備することができます。

このセクションの内容:

システムの要件とサポート対象プラットフォーム [693 ページ]
Ultra Light J では、Android プラットフォームがサポートされます。

Ultra Light J アプリケーションの開発 [693 ページ]
Ultra Light J API は、Java アプリケーションにデータベース機能と同期を提供します。

チュートリアル:Android アプリケーションの構築 [711 ページ]
このチュートリアルでは、Ultra Light J API および Eclipse 環境を使用して、アプリケーションを開発する方法について説明します。このチュートリアルでは、Windows Simulator 上でアプリケーションを実行します。

Ultra LightJ API リファレンス [719 ページ]
Ultra LightJ には豊富な API オブジェクトがあります。

4.1 システムの要件とサポート対象プラットフォーム

Ultra Light J では、Android プラットフォームがサポートされます。

Ultra Light J アプリケーションを開発するには、Android Studio などの Java IDE が必要です。

4.2 Ultra Light J アプリケーションの開発

Ultra Light J API は、Java アプリケーションにデータベース機能と同期を提供します。

API には、Ultra Light データベースに接続し、スキーマ操作を実行し、SQL 文を使用してデータを管理するために必要なすべてのメソッドが含まれます。データの暗号化や同期などの高度な操作もサポートされています。

> **注記**
>
> Ultra Light J API は他のプラットフォーム用の Ultra Light と共通の C++ コードを共有しており、その動作は他のプラットフォームの動作と同様です。Android で提供されていない API 用に SQL 文を使わずにテーブルとローにアクセスできるいくつかの機能が存在します。

このセクションの内容:

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

Ultra Light J アプリケーション開発のクイックスタートガイド [694 ページ]
Ultra Light J アプリケーションを作成するときは、一般にアプリケーションコードでいくつかのデータ管理タスクを実行します。

Android セットアップの考慮事項 [695 ページ]
Android デバイス用のアプリケーションを開発する前に、Ultra Light J API の次の考慮事項に留意してください。

Ultra Light データベースの作成と接続アプローチ [696 ページ]
Java アプリケーションでデータを操作するには、先にデータベースに接続する必要があります。Ultra Light データベースは、Ultra Light J API を使用して作成し、指定されたパスワードで接続することができます。

スキーマ操作とデータ管理のクイックスタートガイド [698 ページ]
SQL 文とクエリを使用してデータベースのテーブル、インデックス、外部キー、パブリケーション、ローの作成、更新、取得を行います。

Ultra Light J でのスキーマ情報 [707 ページ]
データベーススキーマの記述は、プログラムを使用して取得できます。これらの記述は、スキーマ情報と呼ばれ、システムテーブルおよび Ultra Light J API スキーマインタフェースを使用してアクセスできます。

Ultra Light J でのエラー処理 [708 ページ]
ULjException クラスおよび SQLCode クラスを使用して、エラーを処理できます。ほとんどの Ultra Light メソッドは、ULException エラーをスローします。

Ultra Light J を使用した Mobile Link データの同期 [709 ページ]
データの同期は、HTTP ネットワークプロトコルまたは HTTPS ネットワークプロトコルを使用して実行できます。HTTPS 同期を使用すると、Mobile Link サーバに安全な暗号化が提供されます。

Android 用 Ultra Light J アプリケーションの配備 [710 ページ]
適切な作成パラメータ、接続パラメータ、同期パラメータ、プロトコルオプション、メソッド呼び出し、配備ファイルを指定して、Ultra Light J アプリケーションが Android スマートフォンで正常に実行されるようにします。

サンプルコード [711 ページ]
Ultra Light J API を使用するコード例は、*%SQLANYSAMP17%*¥UltraLiteJ¥ ディレクトリにあります。

関連情報

Ultra Light 概要 [8 ページ]
Microsoft Windows Mobile 用の Ultra Light API の利点 [21 ページ]

4.2.1 Ultra Light J アプリケーション開発のクイックスタートガイド

Ultra Light J アプリケーションを作成するときは、一般にアプリケーションコードでいくつかのデータ管理タスクを実行します。

1. Ultra Lght J API パッケージをユーザの Java ファイルにインポートします。
   Ultra Light J パッケージの名前とロケーションは、アプリケーション開発の対象デバイスによって異なります。
2. 新しい Configuration オブジェクトを作成して、データベースを作成するか、データベースに接続します。
   Configuration オブジェクトは、クライアントデータベースの保存場所または作成先を定義します。また、データベースへの接続に必要なユーザ名とパスワードを指定します。異なるデバイスや非永続データベースストア用にさまざまな Configuration オブジェクトがあります。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

3. 新しい Connection オブジェクトを作成します。
   Connection オブジェクトは、Configuration オブジェクトで定義されている指定内容に従ってクライアントデータベースに接続します。
4. SQL 文を使用してデータベーススキーマを作成または変更し、PreparedStatement インタフェースを使用してデータベースを問い合わせます。
   SQL 文を使用して、データベースのテーブル、インデックス、外部キー、パブリケーションを作成または更新できます。PreparedStatement オブジェクトは、Connection オブジェクトに関連付けられているデータベースを問い合わせます。引数には、サポートされている SQL 文を文字列として指定します。PreparedStatement オブジェクトを使用して、データベースの内容を更新できます。
5. ResultSet オブジェクトを生成します。
   ResultSet オブジェクトは、SQL SELECT 文を含む PreparedStatement が Connection オブジェクトによって実行されたときに作成されます。ResultSet オブジェクトを使用して、データベースのテーブルの内容を確認するために、クエリ結果のローを取得できます。

関連情報

Ultra Light SQL 文 [475 ページ]

4.2.2 Android セットアップの考慮事項

Android デバイス用のアプリケーションを開発する前に、Ultra Light J API の次の考慮事項に留意してください。

JAR リソースファイル

Ultra Light J API のアプリケーションを設定する場合、適切な `UltraLiteJ17.jar` または `UltraLiteJNI17.jar` ファイルが使用されるようにプロジェクトが正しく設定されていることを確認してください。

Ultra Light J API は、SQL Anywhere のインストールの `UltraLite¥UltraLiteJ¥Android¥UltraLiteJNI17.jar` ファイルに格納されています。Android 開発プロジェクトを設定し、クラスパスに `UltraLiteJNI17.jar` ファイルを含める必要があります。

次の文を使用して、Ultra Light J パッケージを Java ファイルにインポートします。

```
import com.sap.ultralitejni17.*;
```

このマニュアル内のすべてのサンプルコードとチュートリアルでは、上記の文が指定されていること、開発者が Eclipse での Java アプリケーション開発に精通していることを前提としています。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

4.2.3 Ultra Light データベースの作成と接続アプローチ

Java アプリケーションでデータを操作するには、先にデータベースに接続する必要があります。Ultra Light データベースは、Ultra Light J API を使用して作成し、指定されたパスワードで接続することができます。

> **注記**
>
> Ultra Light J API を使用しないで Ultra Light データベースを作成する場合は、SQL Central または Ultra Light のコマンドラインユーティリティを使用できます。

データベースストアを設定するには、Configuration オブジェクトを使用します。Configuration オブジェクトは、いくつかの方法で実装できます。Ultra Light J API でサポートされているデータベースストアのタイプごとに固有の実装があります。各実装には、データベースストアの設定に使用するメソッドのセットがあります。

次の表に、サポート対象データベースストアで利用できる Configuration オブジェクトの実装を示します。

ストアの型	Ultra Light J API のサポート
Android ファイルシステム	ConfigFileAndroid インタフェース
非永続型 (メモリ内)	ConfigNonPersistent インタフェース

Configuration オブジェクトを作成して設定した後、Connection オブジェクトを使用してデータベースを作成するか、データベースに接続します。また、Connection オブジェクトを使用して、次の操作を実行できます。

トランザクション

トランザクションは、コミットまたはロールバックの間の一連のオペレーションです。永続的なデータベースストアの場合、コミット操作は、最後のコミットまたはロールバックの実行後に行われたすべての変更を永続的にします。ロールバック操作は、最後にコミットが呼び出されたときの状態にデータベースを戻します。

Ultra Light のトランザクションとローレベルの操作は、それぞれアトミックです。複数のカラムを対象とした挿入操作では、すべてのカラムにデータが挿入されるか、どのカラムにもデータが挿入されないかのいずれかです。

トランザクションは、Connection オブジェクトの commit メソッドを使用してデータベースにコミットする必要があります。

SQL 準備文

SQL 文を処理するメソッドが PreparedStatement インタフェースに用意されています。PreparedStatement は、Connection オブジェクトの prepareStatement メソッドを使用して作成できます。

同期

Connection オブジェクトから、Mobile Link の同期を管理するオブジェクトのセットにアクセスできます。

このセクションの内容:

データベースの作成とデータベースへの接続 [697 ページ]
データベースの作成またはデータベースへの接続には、Ultra Light J API と Java アプリケーションを使用します。

関連情報

Ultra Light データベースの作成方法 [27 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

Mobile Link ファイル転送 [85 ページ]
Ultra Light データベースのアンロードユーティリティ (ulunload) [236 ページ]

4.2.3.1 データベースの作成とデータベースへの接続

データベースの作成またはデータベースへの接続には、Ultra Light J API と Java アプリケーションを使用します。

前提条件

Ultra Light J API を実装する Android デバイス用の既存の Java アプリケーション

手順

1. データベース名を参照し、プラットフォームに適した新しい Configuration を作成します。

   次の例では、*config* が Configuration オブジェクトの名前で、*DBname* がデータベースの名前です。

   ```
   ConfigFileAndroid config =
       DatabaseManager.createConfigurationFileAndroid(
           "DBname.udb",
           getApplicationContext()
       );
   ```

2. Configuration オブジェクトのメソッドを使用して、データベースプロパティを設定します。

   たとえば、setPassword メソッドを使用して新しいデータベースパスワードを設定できます。

   ```
   config.setPassword("my_password");
   ```

   setCreationString メソッドと setConnectionString メソッドを使用して、それぞれ追加の作成パラメータと接続パラメータを設定します。

3. データベースを作成する、またはデータベースに接続するために、Connection オブジェクトを作成します。

   ```
   Connection conn = DatabaseManager.createDatabase(config);
   ```

   DatabaseManager.createDatabase メソッドによりデータベースが作成され、そのデータベースへの接続が返されます。

   また、既存のデータベースに接続するには次のコードを使用します。

   ```
   Connection conn = DatabaseManager.connect(config);
   ```

   connect メソッドはデータベース接続処理を完了します。データベースが存在しない場合は、エラーがスローされます。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

結果

Java アプリケーションから SQL 文を実行すると、データベースにテーブルとインデックスを作成できますが、データベース名、パスワード、ページサイズなどの特定のデータベース作成パラメータを変更することはできません。

関連情報

Ultra Light オプション [138 ページ]
Ultra Light 接続パラメータ [175 ページ]

4.2.4 スキーマ操作とデータ管理のクイックスタートガイド

SQL 文とクエリを使用してデータベースのテーブル、インデックス、外部キー、パブリケーション、ローの作成、更新、取得を行います。

スキーマ操作によってデータを管理する場合、一般的にアプリケーションコードで次のタスクを実行します。

1. スキーマ操作を実行します。
   データベース接続に対して CREATE TABLE または CREATE INDEX などの SQL 文を使用することで、スキーマを管理し変更します。
2. ロー操作を管理します。
   データベース接続に対して INSERT、UPDATE、または DELETE などの SQL 文を使用することで、テーブル内のデータを管理します。
3. 結果セットにローデータを取得します。
   SELECT 文を使用して結果セットを取得してから、結果セットのナビゲーションメソッド (*previous* および *next* など) を使用し、ローデータをトラバースします。

このセクションの内容:

例: Android デバイスに対するデータベース操作の管理 [699 ページ]
この例は、Ultra Light J API を使用してデータベースを作成し、基本的なデータオペレーションを実行するサンプルクラスを Android アプリケーションで作成する方法を示しています。

スキーマの操作 [702 ページ]
テーブル作成などの一般的なデータベース操作は、PreparedStatement オブジェクトを使用して実行できます。

ロー操作の管理 [703 ページ]
ロー操作は、Connection オブジェクトと PreparedStatement オブジェクトを使用して管理できます。

ローデータの取り出し [705 ページ]
データは、executeQuery メソッドと、ResultSet オブジェクトのナビゲーションメソッドを使用して取り出すことができます。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

4.2.4.1 例: Android デバイスに対するデータベース操作の管理

この例は、Ultra Light J API を使用してデータベースを作成し、基本的なデータオペレーションを実行するサンプルクラスを Android アプリケーションで作成する方法を示しています。

この例は、以下のオペレーションを示しています。

- データベースにテーブルを作成します。
- テーブルに新しいローを挿入します。
- テーブル内のローを更新します。
- テーブルからローを削除します。
- データベースへの変更をコミットします。
- 結果セットを作成することによって、テーブルのすべてのローを選択します。
- 結果セットをトラバースしてデータベースのローを表示します。

このクラスには、これらの操作に加え、正常な動作をログに出力するために使用される、PrintText という名前のメソッド (Eclipse の *LogCat* タブ) や、Ultra Light J API 操作の実行中に発生する可能性のあるエラーのレポートに使用される HandleError メソッドがあります。

```
package com.sampleapp;
import android.app.Activity;
import android.os.Bundle;
import android.util.Log;
import com.sap.ultralitejni17.*;
public class NewUltraLiteJAppActivity extends Activity {
    Connection _conn = null;
    ResultSet _departments = null;
    PreparedStatement _inserter = null;
    PreparedStatement _updater = null;
    PreparedStatement _deleter = null;
    PreparedStatement _preparer = null;
    public void PrintText(String strText) {
        Log.i("NewUltraLiteJAppActivity", strText);
    }
    public void HandleError(ULjException err) {
        Log.w("NewUltraLiteJAppActivity", "Exception: " + err.toString());
    }
    public Connection GetDatabase(String strFilename) {
        ConfigFileAndroid config = null;
        Connection dbConnection = null;
        try {
            config = DatabaseManager.createConfigurationFileAndroid(
                    strFilename, getApplicationContext()
            );
            dbConnection = DatabaseManager.connect(config);
            PrintText("Successfully connected to the database at: "
                    + strFilename);
        } catch(ULjException ex) {
            if (config != null) {
                try {
                    dbConnection = DatabaseManager.createDatabase(config);
                    PrintText("Successfully created a new database at: "
                            + strFilename);
                } catch(ULjException exception) {
                    HandleError(exception);
                }
            }
            HandleError(ex);
        }
        return dbConnection;
    }
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

```
    public void Commit() {
        try {
            _conn.commit();
        } catch (ULjException e1) {
            HandleError(e1);
        }
    }
    public void CloseDatabase() {
        try {
            _conn.release();
        } catch (ULjException e) {
            HandleError(e);
        }
    }
    public void ExecuteSQLStatement(String strSQLstmt) {
        PreparedStatement ps;
        try {
            ps = _conn.prepareStatement(strSQLstmt);
            ps.execute();
            ps.close();
            PrintText("Successfully executed: " + strSQLstmt);
        } catch (ULjException e) {
            HandleError(e);
        }
    }
    public void InitStatements() {
        String stmt;
        try {
            stmt = "INSERT INTO Department(dept_no, name) VALUES (?,?)";
            _inserter = _conn.prepareStatement(stmt);
            stmt = "UPDATE Department SET dept_no = ? WHERE dept_no = ?";
            _updater = _conn.prepareStatement(stmt);
            stmt = "DELETE FROM Department WHERE dept_no = ?";
            _deleter = _conn.prepareStatement(stmt);
        } catch (ULjException e) {
            HandleError(e);
        }
    }
    public void FiniStatements() {
        try {
            _departments.close();
            _inserter.close();
            _updater.close();
            _deleter.close();
            _preparer.close();
        } catch (ULjException e) {
            HandleError(e);
        }
    }
    public void AddDepartment(int deptID, String deptName) {
        try {
            _inserter.set(1, deptID);
            _inserter.set(2, deptName);
            _inserter.execute();
            PrintText("Successfully executed:"
                    + " INSERT INTO Department(dept_no, name)"
                    + " VALUES (" + deptID + "," + deptName + ")");
        } catch (ULjException e) {
            HandleError(e);
        }
    }
    public void UpdateDepartment(int deptIDold, int deptIDnew) {
        try {
            _updater.set(1, deptIDnew);
            _updater.set(2, deptIDold);
            _updater.execute();
            PrintText("Successfully executed:"
                    + " UPDATE Department SET dept_no = " + deptIDnew
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

```
                    + " WHERE dept_no = " + deptIDold);
        } catch (ULjException e) {
            HandleError(e);
        }
    }
    public void DeleteDepartment(int deptID) {
        try {
            _deleter.set(1, deptID);
            _deleter.execute();
            PrintText("Successfully executed:"
                    + " DELETE FROM Department WHERE dept_no = " + deptID);
        } catch (ULjException e) {
            HandleError(e);
        }
    }
    public ResultSet SelectDepartmentRows() {
        String stmt = "SELECT * FROM Department ORDER BY dept_no";
        _preparer = null;
        _departments = null;
        try {
            _preparer = _conn.prepareStatement(stmt);
            _departments = _preparer.executeQuery();
            PrintText("Successfully executed: " + stmt);
        } catch (ULjException e) {
            HandleError(e);
        }
        return _departments;
    }
    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        PrintText("Starting application...");

        _conn = GetDatabase("test.udb");
        if (_conn == null) {
            return;
        }

        String[] stmt = new String[3];

        stmt[0] = "CREATE TABLE Department("
                + "dept_no INT PRIMARY KEY, "
                + "name CHAR(50) NOT NULL)";
        stmt[1] = "CREATE TABLE Employee("
            + "id INT PRIMARY KEY, "
            + "last_name CHAR(50) NOT NULL, "
            + "first_name CHAR(50) NOT NULL, "
            + "dept_id INT NOT NULL, "
            + "NOT NULL FOREIGN KEY(dept_id) "
            + "REFERENCES Department(dept_no))";
        stmt[2] = "CREATE INDEX ON Employee(last_name, first_name)";

        for(int i = 0; i< stmt.length; i++) {
            ExecuteSQLStatement(stmt[i]);
        }
        InitStatements();
        AddDepartment(101, "Electronics");
        AddDepartment(105, "Sales");
        AddDepartment(109, "Accounting");
        UpdateDepartment(101, 102);
        DeleteDepartment(102);
        Commit();
        _departments = SelectDepartmentRows();
        if (_departments != null) {
            try {
                while(_departments.next()) {
                    int dept_no = _departments.getInt(1);
```


© 2016 SAP SE or an SAP affiliate company. All rights reserved.

```
                    String dept_name = _departments.getString(2);
                    PrintText("Department no.:" + dept_no
                            + " Department name: " + dept_name);
                }
            } catch (ULjException e) {
                HandleError(e);
            }
        }

        FiniStatements();
        CloseDatabase();
        PrintText("Closing application...");
        finish();
    }
}
```

関連情報

Ultra Light SQL 文 [475 ページ]

4.2.4.2 スキーマの操作

テーブル作成などの一般的なデータベース操作は、PreparedStatement オブジェクトを使用して実行できます。

次の一般的なタスクによってスキーマ操作を実行します。

1. 文字列変数に SQL 文を作成します。
2. 文字列変数を Connection.prepareStatement メソッドに渡して PreparedStatement オブジェクトを作成します。
3. PreparedStatement.execute メソッドを呼び出してデータベースの操作を実行します。
4. PreparedStatement オブジェクトを閉じてリソースを解放します。

例

この例に示したコードは、Ultra Light J API を使用してスキーマやデータを管理する基本操作を実行する方法を紹介している完全なコードサンプルの一部です。

個々の CREATE TABLE 文と CREATE INDEX 文が組み立てられ、ExecuteSQLStatement という名前のカスタムメソッドに渡されて、必要なスキーマ操作がすべて実行されます。

```
String[] stmt = new String[3];
stmt[0] = "CREATE TABLE Department("
    + "dept_no INT PRIMARY KEY, "
    + "name CHAR(50) NOT NULL)";
stmt[1] = "CREATE TABLE Employee("
    + "id INT PRIMARY KEY, "
    + "last_name CHAR(50) NOT NULL, "
    + "first_name CHAR(50) NOT NULL, "
    + "dept_id INT NOT NULL, "
    + "NOT NULL FOREIGN KEY(dept_id) "
    + "REFERENCES Department(dept_no))";
stmt[2] = "CREATE INDEX ON Employee(last_name, first_name)";

for(int i = 0; i< stmt.length; i++) {
    ExecuteSQLStatement(stmt[i]);
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

```
}
```

ExecuteSQLStatement メソッドは次のコードで構成されています。

```
public void ExecuteSQLStatement(String strSQLstmt) {
    PreparedStatement ps;
    try {
        ps = _conn.prepareStatement(strSQLstmt);
        ps.execute();
        ps.close();
        PrintText("Successfully executed: " + strSQLstmt);
    } catch (ULjException e) {
        HandleError(e);
    }
}
```

関連情報

例: Android デバイスに対するデータベース操作の管理 [699 ページ]

4.2.4.3 ロー操作の管理

ロー操作は、Connection オブジェクトと PreparedStatement オブジェクトを使用して管理できます。

次の一般的なタスクを実行して、ロー操作を管理します。

1. 文字列変数に SQL 文を作成します。
2. 文字列変数を Connection.prepareStatement メソッドに渡して PreparedStatement オブジェクトを作成します。
3. ホスト変数を設定します。疑問符 (?) で表し、PreparedStatement.set メソッドを使用します。
   各ホスト変数は文内の位置を表す序数で参照できます。たとえば、最初の ? は 1、2 番目の ? は 2 のようになります。下記の例に示されている *set* メソッドでは、変数の位置を表す序数を参照し、新しい値を指定できます。
4. PreparedStatement.execute メソッドを呼び出してデータベースの操作を実行します。
5. Connection.commit メソッドを呼び出してデータベースへの変更をコミットし、永続的な変更にするか、または Connection.rollback メソッドを呼び出します。
   トランザクションは、Connection インタフェースでサポートされているメソッドを使用して明示的にコミットまたはロールバックする必要があります。
6. PreparedStatement オブジェクトを閉じてリソースを解放します。

例

この例に示したコードは、Ultra Light J API を使用してスキーマやデータを管理する基本操作を実行する方法を紹介している完全なコードサンプルの一部です。

グローバルの PreparedStatement オブジェクトが定義され、InitStatements メソッドの呼び出しによってインスタンス化されます。データの挿入、更新、削除の操作はそれぞれ、AddDepartment、UpdateDepartment、DeleteDepartment

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

のカスタムメソッドで実行されます。Commit メソッドはロー操作を永続的なものにします。FiniStatements メソッドは、グローバルの PreparedStatement オブジェクトを閉じて、リソースを解放します。

```
Connection _conn = null;
ResultSet _departments = null;
PreparedStatement _inserter = null;
PreparedStatement _updater = null;
PreparedStatement _deleter = null;
PreparedStatement _preparer = null;
public void InitStatements() {
    String stmt;
    try {
        stmt = "INSERT INTO Department(dept_no, name) VALUES (?,?)";
        _inserter = _conn.prepareStatement(stmt);
        stmt = "UPDATE Department SET dept_no = ? WHERE dept_no = ?";
        _updater = _conn.prepareStatement(stmt);
        stmt = "DELETE FROM Department WHERE dept_no = ?";
        _deleter = _conn.prepareStatement(stmt);
    } catch (ULjException e) {
        HandleError(e);
    }
}
public void FiniStatements() {
    try {
        _departments.close();
        _inserter.close();
        _updater.close();
        _deleter.close();
        _preparer.close();
    } catch (ULjException e) {
        HandleError(e);
    }
}
public void AddDepartment(int deptID, String deptName) {
    try {
        _inserter.set(1, deptID);
        _inserter.set(2, deptName);
        _inserter.execute();
        PrintText("Successfully executed:"
                + " INSERT INTO Department(dept_no, name)"
                + " VALUES (" + deptID + "," + deptName + ")");
    } catch (ULjException e) {
        HandleError(e);
    }
}
public void UpdateDepartment(int deptIDold, int deptIDnew) {
    try {
        _updater.set(1, deptIDnew);
        _updater.set(2, deptIDold);
        _updater.execute();
        PrintText("Successfully executed:"
                + " UPDATE Department SET dept_no = " + deptIDnew
                + " WHERE dept_no = " + deptIDold);
    } catch (ULjException e) {
        HandleError(e);
    }
}
public void DeleteDepartment(int deptID) {
    try {
        _deleter.set(1, deptID);
        _deleter.execute();
        PrintText("Successfully executed:"
                + " DELETE FROM Department WHERE dept_no = " + deptID);
    } catch (ULjException e) {
        HandleError(e);
    }
}
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

```
public void Commit() {
    try {
        _conn.commit();
    } catch (ULjException e1) {
        HandleError(e1);
    }
}
```

注記

1 回しか実行する必要のない SQL 文を作成する場合は、ホスト変数に文字列連結を利用することをお奨めします。

たとえば、次のコードは、DeleteDepartment メソッドの置換に使用できるコードですが、文字列連結で SQL 文が作成されています。

```
public void DeleteDepartment(int deptID) {
    String stmt = "DELETE FROM Department WHERE dept_no = " + deptID;
    PreparedStatement deleter;
    try {
        deleter = _conn.prepareStatement(stmt);
        deleter.execute();
        deleter.close();
        PrintText("Successfully executed: " + stmt);
    } catch (ULjException e) {
        HandleError(e);
    }
}
```

関連情報

例: Android デバイスに対するデータベース操作の管理 [699 ページ]

4.2.4.4 ローデータの取り出し

データは、executeQuery メソッドと、ResultSet オブジェクトのナビゲーションメソッドを使用して取り出すことができます。

次の一般的なタスクを実行して、テーブルからローデータを取り出します。

1. 文字列変数で SELECT SQL 文を作成します。
2. 文字列変数を Connection.prepareStatement メソッドに渡して PreparedStatement オブジェクトを作成します。
3. PreparedStatement.executeQuery メソッドを呼び出して、クエリ結果を ResultSet オブジェクトに割り当てます。
4. ナビゲーションメソッドを使用して ResultSet オブジェクトをトラバースし、ローデータを取り出します。
   結果セットのトラバースには、次のナビゲーションメソッドを使用できます。

   afterLast

   カーソルを最後のローの直後に配置します。
   beforeFirst

   最初のローの直前に配置します。
   first

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

最初のローに移動します。
last

最後のローに移動します。
next

次のローに移動します。
previous

前のローに移動します。
relative(offset)

現在のローを基準にして、符号付きオフセット値で指定された数だけローを移動します。オフセット値を正の値で指定すると、現在の結果セットのポインタ位置から前方に移動します。負の値で指定すると後方に移動します。オフセット値が 0 の場合、カーソルは現在のロケーションから移動しませんが、ローバッファが再配置されます。

5. ResultSet オブジェクトと PreparedStatement オブジェクトを閉じてリソースを解放します。

例

この例に示したコードは、Ultra Light J API を使用してスキーマやデータを管理する基本操作を実行する方法を紹介している完全なコードサンプルの一部です。

カスタム SelectDepartmentRows メソッドで結果セットが取り出されます。

```
public ResultSet SelectDepartmentRows() {
    String stmt = "SELECT * FROM Department ORDER BY dept_no";
    _preparer = null;
    _departments = null;
    try {
        _preparer = _conn.prepareStatement(stmt);
        _departments = _preparer.executeQuery();
        PrintText("Successfully executed: " + stmt);
    } catch (ULjException e) {
        HandleError(e);
    }
    return _departments;
}
```

next ナビゲーションメソッドを使用して結果セットをトラバースすることで、ローデータが取り出されます。

```
_departments = SelectDepartmentRows();
if (_departments != null) {
    try {
        while(_departments.next()) {
            int dept_no = _departments.getInt(1);
            String dept_name = _departments.getString(2);
            PrintText("Department no.:" + dept_no
                    + " Department name: " + dept_name);
        }
    } catch (ULjException e) {
        HandleError(e);
    }
}
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

関連情報

例: Android デバイスに対するデータベース操作の管理 [699 ページ]

4.2.5 Ultra Light J でのスキーマ情報

データベーススキーマの記述は、プログラムを使用して取得できます。これらの記述は、スキーマ情報と呼ばれ、システムテーブルおよび Ultra Light J API スキーマインタフェースを使用してアクセスできます。

システムテーブルを使用したスキーマ情報へのアクセス

データベースのスキーマ情報は、Ultra Light のシステムテーブルに格納されます。この情報にアクセスするには、正規 SQL クエリを実行して適切なシステムテーブルから所定の情報を選択し、結果セットにアクセスします。

スキーマインタフェースを使用したスキーマ情報へのアクセス

一部のスキーマ情報は、システムテーブルの代わりにスキーマインタフェースを使用してアクセスできます。Ultra Light J API には、次のスキーマインタフェースが含まれています。

TableSchema

カラムとインデックスの設定に関する情報を返します。

IndexSchema

インデックス内のカラムに関する情報を返します。IndexSchema オブジェクトは、TableSchema オブジェクトから取得できます。

ColumnSchema

テーブル内のカラムに関する情報を返します。ColumnSchema オブジェクトは、TableSchema オブジェクトから取得できます。

関連情報

ローデータの取り出し [705 ページ]
Ultra Light のシステムテーブル [240 ページ]

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

4.2.6 Ultra Light J でのエラー処理

ULjException クラスおよび SQLCode クラスを使用して、エラーを処理できます。ほとんどの Ultra Light メソッドは、ULException エラーをスローします。

SQLCode エラーは、エラー型を示す負数で、ULjException.SQLE_INDEX_NOT_FOUND などの定数を使用して参照できます。ULjException.getErrorCode メソッドを使用して、エラーに割り当てられた SQLCode 値を取得できます。ULjException.toString メソッドを使用して、エラーの説明文を取得できます。

例

次の例は、Ultra Light データベースへの接続時に発生する可能性があるエラーを、ULjException クラスを使用して処理する Java クラスを示しています。

```
import com.sap.ultralitejni17.*;
import java.util.log;
import android.content.Context;
class DataAccess {
    DataAccess() {
    }

    public static synchronized DataAccess getDataAccess(boolean reset)
        throws Exception
    {
        if (_da == null) {
            _da = new DataAccess();
            ConfigFileAndroid config =
DatabaseManager.createConfigurationFileAndroid("HelloDB.udb",
getApplicationContext());
            if (reset) {
                _conn = DatabaseManager.createDatabase(config);
            }
            else {
                try {
                    _conn = DatabaseManager.connect(config);
                }
                catch (ULjException uex1) {
                    if (uex1.getErrorCode() !=
ULjException.SQLE_ULTRALITE_DATABASE_NOT_FOUND) {
                        Log.e("Exception: " + uex1.toString(), ". Recreating
database...");
                    }
                    _conn = DatabaseManager.createDatabase(config);
                }
            }
        }
        return _da;
    }
    private static Connection _conn;
    private static DataAccess _da;
}
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

4.2.7 Ultra Light J を使用した Mobile Link データの同期

データの同期は、HTTP ネットワークプロトコルまたは HTTPS ネットワークプロトコルを使用して実行できます。HTTPS 同期を使用すると、Mobile Link サーバに安全な暗号化が提供されます。

データを同期するには、アプリケーションで次の手順を実行する必要があります。

1. 統合データベースに関する情報 (サーバ名、ポート番号)、同期するベータベース名、同期するテーブルの定義を含む syncParms オブジェクトをインスタンス化します。
2. syncParms オブジェクトを渡した接続オブジェクトから synchronize メソッドを呼び出して、同期を実行します。

同期するデータはテーブルレベルで定義できます。テーブルの一部分を同期するように設定することはできません。

CustDB チュートリアルは、Ultra Light J アプリケーションを使用したデータの同期を示しています。この例は、*%SQLANYSAMP17%*¥UltraLiteJ¥Android¥CustDB にあります。

このセクションの内容:

Ultra Light J 同期ストリームのネットワークプロトコルのオプション [709 ページ]
Mobile Link サーバと同期するときは、アプリケーション内でネットワークプロトコルを設定する必要があります。各データベースはネットワークプロトコルを通じて同期されます。Ultra Light J では、HTTP と HTTPS の 2 つのネットワークプロトコルを使用できます。

関連情報

チュートリアル:Android アプリケーションの構築 [711 ページ]

4.2.7.1 Ultra Light J 同期ストリームのネットワークプロトコルのオプション

Mobile Link サーバと同期するときは、アプリケーション内でネットワークプロトコルを設定する必要があります。各データベースはネットワークプロトコルを通じて同期されます。Ultra Light J では、HTTP と HTTPS の 2 つのネットワークプロトコルを使用できます。

設定したネットワークプロトコルでは、対応するプロトコルオプションのセットから選択することによって、Ultra Light J アプリケーションが Mobile Link サーバを特定して通信できるようにします。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します。

HTTP ネットワークプロトコルの設定

HTTP ネットワークプロトコルは、Ultra Light J API の StreamHTTPParms インタフェースを使用して設定します。インタフェースのメソッドを使用して、Mobile Link サーバで定義されているネットワークプロトコルのオプションを指定します。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

HTTPS ネットワークプロトコルの設定

HTTPS ネットワークプロトコルは、Ultra Light J API の StreamHTTPSParms インタフェースを使用して設定します。インタフェースのメソッドを使用して、Mobile Link サーバで定義されているネットワークプロトコルのオプションを指定します。

4.2.8 Android 用 Ultra Light J アプリケーションの配備

適切な作成パラメータ、接続パラメータ、同期パラメータ、プロトコルオプション、メソッド呼び出し、配備ファイルを指定して、Ultra Light J アプリケーションが Android スマートフォンで正常に実行されるようにします。

手順

1. 次のファイルを Android プロジェクトに追加します。
   - `%SQLANY17%¥UltraLite¥UltraLiteJ¥Android¥UltraLiteJNI17.jar`
   - `%SQLANY17%¥UltraLite¥UltraLiteJ¥Android¥ARM¥libultralitej17.so`
2. 次のパラメータを指定します。
   - 難読化を使用している場合は、データベース作成時に作成パラメータ *obfuscate=1* を設定します。
   - AES 暗号化を使用する場合は、データベース作成時または接続時に接続パラメータ *DBKEY*=`encryption-key` を設定します。

   これらのパラメータを設定するには、setCreationString メソッドと setConnectionString メソッドを使用します。
3. Ultra Light アプリケーションで同期を使用する場合は、パラメータを適切に設定してください。

   表 1: 結果

同期タイプ	パラメータの設定
TCP/IP	Set the *Stream* synchronization parameter to "tcpip".
HTTP	Set the *Stream* synchronization parameter to "http".
RSA TLS	Set the *Stream* synchronization parameter to "tls".
RSA HTTPS	Set the *Stream* synchronization parameter to "https".

4. RSA エンドツーエンド暗号化 (RSA E2EE) を使用している場合は、プロトコルオプション *e2ee_public_key*=`key-file` を設定します。
5. ZLIB 圧縮を使用している場合は、プロトコルオプション *compression=zlib* を設定します。
6. AES 暗号化を使用している場合は、ConfigPersistent.EnableAesDBEncryption メソッドを呼び出します。
7. RSA TLS、RSA HTTPS、または RSA E2EE を使用している場合は、`%SQLANY17%¥UltraLite¥UltraLiteJ¥Android¥ARM¥libmlcrsa17.so` を配備します。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

結果

Ultra Light J アプリケーションは配備先の Android デバイスで正常に稼働します。

次のステップ

アプリケーションが配備された Android モバイルデバイスに Ultra Light データベースを配備するか、または配備したアプリケーションで新しいデータベースを作成します。

関連情報

Ultra Light データベースの配備方法 [124 ページ]
Ultra Light アプリケーションのビルドと配備の仕様 [118 ページ]

4.2.9 サンプルコード

Ultra Light J API を使用するコード例は、*%SQLANYSAMP17%*`¥UltraLiteJ¥` ディレクトリにあります。

CustDB サンプルデータベースを使用するサンプル Eclipse プロジェクトが *%SQLANYSAMP17%*`¥UltraLiteJ¥Android¥CustDB` ディレクトリに用意されています。ソースコードは、*%SQLANYSAMP17%*`¥UltraLiteJ¥Android¥CustDB¥src¥com¥sap¥custdb` にあります。

例に基づいたチュートリアルを利用できます。

4.3 チュートリアル:Android アプリケーションの構築

このチュートリアルでは、Ultra Light J API および Eclipse 環境を使用して、アプリケーションを開発する方法について説明します。このチュートリアルでは、Windows Simulator 上でアプリケーションを実行します。

前提条件

次の項目が必要です。

- Java の知識
- Eclipse の知識

次のソフトウェアが必要です。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

- Eclipse 3.5.2 以降
- Android SDK Starter Package
- Android Development Tools (ADT) Plug-in for Eclipse 1.1 以降
- SQL Anywhere17 サンプル
- Ultra Light J API

コンテキスト

このチュートリアルで使用する Android アプリケーションは、*%SQLANYSAMP17%*¥UltraLiteJ¥Android¥Eclipse¥CustDB¥ ディレクトリにあります。

> **注記**
>
> このチュートリアルの Android Studio バージョンは、*%SQLANYSAMP17%*¥UltraLiteJ¥Android¥AndroidStudio¥CustDB¥ ディレクトリにあります。

src¥com¥sap¥custdb ディレクトリにあるアプリケーションコードで、Ultra Light J API で行う次の操作を参照できます。

- Ultra Lite のリモートデータベースの作成。
- データベースでの SQL の操作。
- Mobile Link を使用した SQL Anywhere CustDB サンプルデータベースとのデータの同期。

res¥menu および res¥layout ディレクトリは、Android のメニュー項目とインタフェースの作成方法を示します。新しい Android プロジェクトを作成するときに、Eclipse を介して、これらのファイルを表示できます。

Android アプリケーションからネットワークにアクセスできるように AndroidManifest.xml プロジェクトファイルが変更されました。これはデータ同期に必要です。次のパーミッション文が追加されました。

```
<uses-permission android:name="android.permission.INTERNET" />
```

1. レッスン 1:新しい Android プロジェクトの設定 [713 ページ]
   このレッスンでは、Eclipse 統合開発環境を介して、新しい Android プロジェクトを作成します。
2. レッスン 2:Mobile Link サーバの起動 [715 ページ]
   このレッスンでは、Mobile Link サーバを起動して、同期を実行します。
3. レッスン 3: Android アプリケーションの実行 [716 ページ]
   このレッスンでは、Android Simulator 上でアプリケーションを実行します。
4. レッスン 4:Android アプリケーションのテストと同期 [717 ページ]
   このレッスンでは、Android アプリケーションを使用して、Ultra Light のリモートデータベースを更新し、CustDB 統合データベースと同期します。
5. レッスン 5:クリーンアップ [718 ページ]
   最近作成したチュートリアルをコンピュータから削除します。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

関連情報

Android SDK および Eclipse 用 Android Development Tools (ADT) プラグイン
Ultra LightJ API リファレンス [719 ページ]

4.3.1 レッスン 1:新しい Android プロジェクトの設定

このレッスンでは、Eclipse 統合開発環境を介して、新しい Android プロジェクトを作成します。

前提条件

このレッスンでは、必要なソフトウェアをすでにインストールしていることを前提としています。

コンテキスト

このチュートリアルは、開発者が Java と Eclipse に精通していることを前提としています。

手順

1. Android ライブラリを、Android CustDB サンプルディレクトリにコピーします。

   コマンドプロンプトを開いて、*%SQLANYSAMP17%*`¥UltraLiteJ¥Android¥CustDB¥` ディレクトリに移動し、次のコマンドを実行します。

   ```
   setup.bat
   ```

   `UltraLiteJNI`*17*`.jar` ファイルと `libultralitej`*17*`.so` ファイルが `Android¥CustDB¥libs` ディレクトリと `Android¥CustDB¥libs¥armeabi` ディレクトリにコピーされます。
2. Eclipse を実行します。

   デフォルトアプリケーションパスは `C:¥Eclipse¥eclipse.exe` です。
3. *Workspace* フィールドで、CustDB サンプルディレクトリ以外の作業ディレクトリを指定して、*OK* をクリックします。
4. Eclipse への CustDB プロジェクトのインポート
   a. *File* > *Import* をクリックします。
   b. *General* ディレクトリを展開し、*Existing Projects into Workspace* を選択します。次へをクリックします。
   c. *Select Root Directory* フィールドに、***%SQLANYSAMP17%¥UltraLiteJ¥Android¥CustDB*** と入力します。*Copy Projects Into Workspace* を選択して、*Finish* をクリックします。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

5. Eclipse で適切な Android SDK パスが指定されていることを確認します。

   > **注記**
   >
   > パスを指定する前に Android SDK をインストールする必要があります。

   a. *Window* > *Preferences* をクリックします。
   b. 左ウィンドウ枠で、*Android* をクリックします。
   c. *SDK Location* フィールドで、Android SDK のロケーションを入力し、*Apply* をクリックします。

      利用可能なビルドターゲットのリストが表示されます。
   d. *OK* をクリックします。
6. Eclipse で Ultra Light J ライブラリパスが指定されていることを確認します。
   a. ファイル > プロパティ をクリックします。
   b. 左ウィンドウ枠で、*Java Build Path* > *User libraries* をクリックします。
   c. *Libraries* タブをクリックします。
   d. *UltraLiteJNI17.jar* をクリックし、*Edit* をクリックします。
   e. 作業ディレクトリから `¥CustDB¥libs¥UltraLiteJNI`*17*`.jar` を開きます。
   f. *OK* をクリックします。
7. プロジェクトに UltraLiteJNI Javadoc マニュアルへのパスを追加します。
   a. 左ウィンドウ枠で、*Javadoc Location* をクリックします。
   b. *Browse* をクリックし、*%SQLANY17%*`¥UltraLite¥UltraLiteJ¥Android¥html` を開きます。
   c. *OK* をクリックします。
   d. *OK* をクリックして、ウィンドウを閉じます。
8. プロジェクトの構築 *Project* > *Clean* をクリックし、*OK* をクリックします。

   プロジェクトはエラーを発生させずに構築する必要がありますが、問題タブの下に警告が表示されることがあります。

結果

Ultra Light J API が新しい Android アプリケーションで機能します。

次のステップ

次のレッスンに進みます。

タスクの概要: チュートリアル:Android アプリケーションの構築 [711 ページ]

次のタスク: レッスン 2:Mobile Link サーバの起動 [715 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

4.3.2 レッスン 2:Mobile Link サーバの起動

このレッスンでは、Mobile Link サーバを起動して、同期を実行します。

前提条件

このチュートリアルのこれまでのレッスンを完了している必要があります。

このチュートリアルの冒頭に一覧されているロールと権限を持っている必要があります。

手順

Mobile Link を起動するには、*%SQLANYSAMP17%*¥MobiLink¥CustDB¥ から次のコマンドを発行します。

```
mlsrv17 -v+ -zu+ -c "DSN=SQL Anywhere 17 CustDB;UID=ml_server;PWD=sql" -x
http(port=80) -ot ml.mls
```

-c オプションは、Mobile Link を SQL Anywhere CustDB データベースに接続します。-v+ オプションは、高い冗長レベルを設定して、処理中の内容を Mobile Link のサーバメッセージウィンドウで確認できるようにします。-x オプションは、通信に使用されているポート番号を指定します。-ot オプションは、ログファイル (`ml.mls`) が、Mobile Link サーバを起動したディレクトリに作成されるように指定します。Android アプリは、HTTP を使用して Mobile Link に接続します。ポート 80 に加えて別のポートを使用する場合、ファイアウォールでそのポートを開く必要があります。

結果

Mobile Link サーバが起動します。

次のステップ

次のレッスンに進みます。

タスクの概要: チュートリアル:Android アプリケーションの構築 [711 ページ]

前のタスク: レッスン 1:新しい Android プロジェクトの設定 [713 ページ]

次のタスク: レッスン 3: Android アプリケーションの実行 [716 ページ]

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

4.3.3 レッスン 3: Android アプリケーションの実行

このレッスンでは、Android Simulator 上でアプリケーションを実行します。

前提条件

このチュートリアルのこれまでのレッスンを完了している必要があります。

このチュートリアルの冒頭に記載されているロールと権限を持っている必要があります。

手順

1. Eclipse で Android の仮想デバイスを設定します。
   a. *Window* > *AVD Manager* をクリックします。
   b. **新規**をクリックします。

      Create New Android Virtual Device (AVD) ウィンドウが表示されます。
   c. **名前**項目に `my_avd` と入力します。
   d. *Target* フィールドで、*Android 2.2 - API Level 8* をクリックします。
   e. *Create AVD* をクリックします。
   f. *AVD Manager* ウィンドウを閉じます。
2. *Package Explorer* ウィンドウで、`CustDB` を選択します。
3. *Run* メニューで、*Run As* > *Android Application* を選択します。

   Android Simulator がロードされます。
4. *Menu* をクリックします。

   Android アプリケーションがロードされます。

結果

シミュレートされた Android デバイスに Ultra Light アプリケーションがロードされます。

次のステップ

次のレッスンに進みます。

タスクの概要: チュートリアル:Android アプリケーションの構築 [711 ページ]

 © 2016 SAP SE or an SAP affiliate company.All rights reserved.

前のタスク: レッスン 2:Mobile Link サーバの起動 [715 ページ]

次のタスク: レッスン 4:Android アプリケーションのテストと同期 [717 ページ]

4.3.4 レッスン 4:Android アプリケーションのテストと同期

このレッスンでは、Android アプリケーションを使用して、Ultra Light のリモートデータベースを更新し、CustDB 統合データベースと同期します。

前提条件

このチュートリアルのこれまでのレッスンを完了している必要があります。

このチュートリアルの冒頭に一覧されているロールと権限を持っている必要があります。

手順

1. *Employee ID* フィールドが 50、*Host* フィールドが 10.0.2.2、*Port* フィールドが 80 になっていることを確認し、*Save* をクリックします。

   アプリケーションは自動的に同期を実行し、一連の顧客、製品、注文情報が CustDB 統合データベースからアプリケーションにダウンロードされます。
2. Simulator で、**メニュー** > **新規** をクリックします。
3. *Customer* フィールドで *Ace Properties* を選択します。
4. *Product* フィールドで *4x8 Drywall x100* を選択します。
5. *Quantity* フィールドに *999* と入力します。
6. *Discount* フィールドに *25* と入力します。
7. *OK* をクリックし、新規注文を追加します。
8. CustDB 統合データベースとアプリケーションを同期します。

   Simulator で、*Menu* をクリックし、*Sync* を選択します。
9. Interactive SQL を CustDB 統合データベースに接続します。
   a. **スタート** > **プログラム** > *SQL Anywhere17* > **管理ツール** > *Interactive SQL* をクリックするか、次のコマンドを実行します。

      ```
      dbisql
      ```
   b. *ODBC* **データソース名**で、**参照**をクリックし、*SQL Anywhere 17 CustDB* を選択して、*OK* をクリックします。
   c. パスワードフィールドに *sql* と入力し、**接続**をクリックします。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

10. 同期が成功したことを確認します。

    Interactive SQL で次の SQL 文を実行します。

```
SELECT order_id, disc, quant, notes, status, c.cust_id,
    cust_name, p.prod_id, prod_name, price
  FROM ULOrder o, ULCustomer c, ULProduct p
  WHERE o.cust_id = c.cust_id
  AND o.prod_id = p.prod_id
  AND c.cust_name = 'Ace Properties'
  AND p.prod_name = '4x8 Drywall x100'
```

    Interactive SQL に注文にエントリが表示されたら、同期が成功しています。

11. [Simulator] ウィンドウを閉じます。

結果

シミュレータに加えた変更は CustDB 統合データベースと同期されます。

次のステップ

次のレッスンに進みます。

タスクの概要: チュートリアル:Android アプリケーションの構築 [711 ページ]

前のタスク: レッスン 3: Android アプリケーションの実行 [716 ページ]

次のタスク: レッスン 5:クリーンアップ [718 ページ]

4.3.5 レッスン 5:クリーンアップ

最近作成したチュートリアルをコンピュータから削除します。

前提条件

このチュートリアルのこれまでのレッスンを完了している必要があります。

このチュートリアルの冒頭に一覧されているロールと権限を持っている必要があります。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

手順

1. Eclipse を終了します。

   File > *Exit* をクリックします。
2. タスクバー上で、Mobile Link、Interactive SQL、同期クライアントの各ウィンドウを右クリックし、終了またはシャットダウンをクリックします。
3. CustDB データベースをリセットします。

   %SQLANYSAMP17%¥UltraLite¥CustDB ディレクトリから次のコマンドを実行します。

   ```
   makedbs
   ```

結果

チュートリアルがコンピュータから削除され、このチュートリアルを最初のレッスンから再び繰り返すことができます。

タスクの概要: チュートリアル:Android アプリケーションの構築 [711 ページ]

前のタスク: レッスン 4:Android アプリケーションのテストと同期 [717 ページ]

4.4 Ultra LightJ API リファレンス

Ultra LightJ には豊富な API オブジェクトがあります。

次に、よく使用される API オブジェクトの一部を示します。

DatabaseManager

データベースと接続を管理する方法を提供します。

接続

Ultra Light データベースへの接続を表します。Connection オブジェクトは 1 つまたは複数作成できます。

SyncParms

Ultra Light データベースを Mobile Link サーバと同期させます。

PreparedStatement、ResultSet

動的 SQL 文の作成、クエリの記述、INSERT、UPDATE、DELETE 文の実行、プログラムによるデータベースの結果セットの制御を行います。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

パッケージ [Android]

```
com.sap.ultralitejni17
```

> **注記**
>
> API リファレンスマニュアルをお探しですか。マニュアルをローカルにインストールした場合は、Windows のスタートメニューを使用してアクセスするか (Microsoft Windows)、`C:¥Program Files¥SQL Anywhere 17¥Documentation` にナビゲートします。
>
> また、DocCommentXchange の Web で、SAP SQL Anywhere API リファレンスマニュアルにアクセスすることもできます。http://dcx.sap.com

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

5 Ultra Light - WinRT プログラミング

このマニュアルでは、Ultra Light WinRT プログラミングインタフェースについて説明します。Ultra Light for WinRT は、データベース機能を備えており、Windows Phone や Windows ストアアプリとの同期が可能です。

このセクションの内容:

システムの要件とサポート対象プラットフォーム [721 ページ]
Ultra Light for WinRT は、いくつかのターゲットプラットフォームをサポートしています。サードパーティ製ソフトウェアは、Ultra Light for WinRT を使用する必要があります。

Ultra Light for WinRT アプリケーション開発 [722 ページ]
Ultra Light for WinRT は、データベース機能を備えており、Windows Phone や Windows ストアアプリとの同期が可能です。

チュートリアル: Windows Phone アプリケーションの構築 [728 ページ]
Ultra Light for WinRT API と Microsoft Visual Studio 環境を使用して、Windows Phone 用アプリケーションを開発します。このチュートリアルでは、アプリケーションが Windows Phone エミュレータ上で実行されることを前提としています。

Ultra Light for WinRT API リファレンス [733 ページ]
Ultra Light for WinRT には C#、C++、または JavaScript から使用する豊富な API オブジェクトがあります。

このマニュアルの印刷、再生、および再配布 [734 ページ]
次の条件に従うかぎり、このマニュアルの全部または一部を使用、印刷、再生、配布することができます。

5.1 システムの要件とサポート対象プラットフォーム

Ultra Light for WinRT は、いくつかのターゲットプラットフォームをサポートしています。サードパーティ製ソフトウェアは、Ultra Light for WinRT を使用する必要があります。

システムの稼働条件

Ultra Light for WinRT アプリケーションを開発するには、次のソフトウェアが必要です。

- Windows 8.1
- Windows 10
- Windows 10 SDK
- Microsoft Visual Studio 2013 以降
- Windows Phone SDK 8.1
  Windows Phone SDK には、Hyper-V 上で仮想マシンとして動作するエミュレータが含まれます。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

- Windows 8.1 ランタイム依存性: Microsoft Visual C++ 2013 Runtime Package
- Windows 10 ランタイム依存性: Microsoft Visual C++ 2013 Runtime Package for Windows Universal

ターゲットプラットフォーム

Ultra Light for WinRT は、次のターゲットプラットフォームをサポートしています。

- Windows Embedded 8.1 Handheld を含む、Windows Phone 8.1、ARM または x86 CPU
- Windows 8.1 システム、x86 または x64 CPU
- Windows 10 Universal Windows Platform (UWP)、x86、x64、または ARM CPU

関連情報

Windows Phone SDK 8.0 ダウンロードセンタ

5.2 Ultra Light for WinRT アプリケーション開発

Ultra Light for WinRT は、データベース機能を備えており、Windows Phone や Windows ストアアプリとの同期が可能です。

API には、Ultra Light データベースに接続し、スキーマ操作を実行し、SQL 文を使用してデータを管理し、データを暗号化するために必要なすべてのメソッドが含まれます。

このセクションの内容:

Ultra Light for WinRT アプリケーション開発のクイックスタートガイド [723 ページ]
アプリケーションを作成するときは、一般にアプリケーションコードでいくつかのデータ管理タスクを実行します。

Ultra Light for WinRT の設定の考慮事項 [723 ページ]
WinRT アプリケーションを開発する前に、API の次の考慮事項に留意してください。

スキーマ操作とデータ管理のクイックスタートガイド [724 ページ]
SQL 文とクエリを使用してデータベースのテーブル、インデックス、外部キー、パブリケーション、ローの作成または取得を行います。

Windows Phone または Windows ストアアプリ用 Ultra Light アプリケーションの配備 [726 ページ]
適切な作成パラメータ、接続パラメータ、同期パラメータ、プロトコルオプション、リンクライブラリ、メソッド呼び出し、配備ファイルを指定して、Ultra Light アプリケーションが Windows Phone デバイスまたは Windows ストアアプリで正常に実行されるようにします。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

関連情報

Ultra Light 概要 [8 ページ]
Microsoft Windows Mobile 用の Ultra Light API の利点 [21 ページ]

5.2.1 Ultra Light for WinRT アプリケーション開発のクイックスタートガイド

アプリケーションを作成するときは、一般にアプリケーションコードでいくつかのデータ管理タスクを実行します。

Ultra Light for WinRT では、Windows Phone Runtime API モデルが使用されます。C++ または C# を使用してアプリケーションを実装し、目的の言語への API 射影を使用します。

1. DatabaseManager オブジェクトを初期化します。
2. 既存のデータベースへの接続を開きます。または、Database Manager オブジェクトのメソッドを使用して新しいデータベースを作成します。
3. アプリケーションが終了したら、DatabaseManager オブジェクトをファイナライズします。

関連情報

Ultra Light for WinRT の設定の考慮事項 [723 ページ]
Ultra Light SQL 文 [475 ページ]

5.2.2 Ultra Light for WinRT の設定の考慮事項

WinRT アプリケーションを開発する前に、API の次の考慮事項に留意してください。

安全なデータ同期

Windows Phone Runtime API の制限により、安全なデータ同期には次の注意事項があります。

- ネットワークプロトコルには HTTPS を使用する必要がある
- クライアント認証はサポートされていない
- trusted_certificate=filename プロトコルオプションはサポートされていません。アプリケーションは、HTTPS 同期操作の実行前に、信頼できるルート証明書を信頼できる証明書ストアに追加する必要がある

次の Mobile Link ネットワークプロトコルオプションは、クライアントによってサポートされていません。

- certificate_company

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

- certificate_name
- client_port
- e2ee_public_key
- fips
- identity
- identity_name
- identity_password
- identity_adapter_name
- network_leave_open
- network_name
- trusted_certificates
- trusted_certificate_name

Windows Phone アプリケーションの設定

`WMAppManifest.xml` ファイルで次の機能を定義する必要があります。

<Capability name="ID_CAP_IDENTITY_DEVICE"/> この機能により、良質な GUID の生成に使用されるユニークなデバイス識別子が生成されます。
<Capability name="ID_CAP_NETWORKING"/> この機能は、ネットワーク操作の実行に必要です。

Windows ストアアプリケーションの設定

`WMAppManifest.xml` ファイルで次の機能を定義する必要があります。

<Capability name="internetClient"/> この機能は、package.appxmanifest にあります。

関連情報

Ultra Light データベースの作成方法 [27 ページ]

5.2.3 スキーマ操作とデータ管理のクイックスタートガイド

SQL 文とクエリを使用してデータベースのテーブル、インデックス、外部キー、パブリケーション、ローの作成または取得を行います。

スキーマ操作によってデータを管理する場合、一般的にアプリケーションコードで次のタスクを実行します。

1. スキーマ操作を実行します。
   データベース接続に対して CREATE TABLE または CREATE INDEX などの SQL 文を使用することで、スキーマを管理し変更します。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

2. ロー操作を管理します。
   データベース接続に対して INSERT SQL 文を使用することで、テーブル内のデータを管理します。
3. 結果セットにローデータを取得します。
   SELECT 文を使用して結果セットを取得してから、結果セットのナビゲーションメソッド (前や次など) を使用し、ローデータをトラバースします。

このセクションの内容:

例: Windows Phone に対するデータベース操作の管理 [725 ページ]
この例は、データベースを作成し、基本的なデータオペレーションを実行するサンプルクラスを Windows Phone アプリケーションで作成する方法を示しています。

関連情報

例: Windows Phone に対するデータベース操作の管理 [725 ページ]

5.2.3.1 例: Windows Phone に対するデータベース操作の管理

この例は、データベースを作成し、基本的なデータオペレーションを実行するサンプルクラスを Windows Phone アプリケーションで作成する方法を示しています。

この例は、以下のオペレーションを示しています。

- データベースにテーブルを作成します。
- テーブルに新しいローを挿入します。
- テーブル内のローを更新します。
- テーブルからローを削除します。
- データベースへの変更をコミットします。
- 結果セットを作成することによって、テーブルのすべてのローを選択します。
- 結果セットをトラバースしてデータベースのローを表示します。

```
using UltraLite;
if (DatabaseManager.Init()) {
    // DatabaseManager.Init() succeeded.
} else {
    // DatabaseManager.Init() failed.
}
// Use DatabaseManager.Fini() when terminating the app.
private async void Button1_Click_1(object sender, RoutedEventArgs e)
{
    String mypath = ApplicationData.Current.LocalFolder.Path;
    Connection conn = null;
    int code;
    String errorParms;
    try  {
        conn = DatabaseManager.CreateDatabase("dbf=" + mypath + "¥¥test.udb", "");
        conn.ExecuteStatement("CREATE TABLE T (c1 integer primary key, c2
char(20))");
        PreparedStatement ps = conn.PrepareStatement("INSERT INTO T VALUES(?,?)");
        ps.SetParameterInt32(0, 11);
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

```
        ps.SetParameterString(1, "row 11");
        ps.ExecuteStatement();
        ps.SetParameterInt32(0, 37);
        ps.SetParameterString(1, "row 37");
        ps.ExecuteStatement();
        conn.Commit();
        ps.Close();
        ps = conn.PrepareStatement("SELECT * FROM T");
        IAsyncOperation<ResultSet> op = ps.ExecuteQueryAsync();
        ResultSet rs = await op;
        int i;
        string str;
        while (rs.Next()) {
            i = rs.GetInt32(0);
            str = rs.GetString(1);
            // do something with row values
        }
        rs.Close();
        ps.Close();
    }
    catch (Exception ex) {
        // EXCEPTION HRESULT: ex.HResult
        if (ex.HResult == (int)ErrorCodes.E_ULTRALITE_ERROR) {
            if (conn != null) {
                conn.GetLastError(out code, out errorParms);
            } else {
                DatabaseManager.GetLastError(out code, out errorParms);
            }
            // do something with SQL code and error parameters
        }
    }
    finally {
        if (conn != null) {
            conn.Close();
        }
    }
}
```

関連情報

Windows Phone または Windows ストアアプリ用 Ultra Light アプリケーションの配備 [726 ページ]

5.2.4 Windows Phone または Windows ストアアプリ用 Ultra Light アプリケーションの配備

適切な作成パラメータ、接続パラメータ、同期パラメータ、プロトコルオプション、リンクライブラリ、メソッド呼び出し、配備ファイルを指定して、Ultra Light アプリケーションが Windows Phone デバイスまたは Windows ストアアプリで正常に実行されるようにします。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

手順

1. 次のパラメータを指定します。
   - 難読化を使用している場合は、データベース作成時に作成パラメータ *obfuscate=1* を設定します。
   - AES 暗号化を使用する場合は、データベース作成時または接続時に接続パラメータ *DBKEY*=`encryption-key` を設定します。
2. Ultra Light アプリケーションで同期を使用する場合は、パラメータを適切に設定してください。

同期タイプ	パラメータの設定
TCP/IP	*Stream* 同期パラメータを tcpip に設定します。
HTTP	*Stream* 同期パラメータを http に設定します。
RSA HTTPS	*Stream* 同期パラメータを https に設定します。

3. ZLIB 圧縮を使用している場合は、プロトコルオプション *compression=zlib* を設定します。
4. `UltraLite.winmd` ファイルの参照を Microsoft Visual Studio プロジェクトの Ultra Light WinRT コンポーネントに追加します。

   `UltraLite.winmd` の参照を追加すると、`UltraLite.dll` ライブラリがアプリケーションにパッケージされます。

   ARM、x86、および x64 プロセッサ向け WinRT コンポーネントは、SQL Anywhere インストール環境の `UltraLite¥WinRT¥Windows¥8.0¥` および `UltraLite¥WinRT¥WindowsPhone¥8.0¥` ディレクトリにあります。x86 プロセッサ向け Windows Phone 8 エミュレータは、それぞれのディレクトリ内に含まれています。

結果

Ultra Light アプリケーションは、配備先の Windows Phone デバイスまたは Windows ストアアプリで正常に稼働します。

次のステップ

アプリケーションを配備した Windows Phone デバイスまたは Windows ストアアプリに Ultra Light データベースを配備するか、配備したアプリケーションと一緒に新しいデータベースを作成します。

関連情報

Ultra Light データベースの配備方法 [124 ページ]
Ultra Light アプリケーションのビルドと配備の仕様 [118 ページ]

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

5.3 チュートリアル: Windows Phone アプリケーションの構築

Ultra Light for WinRT API と Microsoft Visual Studio 環境を使用して、Windows Phone 用アプリケーションを開発します。このチュートリアルでは、アプリケーションが Windows Phone エミュレータ上で実行されることを前提としています。

前提条件

次の知識が必要です。

- C# の知識
- Microsoft Visual Studio の知識

次のソフトウェアが必要です。

- Microsoft Visual Studio 2013 以降
- Windows Phone 8.1 SDK 以降
- Ultra Light for WinRT API

コンテキスト

このチュートリアルで使用する Windows Phone アプリケーションは、*%SQLANY17%*`¥UltraLite.WinRT¥WindowsPhone¥CustDb¥` ディレクトリにあります。`¥CustDb` ディレクトリにあるアプリケーションコードで、Ultra Light for WinRT API で行う次の操作を参照できます。

- Ultra Light のリモートデータベースの作成。
- データベースでの SQL の操作。
- Mobile Link を使用した CustDB サンプルデータベースとのデータの同期。

XAML ファイルは、Windows Phone のメニュー項目とインタフェースの作成方法を示します。`CustDb.sln` ソリューションファイルを開き、これらのファイルを Microsoft Visual Studio で表示します。

`WMAppManifest.xml` ファイルが修正され、ネットワークへのアクセスと、アプリケーションによる良質な GUID の生成が可能になりました。次の Capability 文が追加されました。

```
<Capability Name="ID_CAP_NETWORKING" />
<Capability Name="ID_CAP_IDENTITY_DEVICE" />
```

このセクションの内容:

レッスン 1: 新しい Windows Phone アプリケーションの設定 [729 ページ]
Microsoft Visual Studio を使用して、新しい Windows Phone ソリューションを作成します。

レッスン 2: Mobile Link サーバの起動 [730 ページ]
Mobile Link サーバを起動して、同期を実行します。

レッスン 3: Windows Phone アプリケーションの実行と同期 [731 ページ]

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

Windows Phone アプリケーションを使用して、Ultra Light のリモートデータベースを更新し、CustDB 統合データベースと同期します。

レッスン 4: クリーンアップ [732 ページ]
チュートリアルをコンピュータから削除します。

5.3.1 レッスン 1: 新しい Windows Phone アプリケーションの設定

Microsoft Visual Studio を使用して、新しい Windows Phone ソリューションを作成します。

コンテキスト

このチュートリアルは、開発者が C# と Microsoft Visual Studio に精通していることを前提としています。

手順

1. Microsoft Visual Studio 2013 で *%SQLANYSAMP17%*`¥UltraLite.WinRT¥WindowsPhone¥CustDB¥CustDb.sln` を開きます。
2. プロジェクトをビルドします。ビルド > ソリューションのビルド をクリックします。

   プロジェクトが正常にビルドされました。

結果

Ultra Light for WinRT API が新しい Windows Phone アプリケーションで機能します。

次のステップ

次のレッスンに進みます。

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

5.3.2 レッスン 2: Mobile Link サーバの起動

Mobile Link サーバを起動して、同期を実行します。

前提条件

このチュートリアルのこれまでのレッスンを完了している必要があります。

手順

1. Mobile Link を起動します。 スタート > *SQLAnywhere 17* > *Mobile Link* > 同期サーバのサンプル の順にクリックします。
2. エミュレータの Windows Phone アプリケーションが Mobile Link サーバに接続できるようにします。Windows ファイアウォールで適切な受信ポートを開きます。TCP/IP のデフォルトは 2439 です。これは、Mobile Link サーバの IANA 登録ポート番号です。
   a. スタート > コントロールパネル > *Windows* ファイアウォール をクリックします。
   b. 詳細設定をクリックします。
   c. 受信の規則をクリックします。
   d. 新しい規則をクリックします。
   e. ポートをクリックします。次へをクリックします。
   f. *TCP* が選択されていることを確認します。特定のローカルポートを選択します。`2439` と入力します。次へをクリックします。
   g. 接続を許可するを選択します。次へをクリックします。
   h. 名前フィールドに `MobiLink Incoming` と入力します。完了をクリックします。

結果

Mobile Link サーバが起動します。TCP ポート 2439 が受信トラフィックを受け付けられるようになります。

次のステップ

次のレッスンに進みます。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

5.3.3 レッスン 3: Windows Phone アプリケーションの実行と同期

Windows Phone アプリケーションを使用して、Ultra Light のリモートデータベースを更新し、CustDB 統合データベースと同期します。

前提条件

このチュートリアルのこれまでのレッスンを完了している必要があります。

手順

1. CustDB アプリケーションの初回実行時には、設定画面が表示されます。
   - *Employee ID* は、アプリケーションを使用している従業員を特定します。CustDB アプリケーションでは、その従業員に関連するデータのみが表示されます。デフォルトは 50 です。
   - *Host* は、Mobile Link サーバを実行しているコンピュータを特定します。ローカルエリアネットワーク上の Mobile Link サーバに接続している場合は、コンピュータのローカル IP アドレスまたはホスト名を使用できます。コンピュータの IP アドレスを特定するには、コマンドラインで `ipconfig` を実行します。
   - *Port* は、Mobile Link サーバで使用される受信ポート番号を表します。デフォルトは 2439 です。
2. *Orders* 画面で、*Sync* をクリックします。

   アプリケーションは自動的に同期を実行し、一連の顧客、製品、注文情報が CustDB 統合データベースからアプリケーションにダウンロードされます。個々の注文情報を上下にフリックしたりタップしたりすることによって注文情報を参照し、*Order Details* 画面を表示します。
3. 新しい注文を追加します。*New* をクリックします。フォームに次の情報を入力します。

オプション	説明
Customer	Ace Properties
Product	4x8 Drywall x100
Quantity	999
Discount	25

4. *OK* をクリックし、新規注文を追加します。
5. *Sync* をクリックして、CustDB 統合データベースとアプリケーションを同期します。
6. 同期が成功したことを確認します。Interactive SQL を使用して、CustDB 統合データベースに接続します。
   a. **スタート** > *SQL Anywhere 17* > **管理ツール** > *Interactive SQL* をクリックします。
   b. *ODBC データソース名*をクリックし、*SQL Anywhere 17 CustDB* を選択します。
   c. **接続**をクリックします。
7. 同期が成功したことを確認します。

   次の文を実行します。

```
SELECT order_id, disc, quant, notes, status, c.cust_id,
```


© 2016 SAP SE or an SAP affiliate company. All rights reserved.

```
    cust_name, p.prod_id, prod_name, price
  FROM ULOrder o, ULCustomer c, ULProduct p
  WHERE o.cust_id = c.cust_id
  AND o.prod_id = p.prod_id
  AND c.cust_name = 'Ace Properties'
  AND p.prod_name = '4x8 Drywall x100'
```

Interactive SQL に注文にエントリが表示されたら、同期が成功しています。

8. エミュレータウィンドウを閉じます。

結果

エミュレータに加えた変更は CustDB 統合データベースと同期されます。

次のステップ

次のレッスンに進みます。

5.3.4 レッスン 4: クリーンアップ

チュートリアルをコンピュータから削除します。

前提条件

このチュートリアルのこれまでのレッスンを完了している必要があります。

手順

1. Microsoft Visual Studio を閉じます。
2. タスクバー上で、Mobile Link、Interactive SQL、同期クライアントの各ウィンドウを右クリックし、終了またはシャットダウンをクリックします。
3. CustDB データベースをリセットします。

   %SQLANYSAMP17%¥UltraLite¥CustDb ディレクトリから次のコマンドを実行します。

```
makedbs
```


© 2016 SAP SE or an SAP affiliate company.All rights reserved.

結果

チュートリアルがコンピュータから削除され、このチュートリアルを最初のレッスンから再び繰り返すことができます。

5.4 Ultra Light for WinRT API リファレンス

Ultra Light for WinRT には C#、C++、または JavaScript から使用する豊富な API オブジェクトがあります。

次に、よく使用される API オブジェクトの一部を示します。

DatabaseManager

データベースと接続を管理する方法を提供します。

接続

Ultra Light データベースへの接続を表します。Connection オブジェクトは 1 つまたは複数作成できます。

PreparedStatement、ResultSet

動的 SQL 文の作成、クエリの記述、INSERT、UPDATE、DELETE 文の実行、プログラムによるデータベースの結果セットの制御を行います。

ネームスペース

```
UltraLite
```

> **注記**
>
> API リファレンスマニュアルをお探しですか。マニュアルをローカルにインストールした場合は、Windows のスタートメニューを使用してアクセスするか (Microsoft Windows)、`C:¥Program Files¥SQL Anywhere 17¥Documentation` にナビゲートします。
>
> また、DocCommentXchange の Web で、SAP SQL Anywhere API リファレンスマニュアルにアクセスすることもできます。http://dcx.sap.com

関連情報

Ultra Light for WinRT の設定の考慮事項 [723 ページ]

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

5.5 このマニュアルの印刷、再生、および再配布

次の条件に従うかぎり、このマニュアルの全部または一部を使用、印刷、再生、配布することができます。

1. ここに示したものとそれ以外のすべての版権と商標の表示をすべてのコピーに含めること。
2. マニュアルに変更を加えないこと。
3. SAP 以外の人間がマニュアルの著者または情報源であるかのように示す一切の行為をしないこと。

ここに記載された情報は事前の通知なしに変更されることがあります。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

重要免責事項および法的情報

コードサンプル

この文書に含まれるソフトウェアコード及び / 又はコードライン / 文字列 (「コード」) はすべてサンプルとしてのみ提供されるものであり、本稼動システム環境で使用することが目的ではありません。「コード」は、特定のコードの構文及び表現規則を分かりやすく説明及び視覚化することのみを目的としています。SAP は、この文書に記載される「コード」の正確性及び完全性の保証を行いません。更に、SAP は、「コード」の使用により発生したエラー又は損害が SAP の故意又は重大な過失が原因で発生させたものでない限り、そのエラー又は損害に対して一切責任を負いません。

アクセシビリティ

この SAP 文書に含まれる情報は、公開日現在のアクセシビリティ基準に関する SAP の最新の見解を表明するものであり、ソフトウェア製品のアクセシビリティ機能の確実な提供方法に関する拘束力のあるガイドラインとして意図されるものではありません。SAP は、この文書に関する一切の責任を明確に放棄するものです。ただし、この免責事項は、SAP の意図的な違法行為または重大な過失による場合は、適用されません。さらに、この文書により SAP の直接的または間接的な契約上の義務が発生することは一切ありません。

ジェンダーニュートラルな表現

SAP 文書では、可能な限りジェンダーニュートラルな表現を使用しています。文脈により、文書の読者は「あなた」と直接的な呼ばれ方をされたり、ジェンダーニュートラルな名詞 (例: 「販売員」又は「勤務日数」) で表現されます。ただし、男女両方を指すとき、三人称単数形の使用が避けられない又はジェンダーニュートラルな名詞が存在しない場合、SAP はその名詞又は代名詞の男性形を使用する権利を有します。これは、文書を分かりやすくするためです。

インターネットハイパーリンク

SAP 文書にはインターネットへのハイパーリンクが含まれる場合があります。これらのハイパーリンクは、関連情報を見いだすヒントを提供することが目的です。SAP は、この関連情報の可用性や正確性又はこの情報が特定の目的に役立つことの保証は行いません。SAP は、関連情報の使用により発生した損害が、SAP の重大な過失又は意図的な違法行為が原因で発生したものでない限り、その損害に対して一切責任を負いません。すべてのリンクは、透明性を目的に分類されています (http://help.sap.com/disclaimer を参照)。

© 2016 SAP SE or an SAP affiliate company.All rights reserved.

go.sap.com/registration/contact.html

© 2016 SAP SE or an SAP affiliate company.All rights reserved.
本書のいかなる部分も、SAP SE 又は SAP の関連会社の明示的な許可なくして、いかなる形式でも、いかなる目的にも複製又は伝送することはできません。本書に記載された情報は、予告なしに変更されることがあります。SAP SE 及びその頒布業者によって販売される一部のソフトウェア製品には、他のソフトウェアベンダーの専有ソフトウェアコンポーネントが含まれています。製品仕様は、国ごとに変わる場合があります。
これらの文書は、いかなる種類の表明又は保証もなしで、情報提供のみを目的として、SAP SE 又はその関連会社によって提供され、SAP 又はその関連会社は、これら文書に関する誤記脱落等の過失に対する責任を負うものではありません。SAP 又はその関連会社の製品及びサービスに対する唯一の保証は、当該製品及びサービスに伴う明示的保証がある場合に、これに規定されたものに限られます。本書のいかなる記述も、追加の保証となるものではありません。
本書に記載される SAP 及びその他の SAP の製品やサービス、並びにそれらの個々のロゴは、ドイツ及びその他の国における SAP SE（又は SAP の関連会社）の商標若しくは登録商標です。本書に記載されたその他すべての製品およびサービス名は、それぞれの企業の商標です。
商標に関する詳細の情報や通知については、http://www.sap.com/corporate-en/legal/copyright/index.epx をご覧ください。